Assessment of Options for Handling Full Unicode in Character Encodings in MARC 21

Part 2: Issues

1 Introduction

MARC 21 adopted Unicode as an alternative character set in 1994, thus providing for encoding many more characters than are in the MARC-8 sets. The decision was made at the time to stop adding characters to MARC-8, as they would be automatically available with the transition to Unicode. Since that time the community has focused on establishing mappings from the 16,000+ characters in the MARC-8 repertoire of characters to Unicode, in order to facilitate migration of data to that new character set.

With a number of systems and installations now able to use Unicode encoding for MARC 21, systems are starting to use the full Unicode character repertoire for records. This paper discusses some of the issues this brings up because of the size and characteristics of the new repertoire, the need to maintain communication with MARC-8 systems, and the consideration that needs to be given to some applications.

In the pre-Unicode environment, the need for characters beyond those in ASCII spawned many coded character sets with overlapping character repertoires. Definition of a fixed number of standard coded character sets for use in MARC records was important for the development of efficient record interchange by the library community. It should be noted, however, that the character sets specified for MARC 21 are used for the **exchange** of records, and many institutions use alternative encodings (e.g., EBCDIC, precomposed diacritics, scripts or characters not specified for MARC 21) **within** systems or specifically defined interchange groups.

In Unicode (ISO/IEC 10646) the library community shares with other communities a single character set, universal in scope. The pre-Unicode need for many character sets addressed by escape sequences that permit redefinition of the code points of a small space goes away.

Unicode has been in use long enough that the Unicode Consortium has developed extensive technical documentation and tables to guide and support the character set and many vendors have developed Unicode-aware software ranging from large application systems to miscellaneous desktop tools, frequently available as free- or shareware.

However, it will take several more years before all library systems become Unicode compliant. In the meantime there will be a need to exchange MARC data encoded either in Unicode or in MARC-8. Users of Unicode-capable systems are eager to take advantage of the larger character repertoire now available to them. They may need to communicate with older systems that are limited to the relatively small MARC-8 repertoire. Senders will want to send as much data as they can in a way that the recipients can process according to their more limited capabilities. Conventions are needed to support this kind of transfer.

This report is a companion to a report by Jack Cain circulated in January 2004. That report, Part 1: New Scripts, discussed the new scripts and new characters in existing scripts that are introduced via Unicode and described various techniques for handling those scripts in existing MARC-8 environments, along with display and font issues. This Part 2 report on some special issues was prepared by Sally McCallum with the collaboration of Joan Aliprand, Joe Altimus, Jack Cain, Charles Husbands, and Gary Smith.

2 Background on the Character Sets Used with MARC 21

The following describes the origins of the MARC-8 character sets and the work of task groups on Unicode and their decisions over the period 1996-2001.

2.1 MARC-8 character encoding and repertoire

The following are the 8-bit character sets that were specified for use with the MARC format in a pre-Unicode environment.

• ASCII (ANSI X3.4 and its international version ISO 646 (IRV)),

• ANSI/NISO Z39.47 (ANSEL) - Extended Latin set (containing 36 special spacing characters and 29 non-spacing diacritics used in Latin languages and transliterations of non-Latin languages into the Latin script).

Libraries and other information agencies had the underlying need to be able to present information in multiple languages, and even different scripts, next to each other. This requirement made an extended repertoire of Latin script-based characters essential. They needed characters that appeared in older and non-mainstream literature, not just those appearing in "current" literature. Libraries also need a large number of diacritic/alphabetic character combinations in order to be able to transcribe information as it is found on bibliographic items and for transliteration. The Latin extensions to ASCII, later codified as ANSEL, were developed in the mid 1960s before the larger IT environment had character sets and tools that could be adopted.

In the 1970s and 1980s, several non-Latin sets were approved for use with the MARC format:

• Chinese, Japanese, and Korean (ANSI/NISO Z39.64, EACC) - ~15,852 characters

- Hebrew (ISO 8957)
- Arabic, basic (ISO 9036) and extended (ISO 11822)
- Cyrillic, basic (ISO Registration #37) and extended (ISO 5427)
- Greek (ISO 5428)

A decision was made in the MARC21 community in the early 1990s not to add to MARC-8 any more script sets or characters to existing sets but to essentially freeze it since the implementation of Unicode would yield an expanded character repertoire. While the community appears to have been overly optimistic about when the practical use of Unicode would be possible, it was the case that adding sets and/or characters to the MARC-8 encoding was expensive to implement across the many complex and varied library systems by then deployed – without assistance from the equipment manufacturers that was expected to be available with Unicode.

