ISO/IEC JTC 1/SC 2 N 3068
Date: 1998-04-10

ISO/IEC JTC 1/SC 2

CODED CHARACTER SETS

SECRETARIAT: JAPAN (JISC)

DOC TYPE: Request for Comments

TITLE: First Working Draft for Project JTC 1.18.02, Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 2: CJK Unified Ideographs Supplementary plane, General Scripts and Symbols Plane, General Purpose Plane (WG 2 N 1717)
Attachment: Plane 14 Characters for Language Tags (WG 2 N 1670)

SOURCE: ISO/IEC JTC 1/SC 2/WG 2

PROJECT: JTC 1.02.18.02

STATUS: In accordance with Resolution M08.16, this document is circulated to the SC 2 members for review and comment. SC 2 members are requested to forward their comments to the SC 2 Secretariat as soon as possible but not later than 1998-07-30.

ACTION ID: COM

DUE DATE: 1998-07-30

DISTRIBUTION: P, O and L Members of ISO/IEC JTC 1/SC 2
WG Conveners and Secretariats
Secretariat, ISO/IEC JTC 1
ISO/IEC ITTF

NO. OF PAGES: 18

ACCESS LEVEL: Open

WEB ISSUE #: 014

Contact: Secretariat ISO/IEC JTC 1/SC 2 - Toshiko KIMURA
IPSJ/ITSCJ (Information Processing Society of Japan/Information Technology Standards Commission of Japan)*
Room 308-3, Kikai-Shinko-Kaikan Bldg., 3-5-8, Shiba-Koen, Minato-ku, Tokyo 105 JAPAN
Tel: +81 3 3431 2808; Fax: +81 3 3431 6493; Telex: 2425340 IPSJ J; email: kimura@itscj.ipsj.or.jp
*A Standard Organization accredited by JISC

ISO/IEC WD 10646-2

Information technology - Universal Multiple-Octet Coded Character Set (UCS) -

Part 2:

CJK Unified Ideographs Supplementary plane

General Scripts and Symbols Plane

General Purpose Plane

Contents

Annexes

<TBD>

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialised system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields or technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organisations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC1. Draft international Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standards ISO/IEC 10646-1 and 10646-2 were prepared by Joint Technical Committee ISO/IEC JTC1, *Information technology*.

ISO/IEC 10646 consists of the following part, under the general title *Information technology - Universal Multiple-Octet Coded Character Set (UCS):*

- *Part 1: Architecture and Basic Multilingual Plane*
- *Part 2: CJK Unified Ideographs Supplementary plane, General Scripts and Symbols Plane, General Purpose Plane*

Additional parts will specify other planes.

Information technology - Universal Multiple-Octet Coded Character Set (UCS) -

Part 2:

- CJK Unified Ideographs Supplementary plane,
- General Scripts and Symbols Plane,
- General Purpose Plane

1

2 Scope

ISO/IEC 10646 specifies the Universal Multiple-Octet Coded Character Set (UCS). It is applicable to the representation, transmission, interchange, processing, storage, input and presentation of the written form of the languages of the world as well as additional symbols.

ISO/IEC 10646 Part 1 specifies the overall architecture and the Basic Multilingual Plane (BMP) of the UCS.

This second part:

- specifies the CJK Unified Ideographs Supplementary Plane (UISP) of the UCS and defines a set of graphic characters that are used in East Asia. These are defined as Chinese/Japanese/Korean (CJK) unified ideographs (unified East Asian ideographs);
- specifies the General Scripts and Symbols Supplementary Plane (GSP) of the UCS and defines a set of graphic characters used in all other scripts not covered by the previous Plane and the BMP;
- specifies the General Purpose Plane (GPP) and defines a set of non-graphic characters.
- specifies the names for graphic characters of the two first planes, and the coded representation using the four-octet (32-bit) canonical form of the UCS.
- specifies the names for non-graphic characters of the GPP, and the code representation using the four-octet (32-bit) canonical form of the UCS.
- Graphic characters that are already encoded in the Part 1 shall not be duplicated in these supplementary planes. In addition, these planes do not have duplicated encoding of graphic characters within themselves

3 Conformance

The Conformance clauses of Part 1 also apply to this part.

