The IDN Variant Issues Project: A Study of Issues Related to the Delegation of IDN Variant TLDs

Discussion Draft - 5 December 2011

Executive Summary

[TBD]

1. Project Overview

The internationalization of the DNS space is a critical area of work for the ICANN community. Historically, the DNS root zone has been limited to a subset of the characters in the US-ASCII (American Standard Code for Information Interchange) character set. This is changing with the introduction of Internationalized Domain Names (IDNs), including the introduction of new top-level domains (TLDs) in multiple scripts, enabling Internet users to access domain names using writing systems familiar to them.

The opening of the IDN country code Top-Level Domain (ccTLD) Fast Track Process[1] by the ICANN Board in October 2009 enabled countries and territories to submit requests to ICANN for IDN ccTLDs representing their respective country or territory names in scripts other than US-ASCII characters.

The new generic Top-Level Domain (gTLD) Program[2], approved in June 2011 and opening for applications in January 2012, will allow for the first time the addition of IDN gTLDs into the root zone.

1.1 The Variant Issues Project

IDNs can serve as powerful tools for broadening the Internet's capacity and accessibility; however, for a good user experience they also raise unique issues. One important issue concerns the use of "variants," which, according to one technical definition, occur when a single conceptual character can be identified with two or more different Unicode Code Points.[3] TLDs containing one or more such characters might be considered “variant TLDs,” and unless carefully implemented, might result in user confusion or a poor user experience. While the concept of “variants” is raised in a number of contexts, and in some cases is regarded as critical for the successful adoption of IDN TLDs to meet user needs, there is no single definition or rule for determining whether TLDs can be considered variants of one another.

[1] http://www.icann.org/en/topics/idn/fast-track/

[2] http://newgtlds.icann.org/

[3] http://www.ietf.org/rfc/rfc3743.txt

In an effort to develop potential solutions for the delegation of IDN variant Top-Level Domains (TLDs), the ICANN Board in 2010 passed a resolution directing the development of a preliminary report on the viability, sustainability and delegation of IDN variants.[4]

The IDN Variant Issues Project plan was published in April 2011[5], along with a call for volunteers, and work commenced shortly thereafter. Six script case study teams (Arabic, Chinese, Cyrillic, Devanagari, Greek, and Latin) worked to identify the set of issues that, if resolved, could enable the delegation of IDN variant TLDs for the benefit of the respective user communities. The case study teams comprised a total of 66 experts from 29 countries and territories, and offered expertise in the areas of: DNS, IDNA, linguistics, security & scalability, policy, registry/registrar operations, and community representation. The case study team reports were produced on schedule and published for public comment in October 2011.[6]

ICANN, assisted by a coordination team comprised of representatives from the case study teams, has worked to build on these team reports to develop this integrated issues report, to cover both common issues germane across the cases studied and issues particular to specific cases. An integrated analysis of the issues associated with IDN variant TLDs will be an important milestone toward considering follow-on work in this area.

1.2 The Script Case Studies

The six case study reports identify relevant variant issues in six different scripts: Arabic, Chinese, Cyrillic, Devanagari, Greek and Latin.

Broadly the reports share a common structure, beginning with an introduction of the report's authors, and presentation of the history, distribution, and structure of the relevant script. The reports propose specific terminology to give details on the kinds of issues affecting their script which may suggest treatment by establishment of a variant management mechanism. Most of the reports include an exhaustive listing of the Unicode code points (with character names and glyphs) which make up their script (though not Chinese with over 5000 characters to organize) and which they recommend for use in TLD labels. There is then a review of the kinds of relations

[4] The Board resolution provided that "The CEO is directed to develop (in consultation with the board ES-WG) an issues report identifying what needs to be done with the evaluation, possible delegation, allocation and operation of gTLDs containing variant characters IDNs as part of the new gTLD process in order to facilitate the development of workable approaches to the deployment of gTLDs containing variant characters IDNs. The analysis of needed work should identify the appropriate venues (e.g., ICANN, IETF, language community, etc.) for pursuing the necessary work. The report should be published for public review." See http://www.icann.org/en/minutes/resolutions-25sep10-en.htm#2.5

[5] http://www.icann.org/en/announcements/announcement-20apr11-en.htm

[6] http://www.icann.org/en/announcements/announcement-4-03oct11-en.htm

between characters (and, in a very few cases, words) which might be modelled with the mechanism of variants. The language-related and contrastive complexities of individual characters are later supplemented with issues concerning the ease of character recognition, and other user issues deriving from constraints on the software and hardware environment in which the script is currently used. There are, in some cases, examinations of technical issues not centrally focused on variants (e.g., the degree of acceptability of invisible code points, apostrophes and other characters which are not members of the code block associated with the script).

The focus then moves to the procedures for applying to register TLDs in the script, and the administrative apparatus which remains installed to support them. This last includes operational security concerns and procedures for dispute resolution.

Most of the reports (4 out of 6) also have a summary of conclusions, to emphasize certain of their theses.

1.2.1 Statements of General Principles

The focus of the work is the identification of issues concerning the potential use of variant characters within IDN scripts, to define variant top-level domains (TLDs), whether generic or country-code. The IDNA Protocol determines whether a code point is Protocol-Valid (PVALID) by derivation from certain Unicode properties. It was noted with caution that the teams of experts might not have expertise on every language using the scripts considered. (Arabic §3, Devanagari §1, Cyrillic §1).

1.2.2 Distinctive Terminologies

To define the field of character variants, the teams initiated their work around a basic set of definitions for terms as provided by ICANN.[7] These were supplemented by terms from the Unicode website, RFC 6365, and RFCs 5890, 5892 and 5893. (Arabic §4, Latin §2)

The reports also defined additional terms to address properties of their own scripts. All except the Devanagari and Latin teams found it necessary to define such terms. (Arabic Appendix E, Chinese §3, Cyrillic §2, Greek §2). The coordination team has worked to arrive at a set of generally agreed terminology for the IDN Variant Project.

1.2.3 Code Blocks in Extenso; Label Generation Policy

The IDNA Protocol has a very broad-based filter to determine what code points are permitted under the protocol; the rules are defined in RFC 3892.[8] Many teams considered or proposed

[7] https://community.icann.org/download/attachments/16842778/Draft+Definitions.pdf?version=1&modificationDate=1310669168000

[8] http://tools.ietf.org/html/rfc5892

further restrictions (e.g. to exclude free-standing diacritics), and listed the resulting subsets of code points which would be available for use in TLDs. (Arabic §5; Chinese §2.1; Greek Appendix A; Latin Appendix B2).

The Arabic report (§7) notes that in addition to these lists of label-valid code points, a policy on defining character variants, and a set of other rules and meta-information must also be added, in order fully to identify the set of possible labels. Taken together, these constitute a Label Generation Policy. The Chinese report (§7) covers similar ground on the full integration of character repertoire and variant linkages, but basing its discussion on the concept of a Language Variant Table, in keeping with RFC 3743.[9]

1.2.4 Defining the Scope of Variants within a Script

This was the central concern of each case study team. Inevitably, at the outset of the project, it could not be precisely foreseen what may turn out to be the ultimately acceptable boundary conditions.

So the Arabic report (§6) and the Devanagari report (§3.2) distinguish cases of identical and similar glyphs (which might ultimately be seen as cases not of Variance but Visual Similarity). The Arabic report also notes the existence of interchangeable characters (where the basis for equivalence is linguistic functions) and optional cases (where the writing system allows some degree of choice in the exactness of a written form). The Devanagari report does not consider these latter as potential variants (§4.2). The Cyrillic report (§3) looks at a number of concrete issues in various Cyrillic-using languages, where additions and refinements to Cyrillic have created inconsistent usage. Specific examples of these types are listed in Appendices Arabic A and Cyrillic A.