Initial steps toward the introduction and use of Unicode with MARC records were taken in 1994 with the first of a series of task groups. This work focused on defining the mapping of the MARC-8 character repertoire (those 16,000+ characters specified above) to Unicode and establishing how Unicode encoding was to be used with MARC 21. As we move into a fuller use of Unicode it is useful to review those decisions in order to consider their continuing validity.

2.2 Mapping the MARC-8 repertoire to Unicode

Between 1994 and 2001 all 16000+ MARC-8 characters were mapped to the Unicode. (See http://www.loc.gov/marc/specifications/speccharintro.html) This was an important step as it enabled systems to easily and consistently convert existing data to Unicode encoding as institutions began implementations. These mappings tried to ensure that round trip movement of MARC-8 characters would be supported wherever possible. It was recognized that conversion of bibliographic data between the MARC-8 encoding and Unicode encoding would be needed for various purposes for many years as systems and equipment became able to handle Unicode.

Letters with diacritics were mapped to the Unicode decomposed forms, continuing the MARC-8 preference. Use of the Private Use Area (PUA) of Unicode was largely avoided. In the initial mappings only 296 characters were mapped to the PUA, and in 2004, many of those characters were unified and remapped to their related variants or to newly defined characters in Unicode. Use of the PUA limits interoperability with applications outside the library community and limits library use of standard software, since others can and do use the private use codes in different ways. (See recommendation on the remaining PUA in Section 7 below.)

Many mapping issues such as one to many matches, near matches, character appearance differences, duplication in MARC-8 numbers and punctuation, etc., were worked out and mappings made for all MARC-8 sets. Round trip mapping from MARC-8 to Unicode and back to MARC-8 was largely but not totally enabled.

2.3 Record and identification issues

A number of decisions and recommendations were also proposed by a task group and approved in the 1990s that related to using Unicode in MARC 21 communications records.

Task group decisions:

- The Unicode encoding used in MARC21 shall be UTF-8.
- Positionally defined data, such as in the Leader and Directory, in fields 006, 007, and 008, in certain subfields of other fields, in indicators, and in subfield codes, shall be restricted to ASCII characters to ensure that these parts of the record can be readily interpreted in either MARC-8 or UTF-8.
- Lengths in the MARC record shall be specified by number of octets, rather than number of characters. Lengths are specified in the MARC Leader (length of record, length of indicators, length of area before the first data field, etc.) and in the MARC Directory. This stipulation means that a record with hex values above U+00FF will have reduced character capacity, since those characters require more than one UTF-8 octet while the maximum number of octets in a field or record cannot change.
- Text in UTF-8 MARC records shall follow the Unicode rule that diacritics follow rather than precede the character they modify.
- A UTF-8 MARC record shall be identified by a code in Leader/09.
- Explicit indication of the use of subsets of Unicode in a record shall not be supported.
- Field 066, Character Sets Present, shall not be used in Unicode records. This field is used in MARC-8 to indicate which pre-Unicode character sets to expect in the record, but Unicode is a single unitary set.

Task group recommendations:

- A MARC <u>record</u> should not mix encodings - it should be either MARC 8 or Unicode (UTF-8)
- A MARC <u>file</u> of records should not mix encodings, all records in a transmitted file should be MARC or Unicode (UTF-8).

2.4 Since 2001

Various small adjustments have been made to the MARC-8 to Unicode mapping as Unicode editions were published and MARC based systems gained implementation experience. For example, all but 61 of the PUA characters were mapped to equivalents in Unicode. A decision was also made not to map any MARC-8 CJK characters to the Unicode Compatibility Ideographs so a few characters were remapped to the CJK Unified Ideographs in Unicode.

The above decisions and mappings served as the foundation for many systems to move all or parts of their data and system components to a Unicode basis. With the adoption of Unicode, the library community went beyond the MARC-8 repertoire, but there are a number of characteristics of Unicode that must be understood in order to adjust to the new character set behaviors in the MARC 21 record exchange environment. It is also the case that the community will not be in a total Unicode environment for many years, so interchange of records that are "full" Unicode, MARC-8 repertoire in Unicode, and MARC 8 encoding needs to be analyzed.

3 Characteristics of Unicode

Using all of Unicode is often viewed as simply adding additional scripts. However, while Unicode does that admirably, the original ideal of one shape (glyph), one character, one encoding, was not always possible in Unicode for a variety of practical reasons. Therefore, for example, the Latin capital letter H has several separately coded related representations in Unicode: script capital H, black letter capital H, and double-struck capital H. The colon (:) has several other Unicode characters that look like it in shape: Armenian full stop, Hebrew punctuation sof pasuq, and the ratio sign. Some of the characteristics of Unicode that need to be considered in practical use are the following.