4 Normative references

See Part 1.

5 Definitions

In addition to the definitions specified by Part 1, the following definitions apply:

5.1 General Scripts and Symbols Plane (GSP)
Plane 01 of Group 00.

5.2 CJK Unified Ideographs Supplementary Plane (UISP)
Plane 02 of Group 00.

5.3 General Purpose Plane (GPP)
Plane 14 of Group 00.

5.4 Tagging
The association of attribute of text with a point or range of a text sequence.

NOTE - The value of a particular tag is not generally considered to be part of the content of the text. Typical examples of tagging are to mark language or font of a portion of text.

5.5 Tag character

A coded character used for text tagging. A Tag character can only express a tag value and has no textual content by itself, and as such, has no graphic character equivalent.

6 General Supplementary Plane

The plane 01 of Group 00 shall be the General Supplementary Plane (GSP). Unlike the Basic Multilingual plane (BMP), the GSP cannot be used as a two-octet coded character set. It can only be used in the four-octet canonical form. <Note about UTF8 and UTF16>

As a special plane is reserved for CJK Unified Ideographs, the GSP shall not be used to encode them. The main purpose of the GSP is to specify a set of coded graphic characters used in all other significant scripts of the world (mostly extinct) which are not already encoded in the BMP.

NOTE - The following decomposition of the GSP has been proposed:

- Alphabetic,
- Hieroglyphic, Ideographic and Miscellaneous Syllabaries
- CJK Ideographic derived
- Newly Invented Scripts
- Symbol sets

9.2 General Supplementary Plane

<TBD>

9.3 VJK Unified Ideographs Supplementary Plane

<TBD>

9.4

7 CJK Unified Ideographs Supplementary Plane

The plane 02 of Group 00 shall be the CJK Unified Ideographs Supplementary Plane (UISP), and unlike the BMP, the UISP cannot be used as a two-octet coded character set.

The UISP is used for CJK unified ideographs (unified East Asian ideographs) that are not encoded in the BMP.

8 General Purpose Plane

The plane 14 of Group 0 shall be the General Purpose Plane (GPP). The GPP is used for non-graphic characters. For example it includes the Tag Characters.

9 Code Tables and lists of character names

Detailed code tables and lists of character names for the planes are shown on the following pages.