The Greek report (§6-9) also focuses on inconsistent orthographical practices (e.g. in use of upper/lower case and the *tonos* accent), but also (§5) notes the need for a policy on the orthography of the letter *sigma*, which (by historic convention) requires positional alternation between different code points. It also (§13.1) favors recognition of some dialectal word equivalences as variants, specifically between words in the archaizing *katharevousa* standard and the modern *dimotiki* standard language. The Latin report (§6) – although it ultimately requests no variants – considers upper/lower case, display forms, glyph identity, decorative forms, issues with diacritics, and punctuation marks.

The Chinese report (§5) has a different set of preoccupations since its script is ideographic rather than alphabetic, especially so since the report largely excludes Japanese and Korean with their additional phonetic systems hiragana, katakana and hangul. It excludes half-width

---

[9] http://www.ietf.org/rfc/rfc3743.txt

characters (as not used for Chinese) and also homographs: here a single character has the same code, glyph and pronunciation, but appears to have multiple meanings. It recognizes variant characters, however, in two issues: the nexus between Simplified and Traditional characters (as a species of "regional variation"); and the Generic variants, where (owing to Unicode's historic policies in defining Unified Han script) subtly different forms of glyphs (derived from differing authoritative sources) have been assigned different code points, although they are functionally non-distinct in Chinese.

Further comparison of the instances and generalizations made by the various groups can be found in section 4 of this report.

1.2.5 The Role of Visual Similarity as a threat to Glyph Recognition

Many reports refer to the role of visual similarity, which concerns the relationship between the form of a character as presented for recognition by users, and the specific character identities assigned by the Unicode code points. (Arabic §6; Chinese §5; Cyrillic §4, 9.1; Devanagari §3.2; Greek §7; Latin §6.2).

The issue includes potential confusions between glyphs in different scripts: especially among Latin, Greek and Cyrillic, but also between Devanagari and other closely related Indian scripts. In fact, it has to do with problems of recognition by users rather than (directly) with clashes of rights inherent in registration. Potential for cross-script confusion is addressed in Cyrillic §4, 9.1; Devanagari §3.4, 4.1; Greek §7; Latin §6.8, 7.

1.2.6 Non-Variant Script Issues

A number of technical issues have arisen in considering the need for variants which are not, strictly speaking, directed at variants. These include:

- available fonts and their impact on glyph shape and recognition (Arabic Appendix C);
- the admissibility of zero-width characters ZWJ and ZWNJ, for the control of glyph rendering (Arabic §5.21 and Appendix D, Devanagari §4.3);
- the admissibility of characters outside the code-block assigned to a given script, notably the (extremely similar) apostrophe, saltillo or turned comma (Cyrillic §3.6, 9.2, Appendix A; Devanagari 3.4, Latin 6.7).

1.2.7 Other User Experience Issues

Some of the reports dwell on specifics of the user situations for Internet use and connections in their specific script areas.

Arabic §13 considers the inadequacies of computer systems (especially their keyboards and operating systems) to input and process the full range of Arabic characters in different regions (cf. Devanagari §5.3); the confusion of font differences resulting from the historic variety of writing styles (and cf. Cyrillic §3.2, 3.3, 9.1); the intrinsic problems of reversing text direction in

using a right-to-left script with many left-to-right elements; and the current lack of penetration of IDNs into a variety of computer applications used by Arabic-script users.

Chinese §6.1, 6.2 emphasizes the steep learning curve for users of information technology in China, resulting from historically low levels of technical education, and a dramatic increase in take-up over recent years. The result is inferred to be a wide-scale requirement of Chinese users to have computer systems for use in which they can make the same assumptions as in the rest of their literate practice (and cf. the characterization of the Indian situation in Devanagari §5.5, and of the Greeks in Greek §7).

Latin §4 highlights the implications of this for policy on upper/lower case sensitivity and web-browser behavior. Latin §5 points out that the constraints on usability of available characters which apply to an IDN environment have no precedent in people's prior experience with ASCII-coded information technology.

1.2.8 Evaluation of Applications, Registration and Operations

Towards the end of all the reports, consideration is given to administrative concerns: how are applications for registration to be evaluated and charged, how are registry records to be kept, how is security to be managed and any disputes to be resolved?

Arabic §8-11, Chinese §8-9, Devanagari §4.4-5, §5 and Latin §9 consider this conglomerate of issues for their respective script areas, but in virtue of their administrative nature, the points at issue are much the same.

The reports all emphasize the multiplication of entities which comes about as variant-generating rules are authorized. As well as there being more potential labels in existence, the root is working with scripts that are each typically used by many languages, whilst many of the languages in turn are used in multiple national administrations. Hence there will be requirements for mutual updating on a massive scale, and contact even before labels are reserved (Latin §8).

In this new, much larger universe of discourse, it will be more challenging to keep track of what is available for users to apply for and registries to reserve, allocate, delegate or block, with legal as well as technical questions to be answered. Maintenance of labels once they are activated will be more demanding. An appropriate fee structure, as well as security procedures, will need to be defined.

As to procedures for dispute resolution, this seems to be a point of decided interest, since it is addressed at length in Arabic §12, Chinese §8.2 and Latin §10.

1.2.9 Conclusions

Each of the reports took the opportunity to stress particular theses identified for the relevant scripts.

Chinese §10 dwells on the need for Chinese IDLs and their variants to be delegated to the same entity both in Simplified and Traditional character versions.

Cyrillic §10 stresses the need for a conservative (perhaps even precautionary) approach to the admission of variants. Furthermore, it urges that selective blocking (rather than joint delegation, or some form of aliasing) is the best mode to take account of any variants which are admitted.

Greek §14 gives some explanation of the thinking behind the report's recommendations, but finishes with two "red lines": the requirement that *tonos* accentuation and distinctive final *sigma* be recognized in IDL, and the requirement that a string and all its variants be reserved for the same registrant.

Latin §11 notes that the reception of labels in Latin script is language-dependent, but the script itself is language-independent; it is therefore impractical to signalize any particular relation between two code points in the Latin repertoire as variants of one another. There must furthermore be specific linguistic justification for any Latin character whose inclusion is sought in a label for registration.

2. Integrated Issues Report Overview

The report considers the issues associated with delegation of IDN variant TLDs. The report is organized according to two main sets of issues: those that concern how variant TLDs are established (discussed in section 5), and those that concern how variant TLDs are treated once established (explored in section 6. Both sets of issues are analyzed with certain overriding considerations: the security and stability of the DNS, and user experience. Maintaining the security and stability of the DNS is central to ICANN's core mission. It is of the utmost importance that the actual operation and maintenance of the DNS, on which many services rely, are not adversely impacted by the introduction of IDN variant TLDs. Where relevant, these issues are discussed. Secondly, user experience considerations are borne in mind throughout the discussion, although there is a special section devoted to issues that primarily impact end users. This would include not delegating variant TLDs in a manner that creates user vulnerabilities or a probability of confusion, as well as an interest in functionality and efficiency for the user experience.

2.1 Objectives

As detailed in the project plan, this issues report is intended to describe each of the general and case-specific issues to be resolved for the cases studied. It will also provide a detailed roadmap that can be used for studying additional cases, so that the experience gained from these initial case studies can support the efficient development of similar issues reports for additional cases.

In accordance with the scope defined for the project, the following objectives have been established for this issues report:

- Identify the set of issues relevant across scripts
- Identify any sets of issues relevant to specific scripts

- Provide a brief analysis of the issues, including the benefits and risks of possible approaches identified
- Document the levels of support for issue definition and analysis, based on collaborative inputs from case study teams and subject matter experts
- Identify areas where further study or work are needed
- Create a "roadmap for additional cases," i.e., steps that could be taken to perform case studies for additional scripts

2.2 Scope

This report is designed to provide a review of issues concerning the delegation of IDN variant TLDs. Other related issues are discussed in the report to the extent they are relevant.