Compatibility characters - A major principle in Unicode was to unify characters within scripts across language groups. Unified characters would then be given a single encoding. This rule was followed as much as possible but even after unification some existing national sets still had characters that they deemed unique and that they needed to be able to code for Unicode to be useful to them. Thus a number of "compatibility" characters were added, largely to assure round trip convertibility with prominent national or international standards. The compatibility characters are really variants of nominal characters. Examples include half-width or full width characters found in East Asian encoding standards, Arabic contextual form glyphs from preexisting Arabic standards, CJK ideographs that are variants or duplicates of unified Han ideographs. These characters are generally in areas of Unicode labeled Compatibility or Specials, but there are some similarly duplicative characters in other sections.

Precomposed characters with diacritics - Unicode provides letters-with-diacritics precomposed as single characters, and also the diacritics alone (called combining diacritical marks) for use in combination with base alphabetic letters. Thus there are two ways to encode many of the more common letter+diacritic combinations in Unicode. MARC-8 specifies only the use of decomposed diacritic strings in most cases, in order to accommodate highly unusual letter+diacritic combinations that occur in bibliographic data.

Digraphs - Unicode includes a number of digraphs (e.g., lj, nj, ij), which are encoded as separates in the MARC 8 repertoire.

Punctuation. Many of the punctuation signs have multiple encodings in Unicode as they are visually different in different scripts and even in different languages. Thus the quote mark, for example has several different encodings in Unicode, e.g., neutral quotation mark, angle bracket, double angle bracket, and left and right double quotation marks.

Case - For the scripts that have case (Latin, Greek, Cyrillic, Armenian, Deseret, and archaic Georgian), Unicode separately encodes the upper and lower case forms of the letters. MARC-8 also separately encodes upper and lower case in Latin, Greek and Cyrillic scripts.

Numerals - Numbers are sometimes represented in a different ways in different languages and scripts and for different purposes. These are all given different encodings in Unicode. For example the Roman numerals are given separate encoding in Unicode.

Text characters - There are many single characters in Unicode that are more relevant for textual data than for bibliographic data. Examples are the outline characters such as the numbers and letters in parentheses, in circles, or with periods (e.g., 6.).

Other characters - Unicode also provides encoding for a number of other character items, such as geometric shapes, dingbats, mathematical symbols, braille patterns, symbols, arrows, box drawing, block elements, and miscellaneous controls for various scripts.

With this background, this report examines the following four areas for bibliographic use of full Unicode encoding.
1) Clarification of normalization in the Unicode environment, and understanding of the tools that support different aspects of normalization (Section 4).
2) Techniques for reducing the Unicode set to MARC-8 and/or inclusion of Unicode characters in MARC-8 records (Section 5).
3) Clarification of the use of 880 fields in the Unicode environment (Section 6).
4) Standardization of the library use of Unicode by deprecating the Private Use Area characters (Section 7).

These are discussed below with suggested adjustments or tools that enable efficient use of Unicode.

4 Normalization

MARC 21 does not specify normalization of character strings, however, there are several key functions that the community expects to carry out on bibliographic data that have a dependency on various forms of what is commonly called “normalization” of the characters in a character string. Unicode presents new challenges to a library’s normalization routines worked out under MARC-8. Since the word normalization is used differently in different contexts, “bibliographic normalization” below is defined as harmonizing a character string to a specific subset of characters by, for example, transforming case, diacritics, punctuation, digraphs, certain composite characters, and taking into account certain content designation. “Unicode normalization” is more narrowly defined (see below) and other transformations such as “case folding” are specified separately in Unicode technical reports. Unicode provides useful tools and a common vocabulary for carrying out "normalization" for a variety of purposes that relate first to the operation of individual systems, and second, to the interoperability of systems. The bibliographic community needs to examine the Unicode components of normalization and collation and consider whether they can be adopted across scripts.

4.1 Areas for bibliographic normalization agreement

In the bibliographic environment, some applications are facilitated by multi institutional agreement on normalization, while for others, bibliographic normalization can be a local option.

The following table suggests where the requirement for multi institutional agreement is most important.

