Unicode Support in Enterprise COBOL

Nick Tindall
Stephen Miller
Sam Horiguchi
August 13, 2003

What is Unicode?

- Industry standard for coded character set – defined by Unicode Consortium and ISO
- Covers all commonly used characters in the world in one code page (vs. one "language" per ASCII, EBCDIC, or EUC code page)
- Characters: text, digits, special characters, symbols, control characters, ...
- Multiple Unicode encoding formats: UTF-8, UTF-16, UTF-32
- "Stateless" encoding: meaning of an encoding unit is self defining

Why Unicode?

- Support global e-business environment:
  - Applications for multi-cultural/multi-geographic businesses
  - Networks of heterogeneous systems
- Enables a common implementation for global applications
  vs.
  separate code page for each geographic area or system platform
- Supported by all key operating system and middleware platforms
- Required by: XML, HTML, Java, ...

UTF-8 and UTF-16 Encoding

- UTF-8 encoding unit: one byte
  - "character": 1 to 4 encoding units
- UTF-16 encoding unit: two bytes
  - "character": 1 or 2 encoding units
  - All characters defined in commonly used EBCDIC and ASCII code pages represented in one encoding unit
- Characters in Latin-1 (Western European) ASCII code page represented consistently
  - "A" = X"41" in UTF-8 or ASCII,
  - "A" = X"0041" in UTF-16

Unicode support in Enterprise COBOL for z/OS and OS/390

- Enable basic Unicode processing for COBOL applications
- Consistent with COBOL 2002 standard
- Interoperate with:
  - DB2 Unicode support
  - Java
  - COBOL XML support

Unicode support overview

- Unicode literal and value clause
- Unicode data type
- New compiler options
  - CODEPAGE()
  - NSYMBOL()
- Collation: binary
- Implicit conversions for EBCDIC data assigned to or compared with Unicode data
- Explicit conversions via intrinsic functions

Unicode literals

- N'αβγ', N'Straße'
  - Literal value in source is encoded in some EBCDIC code page
  - Value is converted to UTF-16 for execution
  - Value limited to those representable by the source program code page
- NX'03B103B203B3'
  - Can be used for characters
    - not supported by editor, or
    - not in code page of source program

Unicode data type

- USAGE NATIONAL, Picture character N

```
01 Japan pic N(20) usage national value N'日本'.
```

- One UTF-16 encoding unit (2 bytes) per PICTURE N character
- "Character" defined in terms of PICTURE symbol positions, for reference modification, character counts, etc.

Compiler options

- CODEPAGE (*ccsid*)
  - Specifies EBCDIC CCSID for:
    - literals in source program
    - contents of alphanumeric and DBCS data items
  - Shipped default is 1140 (Latin-1 with Euro)
- NSYMBOL (DBCS | NATIONAL)
  - 01 X PIC NN. and N'...' are ambiguous: Unicode or DBCS?
  - NSYMBOL option controls default interpretation
  - Note: PICTURE G and G'...' are treated as DBCS regardless of NSYMBOL

Assignment

- NATIONAL, DISPLAY or DISPLAY-1 item may be assigned to NATIONAL item

```
01 Country pic N(20) usage national.
01 USA pic X(13) value 'United States'.
01 Greece pic X(6) value 'Ελλάδα'.
Move USA to Country.
Move Greece to Country.
```

- Numeric integer may be assigned to NATIONAL
- Padding with Unicode space character: X'0020'
- Truncation by 2-byte encoding units
  - Application logic responsible for avoiding partial character truncation when dealing with characters represented in two encoding units

Unicode Compares

- National item may be compared with: national, alphanumeric, DBCS, or numeric integer operand.

```
If Country = N'日本' …
```

- Non-Unicode operand converted to Unicode
- Shorter operand value padded with Unicode blanks
- Byte for byte compare in binary order
  - No culturally sensitive compares
    - e.g. **N'ç'** is not equal **N'c'**, regardless of locale
  - No normalization
    - e.g. **á** (composed) is not equal to **a´** (decomposed)