It is important to note that there are a number of scripts not represented by the six case studies. In addition, as noted by several of the teams in their reports, some of the scripts studied represent a number of languages, and not all languages were represented on the study teams. Accordingly, analysis of issues has not been undertaken for all possible cases and is not being represented as comprehensive. However, it is expected that this gathering of DNS, language expertise represents a significant percentage of the world's Internet users, and will be an important milestone in the work in this area.

Devising rules or proposing variant management solutions are not within the scope of this report. In some instances, a range of possible solutions are considered and analyzed with a view toward informing potential later phases of the project that are focused on solutions.

2.3 Guiding Principles

The mission of ICANN is to coordinate, at the overall level, the global Internet's systems of unique identifiers, and in particular to ensure the stable and secure operation of the Internet's unique identifier systems. When considering the possible delegation of IDN variant TLDs, ICANN has the responsibility to undertake these activities in a manner that will not adversely affect the security or stability of the DNS.

At the current time, a variant management mechanism for the top level does not exist. In considering the issues associated with developing such a mechanism, certain existing references are used as an underlying foundation. The Unicode standard[10] (currently version 6.0) provides a

[10] The Unicode Standard is a character coding system designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages and technical disciplines of the modern world. See http://unicode.org/standard/standard.html. Note that The Unicode Technical Standard is being prepared for an update to align with Unicode 6.1. Public review and comment are invited on the drafts Issue #208: Proposed Update UTR #36: Unicode Security Considerations (see http://www.unicode.org/review/pri208/) and Issue #209 : Proposed Update Unicode Technical Standard #39 Unicode Security Mechanisms (see http://www.unicode.org/review/pri209/.

repertoire of code points used in world scripts, including various classifications of character properties, and normalization rules. The Internationalizing Domain Names in Applications (IDNA) protocol (RFCs 5890-5)[11] specifies rules for determining whether a code point, considered in isolation or in context, is a candidate for inclusion in a domain name.

It is assumed that these reference points will continue to be applicable to the DNS. In addition to the assumption of stable references, ICANN has also distilled from the case study team reports and the mission and core values of ICANN a set of guiding principles, described below, which has been adopted in the development of this report.

1. The root zone is a shared resource, and the management of the root zone should accommodate, to the maximum extent possible, the needs of users of multiple global scripts. A principle of equivalent treatment should be adhered to, avoiding undue consideration for users of particular scripts in a space that is used by all. In addition, both gTLDs and ccTLDs exist in the root zone and are relied upon by Internet users around the world. While gTLDs and ccTLDs may involve different operating environments, it is critical that a reliable user experience is produced across the TLD space. As a result, any restrictions on (Unicode) code point repertoires for TLDs will need to be adhered to by both ccTLDs and gTLDs.

2. A conservative approach should be adopted in the definition of variants; if necessary, a more liberal approach may be adopted later. Given that experience in this area is limited, and actions taken will create a precedent that cannot be undone, variant TLD labels should be narrowly defined, and precise rules for active use of variant labels in the DNS should be adopted. Wherever possible, instead of adding a new type of variant TLD label, an alternative approach should be used – for example, using an existing ICANN evaluation or objection process that delivers an appropriate way of blocking undesired TLD strings. If there is a process already in existence that delivers a similar result to what is desired, that process should be used rather than establishing a new type of variant label.

3. The root zone is a special case, and the approach taken for variant management in the root need not prescribe that taken by individual TLD registries. ICANN must adopt and a consistent policy for consideration of requests for IDN variant TLDs, and this may entail specific criteria and rules applicable for variant strings to be allowed at the top level. However, formulation of TLD policy often takes into account the specific user context, and there may be corresponding reasons for different criteria or rules relating to variant label generation and use, subject to certain minimum requirements necessary for security or

---

[11] See http://tools.ietf.org/html/rfc5890; http://tools.ietf.org/html/rfc5891; http://tools.ietf.org/html/rfc5892; http://tools.ietf.org/html/rfc5893; http://tools.ietf.org/html/rfc5894; http://tools.ietf.org/html/rfc5895.

stability reasons. Accordingly, TLD registry operators should not have an automatic obligation to abide by all of the same variant tables and policies used at the top level.

4. User behavior can adapt as necessary to the circumstances, and thus all natural language uses need not be reflected in TLDs. There are two important ways that this is true. First, names in the DNS, and at the top level especially, have always been restricted. RFC 1123[12] notes that the top-level names "will be alphabetic," (which in the context meant “only the ASCII characters a-z and A-Z”) and that convention has held up through the history of the DNS. Additionally, characters that are typically used in written language (such as <?> or <+>) have always been excluded from the traditional letters, digits, hyphen (“LDH”) range normally used in DNS labels. Just as these restrictions prevent the expression of many useful ASCII labels at the top level, the internationalization of the rules for the top level should limit its scope to the minimum necessary, and not to the maximum possible. Secondly, it is not reasonable to suppose that a naive user will approach the DNS with only the rules of his or her natural language in mind. A general principle of machine usage is that there are special words and conventions for interacting with the machine. While reasonable approximations of natural language usage are desired in any approach, users are not dependent on the ability to use the full natural language without restrictions and should be able to accommodate certain limitations where necessary.

3. Discussion of Issues: Possible Types of Variant Labels

A review of the case study team reports yielded a universe of possible variant types detailed as discussed in this section. This chart attempts to place all “variant issues” found in the six different scripts within a framework of abstract types. As such, it may be useful as a first grouping of the issues, suggesting which may yield to common approaches, even though the specific variant issues which crop up in the different scripts may appear very different.

Each of the headings is here discussed with examples, as a further attempt at clarification.

The key role played by the concept of “Abstract Characters” in this taxonomy may entail some difficulties of interpretation. The assignment of code points and representative glyphs is definite within Unicode, but the recognition of abstract characters is implicit (and stated, if at all, only in the word description of the character). In fact, its identity depends on linguistic analyses of all the languages in question. Essentially, a linguistic test of whether there is one or more abstract character depends on whether there is a discernible contrast within a language.

The presumption is that, within a language, every abstract character corresponds to a single phoneme or grapheme (or in UniHan a single sememe, i.e. an interpretable meaning, usually with determinate phonic pronunciations). However, although a single character may be consistent within a language, as

[12] http://www.ietf.org/rfc/rfc1123.txt

between languages it may represent very different phonemes, graphemes and sememes. [Consider, e.g., “a” in English and Spanish spelling, pronounced very differently; standard and swash kaf in Sindhi and Arabic, which marks a linguistic contrast in one language but is stylistic in the other; and as a representative Chinese character, 人 (U+4EBA) has a basic meaning ‘human being’, but in Chinese is pronounced rén and in Japanese jin, nin, hito etc.; each language then uses it with different (but overlapping) combinatory properties.]

Nevertheless, “abstract character” is a concept that is needed and is reasonably clear within a given language, giving a sense of an underlying unity in cases where code points or glyphs seem to have been multiplied beyond necessity, in the process of defining Unicode.

The references used in the table below are to the corresponding section in that team’s case study report.