Function	**Agreement needed ?**
• Uniqueness matching (e.g., in the authority file context)	Shared, multi-institutional?
• Indexing/searching • Sorting • Record matching	Individual institution
• Display	Individual institution/system

Normalization of indexed data enables searching, and agreements can make it more consistent across systems. For example, will punctuation be kept or dropped in a string? If kept, will the search string keyed by the users also be required to have the punctuation in order to match? In typical authority work, determining the uniqueness of headings, requirements for additions to headings in cases of non-uniqueness, and decisions concerning references are dependent on expected normalization of the heading string. When matching a heading to determine uniqueness, will a diacritic difference in a heading be used to determine uniqueness?

4.2 Unicode technical tables

The Unicode Consortium specifies character property tables as well as algorithms to provide for consistent implementation of the Unicode character set, including components of normalization, folding, collation, etc. The IT industry has incorporated these tables and algorithms into readily available software tools.

Unicode normalization. The Unicode Standard specifies four Unicode Normalization Forms, designated NFC, NFD, NFKC, and NFKD. A particular Normalization Form converts all equivalent sequences of characters to a single preferred representation, according to a particular set of rules for that Form. For further information, see Unicode Standard Annex #15, *Unicode Normalization Forms* (http://www.unicode.org/reports/tr15/). Briefly, the four Unicode normalization forms may be summarized as follows.

Normalization Form D (NFD) which decomposes characters to *exact equivalents*, primarily affecting precomposed letter-with-diacritic combinations.

Normalization Form KD (NFKD) which decomposes characters to *exact equivalents* and also converts a number of other differences to a "*compatible equivalent*". For example, all of the characters labeled compatibility in Unicode (especially in the Fxxx range) are converted to corresponding nominal forms, circled characters to non-circled forms, width variants and size variants to nominal width and size, and super and subscripts to nominal characters. Also digraphs, character/punctuation combinations, and fractions are decomposed to character strings (e.g., "c/o", "II", "ffi", "5.", "(7)", "1/4").

Normalization Form C (NFC) which *composes* characters to exact equivalents, primarily decomposed letters-with-diacritics.

Normalization Form CD (NFCD) which essentially decomposes and converts as in NFKD but then composes the exact equivalents (primarily letter-with-diacritic combinations) as in NFC.

These normalizations are worked out for all characters in Unicode. For the Latin character subset, NFKD would be the one most like the current normalization used for indexing and searching and for uniqueness matching. This would provide digraph expansion and *enable* the application of components of bibliographic normalization such as eliminating diacritics.

Unicode case folding specifications. The scripts that have case are the Latin, Greek, Cyrillic, Armenian, Deseret, and archaic Georgian. The Unicode Consortium provides tables that convert the case in those scripts to capital or small letters.

Other Unicode tables. In addition to normalization and case folding, there are Unicode-supplied tables for removing diacritics, including strokes, hooks, and descenders (DiacriticFolding) and for folding Han radicals, Hiragana, Katagana, and Simplified Han. Unicode makes clear that desirable character transformations for search/indexing and uniqueness may be different for different languages, requiring local adjustments to the published tables.

Unicode Collation Algorithm. The Unicode Consortium specifies a complete, unambiguous, specified ordering for all characters in Unicode through the Unicode Collation Algorithm. This algorithm incorporates many of the actions of the tables mentioned above, and specifies meaning and assignment of collation levels, including whether a character is ignorable by default in collation; and a complete specification of the rules for using the level of weights to determine the default collation order of strings. Significantly, it includes override mechanisms for creating language-specific orderings. The Consortium has tuned this algorithm so that it can be efficiently implemented, both in terms of performance and in terms of memory requirements. This will be a key component of library systems using Unicode.

4.3 Bibliographic normalization

Other current bibliographic normalization conventions. Current library normalization may also include other conventions transforming some characters. For example, in the Latin alphabet area, the NACO normalization transforms the O with a stroke to O; discards punctuation; and closes up spaces left by certain characters. It also may drop characters such as % or * and keep others such as & and musical flat. However, Unicode resources can be exploited to define the normalization that the NACO rules are now used to govern, and they can provide a framework for managing other locale-based normalization.

Possible Action: Bibliographic programs that depend on shared bibliographic normalization for efficient cooperation and for consistency for end users will need to determine the normalization conventions they will follow for the larger Unicode repertoire. The library community will want to adopt as far as possible normalization, folding, etc. specified by the Unicode Consortium tables, and especially the Unicode Collation Algorithm. These specifications will be implemented in IT industry software, and their use will minimize exceptions that will require customization.