9.1

9.5 General Purpose Plane

TABLE 1 - Row 00: TAGS

dec	hex	Name	dec	hex	Name
000	00	(This position shall not be used)	064	40	TAG COMMERCIAL AT
001	01	LANGUAGE TAG	065	41	TAG LATIN CAPITAL LETTER A
002	02	(This position shall not be used)	066	42	TAG LATIN CAPITAL LETTER B
003	03	(This position shall not be used)	067	43	TAG LATIN CAPITAL LETTER C
004	04	(This position shall not be used)	068	44	TAG LATIN CAPITAL LETTER D
005	05	(This position shall not be used)	069	45	TAG LATIN CAPITAL LETTER E
006	06	(This position shall not be used)	070	46	TAG LATIN CAPITAL LETTER F
007	07	(This position shall not be used)	071	47	TAG LATIN CAPITAL LETTER G
008	08	(This position shall not be used)	072	48	TAG LATIN CAPITAL LETTER H
009	09	(This position shall not be used)	073	49	TAG LATIN CAPITAL LETTER I
010	0A	(This position shall not be used)	074	4A	TAG LATIN CAPITAL LETTER J
011	0B	(This position shall not be used)	075	4B	TAG LATIN CAPITAL LETTER K
012	0C	(This position shall not be used)	076	4C	TAG LATIN CAPITAL LETTER L
013	0D	(This position shall not be used)	077	4D	TAG LATIN CAPITAL LETTER M
014	0E	(This position shall not be used)	078	4E	TAG LATIN CAPITAL LETTER N
015	0F	(This position shall not be used)	079	4F	TAG LATIN CAPITAL LETTER O
016	10	(This position shall not be used)	080	50	TAG LATIN CAPITAL LETTER P
017	11	(This position shall not be used)	081	51	TAG LATIN CAPITAL LETTER Q
018	12	(This position shall not be used)	082	52	TAG LATIN CAPITAL LETTER R
019	13	(This position shall not be used)	083	53	TAG LATIN CAPITAL LETTER S
020	14	(This position shall not be used)	084	54	TAG LATIN CAPITAL LETTER T
021	15	(This position shall not be used)	085	55	TAG LATIN CAPITAL LETTER U
022	16	(This position shall not be used)	086	56	TAG LATIN CAPITAL LETTER V
023	17	(This position shall not be used)	087	57	TAG LATIN CAPITAL LETTER W
024	18	(This position shall not be used)	088	58	TAG LATIN CAPITAL LETTER X
025	19	(This position shall not be used)	089	59	TAG LATIN CAPITAL LETTER Y
026	1A	(This position shall not be used)	090	5A	TAG LATIN CAPITAL LETTER Z
027	1B	(This position shall not be used)	091	5B	TAG LEFT SQUARE BRACKET
028	1C	(This position shall not be used)	092	5C	TAG REVERSE SOLIDUS
029	1D	(This position shall not be used)	093	5D	TAG RIGHT SQUARE BRACKET
030	1E	(This position shall not be used)	094	5E	TAG CIRCUMFLEX ACCENT
031	1F	(This position shall not be used)	095	5F	TAG LOW LINE
032	20	TAG SPACE	096	60	TAG GRAVE ACCENT
033	21	TAG EXCLAMATION MARK	097	61	TAG LATIN SMALL LETTER A
034	22	TAG QUOTATION MARK	098	62	TAG LATIN SMALL LETTER B
035	23	TAG NUMBER SIGN	099	63	TAG LATIN SMALL LETTER C
036	24	TAG DOLLAR SIGN	100	64	TAG LATIN SMALL LETTER D
037	25	TAG PERCENT SIGN	101	65	TAG LATIN SMALL LETTER E
038	26	TAG AMPERSAND	102	66	TAG LATIN SMALL LETTER F
039	27	TAG APOSTROPHE	103	67	TAG LATIN SMALL LETTER G
040	28	TAG LEFT PARENTHESIS	104	68	TAG LATIN SMALL LETTER H
041	29	TAG RIGHT PARENTHESIS	105	69	TAG LATIN SMALL LETTER I
042	2A	TAG ASTERISK	106	6A	TAG LATIN SMALL LETTER J
043	2B	TAG PLUS SIGN	107	6B	TAG LATIN SMALL LETTER K
044	2C	TAG COMMA	108	6C	TAG LATIN SMALL LETTER L
045	2D	TAG HYPHEN-MINUS	109	6D	TAG LATIN SMALL LETTER M
046	2E	TAG FULL STOP	110	6E	TAG LATIN SMALL LETTER N
047	2F	TAG SOLIDUS	111	6F	TAG LATIN SMALL LETTER O
048	30	TAG DIGIT ZERO	112	70	TAG LATIN SMALL LETTER P
049	31	TAG DIGIT ONE	113	71	TAG LATIN SMALL LETTER Q
050	32	TAG DIGIT TWO	114	72	TAG LATIN SMALL LETTER R
051	33	TAG DIGIT THREE	115	73	TAG LATIN SMALL LETTER S
052	34	TAG DIGIT FOUR	116	74	TAG LATIN SMALL LETTER T
053	35	TAG DIGIT FIVE	117	75	TAG LATIN SMALL LETTER U
054	36	TAG DIGIT SIX	118	76	TAG LATIN SMALL LETTER V
055	37	TAG DIGIT SEVEN	119	77	TAG LATIN SMALL LETTER W
056	38	TAG DIGIT EIGHT	120	78	TAG LATIN SMALL LETTER X
057	39	TAG DIGIT NINE	121	79	TAG LATIN SMALL LETTER Y
058	3A	TAG COLON	122	7A	TAG LATIN SMALL LETTER Z
059	3B	TAG SEMICOLON	123	7B	TAG LEFT CURLY BRACKET
060	3C	TAG LESS-THAN SIGN	124	7C	TAG VERTICAL LINE
061	3D	TAG EQUALS SIGN	125	7D	TAG RIGHT CURLY BRACKET
062	3E	TAG GREATER-THAN SIGN	126	7E	TAG TILDE
063	3F	TAG QUESTION MARK	127	7F	CANCEL TAG