	Arabic	Chinese	Cyrillic	Devanagari	Greek	Latin
I. Character substitutability						
A. Intra-script						
1. Same abstract character						
1a. Abstract characters that have more than one encoding						
	Identical: decomposables, esp. if not normalized (6.1) 6.1 Appendix A.2 (A.2.1 – A.2.2)	Non-identical: Generic variant characters (5)	Identical: decomposables, esp. if not normalized (3.8)			(Discussed at 6.5: Precomposed characters) (Discussed at 6.4: decorative and contrastive variants)
1b. Apparently the same abstract character in some contexts						
	KAF Group, HEH Group, and YEH Group (6.1.abc), MARBUTA Group (6.1g) HEH with HAMZA Group (6.1h) NOON Group (6.1j); KAF Group, YEH Group, YEH with HAMZA Group; variants of dot orientation (all in 6.2) 6.1 a thru k – Appendix A.1			Correct spelling of inflected forms in Nepali requires use of ZWNJ (4.3.2)	SIGMA and FINAL SIGMA (5, 12)	SMALLER TURNED E (U+01DD), SMALL LETTER SCHWA (U+0259) (6.2)

	Arabic	Chinese	Cyrillic	Devanagari	Greek	Latin
2. Different abstract characters						
2a. Different abstract characters interchangeable for users						
	6.3 a thru d vowels with/without HAMZA (U+0621), All forms of ALEF (e.g., U+0627) with/without composed form (6.3ab); TEH MARBUTA (U+0629) and HEH (U+0647) (6.3c); Arabic-Indic and extended Arabic-Indic digits (6.3d)	Simplified characters (SC) / Traditional characters (TC) (5) Japanese-only Kanji (6.3)				(Discussed at 6.6: combining marks)
2b. Simple visual confusability						
	Non-identical: GHAIN/FEH Group, QAF/AIN with 2 DOTS ABOVE Group, AIN/FEH/QAF with 3 DOTS ABOVE		GHE/GHE (U+0433) WITH UPTURN (U+0491) in Ukrainian (3.2); Old Letters, e.g., YAT/SEMISOFT SIGN (U+048C / U+048D) (3.4); ZE (U+0437) / DIGIT	GHA (U+0918) / DHA (U+0927) / BHA (U+092D) / MA (U+092E) etc (3.2.1 + Appx III); composite characters DGA/DNA/DRA etc. (3.2.2 + Appx IV); Potential mistaken		(Discussed at 6.7: Punctuation: Latin characters confusable with APOSTROPHE)

	Arabic	Chinese	Cyrillic	Devanagari	Greek	Latin
	Group (6.2) 6.2 Appendix B		THREE (U+0033) (3.7); various forms of GHE, KA and EN (e.g., U+0433, 0491, 0493, U+043A,049B, 049D, 04A3, 04A5) (3.9); Appx I passim	characters due to rendering errors of poor font design (3.3.1)		
2c. Compatibility mappings						
			Rarely used character/sequence of characters, e.g., EN WITH HOOK (U+04C7 / U+04C8) /EN + GHE (U+043D U+0433) (3.5); C WITH ACUTE (U+0107)/ C + J in Montenegrin (U+0441 U+0301) (9.1)			(Discussed for Swedish and German umlaut in 6.3)
3. Character substitutability dependent on the string						
	Identical: ZWNJ/zero (5.21)			EYELASH RA in Nepali represents a different phoneme from RA, and is currently only rep'd by combination of RA with ZWJ (U+0930 U+094D		

	Arabic	Chinese	Cyrillic	Devanagari	Greek	Latin
				U+200D) (4.3.1); Homophonous spellings (4.2)		
4. Upper/lower case and underspecified information						
	Optional diacritics (notably vowels) (6.4)	Instances of one-for-many substitution of SC for TC, often context dependent (2.1 final para.)	IE/IO in Russian (3.1); SMALL LETTER I WITH GRAVE (3.3);		TONOS may be absent (11)	Case folding (6.3) (Discussed at 6.7: Punctuation: Latin characters correctly substituted for APOSTROPHE)
5. Improper characters			APOSTROPHE in Ukrainian (3.6)	MODIFIER LETTER APOSTROPHE (U+02BC) / zero in Boro/Dogri/ Maithili languages (3.4)		
B. Inter-script						
			Especially of concern for WHOLE-SCRIPT CONFUSABLE strings with Greek and Latin (4)	Especially of concern for WHOLE-SCRIPT CONFUSABLE strings with other Brahmi scripts, e.g. Gujarati (3.5, 4.1)	Especially of concern for WHOLE-SCRIPT CONFUSABLE strings with Cyrillic and Latin	(Discussed at 6.1, 7 and 8.) Especially of concern for WHOLE-SCRIPT CONFUSABLE strings with Greek and Cyrillic
II. Linguistic variants						

	Arabic	Chinese	Cyrillic	Devanagari	Greek	Latin
					Equivalence between corresponding words in Dimotiki and Katharevousa dialects (13)	

I. Character substitutability

The variant issues in Section I are all based on indeterminacy in the representation of single characters, not strings.

Unicode typically assigns a unique code point to what it recognizes as an "abstract character" (i.e., a unit of information used for the organization, control, or representation of textual data), and this combination of code point and character in turn corresponds at a single glyph in the abstract. Here we consider cases where this fails for some reason.

A. Intra-script

Here we only consider variant issues which show up within the glyphs and rendering of a given script. However, the confusions often show up primarily because we are comparing the differing uses of a script (and glyphs with in it) made by different languages.

1. Same abstract character

1a. Abstract characters that have more than one encoding

A clear example that many composite characters (e.g., vowels and consonant signs that carry diacritic marks, such as <é>, <ā>, < ç>, <ṭ>) have more than one representation as code points, namely, as pre-composed wholes, and as the simple letter with a separate combining diacritic. This occurs in Arabic, Cyrillic and Latin. Many of these cases (including some that were raised in the case study reports) do not bear on IDNs, because of the requirements for what qualifies to be a code point in a U-label. In particular, where there are different standard ways of encoding the same abstract character as precomposed and decomposed forms, normalization will render them all the same. There are nevertheless some cases where normalization does not completely cover the range of potential inputs. (See [case study refs])

A rarer case is where the glyphs assigned to distinct code points turn out to be indiscernible. It is unclear in these cases whether there is more than one abstract character involved: in any case, the variants do not contrast with one another, and it is a matter of indifference, in ordinary use, which one is intended. There is, however, a security danger comes from code points which are distinct, but invisibly so.)

This is the case of Chinese generic variant characters (often resulting from the unification of sources to create UniHan), but also of the two ways of expressing 'turned e' or 'schwa' in Latin script.

The decorated characters in Latin provide many examples of distinct (sets of) code points which represent the same characters, although the Latin report (with one exception) recommends banning them all from use in TLD labels.

1b. Apparently the same abstract character in some contexts

These are like the cases in 1a, but differ in that the contrasting code points only have differing glyphs in some contexts.

This is frequent in the languages written in Arabic script, where the different glyphs may represent what are historically different ways of writing a single given character (but have been given varying code points in Unicode).

Conversely, the different forms of an Arabic character which are seen at different word-positions (initial, medial, final and isolated) are all generated by renderings of the same character (at a single code point). It will be difficult, therefore, to relate them as variant characters, since they have no identity as separate characters.

In Greek, the positional form of lower-case s (sigma) occurring finally in a string has been assigned a separate code point (U+03C2 **ς**, not U+03C3 σ). Nonetheless, it clearly represents the same conceptual character, and in upper case <Σ> shows no variation.)

2. Different abstract characters

These examples are cases where there are two characters, and two code points, but they are treated as variant characters for a particular reason.

2a. Different abstract characters interchangeable for users

The classic case is the Chinese equivalence between Simplified and Traditional characters. These are defined as quite separate code points and hence (prima facie) separate characters. Yet in terms of their contribution to writing particular words, there is no difference between the members of a corresponding Simplified-Traditional pair of characters. The only way to make them contrast is to talk about the actual form or identity of the characters themselves, rather than what they mean or how they sound.

