IETF
Internet-Draft
Intended status: Best Current Practice
Expires: August 18, 2014

A. Sullivan
Dyn
D. Thaler
Microsoft
J. Klensin

February 14, 2014

IETF Policy on Character Sets and Languages
draft-sullivan-rfc2277-bis-00

Abstract

This is a proposed new policy for the IETF on Character Sets and Languages.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Internet is international.

With the international Internet follows an absolute requirement to interchange data in a multiplicity of languages, which in turn utilize a bewildering number of characters.

The document is very much based upon RFC 2277 [RFC2277] which is the current policy being applied by the Internet Engineering Steering Group (IESG) towards the standardization efforts in the Internet Engineering Task Force (IETF) in order to help Internet protocols fulfill these requirements.

RFC 2277 in turn was based on the recommendations of the IAB Character Set Workshop of February 29-March 1, 1996, which is documented in RFC 2130 [RFC2130]. This document is a proposed replacement for RFC 2277 and attempts to be explicit and clear, and as concise as possible without leaving out necessary detail.[[CREF1: What other references do we want to add? --ajs@anvilwalrusden.com]]

1.1. Terminology

This document uses the terms "character", "charset", "coded character set", "language", "locale", and "protocol elements" as defined in RFC 6365 [RFC6365]. IDNA terminology is defined in RFC 5890 [RFC5890]. Any of those definitions may be used below, and the reader is expected to be familiar with them. [[CREF2: That last sentence makes this document much less accessible. I think at a minimum we need to list which terms used in this document are defined in each other RFC. I've now added a list above for 6365, but it may be missing some and the list of terms used from 5890 is needed. --dthaler@microsoft.com]][[CREF3: This is fair. I suggest we leave this as is and do an exhaustive pass for terminology later and updates these lists. --ajs@anvilwalrusden.com]]

This document uses the terms 'MUST', 'SHOULD' and 'MAY', and their negatives, in the way described in RFC 2119 [RFC2119]. In this case, 'the specification' as used by RFC 2119 refers to the processing of protocols being submitted to the IETF standards process.

2. Where to do internationalization

Internationalization is necessary because of the way natural language is written. It enables localization, which is for humans. This means that protocols are not subject to internationalization; text strings are. Where protocol elements look like text tokens, such as in many IETF application layer protocols, protocols MUST specify which parts are protocol and which are text (see Section 2.2.1.1 of [RFC2130]).

It is helpful to distinguish among four different types of strings for these purposes: domain names whether in the DNS or not, other protocol elements that are not normally visible to users, other protocol elements that are (even sometimes) normally visible to users, and data (in most cases, the protocol payload).

2.1. Domain names

Domain names (or strings of domain-name-like things) are used in a number of protocols, and not all of those names are intended to be looked up in the DNS. This raises a number of issues explored at length in [RFC6055].

Given this state of affairs, it is possible to recommend the following. These recommendations are consistent with RFC 6055:

- At resolution time, names that are to be looked up in the global DNS SHOULD be transmitted as A-labels.

- At resolution time, names that are not to be looked up in the global DNS ought to be transmitted in the form appropriate to the name resolution protocol. This is often UTF-8.

- Storage of internationalized domain names ought generally to be in the form of U-labels.

- Any protocol that needs to use domain names ought to use U-labels or A-labels consistently, and ought to prefer U-labels.

- Storage of U-labels (or putative U-labels) should be in the encoding form appropriate to the context. For instance, on a system that normally encodes UTF-8 using NFD, that is how the strings should be stored; similarly, a system that uses UTF-16 should store the strings in that form.

[[CREF4: This in the end will need to be checked carefully for its consistency with 6055. --ajs@anvilwalrusden.com]]

2.2. Non-DNS, "invisible" protocol elements

Many protocols include elements that are either words or word-like in some natural language (usually English), but that are never exposed to users under normal circumstances. Users might encounter these protocol elements in log messages and so on, and system administrators might regularly encounter them as part of the ordinary support burden. But these elements are no more candidates for internationalization than are hexadecimal protocol parameters. Because they are not intended for user consumption, they should not be treated as any part of a user interface. Internationalization considerations do not apply to them.