----End--

Unicode Technical Committee

N 1670
December 12, 1997

Title:	Plane 14 Characters for Language Tags
Source:	Unicode Technical Committee
Status:	Proposal
Action:	For the consideration of WG2
Distribution:	WG2 and National Body Members

Overview of the Proposal

The attached technical report from the Unicode Technical Committee is submitted for the consideration of WG2.

A mechanism for language tagging has been requested by the Internet Engineering Taskforce (IETF), in conjunction with the requirements for developing Internet protocols that interoperate with ISO/IEC 10646 as the basic character encoding.

The Unicode Technical Committee has worked with the IETF to draft a proposal which meets the IETF requirements and which will also work with existing implementations of ISO/IEC 10646. Details of that proposal and the background for the requirement are addressed in the attached technical report.

In parallel with this proposal, the document “Plane 14 Characters for Language Tags” has been formatted and posted as an Internet Draft, for discussion and use by the Internet standards community.

Unicode Technical Report #7
Plane 14 Characters for Language Tags

September 18, 1997

Authors: Ken Whistler, Sybase; Glenn Adams, Spyglass
References: See end of this document

ABSTRACT

This proposal addresses the need for a mechanism for generic tagging in Unicode plain text. A set of special-use tag characters on Plane 14 of ISO/IEC 10646 (accessible through UTF-8, UTF-16, and UCS-4 encoding forms) are proposed for encoding to enable the spelling out of ASCII-based string tags using characters which can be strictly separated from ordinary text content characters in 10646 (or Unicode).

One tag identification character and one cancel tag character are also proposed. In particular, a language tag identification character is proposed to identify a language tag string specifically; the language tag itself makes use of RFC 1766 language tag strings spelled out using the Plane 14 tag characters. Provision of a specific, low-overhead mechanism for embedding language tags in plain text is aimed at meeting the need of Internet protocols such as ACAP, which require a standard mechanism for marking language in UTF-8 strings.

This proposal is the result of an intense email discussion regarding language tagging and related issues, occasioned by the review of draft-ietf-acap-mlsf-01.txt and of draft-ietf-acap-langtag-00.txt, which proposed different mechanisms for language tagging in plain text. The Plane 14 proposal represents the consensus of a meeting of the UTC Working Group on Tagging and Annotation and of IETF representatives which took place on June 24, to be documented in an informational RFC.

DEFINITION OF TERMS

No attempt is made to define all terms used in this document. However, four terms which are used in special senses here require some clarification.

Tagging: The association of attributes of text with a point or range of the primary text. (The value of a particular tag is not generally considered to be a part of the "content" of the text. Typical examples of tagging is to mark language or font of a portion of text.)

Annotation: The association of secondary textual content with a point or range of the primary text. (The value of a particular annotation *is* considered to be a part of the "content" of the text. Typical

examples include glossing, citations, exemplication, Japanese yomi, etc.)

Out-of-band: An out-of-band channel conveys a tag in such a way that the textual content, as encoded, is completely untouched and unmodified. This is typically done by metadata or hyperstructure of some sort.

In-band: An in-band channel conveys a tag along with the textual content, using the same basic encoding mechanism as the text itself. This is done by various means, but an obvious example is SGML markup, where the tags are encoded in the same character set as the text and are interspersed with and carried along with the text data.