In Arabic script, many combinations of alif and hamza with vowels are assigned their own separate code points: they are distinct from those vowels without alif and hamza, but the conventions of the Arabic language (and some other languages') spellings allow the forms with and without alif/hamza to be equivalent. In other languages that use Arabic script, this equivalence is not present.

A very different instance is the treatment in China of Japanese TLDs which are in kanji (Japanese characters) only. Although no simplified characters can be involved here, it is suggested that – to prevent confusion among Chinese users – use of any characters which look like Chinese simplified should trigger in the root the blocking of TLDs which differ only by using their corresponding Traditional equivalents.

2b. Simple visual confusability

These cases are self-explanatory. Glyphs are not sufficiently distinct to represent reliably their associated code points (nor abstract characters).

2c. Compatibility mappings

By this is meant cases where, within a particular language's spelling system, a digraph or trigraph (sequence of two or three glyphs) is taken as a conventional equivalent for another, usually rather rare and distinctive, glyph. If any of these equivalences are directly represented in the root, the effect will be to impose this equivalence on all languages which use the given script.

3. Character substitutability dependent on the string

There are certain characters (notably ZWNJ and ZWJ – and since their effects are indirect, they may be better considered as control characters) which may, under the rules of the IDNA Protocol, only be inserted when in positions in a string that trigger a particular effect.

In Nepali: a. to show "eyelash ra", a separate character from ra, not represented by a separate code point (rendered through the insertion of ZWJ); or b. to show presence of a morpheme-boundary (rendered through the insertion of ZWNJ).

Syllable-final nasal (anusvara) in Hindi representing a nasal consonant before a following syllable-initial consonant (e.g. writing ambā, hindī gaṇgā as "ãbā, hĩdī, gaga," as is normal in Devanagari). These are the "homophonous spellings."

4. Upper/lower case and underspecified information

While DNS labels are octets and can hold any octet desired, they are defined in RFCs 1034[13] and 1035[14] such that they treat ASCII in a special way. While the DNS preserves ASCII case differences, the difference is not considered in the matching rules. For instance, the label "example" in the DNS is supposed to be preserved just as entered in the authoritative zone file during transmission and caching, and also the label "ExAmple" is also supposed to be so preserved. These two labels, however, could not both be in the same zone for the same resource record, because for the purposes of matching they are the same.

The IDNA Protocol uses a different rule: a code point must be stable under NFKC and case folding operations to be a valid code point in the protocol (see RFC 5892[15]). Therefore, upper-case characters in scripts that use them (Latin, Greek, and Cyrillic) are not allowed in U-labels – even if the character would have the difference preserved in a traditional LDH label (to use an artificial example from above, the

[13] http://www.ietf.org/rfc/rfc1034.txt

[14] http://www.ietf.org/rfc/rfc1035.txt

[15] http://tools.ietf.org/html/rfc5892

string “ExAmplé” is not a U-label, because of the two upper-case characters E and A. This means that a distinction is present in usual Latin/Greek/Cyrillic text is routinely neutralized in U-labels and hence IDN TLDs.

5. Improper characters

There is a requirement across languages for characters which are in a block of code points distinct from those reserved for the relevant scripts: e.g. the apostrophe character used in Cyrillic for Ukrainian and Belarusian; the apostrophe character used in Devanagari for Boro, Dogri and Maithili.

Although these characters are not variants in themselves, their presence or absence in the spelling of specific names and words is not straightforward. Because of the characters' status outside the regular code block for the script, the prospects for their routine use are unclear. If both presence and absence of these characters are ruled as acceptable spellings in TLDs, names and words where these characters are used will become thereby open to use as variant labels.

B. Inter-script

If labels are to be restricted to a single script, the only case of this is the whole-script confusable case. It is expected that there will be mechanisms to detect these, and block them where necessary, but no variant cases will arise. There is a bigger issue to consider if labels are to be constructed out of sets of code points not restricted to those sharing a single script property; see the discussion in section 4.1 for some alternatives.

II. Linguistic variants

This raises a different set of issues from the Character Substitutability cases above. Potential variants are whole strings (usually words) rather than single characters within a string.

4. Discussion of Issues: Establishing Variant Labels

The root is a shared resource, and that means that ICANN cannot in principle privilege one group of language users over another in the administration of the root zone. Accommodating everyone is necessarily complicated. However, it is necessary to have an agreed and accepted mechanism for defining code point variants for the root that takes into account the needs of the language communities. ICANN must have a way to validate IDN TLD variants when submitted, and to validate all IDN TLDs requested for variants and variant conflicts.

Identification of an appropriate authority for code point repertoires (script tables) for TLDs is a difficult undertaking. To the maximum extent possible, the relevant language communities need to agree on a shared code point repertoire. It is not clear on what authority ICANN would rest its claim to being able to decide between different code point repertoires in the event of a dispute. If, on the other hand, no dispute exists, then it does not seem that there would be different repertoires submitted. It would be

better for all to co-ordinate a single repertoire for the code points in question. Ideally there should be no advantage in creating subtly different repertoires for code points allowed in TLDs.

4.1 Options for a Label Generation Policy

It appears that there are four broad methods for building a Label Generation Policy for the root zone. In general, only one of these strategies can be adopted, although any of them could be modified somewhat in the particulars.

1. Complete generation for every Script Table to be used

   The first approach is to use the Unicode Script Tables, and build a full policy for every code point in every Script Table that is to be used. (A policy would not be needed for scripts that are not going to be supported in the root zone, though of course it would be necessary to build such a policy on the introduction of a new script.) This would likely require the compilation of complete lists from different language communities using the given scripts, and also asking those communities to specify the variant rules for the code points. The committees that come together for a given script might be called the "script expert panel." In the end, a single Label Generation Policy would be merged from each of the authoritative lists from different script expert panels.

   Using this method, there is some question about what to do about code points that are needed by some language, but that are not included in the script the panel is considering. For instance, several different languages may depend on code points in category Common or Inherited; in such cases, it is not clear which script expert panel should be consulted. Along the same lines, dealing with languages written in more than one script (as e.g. Japanese is) could be extremely tricky. Since the deliberation is at the level of a script, it would seem necessary to require that labels always be in exactly one script, with exceptions to be worked out where need be (though it is not clear on what basis the exceptions would be determined).

   It is also not clear what to do in case a script expert panel cannot come to consensus and deliver the subset of the Label Generation Policy for the Script Table in question. If consensus were required, it would permit a single user of the script to block registrations for everyone. While we do not have examples of such fractiousness in our work so far, how languages are written is often an intensely (politically) fraught question.

   Three of the case study teams noted that they did not have complete expertise in at least some languages that use the script under consideration, and it is not clear how teams could be constructed that would ensure such expertise for every language that uses a given Script Table. It would appear that the script teams that studied the different scripts are the best examples of expert teams ever assembled on this topic, and if those teams were unable to deal with some of the languages using the script, prospects for future teams that will be

more comprehensive may not be good. It might be that such an effort could not proceed on a volunteer basis, but would need paid staff. There is always the possibility of overlooking some language that uses the Script Table, also, but perhaps with sufficient guidance from linguistic experts such an eventuality could be avoided.

In the event Unicode added or removed a code point from a script, it would be necessary immediately to convene a new script expert panel to decide what to do about the new code point. It would be critical to prevent any registration with that code point until the panel had ruled, and probably necessary to have grandfathering rules for handling already-registered labels. If a code point changed scripts -- unlikely but not impossible under the Unicode rules -- then a similar immediate convention of the expert script panel would be needed. It is worth noting that the most common code points available under IDNA2008 have been stable for a very long time, and are extremely unlikely to change in this way; all the examples are likely to be very rare characters. (To all but eliminate this risk, it might be worth adopting a stability criterion so that a code point is ineligible for inclusion in the Label Generation Policy unless it has been completely stable in several Unicode iterations.)