5 Record Distribution

Initial principles for use of Unicode with MARC specified that 1) UTF-8 encoding would be used for Unicode interchange, and 2) MARC-8 and Unicode would not be mixed in a record. For the community's transition period from MARC-8 to Unicode, which may be long, and for standardization and efficiency in the community, there are several character set transformations that might be standardized for MARC 21 record exchange. Also, within a cooperative group of users, what controls, if any, on the expansion of the characters used in Unicode need to be considered. For example, the used repertoire might be expanded only as conventions for the new characters are established and, in particular, their treatment by standard software based on the Unicode normalization and folding tables and the Unicode Collation Algorithm fully understood.

5.1 Unicode distribution

5.1.1 Unicode origin records to Unicode system

As agreed, the preferred encoding for MARC 21 records in the interchange environment from a Unicode system is UTF-8. There are few MARC 21 restrictions on the Unicode characters included, although distributors may want to impose restrictions in deference to their client community. The Unicode standard itself specifies that two specific classes of characters that are not to be used: those designated as "noncharacters" on the properties list and those that are deprecated in Unicode. Appendix C proposes additional characters that should not be used, in particular a number of control characters.

5.1.2 MARC-8 origin records to Unicode system

Character conversion from MARC-8 to Unicode is required to convert records received for load from a MARC-8 system or to convert records on the fly received through search protocols such as Z39.50 from a MARC-8 system. Mapping tables for converting all of the MARC-8 characters to Unicode have been developed and are maintained on the MARC 21 web site.

In the MARC-8 environment, conventions have been developed for characters not in the MARC-8 set. Cataloging guidelines and community conventions may specify substitution of a word for an unavailable character. For example, the preferred encoding for the Greek symbol gamma in a Latin alphabet string may be [gamma], rather than the Greek character. These strings could be identified and converted in some cases, but are not part of the MARC site conversion tables. For characters that are not in the MARC-8 repertoire of the CJK ideographs one of the following substitution techniques is commonly used: a "geta" character is substituted; a romanization of the character, in square brackets, is substituted; or a "geta" followed by a similar ideograph in square brackets are substituted. Since only the "geta" is present in the data and not the real character it stands for, the correct Unicode character cannot be supplied by the software in the conversion to Unicode.

5.2 MARC-8 distribution

At the current time most large vendor systems have implementations of Unicode, even though not all of their users have moved to them. Additionally while all of these systems claim to be able to load UTF-8 records, many of their customers who have the Unicode versions of vendor system software have not yet tested UTF-8 loading. Since RLG and OCLC have not had requests for UTF-8 encoded records they feel that they must continue to supply the records using MARC-8, and in the future these records will contain Unicode characters that are not convertible to MARC-8. Even though the receiving systems do not handle Unicode, they may want the Unicode preserved in the records.

It may therefore be useful to agree on a standard substitution scheme that collapses the Unicode repertoire to MARC-8, preserving as many of the original characters as possible by folding them to near equivalents or plausible substitutes. The characters are generally not to be omitted but a suitable substitution made if the Unicode character is not in MARC-8. It could be useful to first apply some parts of NFKD to a record, in order to reduce the number of Unicode characters not converting to MARC-8. (If NFKD were not applied, for example, an institution that used precomposed forms in Latin, would need a substitution for each of these precomposed Unicode characters since since MARC-8 only supports decomposed letter/diacritic combinations.)

5.2.1 Converting with individual character markers

Part 1 of these reports (by Jack Cain) enumerated several options for including non-MARC-8 characters in MARC-8 records, several of which involved various substitutions. One of those presented and others suggested on the Unicode web site are summarized in Annex B. These preserve the Unicode information, but they are not overly complex to implement.

5.2.2 Converting with escape sequences to Unicode

Unicode characters not in the MARC-8 repertoire, when encountered in a record to be distributed to a MARC-8 system, could be embedded in the MARC-8 record using a standard escape sequence mechanism. A description of this technique is included in Annex B.

5.3 Where transformations take place

It should be noted that although MARC-8 character sets have been stable, the character set capabilities of MARC 21 based systems have been varied even in the MARC-8 environment. For Latin script oriented implementations, some smaller systems employ only ASCII characters for records, while most systems can accommodate the ASCII and ANSEL characters, and some can handle all or parts of the non-Latin characters defined in MARC-8. These various systems handle MARC-8 according to their capabilities. For example, if an ASCII/ANSEL record came to an ASCII-only system, then a mapping may reduce the set to ASCII. Non-Latin characters entering systems that cannot support MARC-8 non-Latin may be ignored or discarded. It should also be noted that internally systems may use another encoding for the MARC-8 repertoire, such as EBCDIC, so a transformation always occurs when a record is loaded.