Introduction

There has been much discussion over the last 8 years of language tagging and of other kinds of tagging of Unicode plain text. It is fair to say that there is more-or-less universal agreement that language tagging of Unicode plain text is required for certain textual processes. For example, language "hinting" of multilingual text is necessary for multilingual spell-checking based on multiple dictionaries to work well. Language tagging provides a minimum level of required information for text-to-speech processes to work correctly. Language tagging is regularly done on web pages, to enable selection of alternate content, for example.

However, there has been a great deal of controversy regarding the appropriate placement of language tags. Some have held that the only appropriate placement of language tags (or other kinds of tags) is out-of-band, making use of attributed text structures or metadata. Others have argued that there are requirements for lower-complexity in-band mechanisms for language tags (or other tags) in plain text.

The controversy has been muddied by the existence and widespread use of a number of in-band text markup mechanisms (HTML, text/enriched, etc.) which enable language tagging, but which imply the use of general parsing mechanisms which are deemed too "heavyweight" for protocol developers and a number of other applications. The difficulty of using general in-band text markup for simple protocols derives from the fact that some characters are used both for textual content and for the text markup; this makes it more difficult to write simple, fast algorithms to find only the textual content and ignore the tags, or vice versa. (Think of this as the algorithmic equivalent of the difficulty the human reader has attempting to read just the content of raw HTML source text without a browser interpreting all the markup tags.)

The Plane 14 proposal addresses the recurrent and persistent call for a lighter-weight mechanism for text tagging than typical text markup mechanisms in Unicode. It proposes a special set of characters used *only* for tagging. These tag characters can be embedded into plain text and can be identified and/or ignored with trivial algorithms, since there is no overloading

of usage for these tag characters--they can only express tag values and never textual content itself.

The Plane 14 proposal is not intended for general annotation of text.

BASIC PROPOSAL

This proposal suggests the use of 95 dedicated tag characters, comprising a clone of 7-bit ASCII, plus a language tag character and a cancel tag, for a total of 97 characters, encoded at the start of Plane 14 of ISO/IEC 10646.

These tag characters are to be used to spell out any ASCII-based tagging scheme which needs to be embedded in Unicode plain text. In particular, they can be used to spell out language tags in order to meet the expressed requirements of the ACAP protocol and the likely requirements of other new protocols following the guidelines of the IAB character workshop (RFC 2130).

The suggested range in Plane 14 for the block reserved for tag characters is as follows, expressed in each of the three most generally used encoding schemes for ISO/IEC 10646:

UCS-4

U-000E0000 .. U-000E007F

UTF-16

U+DB40 U+DC00 .. U+DB40 U+DC7F

UTF-8

0xF3 0xA0 0x80 0x80 .. 0xF3 0xA0 0x81 0xBF

Of this range, U-000E0020 .. U-000E007E is the suggested range for the ASCII clone tag characters themselves.

NAMES FOR THE TAG CHARACTERS

The names for the ASCII clone tag characters should be exactly the ISO 10646 names for 7-bit ASCII, prefixed with the word "TAG".

In addition, there is one tag identification character and a CANCEL TAG character. The use and syntax of these characters is described in detail below.

The entire encoding for the proposed Plane 14 tag characters and names of those characters can be derived from the following list. (The encoded values here and throughout this proposal are listed in UCS-4 form, which is easiest to interpret. It is assumed that most Unicode applications will, however, be making use either of UTF-16 or UTF-8 encoding forms for actual implementation.)

U-000E0000 <reserved>

```
U-000E0001  LANGUAGE TAG
U-000E0002  <reserved>
...
U-000E001F  <reserved>
U-000E0020  TAG SPACE
U-000E0021  TAG EXCLAMATION MARK
...
U-000E0041  TAG LATIN CAPITAL LETTER A
...
U-000E007A  TAG LATIN SMALL LETTER Z
...
U-000E007E  TAG TILDE
U-000E007F  CANCEL TAG
```

RANGE CHECKING FOR TAG CHARACTERS

The range checks required for code testing for tag characters would be as follows. The same range check is expressed here in C for each of the three significant encoding forms for 10646.