2. Create policies only for code points actually requested in a Script Table

   A second approach is to again depend on the Unicode Script Tables, but to build up the set of permitted code points in the Label Generation Policy on the basis of submissions by would-be users of the code point. In this case, the same expert team as above would work on the script, but it would not have to complete a policy for every code point in the script; instead, it would merely have to complete a policy for every code point submitted in a request. Would-be users of the code points would need to make a request in advance of the script expert panel's convening.

   A user desiring to use a code point would need to ensure that it was already included in the Label Generation Policy. A code point not so included would be excluded from registration in the zone until the script expert panel could review it and add it to the Label Generation Policy (along with any rules defining variant behavior).

   This mechanism has most of the disadvantages and advantages of the approach outlined in the previous section. Using this method, however, no decision at all needs to be taken about code points nobody has requested for registration (or for handling under variant rules), which means that there is no difficulty at the outset in dealing with code points used by language communities not engaged with ICANN. It also reduces the chances of serious issues arising from changes in Unicode, because code points that are likely to have any changes are also the ones least likely to be in use. On the other hand, it requires a more or less permanent standing committee of script experts for every script reflected in the root zone. The Label Generation Policy is more likely to experience some instability under this approach than under the first.

Exactly how the expert panel would be retained and how it would be compensated if it needed to do work present a related set of issues. There are other mechanisms within ICANN for other issues where expert panels are occasionally convened or consulted, and those mechanisms might provide models for how to arrange this effort. It would be critical to ensure that any panel included expertise both on all of the languages affected by the Script Table, and expertise on the operation of DNS and IDNA.

3. Create policies for arbitrary lists of code points

Because of the possible difficulties with "one Script Table" policies, it might be desirable instead to permit language communities to specify the code points they want to use to write their language. Assuming the listed code points were PVALID or CONTEXTO or CONTEXTJ and otherwise conformed to root-zone policies, these code points and associated rules could be combined into a single Label Generation Policy.

In order to generate the initial lists and rules, bodies similar to the script expert panels described above would need to be convened. These might be organized around broad script categories (as the Variant Issues Project case study teams were), around a specific language or multiple languages, or even across linguistic lines in the event that seemed desirable (e.g., to address cross-script Latin/Greek/Cyrillic issues). Panels would work on code points with which they were familiar, and would usually need representatives from all the relevant language communities. The set of code points that they would generate could be termed a "repertoire." As in the previous section, code points that did not seem useful to the panel, or which nobody understood, could be ignored (and disallowed for registration until a policy were developed to address those code points).

It would be advisable for some initial panels to be convened at the outset, to deal with those code points judged most likely to be desired (determined for example by a survey of existing and prospective TLD operators). Each panel would prepare a repertoire and an associated set of rules for any code point in the repertoire that was supposed to generate variants. The different initial repertoires would be combined into a single Label Generation Policy, with conflicts in any variant rules always being resolved by a "Block" rule.

The individual repertoires would be associated with some identifier (such as a Language Tag), and would be registered with IANA so that they could be referred to later. Subsequent IDNA registrations in the root zone would need to refer to a Language Tag in the IANA registry, and all and only the code points from the relevant repertoire would be permitted in the label.

If a suitable existing repertoire were not available, then those wishing to adjust an existing repertoire, or to create a new one, would need to request a new panel to create a suitable repertoire. The mechanisms for setting up and maintaining such a panel would be similar to those outlined in the previous section. It would be important, however, to create a process

that tended to discourage the creation of new repertoires or many modifications to existing repertoires. The reason for this is the same as the reasoning behind creating variants in the first place: if the goal is to ensure that users are not surprised, then rules that are different depending on when a name was added to the root zone would violate that principle of least surprise. Once a Label Generation Policy permits a code point or establishes a variant rule, it should be difficult (although not impossible) to change that.

A notable advantage of this approach is that it avoids the somewhat artificial link between Unicode Script Tables and any given language. In addition, because it does not require any treatment of code points for which there is no demand until that demand appears, it avoids having to address code points for which the expertise is not available. A significant disadvantage to this approach is that it relies less on existing attributes (such as Unicode script properties) and relies more on expert groups to make findings about the code points. It also has the potential for "deadlock" and competing repertoires where different groups of users cannot come to agreement about how to deal with a subset of code points.

4. Build up repertoires *ad hoc*

The approach outlined above might be altered to allow anyone to create and register a repertoire (instead of requiring an expert panel to create one). In this case, instead of ICANN selecting an expert group, repertoires would be created by interested parties, organized according to whatever principle they liked. The combination of these repertoires into a single Label Generation Policy would still need rules in order to deal with conflicts; in general, the rules should always pick the most conservative action, in order to avoid controversial or possibly harmful entries in the root zone.

The principal advantage of this approach is that it takes ICANN completely out of the business of deciding whose expertise counts with respect to a language or script. A significant disadvantage of this approach is that it is subject to denial of service: someone who is opposed to variants in principle could always create a conflicting rule, and the conflict-resolution mechanism (in order to do the most conservative thing) would therefore tend to prevent any variant labels from being Activated. Moreover, because this approach tends to encourage more innovative repertoires and associated rules (as there is little cost to adding one), it is more likely to lead to instability in the Label Generation Policy.

4.2 Distinguishing Variant Labels and Visual Similarity

Labels that are **visually confusable**, as noted by Unicode,[16] refer to two different strings of Unicode characters whose appearance in common fonts in small sizes at typical screen resolutions is sufficiently close that people easily mistake one for the other. "Small sizes at

[16] http://unicode.org/reports/tr36/

screen resolutions" means fonts whose ascent plus descent is from 9 to 12 pixels for most scripts, and somewhat larger for scripts, such as Japanese, where the users typically have larger sizes.

To further explore variant versus visual similarity issues, the following analysis can be pursued using two parameters:

a) Linguistic entity (same, different), and

b) Shape (same, similar, and different).

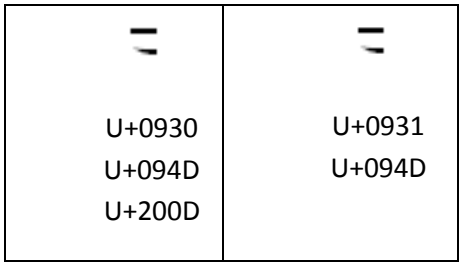
In this context, a "linguistic entity" refers to a character (i.e., one or more code points) that in some language that uses the script is non-distinct (i.e. substitutable in some word without change in the word's identity).

This covers (in the context of the broadly alphabetic scripts such as Arabic, Cyrillic, Devanagari, Greek, or Latin), code points which represent the same letter, and (in the context of ideographic scripts such as Chinese), code points which represent a character with the same pronunciation and meaning. Specifically, Simplified and Traditional characters which are equivalent in writing Chinese represent the same linguistic entity.

The analysis produces the following six categories possible for a given IDN TLD label:

1. Same linguistic entity, same shape
2. Same linguistic entity, similar shape
3. Same linguistic entity, different shape
4. Different linguistic entity, same shape
5. Different linguistic entity, similar shape
6. Different linguistic entity, different shape

Table 1 contains examples from within different scripts for each of these categories.