Other language syntax supporting Unicode

- Statements involving comparisons
  - EVALUATE, IF, INSPECT, PERFORM
  - SEARCH, STRING, UNSTRING
  - SORT, MERGE, Indexed file keys
- Class condition on Unicode data
  - NUMERIC, ALPHABETIC, ALPHABETIC-LOWER, ALPHABETIC-UPPER, class-name
- Unicode arguments for CALL or INVOKE
- INITIALIZE ... REPLACING NATIONAL ...
- Reference modification

Intrinsic conversion functions

- **`FUNCTION DISPLAY-OF(`*`national-data`*` [`*`ccsid`*`])`**
  - returns alphanumeric (EBCDIC) representation of NATIONAL argument.
- **`FUNCTION NATIONAL-OF(`*`ebcdic-data`*` [`*`ccsid`*`])`**
  - returns UTF-16 representation of EBCDIC (DISPLAY or DISPLAY-1) argument.
- If *ccsid* omitted, defaults to value from CODEPAGE() compiler option
- *ccsid* may represent an EBCDIC, ASCII, EUC or UTF-8 code page

Recommendation: use only one EBCDIC code page in a program.

Converting to Unicode

```
01  Unicode-Data   pic  N(20) usage national.
01  EBCDIC-Data    pic  X(20).
01  Greek-Data     pic  X(20).
01  UTF8-Data      pic  X(20).
01  Japanese-Data  pic  G(20) usage display-1.

1) Move EBCDIC-Data to Unicode-Data
2) Move function National-of(EBCDIC-Data) to Unicode-data
3) Move function National-of (Greek-Data, 4971)
      to Unicode-Data
4) Move function National-of (UTF8-Data, 1208)
      to Unicode-Data
5) Move function National-of (Japanese-Data, 1399)
      to Unicode-Data
```

1, 2) Converts EBCDIC-data represented in CCSID in effect via CODEPAGE compiler option to UTF-16

3) Converts EBCDIC Greek (CCSID 4971)data to UTF-16

4) Converts UTF-8 (CCSID 1208)data to UTF-16

5) Converts EBCDIC Japanese (CCSID 1399)data to UTF-16

Converting from Unicode

```
01  Unicode-Data   pic  N(20) usage national.
01  EBCDIC-Data    pic  X(20).
01  Greek-Data     pic  X(20).
01  UTF8-Data      pic  X(20).
01  Japanese-Data  pic  G(20) Usage Display-1.

1) Move function Display-of (Unicode-Data)
      to EBCDIC-DATA
2) Move function Display-of (Unicode-Data, 4971)
      to Greek-Data
3) Move function Display-of (Unicode-Data, 1208)
      to UTF8-Data
4) Move function Display-of (Unicode-Data, 1399)
      to Japanese-Data
```

1) Converts UTF-16 (CCSID 1200) to EBCDIC-data represented in CCSID in effect via CODEPAGE compiler option
2) Converts UTF-16 to EBCDIC Greek (CCSID 4971)
3) Converts UTF-16 to UTF-8 (CCSID 1208)
4) Converts UTF-16 to EBCDIC Japanese (CCSID 1399)

ACCEPT and DISPLAY

- **ACCEPT *national-data* FROM CONSOLE**
  - input data implicitly converted from EBCDIC to Unicode
- **DISPLAY *national-data* UPON CONSOLE**
  - output data implicitly converted from Unicode to EBCDIC
- ACCEPT or DISPLAY from/to devices other than CONSOLE done without implicit conversion
  - Use DISPLAY-OF function to control conversion:
    Display function display-of(Country, 930)

Unicode support in DB2 (V7 and later)

- EBCDIC, ASCII and Unicode data types
- UTF-8 & UTF-16 for Unicode
- Stored data representation at table space level
- Host variables declared as EBCDIC, ASCII or Unicode
  - SBCS mapped to UTF-8
  - DBCS mapped to UTF-16
- Automatic conversion between stored representation and host variable declarations
- Collation order: binary