Range check expressed in UCS-4:

```
     if ( ( *s >= 0xE0000 ) || ( *s <= 0xE007F ) )
```

Range check expressed in UTF-16 (Unicode):

```
    if ( ( *s == 0xDB40 ) && ( *(s+1) >= 0xDC00 ) && ( *(s+1) <= 0xDC7F
) )
```

Expressed in UTF-8:

```
    if ( ( *s == 0xF3 ) && ( *(s+1) == 0xA0 ) && ( *(s+2) & 0xE0 == 0x80
)
```

Because of the choice of the range for the tag characters, it would also be possible to express the range check for UCS-4 or UTF-16 in terms of bitmask operations, as well.

SYNTAX FOR EMBEDDING TAGS

The use of the Plane 14 tag characters is very simple. In order to embed any ASCII-derived tag in Unicode plain text, the tag is simply spelled out with the tag characters instead, prefixed with the relevant tag identification character. The resultant string is embedded directly in the text.

The tag identification character is used as a mechanism for identifying tags of different types. This enables multiple types of tags to coexist amicably embedded in plain text and solves the problem of delimitation if a tag is concatenated directly onto another tag. Although only one type of tag is currently specified, namely the language tag, the encoding of other tag identification characters in the future would allow for distinct tag types to be used.

No termination character is required for a tag. A tag terminates either when the first non Plane 14 Tag Character (i.e. any other normal Unicode value) is encountered, or when the next tag identification character is encountered.

All tag arguments must be encoded only with the tag characters U-000E0020 .. U-000E007E. No other characters are valid for expressing the tag argument.

A detailed BNF syntax for tags is listed below.

LANGUAGE TAGS

Language tags are of general interest and should have a high degree of interoperability for protocol usage. To this end, a specific LANGUAGE TAG tag identification character is provided. A Plane 14 tag string prefixed by U-000E0001 LANGUAGE TAG is specified to constitute a language tag. Furthermore, the tag values for the language tag are to be spelled out as specified in RFC 1766, making use only of registered tag values or of user-defined language tags starting with the characters "x-".

For example, to embed a language tag for Japanese, the Plane 14 characters would be used as follows. The Japanese tag from RFC 1766 is "ja" (composed of ISO 639 language id) or, alternatively, "ja-JP" (composed of ISO 639 language id plus ISO 3166 country id). Since RFC 1766 specifies that language tags are not case significant, it is recommended that for language tags, the entire tag be lowercased before conversion to Plane 14 tag characters. (This would not be required for Unicode conformance, but should be followed as general practice by protocols making use of RFC 1766 language tags, to simplify and speed up the processing for operations which need to identify or ignore language tags embedded in text.) Lowercasing, rather than uppercasing, is recommended because it follows the majority practice of expressing language tag values in lowercase letters.

Thus the entire language tag (in its longer form) would be converted to Plane 14 tag characters as follows:

U-000E0001 U-000E006A U-000E0061 U-000E002D U-000E006A U-000E0070

The language tag (in its shorter, "ja" form) could be expressed as follows:

U-000E0001 U-000E006A U-000E0061

The value of this string is then expressed in whichever encoding form (UCS-4, UTF-16, UTF-8) is required and embedded in text at the relevant point.

ADDITIONAL TAG TYPES

Additional tag identification characters might be defined in the future. An example would be a CHARACTER SET SOURCE TAG, or a GENERIC TAG for private definition of tags.

In each case, when a specific tag identification character is encoded, a corresponding reference standard for the values of the tags associated with the identifier should be designated, so that interoperating parties which make use of the tags will know how to interpret the values the tags may take.

TAG SCOPE AND NESTING

The value of an established tag continues from the point the tag is embedded in text until either:

A. The text itself goes out of scope, as defined by the application. (E.g. for line-oriented protocols, when reaching the end-of-line or end-of-string; for text streams, when reaching the end-of-stream; etc.)

or

B. The tag is explicitly cancelled by the CANCEL TAG character.

Tags of the same type cannot be nested in any way. The appearance of a new embedded language tag, for example, after text which was already language tagged, simply changes the tagged value for subsequent text to that specified in the new tag.