With the use of Unicode in MARC 21, any Unicode character that is valid for exchange may occur in a MARC 21 record, whether the Unicode character has a MARC-8 equivalent or not. The receiver would be responsible for converting the characters into an internal set, which might also be less extensive than full Unicode. This is similar to the current practice with MARC-8 in which records may include any of the 16,000+ MARC-8 characters and the receiving site makes the necessary adjustments to the local situation: an internal set, reduced Unicode, ASCII, MARC-8, etc.

Possible Action: Further specify the substitution and/or escape sequence options for transmitting non-MARC-8 characters in a MARC-8 record.

6 Multi Script Records and 880 Fields

In MARC 21, Latin and non-Latin data content may occur in any field. The only character set restrictions are for the leader, directory, indicators, fixed length data element which occur in certain fields (006, 007, and 008) and subfields (in firlds 533, 7XX, etc.), and subfield codes, which are all required to be in the ASCII repertoire and encoded in single octets. Because of the need to carry the same information in multiple scripts, e.g., vernacular and transliteration, the 880 fields were defined for alternate representations of the data in a corresponding regular field (described in the MARC 21 Bibliographic Format, Appendix D as Model A). In a Model A record, there are preferred scripts and alternate scripts. The preferred script is generally used in the regular fields and the alternate occurs in the 880 fields. In Greece, for example, Greek would be one of the preferred scripts. Latin script, limited to ASCII, must also be a preferred script because it is used for the structure of MARC records (leader, directory, etc.) as described above. Under Model A, a library in Greece might use Greek characters in the regular fields and if any of those fields require transliterated equivalents, they can include transliteration in an 880 field when records are distributed. It is also the case that different scripts may occur incidentally in any field.

In the MARC-8 environment, the Latin script-oriented MARC 21 users have specialized the use of the 880 fields to a pattern that was useful for processing the data in Latin alphabet-oriented systems. For example, RLG and OCLC require that the “regular” fields contain Latin data and that non-Latin data be restricted to 880 fields when records are distributed. This has led to the expectation by the receivers of MARC 21 records that the non-Latin will be in the 880s where it can be used, ignored, or discarded as needed. This practice also led to the need to create unlinked 880 fields when the data in the non-Latin script did not have a corresponding field in the regular sequence. Other script areas may have used the same approach with their vernacular script in the regular fields and all non-vernacular in the 880 fields. Latin script oriented libraries may need to clearly circumscribe the characters that make up the Latin script in the Unicode environment if they plan to continue this practice. An example of a Unicode Latin script subset definition is given in Annex A.

With Unicode, while these practices could continue, there may be an opportunity for less of the record to be transliterated, through the use of more mixed scripts in the regular fields (as described in Model B). Logically the two models would be merged and the 880 fields used only for *duplicated* data in different scripts.

7 Standardization of Library Use of Unicode

When the MARC-8 character set was mapped to Unicode, around 300 CJK MARC-8 characters could not initially be found and were mapped to what is called the Private Use Area (PUA) encoding of Unicode. The characters in this area will not be supported by IT industry-wide software and hardware as the character definitions vary for different communities choosing to use the PUA encodings. Since the initial mapping work, corresponding characters have been found for all but a few of the PUA characters. Remaining in this unsupported area are two small sets of characters. One set contains 35 ideograph components that were included in MARC-8 but were used by RLG for special internal purposes more than a decade ago. RLG says that they should not appear in any MARC records and if they do they are errors and do not have meaning as they are not ideographs. OCLC reports one occurrence of the ideograph component, their new system will not allow the input of these characters.

The rest of the characters remaining in the PUA area are 26 ancient Korean Hangul characters. These characters comprise only a small subset of ancient Korean Hangul, and are inadequate for writing ancient Korean. Furthermore, two experts in Korean have said that about half of the characters are spurious. The source of these characters cannot be identified. They have been very rarely used in both OCLC and RLG databases and many of those uses are thought to be errors. OCLC reports 8 occurrences of these ancient Hangul in their data bases. These will not be able to be displayed in their new Unicode system. If any of the Ancient Hangul characters are documented in the future they will be added to the Unicode repertoire, then they could be available in a standard Unicode environment and supported by standard tools.

Possible Action: Disallow input of these characters in future. Publish instructions for implementers on the MARC 21 site specifying how to deal with these PUA characters when encountered in data. The net result of these proposed actions is deprecation of the PUA mappings.

8 Glossary

MARC-8 - the character encodings currently used for MARC 21 records. It uses 8 bits per character except for CJK ideographs, which have 24 bits per character, and uses escape sequences to switch sets.