It is important to recognize that some of this class of protocol element sometimes appears to be exposed to users -- for instance, many user agents for mail display headers. In these cases, it is important to distinguish between the protocol element itself, and the user cues it may provide. The protocol element does not need to be internationalized. The user interface might. In general, it is best to internationalize (or localize) strings that are encountered by the user and to keep those that are passed between computer systems and interpreted by them as simple and unambiguous as possible. Even for names or strings that provide the underpinnings for the strings that users type or with which they interact, it is important to keep their forms as simple as possible. Examples of such strings include the results of a search or material that must be translated into several different languages.

2.3. Non-DNS, "visible" protocol elements

Sometimes, protocol elements are expected to be visible or, as likely, manipulable by users. [[CREF5: Sorry, the following bit needs some more references, which I've failed to get right in the interests of expediency. This is here to remind me. --ajs@anvilwalrusden.com]] For instance, many values of SMTP [RFC5321] commands are parts of mail addresses that users are expected to type. In the presence of EAI, those addresses may well be internationalized.

In general, there are two ways to handle these sorts of strings. One is to use an ASCII-compatible encoding in the way that IDNA does. Another is to internationalize the protocol. If an internationalized protocol is to be undertaken, agility among coded character sets appears to cause more problems than it solves. Therefore, for the purposes of transmission, it is best to transmit protocol elements as UTF-8 strings in "Net-Unicode" [RFC5198] form, with an appropriate profile. All ASCII-only strings meet this criterion. [[CREF6: Maybe the profile stuff needs to refer to PRECIS anyway. --ajs@anvilwalrusden]]

Merely requiring Net-Unicode is not enough. The PRECIS working group documents outline a number of considerations for how protocol elements and data need to be handled in the face of internationalization concerns. These kinds of considerations are especially important for protocol elements that may be influenced by user action. For instance, if comparisons are to be used, good PRECIS profiles for those elements are critical.

In the design of protocols for use on the Internet (or in other communications systems) that use textual keywords, there is a tradeoff between strings that have high mnemonic value (i.e., the identifiers are easily remembered by those who will use them) in local environments and those that are easily recognized and used internationally. Most cases are (and should be) resolved in favor of the latter, because these are strings used in protocols, a single set can easily be translated, and because it is possible to choose a single well-known script with good properties for those strings. But there are cases when other considerations are more important and each case and protocol should be carefully and separately considered. [[CREF7: I think I'd remove the last of those sentences unless we want to say when. --ajs@anvilwalrusden.com]]

2.4. Protocol data

Protocol data is very frequently user visible, and to the extent there are highly variable internationalization principles, they appear more commonly here.

In general, protocol data needs to carry an indicator of its coded character set. A protocol MUST identify, for all character data, which coded character set is in use. Protocols MUST be able to use UTF-8. New protocols SHOULD use UTF-8, and UTF-8 only, unless strong motivation is given for exceptions. The identification methods discussed in this section are for use with legacy protocols and situations.

NOTE: In the protocol stack for any given application, there is usually one or a few layers that need to address these problems.

It would, for instance, not be appropriate to define language tags for Ethernet frames. It is the responsibility of protocol designers to ensure that whenever responsibility for internationalization is left to "another layer", those responsible for that layer are in fact aware that they have that responsibility. The precis framework provides more guidance. [[CREF8: Surely this is too hand-wavy? Should we refer to particular bits? --ajs]]

3. General charset policy

The general policy of the IETF is that all data should be transmitted on the wire as UTF-8. Any protocol that does not conform to this policy but that is intended for the IETF standards track MUST justify it to the IETF.

When the protocol allows a choice of multiple charsets, someone must make a decision on which charset to use.

In some cases, like HTTP, there is direct or semi-direct communication between the producer and the consumer of data containing text. In such cases, it may make sense to negotiate a charset before sending data.

In other cases, like E-mail or stored data, there is no such communication, and the best one can do is to make sure the charset is clearly identified with the stored data, and choosing a charset that is as widely known as possible.