	Parameter	Arabic Examples	Devanagari Examples	Chinese Examples
1	Same linguistic entity Same shape	ہ , ھ 0647, 06BE	None[17] unless the eyelash ra is considered  U+0930 U+094D U+200D U+0931 U+094D	黄黃 U+9EC4, U+9EC3
2	Same linguistic entity Similar shape	ی, ى 0649, 06CC	Not possible in Devanagari and is possible only if cross-script similarities are permissible. 4.1. 0f Devanagari report	户 戶 U+6237, U+6236
3	Same linguistic entity Different shape	ک, گ,(all voiced velar stops) 06AB, 06AF	ऍ अॅ[18] U+090D U+0972	学 學 U+5B66, U+5B78

[17] This comprises Nukta "normalization" already handled by IDNA 2008. The case of eyelash ra also figures here cf. 4.3.1.. of the report. Both these cases are not listed here since the first is handled by IDNA and in the second case ZWJ is not permissible for GTLD's for a single string.

[18] अॅ can be generated out as a single character and also as a combination of अ+ZWJ+ॅ i.e. U+0905 U+200D U+0945 in which case it would also fit in Category 1

	Parameter	Arabic Examples	Devanagari Examples	Chinese Examples
4	Different linguistic entity Same shape	initial and medial positions of ف, ق 0641, 06A7	ख र्व U+0916, U+0930 U+0935	士 土 U+58EB, U+571F
5	Different linguistic entity Similar shape	ى ی 06CC, 06CD	At single character level ध,घ : U+0927, U+0918 द्ग द्न द्र U+0926 U+094D U+0917 U+0926 U+094D U+0928 U+0926 U+094D U+0930	力 刀 U+529B, U+5200
6	Different linguistic entity Different shape	most of the alphabet	most of the alphabet excluding the types above	Most pairs of ideograms

Table 1: Parametric Analysis based on linguistic entity and shape and examples

From this exercise, it is clear that category 6 is not relevant in the discussion because it represents both a different linguistic entity, and a different shape. If there are confusability issues, they can be addressed by current processes.

It is also clear from the analysis that there are two types of user expectations: for categories 1, 2 and 3, different code points are linguistically the same unit, and thus from a user perspective, they could be somehow equivalent. Categories 4 and 5 are different linguistic units, so there is no user expectation that they be equivalent; however, strings that are presented as candidate alternatives in this category could be blocked in the root zone for security reasons.

To highlight this distinction, we define two categories: activatable variants, and block-only variants.

- **Activatable variants** that would include categories 1, 2, and 3 below. These categories are calling for a linguistic relation between the characters that makes them equivalent somehow. These categories of variants can be blocked, reserved, allocated, delegated, activated.
- **Block-only variants** would be to include categories 4 and 5 which would be block-only variants (variants that can only be blocked, not even reserved).

It is clear that today there is no process to treat activatable variant TLDs, therefore a new process needs to be developed.

To treat blocked variants, there are several options:

Option 1: Define a new process to treat blocked variants.

Option 2: Enhance the string similarity review process to handle blocked variants.

Discussion:

1) The string similarity review process is established with the objective to prevent user confusion and loss of confidence in the DNS resulting from delegation of many similar strings. Note, according to AGB, “Similar” is defined as strings so similar that they create a **probability** of user confusion if more than one of the strings is delegated into the root zone.

2) From the discussions, it seems that in some script community, it is possible to find an acceptable table that would be able to generate blocked variants deterministically.

3) If 2) is possible, it seems that the string similarity review can be improved to take into account the tables.

4.3 Cross-Script Visual Similarity

Cross-script visual similarity is also possible, especially among Cyrillic, Greek and Latin scripts, but also as between Devanagari and closely related Brahmi scripts (e.g. Bangla, Gurmukhi). It is possible for two candidate U-labels to appear to be the same in different scripts. Even excluding

Unicode properties like Common and Inherited, it is possible to create two IDNA-valid strings in two different scripts where competent users of each script are likely to regard the two strings as the same one. This issue is highlighted in some team reports as "whole-script confusables," which is also what a similar phenomenon is called by Unicode.

There are two possible ways to address this problem during registration of TLDs in the root. The first is to treat this as a straightforward matter of string similarity; in that case, the issue is outside the matter of IDN variants, and it can be treated according to other policies. The second way is to treat these candidate strings as variant TLD labels. If the latter strategy is to be adopted, then any mechanism for generating variant labels will, at minimum, need to be able to detect and correctly block or reserve such whole-script cases. Note that, because this is an issue of all the component code points being confusable with one another, it is distinguished from the whole string issues discussed in section 4.4 below.

4.4 Whole String Issues

As opposed to character-level alternatives (in which the locus of difference lies in individual characters within a string), whole-string level alternatives would set whole strings (potential DNS labels) in contrast. In this section discusses the issues concerning recognition of alternatives of this kind as variants.

As defined above, however, whole-string alternatives may be divided again into:

a. **string alternatives proper**, where whole strings of characters are transformed or related by some formal process. Effectively, these include all potential transformations of whole strings, if they can be performed algorithmically. Examples include:

    1. Transliterated strings (e.g. Ευαγγελία (Greek) and Euaggelía)
    2. Whole-script confusable strings (e.g., *peace* (Cyrillic: U+0440, U+0435, U+0430, U+0441, U+0435) and *peace* (Latin: U+0070, U+0065, U+0061, U+0063, U+0065)
    3. Inverted strings (e.g. Yentob and Botney)
    4. Alphabetically shifted strings (e.g. IBM and HAL)

b. **linguistic alternatives**, where the transformation appeals to some part of the rule-set of a language, whether its orthography, its phonology, its grammar, or its lexicon. Examples of what might be considered linguistic alternatives include:

    1. Transcriptions (e.g., Ευαγγελία (Greek) and Evangelia), 网络 (Simplified Chinese) and *wang luo* (Pinyin transcription), which means *network* in English), which involve constructive use of another language's phonology and orthography to reconstruct the effect of an expression's pronunciation in a given language.

2. Homophonic strings within a language's spelling system (e.g., English too and two); this will include different spellings of the same word (lexeme) in the same language and script (e.g., in English: color and colour); in the extreme case, this may extend to strings representing the same word in the scripts of a multi-script language (e.g., in Japanese: ほんだ (Hiragana), ホンダ (Katakana), 本田 (Kanji), Honda (Romaji ~ Latin)).

3. Corresponding terms in different dialects of the same language (e.g. Ευαγγέλιος (Greek) and Βαγγέλης, which are accepted forms of what was (historically) the same name (in Katharevousa and Dimotiki respectively); *diaper* (US English) and *nappy* (UK/AU English).

4. Synonyms within a language (e.g., in English: car and automobile).

5. Different identifying expressions within a language (often dependent for their interpretation on conventions, or other shared information): these may be personal (e.g. *Elizabeth II, The Queen, the present queen, the Queen of England, the Head of the Commonwealth* etc.), geographic (e.g., in English: *New York City, NYC, New York New York, the Big Apple* etc.), or in any other field.

6. Translated strings and inter-linguistic correspondences: representations of the same concept/notion in two languages (e.g., green (English) and vert (French))

While the formal rules that define string alternatives proper may be represented as well-defined automata, this is not possible for linguistic alternatives, since generalizations within and between languages are subject to arbitrary exceptions, as well as being hard to define formally, at any level of abstractness.

In this breakdown, it becomes clear that whole-string alternatives are (potentially) the result of any formal transformation, including any form of encipherment, with little or no relation to user experience (e.g., inverted strings, alphabetically shifted strings). On the other hand, linguistic variant rules are never exhaustively and formally predictable, being dependent on equivalences made at various levels (from orthography through grammar and meaning to irony) by users of a given language.

Given the analysis of linguistic alternatives above, it is clear that some people might want some of them to be considered variant labels. Writing system and spelling reforms had implications for languages that have at times been written in Cyrillic and sometimes also either Arabic script or Latin script. Even English has undergone significant spelling reforms, such that some words can be spelled more than one way (e.g. 'color' and 'colour').