MARC-8 repertoire - the set of characters represented by the MARC-8 character encoding. It includes approximately 16,000 characters belonging to Latin, Chinese, Japanese, Korean, Arabic, Greek, and Hebrew scripts.

Full Unicode - All the characters encoded in the Unicode standard (64,000 and growing). There are various encodings for Unicode characters, one being UTF-8.

Unicode nominal characters - the subset of Unicode characters considered the nominal characters excludes the compatibility characters, the specials, and the PUA.

Normalization - used to encompass any systematic adjustment of a character string to make it conform to defined goals, e.g., case folding, digraph expansion, diacritic treatment. See Bibliographic normalization and Unicode normalization in Section 4.

Folding - operations that map similar characters to a common target. Folding operations are most often used to temporarily ignore certain distinctions between similar characters.

Annex A. Latin Script Character Set in Unicode

A.1 MARC-8 Latin subset

For comparison, recall that the Latin script in MARC-8 is the following repertoire:
• ASCII - includes A-Z alphabetics, numerals, common punctuation, symbols, and control characters.
• ANSEL - includes additional alphabetics and other spacing characters and non-spacing diacritics.
• 14 superscripts, 14 subscripts, 3 Greek symbols

The above characters may be classed into the following categories:

- Latin script letters
- Digits (regular, superscript, and subscript)
- Punctuation marks
- Symbols (such as % and $)
- Non-spacing marks
- Greek symbols

(The Greek symbols that are included in the MARC-8 Latin set have been recommended for disuse, with the name of the character (alpha, beta or gamma) in brackets used instead.)

A.2 Defining the Latin script subset of Unicode characters

Proposed Latin script definition:
• Unicode characters with Script Name property *Latin* or *Common*
• Unicode characters has Script Name property *Inherited*, **and** preceding base character has Script Name property Latin or Common

The definition of characters for the preferred Latin script could logically be based on the Unicode character property *Script Name.* The advantage of using a Unicode character property as the basis for this definition is that the Unicode Consortium is responsible for the updates whenever additional characters are added to Unicode. If another basis was used, the library community would have to examine the additions to determine whether the scope of "Latin" had expanded, and promulgate revised information.

Digits, punctuation marks, and symbols are used with scripts other than Latin. Unicode therefore assigns the Script Name "Common" to the Unicode equivalents of the MARC-8 characters in this group. Unicode's combining diacritical marks have the Script Name "Inherited," which means that they inherit their Script Name property from their base character (normally Latin, although Common is not forbidden). Latin script letters have the Script Name property "Latin" (as might be expected). If the Greek symbols are ignored, all the characters currently considered the Latin script in MARC-8 are covered by the three Script Name properties Latin, Common, or Inherited.

The Unicode Character Database files include www.unicode.org/Public/UNIDATA/Scripts.txt which lists the characters with the above Script Name property. It lists 1,037 Latin characters, 448 Inherited, and a very large number of Common characters.

Annex B: Unicode to MARC-8 Techniques

Non-MARC-8 characters are Unicode characters which cannot be mapped to individual MARC-8 characters, even after normalization and compatible substitutions. Prominent examples are totally new character sets, but also characters for scripts included in MARC-8 but beyond the MARC-8 defined repertoire.

B.1 Converting using individual character markers

In Part 1 of this report, several options were presented. Three would not support round-tripping, but the fourth would. That technique was to encode each Unicode character that would not convert to U+nnnn (where nnnn is the hex value of the Unicode character), e.g., "U+03B1". An FAQ on the Unicode Consortium website (www.unicode.org/faq) suggests several more methods including: XML/HTML conventions using &#xnnnn; e.g., "&#x3B1;" to encode each Unicode character and Java conventions using \unnnn, e.g. "\u03B1". The XML convention has the advantage of accommodating variable length hex codes. All of these conventions add two or more octets to each Unicode character.

B.2 Converting Unicode to MARC-8 using escape sequences

Non-MARC-8 characters may be isolated or part of a string of non-MARC-8 characters. When the first non-MARC-8 character is encountered a marker is inserted (see below). All non-MARC-8 characters following the marker are considered a continuation of the string. Incidental spaces or common punctuation (ASCII punctuation? ASCII numerals?) in the string are treated as part of the string. A string of non-MARC-8 characters terminates when a character for which there is a MARC-8 mapping, the subfield delimiter (hex 1F), or the end of field (hex 1E) is encountered. A marker is inserted at the end of the string.