Tags of different type can have interdigitating scope, but not hierarchical scope. In effect,
tags of different type completely ignore each other, so that the use of language tags can be completely asynchronous with the use of character set source tags (or any other tag type) in the same text in the future.

CANCELLING TAG VALUES

U-000E007F CANCEL TAG is provided to allow the specific cancelling of a tag value. The use of CANCEL TAG has the following syntax. To cancel a tag value of a particular type, prefix the CANCEL TAG character with the tag identification character of the appropriate type. For example, the complete string to cancel a language tag is:

U-000E0001 U-000E007F

The value of the relevant tag type returns to the default state for that tag type, namely: no tag value specified, the same as untagged text.

The use of CANCEL TAG without a prefixed tag identification character cancels *any* Plane 14 tag values which may be defined. Since only language tags are currently provided with an explicit tag identification character, only language tags are currently affected.

The main function of CANCEL TAG is to make possible such operations as blind concatenation of strings in a tagged context without the propagation of inappropriate tag values across the string boundaries. For example, a string tagged with a Japanese language tag can have its tag value "sealed off" with a terminating CANCEL TAG before another string of unknown language value is concatenated to it. This would prevent the string of unknown language from being erroneously marked as being Japanese simply because of a concatenation to a Japanese string.

DISPLAY ISSUES

All characters in the tag character block are considered to have no visible rendering in normal text. A process which interprets tags may choose to modify the rendering of text based on the tag values (as for example, changing font to preferred style for rendering Chinese versus Japanese). The tag characters themselves have no display; they may be considered similar to a U+200B ZERO WIDTH SPACE in that regard. The tag characters also do not affect breaking, joining, or any other format or layout properties, except insofar as the process interpreting the tag chooses to impose such behavior based on the tag value.

For debugging or other operations which must render the tags themselves visible, it is advisable that the tag characters be rendered using the corresponding ASCII character glyphs (perhaps modified systematically to differentiate them from normal ASCII characters). But, as noted below, the tag character values are chosen so that even without display support, the tag characters will be interpretable in most debuggers.

UNICODE CONFORMANCE ISSUES

The basic rules for Unicode conformance for the tag characters are exactly the same as for any other Unicode characters. A conformant process is not required to interpret the tag characters. If it does interpret them, it should interpret them according to the standard, i.e. as spelled-out tags. If it does not interpret tag characters, it should leave their values undisturbed and do whatever it does with any other uninterpreted characters.

So for a non-TagAware Unicode application, any language tag characters (or any other kind of tag expressed with Plane 14 tag characters) encountered would be handled exactly as for uninterpreted Tibetan from the BMP, uninterpreted Linear B from Plane 1, or uninterpreted Egyptian hieroglyphics from private use space in Plane 15.

A TagAware but TagPhobic Unicode application can recognize the tag character range in Plane 14 and choose to deliberately strip them out completely to produce plain text with no tags.

The presence of a correctly formed tag cannot be taken as an absolute guarantee that the data so tagged is actually correctly tagged. For example, nothing prevents an application from erroneously labelling French data as Spanish, or from labelling JIS-derived data as Japanese, even if it contains Greek or Cyrillic characters.

NOTE ON ENCODING LANGUAGE TAGS

The fact that this proposal for encoding tag characters in Unicode includes a mechanism for specifying language tag values does not mean that Unicode is departing from one of its basic encoding principles:

> Unicode encodes scripts, not languages.

This is still true of the Unicode encoding (and ISO/IEC 10646), even in the presence of a mechanism for specifying language tags in plain text.

Language tagging in no way impacts current encoded characters or the encoding of future scripts.

It is fully anticipated that implementations of Unicode which already make use of out-of-band mechanisms for language tagging or "heavy-weight" in-band mechanisms such as HTML will continue to do exactly what they are doing and will ignore Plane 14 tag characters completely.