Note that a charset is an absolute; text that is encoded in a charset cannot be rendered comprehensibly without supporting that charset.

This also applies to English texts; charsets like EBCDIC do NOT have ASCII as a proper subset.

Negotiating a charset may be regarded as an interim mechanism that is to be supported until support for interchange of UTF-8 is prevalent. Despite the wide adoption of Unicode and UTF-8, the timeframe of "interim" may remain long, though perhaps not permanent.

4. Languages

4.1. The need for language information

All human-readable text has a language.

Many operations, including high quality formatting, text-to-speech synthesis, searching, hyphenation, spellchecking and so on benefit greatly from, or are all but impossible without, access to information about the language of a piece of text (Section 3.1.1.4 of [RFC2130]).

Humans have some tolerance for foreign languages, but are generally very unhappy with being presented text in a language they do not understand; this is why negotiation, or at least negotiation, of language is needed.

In most cases, machines will not be able to deduce the language of a transmitted text by themselves; the protocol must specify how to transfer the language information if it is to be available at all. It is sometimes possible to guess the langage of a block of text, but such guessing is usually unreliable and becomes dramatically less reliable the shorter the block of text.

4.2. Requirement for language tagging

Protocols that transfer text MUST provide for carrying information about the language of that text.

Protocols SHOULD also provide for carrying language information about visible protocol elements (especially if they are names), where appropriate.

Note that this does not mean that such information must always be present; the requirement is that if the sender of information wishes to send information about the language of a text, the protocol provides a well-defined way to carry this information. Nevertheless, if the data originator does not supply that information, it is generally impossible to make it up later.

4.3. How to identify a language

The language tag [RFC5646] is at the moment the most flexible tool available for identifying a language; protocols SHOULD use this, or provide clear and solid justification for doing otherwise in the document. Language tags are in general not useful without profiling appropriate to the case, and there is significant danger of over-specification with tags. See Section 4.1 of RFC 5646.

Note also that a language is distinct from a POSIX locale (see Section 5); a POSIX locale identifies a set of cultural conventions, which may imply a language (the "POSIX" and "C" locales of course do not), while a language tag identifies only a language.

4.4. Considerations for language negotiation

Protocols where users have text presented to them in response to user actions MUST provide for support of multiple languages.

How this is done will vary between protocols; for instance, in some cases, a negotiation where the client proposes a set of languages and the server replies with one is appropriate; in other cases, a server may choose to send multiple variants of a text and let the client pick which one to display.

Negotiation is useful in the case where one side of the protocol exchange is able to present text in multiple languages to the other side, and the other side has a preference for one of these; the most common example is the text part of error responses, or Web pages that are available in multiple languages.

Users do not, of course, actually use protocols, but instead user interfaces that in turn use the protocols. Therefore, what is necessary to support is not the full internationalization of everything in the protocol, but enough that the user-visible components can be localized appropriately. See Section 2.3.

Negotiating a language should be regarded as a permanent requirement of the protocol that will not go away at any time in the future.

In many cases, it should be possible to include it as part of the connection establishment, together with authentication and other preferences negotiation.

4.5. Default language

For the purposes of display, it may be necessary to pick a default language to use when it is not possible to determine the language. It is evident that picking a default may lead to user dissatisfaction or confusion, but when language cannot be determined such fallbacks may be necessary.

Section 4.1 of [RFC5646], numbers 5 and 7, outline the considerations for language identification when the language cannot be determined.

5. Locale

The POSIX standard [ISO.9945-2.1993] defines a concept called a "locale", which includes a lot of information about collating order for sorting, date format, currency format and so on.

In some cases, and especially with text where the user is expected to do processing on the text, locale information may be usefully attached to the text; this would identify the sender's opinion about appropriate rules to follow when processing the document, which the recipient may choose to agree with or ignore.

This document does not require the communication of locale information on all text, but encourages its inclusion when appropriate.

Note that language and character set information will often be present as parts of a locale tag (such as no_NO.iso-8859-1; the language is before the underscore and the character set is after the dot); care must be taken to define precisely which specification of character set and language applies to any one text item.