ISO/IEC 2022, *Character Code Structure and Extension Techniques*, identifies escape sequences for escaping to and from Unicode when using 7/8 bit character sets. The string of non-MARC-8 character(s) is enclosed by leading (at the beginning of the string) and following (at the end of the string) markers. The leading marker specified in ISO/IEC 2022 is ESC%G, and the following marker is ESC%@ . The encoding of the enclosed non-MARC-8 characters is UTF-8.

Summary table of cases of non-MARC-8 characters:

Location of non-MARC-8 string	*Source Data*	*Data in MARC-8 Target Record*
Entire subfield	<1F><sfc>uuu<1F|1E>	<1F><sfc>ESC%GuuuESC%@<1F|1E>
Beginning of subfield	<1F><sfc>uuummm …<1F|1E>	<1F><sfc>ESC%GuuuESC%@mmm … <1F|1E>
End of subfield	<1F><sfc>… mmmuuu<1F|1E>	<1F><sfc>… mmmESC%GuuuESC%@<1F|1E>
Middle of subfield	<1F><sfc>mmmuuummm<1F|1E>	<1F><sfc>mmmESC%GuuuESC%@mmm<1F|1E>

sfc = subfield code

uuu = non-MARC-8 characters
mmm = MARC-8 characters

B.3 Use of 066?

The presence of non-MARC-8 characters in a MARC-8 record could be indicated in the 066 field. Alternatives for identifying the presence of non-MARC-8 characters are: the value "%G", for example, in an occurrence of subfield c, or definition of a new subfield d, to provide a more explicit identification. The advantage of a new subfield over use of subfield c that it would be easier to distinguish a record with MARC-8 alternate graphic representations from a record non-MARC-8 characters. This could facilitate processing of incoming records.

Annex C: Proposed Code Point Restrictions in Unicode MARC 21 Records

Certain code points of the Unicode standard not appropriate for inclusion in MARC 21 records are outlined below.

C.1 Noncharacters

Definition from the *Unicode Standard (4.0)*: A code point that is permanently reserved for internal use, and that should never be interchanged. Noncharacters consist of the values U+nFFFE and U+nFFFF (where n is from 0-9, A-F) [a total of 34 code points] and the values U+FDD0 through U+FDEF" [32 code points]. For more information see the *Unicode Standard 4.0*, Section 15.8.

C.2 Deprecated Characters

Definition from the *Unicode standard (4.0)*: "Deprecated character: A coded character whose use is strongly discouraged. Such characters are retained in the standard, but should not be used. Deprecated characters are retained in the standard so that previously conforming data stay conformant in future versions of the standard. Deprecated characters should not be confused with obsolete characters, which are historical. Obsolete characters do not occur in modern text, but they are not deprecated; their use is not discouraged."

Users need to consult the current Unicode standard since such characters appear in various places in the standard and a given code point may be deprecated at any time if a problem with that position is discovered. It should be stated however that deprecated characters are highly uncommon in the standard. Examples can be found in the Tibetan range U+0F71 to U+0F79 resulting from a major revision to the Unicode encoding of this script that took place between versions 2 and 3 of Unicode.

C.3 Control Characters

In UTF-8, ASCII values retain their original one octet values. The values not appropriate for transmission in MARC 21 records are values in the control range from hex 00 through hex 1A, plus hex 1C. Although the values of hex 1B, 1D, 1E and 1F are also control codes, these four values may be present. 1B is the Escape character used in MARC-8, while 1D, 1E, and 1F are part of MARC 21 structure.

Another control code range is the area from U+0080 to U+009F. Only two values in this range are valid for transmission in MARC 21 records in Unicode encoding; these are:

0098	NON-SORT BEGIN / START OF STRING
009C	NON-SORT END / STRING TERMINATOR

These two values correspond to hex 88 and 89 in MARC-8.

C.4 Formatting Characters

Generally formatting characters are not appropriate for use in the transmission of MARC 21 records. Therefore, for example, the value of U+00A0 for non-breaking space would not be appropriate. However, formatting characters related to directionality such as U+200E LEFT-TO-RIGHT MARK and U+200F RIGHT-TO-LEFT MARK may be used.

C.5 PUA Characters

Initially a number of PUA (Private Use Area) characters were defined for CJK ideographs. These should no longer be used. Consult tables on the MARC site for their re-assigned values. (See http://www.loc.gov/marc/specifications/speccharintro.html)

Primary Private Use Area Range: U+E000—U+F8FF
Supplementary Private Use Area Range: U+FFFFE and U+FFFFF.