There is nothing obligatory about the use of Plane 14 tags, whether for language tags or any other kind of tags. This proposal for Plane 14 tags is, instead, aimed at removing a significant barrier to the universal adoption of Unicode in such arenas as Internet protocol development.

TAG SYNTAX DESCRIPTION

An extended BNF (Backus-Naur Form) description of the tags specified in this proposal is found below. Note the following BNF extensions used in this formalism:

1. Semantic constraints are specified by rules in the form of an assertion specified between double braces; the variable $$ denotes the string consisting of all terminal symbols matched by the this non-terminal.

   Example: {{ Assert ($$[0] == '?'); }}

   Meaning: The first character of the string matched by this non-terminal must be '?'

2. A number of predicate functions are employed in semantic constraint rules which are not otherwise defined; their name is sufficient for determining their predication.

   Example: HasValidSOA (qualified-domain-name)

   Meaning: qualified-domain-name has a valid SOA DNS record

   The function ReverseDomainName() takes a reversed domain name and reverses it to produce a standard domain name.

   Example: ReverseDomainName ("org.iso")

   Meaning: return reversed domain form of argument; in this case, returning "iso.org".

3. A lexical expander function, TAG, is employed to denote the tag form of an ASCII character; the argument to this function is either a character or a character set specified by a range or enumeration expression.

   Example: TAG('-')

   Meaning: TAG HYPHEN-MINUS

   Example: TAG([A-Z])

```
   Meaning:   TAG LATIN CAPITAL LETTER A ... TAG LATIN CAPITAL LETTER Z

4. A macro is employed to denote terminal symbols that are character
   literals which can't be directly represented in ASCII. The argument
   to the macro is the UNICODE (ISO/IEC 10646) character name.

   Example:   '${TAG CANCEL}'

   Meaning:   character literal whose code value is U-000E007F

5. Occurrence indicators used are '+' (one or more) and '*' (zero
   or more); optional occurrence is indicated by enclosure in '[' and
']'.

6. An array subscript of '*' indicates any element of the array.

   Example:   Assert ( $$[*] != '?' )

   Meaning:   the character '?' may not appear in the string matched
              by this non-terminal


FORMAT TAG SYNTAX


tag                             :   language-tag
                                |   cancel-tag
                                ;

language-tag                    :   language-tag-introducer language-tag-
argument
                                ;

cancel-tag                      :   cancel-tag-argument cancel-tag-marker
                                ;

language-tag-argument           :   tag-argument
                                    {{ Assert ( IsRFC1766LanguageIdentifier ( $$
); }}
                                ;

cancel-tag-argument             :   /* empty */
                                |   language-tag-introducer
                                ;

tag-argument                    :   tag-character+
                                ;

tag-character                   :   { c : c in TAG( { a : a in printable ASCII
                                        characters or SPACE } ) }
                                ;

language-tag-introducer         :   '${TAG LANGUAGE}'
                                ;

cancel-tag-marker               :   '${TAG CANCEL}'
                                ;


************************************************************
```

References

[L2/97-171R2]

K. Whistler and G. Adams, “Plane 14 Characters for Generic Tags (Revised)”,
UTC position paper. September 18, 1997.

[RFC1766]

Alvestrand, H., "Tags for the Identification of Languages", RFC 1766.

[RFC2070]

F. Yergeau, G. Nicol, G. Adams, and M. Duerst, "Internationalization of the Hypertext Markup Language", RFC 2070, January 1997.

[RFC 2130]

C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R. Atkinson, M. Crispin, and P. Svanberg, "The Report of the IAB Character Set Workshop held 29 February - 1 March, 1996", RFC 2130, April 1997.

draft-ietf-acap-spec-03.txt

C. Newman and J. Myers, "Application Configuration Access Protocol", March 1997.

draft-ietf-acap-mlsf-01.txt

Newman, C., "Multi-Lingual String Format", May 1997.

draft-ietf-acap-langtag-00.txt

Duerst, M., "Two Alternative Proposals for Language Tagging in ACAP", June 1997.