The default locale is the "POSIX" locale.

6. Documenting Internationalization Decisions

In documents that deal with internationalization issues at all, a synopsis of the approaches chosen for internationalization SHOULD be collected into a section called "Internationalization considerations". This practice has historically not been followed regularly, but it remains a good idea. The goal is to provide an easy reference for those who are looking for advice on these issues when implementing the protocol.

7. Security Considerations

Security warnings in a foreign language may cause inappropriate behaviour (such as ignoring the warning entirely) from the user. In addition, the issues raised in [RFC6943], especially in its section 4.2 and section 5, are of particular relevance to internationalization.

8. Acknowledgements

Much of the text comes from [RFC2277]. Harald Alvestrand was the primary author of that RFC.

Most of the discussion above was initiated as part of the IAB's internationalization program. At the time of writing, the program members were (in alphabetical order) Marc Blanchet, Stuart Cheshire, Leslie Daigle, Patrik Faltstrom, Heather Flanagan, John Klensin, Olaf Kolkman, Barry Leiba, Xing Li, Pete Resnick, Peter Saint-Andre, Andrew Sullivan, and Dave Thaler.

Significant text in Section 2.2 and Section 2.3 was derived from a forthcoming Internet Society education module for next-generation Internet leaders and future influencers and used with permission. The contributions and support for that work of Toral Cowleson and Niel Harper of the Internet Society are gratefully acknowledged.

9. IANA Considerations

This document makes no requests of IANA.

10. Informative References

[ISO.10646-1.1993]
International Organization for Standardization, "Information Technology - Universal Multiple-octet coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane", ISO Standard 10646-1, May 1993.

[ISO.9945-2.1993]
International Organization for Standardization, "ISO/IEC 9945-2:1993 Information Technology -- Portable Operating System Interface (POSIX) -- Part 2: Shell and Utilities", ISO Standard 9945-2, 1993.

[RFC1033] Lottor, M., "Domain administrators operations guide", RFC 1033, November 1987.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2130] Weider, C., Preston, C., Simonsen, K., Alvestrand, H., Atkinson, R., Crispin, M., and P. Svanberg, "The Report of the IAB Character Set Workshop held 29 February - 1 March, 1996", RFC 2130, April 1997.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, March 2008.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.

[RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, August 2010.

[RFC5892] Faltstrom, P., "The Unicode Code Points and Internationalized Domain Names for Applications (IDNA)", RFC 5892, August 2010.

[RFC5893] Alvestrand, H. and C. Karp, "Right-to-Left Scripts for Internationalized Domain Names for Applications (IDNA)", RFC 5893, August 2010.

[RFC5894] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Background, Explanation, and Rationale", RFC 5894, August 2010.

[RFC5895] Resnick, P. and P. Hoffman, "Mapping Characters for Internationalized Domain Names in Applications (IDNA) 2008", RFC 5895, September 2010.

[RFC6055] Thaler, D., Klensin, J., and S. Cheshire, "IAB Thoughts on Encodings for Internationalized Domain Names", RFC 6055, February 2011.

[RFC6365] Hoffman, P. and J. Klensin, "Terminology Used in Internationalization in the IETF", BCP 166, RFC 6365, September 2011.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, February 2013.

[RFC6943] Thaler, D., "Issues in Identifier Comparison for Security Purposes", RFC 6943, May 2013.

Appendix A. Version History

A.1. 00

Initial version. Contains a number of xml2rfc warnings.

Authors' Addresses

Andrew Sullivan
Dyn
150 Dow St.
Manchester, NH 03101
U.S.A.

Email: asullivan@dyn.com

Dave Thaler
Microsoft Corporation
One Microsoft Way
Redmonad, WA 98052
USA

Phone: +1 425 703 8835
Email: dthaler@microsoft.com

John C Klensin
1770 Massachusetts Ave, Ste 322
Cambridge, MA 02140
USA

Phone: +1 617 245 1457
Email: john-ietf@jck.com