Using Unicode in DB2 COBOL programs

- Consistent support for Unicode in DB2 and COBOL
  - Same Unicode conversion facility
  - Binary collation
- With DB2 coprocessor (SQL compiler option) CCSID information for NATIONAL, DISPLAY, or DISPLAY-1 host variables is automatically coordinated between COBOL and DB2.

e.g.

```
EXEC SQL DECLARE :X VARIABLE CCSID 1140 END-EXEC
```

is no longer required

COBOL Unicode support and Java interoperability

- Java is based on Unicode
- COBOL:Java interoperability support heavily uses Unicode implicitly, under the covers
- COBOL programmer can use Unicode at application level, to communicate String data to/from Java

COBOL Unicode support and Java interoperability …

Example: Invoke Java, passing String object

```
Class String is 'java.lang.String'.
Class Vendor is 'com.acme.Vendor'.
…
01 Greece pic N(6) usage national value N'Ελλάδα'.
01 CountryString usage object reference String.
01 aVendor usage object reference Vendor.
…
Call 'NewString' using by value JNIEnvPtr
                                address of Greece
                                length of Greece
                    returning CountryString

Invoke aVendor 'setLocation' using by value CountryString
```

COBOL Unicode support and XML processing

- COBOL now contains built-in syntax for processing XML documents
- XML PARSE statement parses XML documents, drives processing procedure for each event
- XML documents may be encoded in UTF-16 Unicode
  - XML documents encoded in UTF-8 may be converted to UTF-16 using the NATIONAL-OF function, then parsed
- XML-NTEXT special register returns to the program the Unicode content from the document, that is associated with each event

COBOL Unicode support and XML processing – example

```
01 XMLdocument  pic N(10000) usage national.

  XML PARSE XMLdocument
    Processing procedure XMLproc
  End-XML.
  …
XMLproc.
  Evaluate XML-Event
    When 'START-OF-ELEMENT'
      If  XML-NText = N'Ελλάδα'
        Display 'Processing <Greece> element'
        …
      End-if
    …
  End-evaluate.
```

System configuration for Unicode

- Unicode support in COBOL and DB2 is based on the package:
  - z/OS Support for Unicode, or
  - OS/390 Support for Unicode
- This support must be installed and configured, on both development and production systems that use:
  - COBOL Unicode support,
  - COBOL Object-Oriented language syntax for Java interoperability, or
  - DB2 Unicode support

Installing "Support for Unicode"

- z/OS V1R2 or later
  - *Support for Unicode* is part of the operating system
  - Documentation:
    - z/OS Support for Unicode: Using Conversion Services (SA22-7649)
    - http://publibz.boulder.ibm.com/epubs/pdf/iea2un20.pdf
- z/OS V1R1 or OS/390 V2R10
  - Install *OS/390 Support for Unicode* from the web:
    - https://www6.software.ibm.com/dl/os390/unicodespt-p
  - Documentation:
    - OS/390 Support for Unicode: Using Conversion Services (SC33-7050)
    - http://publibfp.boulder.ibm.com/pubs/pdfs/os390/cunuge00.pdf
- See Enterprise COBOL Customization Guide

References

- Unicode

  Unicode Consortium: http://www.unicode.org
  IBM DeveloperWorks – Unicode:
  http://www.ibm.com/developerworks/unicode/
  ICU: http://oss.software.ibm.com/icu/

- z/OS Support for Unicode: Using Conversion Services

  http://publibz.boulder.ibm.com/epubs/pdf/iea2un20.pdf

- DB2 for OS/390 and z/OS V7: Installation Guide

  http://www.ibm.com/software/data/db2/os390/library.html

- COBOL for z/OS &OS/390 V3R2 books (LRM, PG, CG)

  http://www.ibm.com/software/awdtools/cobol/zos/library