One of these types of alternatives, "Corresponding terms in different dialects of the same language," was recommended for consideration by the Greek case study team. The Greek case

study team report recommended that Katharévousa strings and Dimotikí strings be treated as variants to one another, such that the registration of one such string should cause the corresponding string to be Blocked.

The Greek case study team report noted:

> Alternative, same[-]meaning words, if the domain consists of words[,] are to be disallowed. This rule is especially put in place to deal with words in "Katharevousa" and "Dimotiki". Dimotiki is the contemporary Greek language, where Katharevousa was in use before 1976. However, many word types from Katharevousa are still in use in Dimotiki and the user should be protected of confusability issues between these same meaning words. For example "Πειραιάς" and "Πειραιεύς" are two names of the same city in Greece, respectively in Dimotiki and Katharevousa – The applicant will only choose to register one of these and the other one should be excluded of registration. The team recognizes the difficulties this procedure presents for automated systems, however, since TLD registrations are not expected to be of great volume this rule could be implemented with relatively low cost for ICANN. (p.14)

The first sentence of this text suggests that the Greek Case Study group has not made a distinction between the requested inter-dialectal equivalence of words and a more general synonymy, which would be highly subjective in detailed application. Since the recommendation is only for the inter-dialectal equivalence, this may not matter. However, even if check-lists of all the relevant words and names are available, the relationship is less neat than may be assumed.

Although Katharevousa and Dimotiki are largely related formally, this is overall a complex and sometimes unruly linguistic relationship.

For example there is a general principle (among many other equivalence rules) that ancient (Katharevousa) names ending in -εύς have a modern (Dimotiki) form ending in –εάς or –ιάς: e.g. Ατρεύς Ατρέας: βασιλεύς βασιλιάς: Ζεύς Δίας: Ιδομενεύς Ιδομενέας: Πειραιεύς Πειραιάς etc. The list is long, and not clearly finite. But at the same time the equivalence, in the most salient cases at least, is not exact, but highly context- and referent-dependent: so Ατρεύς not Ατρέας is the asteroid Atreus, whereas either can refer to the mythical hero of that name. On the other hand, Ζεύς and Δίας can both refer to the god Zeus, but only Δίας applies to the planet Jupiter. If the Dimotiki and Katherevusa forms were to be judged equivalent in TLDs, implementers would find it necessary to refer to a specific fixed list of names (perhaps even excluding Ατρεύς and Ατρέας, Ζεύς and Δίας). This might have the effect that if new names in -εύς or –εάς or –ιάς were invented, they would not automatically get a variant label. On the other hand, native speakers of Greek would largely be able to predict a possible Katharevousa form, when presented with a new Dimotiki example ending in –εάς or –ιάς.

A further problem stems from the need to agree a specific fixed list, not only of names, but also of every word in the Greek language (onomasticon), and specifically one which specified the

Dimotiki and Katharevusa equivalent for each headword (lexicon). Besides the need to maintain both of these, there is the further complication that Greek is a highly inflected language, and policies would need to be defined on which precise parts of each word's alternations would be recognized.

The aforementioned paragraph from the Greek team report is requesting to "exclude from registration" the second alternative string, i.e., blocking it. The gTLD Applicant Guidebook already defines two processes that could take care of the issue in a similar way without the need to establish such alternatives as variant labels. The String Similarity review described in Module 2 compares any two applied-for gTLD strings for visual similarity. If the strings were found to be confusingly similar, they would be placed in contention sets for later evaluation and resolution. Additionally, the review compares each applied-for gTLD string against existing TLDs. If the applied-for gTLD string failed the String Similarity review due to similarity to an existing TLD, the application would not pass the Initial Evaluation.

Secondly, Module 3 describes the public objection process, in which string confusion is one possible grounds for a formal objection to be filed to a gTLD application. Such confusion is not limited to visual similarity. Rather, confusion based on any type of similarity (including visual, aural, or similarity of meaning) may be a basis for an objection. The use of any of these two processes could yield the outcome requested by the Greek case study report for the Dimotiki/Katharevousa issue.

On the more general topic, as noted above, these sorts of linguistic alternatives are not amenable to algorithmic treatment, because the linguistic principles that cause them do not usually exhibit complete regularity. In order to produce a general solution to any general case of linguistic alternatives (even in a single language), it would be necessary to generate (in advance) a dictionary or set of dictionaries that would govern the handling of submitted strings; this would require the engagement of relevant expertise for each language with linguistic alternatives. Presumably, the dictionaries would need to be reconciled with one another, in order to ensure that any possible conflicts would be dealt with.

Since there is no requirement that DNS labels actually be words, submitted candidate labels would need to be checked to see whether they matched anything in the relevant dictionary or dictionaries, probably including checking for substrings in the dictionaries and substrings in the submitted candidate label. In the event a match was detected, then zone policy would apply as to what to do with the resulting matching label (e.g., to Block it, Reserve it, or Activate it).

It would be necessary to develop a policy for maintaining the dictionaries. It is not clear how ICANN could make effective exclusion rules permitting one type of linguistic alternative to be treated as variant, but another not to be. It is also not clear that the system would be subject to practical automation, given the very large number of potential combinations and the likely requirement to perform substring matching. It would be necessary to formulate a policy that

could accommodate future changes to dictionaries, and to formulate a policy that accommodated such changes as to create a conflict between then-existing root zone delegations. Finally, it seems it would be necessary to have a dispute resolution procedure to handle cases where different applicants' desires went unmet.

Conclusions regarding whole-string alternatives

1. There is no historical precedent, and apparently little demand, for whole-string alternatives to be considered as variant labels.
2. String alternatives proper would be easier to implement as variants than linguistic alternatives, since they would be implementable by some formal rule of transformation. However, they probably make less sense from a user-experience perspective given their lack of ties to language.
3. If linguistic alternatives were desired as variants, generalizing them would be expensive, because of the irregular nature of the linguistic principles in use.
4. Placing any such variants in the root will create indeterminacy in the system, and uncertainty for users. If a user knows that linguistic variants (with all their intrinsic unpredictability) are part of the system, it spreads the sense that what is typed (or seen on the screen) is not necessarily what others get. This will be a pain. And it may be a bigger pain than the convenience that others will derive in having a facility of linguistic whole-string variants. Such a facility will inevitably be limited, and we do not yet know how its limits could be set.
5. There are processes in place that can produce the blocking action for some of the whole-string alternative issues described, obviating the need of widening the taxonomy of variants for the root zone.

5. Discussion of Issues: Treatment of Variant Labels

Once variant labels are identified, a range of possible actions may be taken on such labels, such as:

Block: An administrative action by a registry over a particular string (a potential domain name) rendering such string unavailable for allocation to anyone.

Reserve: To set aside a name for potential allocation to a particular entity (TLD registry in the case of the root). The name is not yet allocated, but could be (to that particular entity) if/when certain conditions are met.

Allocate: It is the first step on the way to activation. The registry makes an administrative association between a string and some entity that requests the string, making the string a potential label inside the zone, and a candidate for activation. Allocation alone does not affect the DNS at all.

Delegate: The act of entering parent-side NS (name server) records in a DNS zone, thereby creating a subordinate namespace with its own SOA (start of authority) record. See RFC 1034[19] for detailed discussion of how the DNS name space is broken up into zones.

Mirror: To activate two or more domain names such that for a namespace starting with one, the namespace starting with the other is isomorphic to the first, subject to the usual DNS loose consistency strictures. Currently, there are two different techniques for this. The first is aliasing: CNAME, DNAME, and other such techniques that redirect a name or a tree, effectively substituting one label for another during DNS lookup. The second is by using provisioning constraints, such that an underlying provisioning system always effects a change in all of the names whenever that change is effected in one of the names.

Activate: The act of entering parent-side DNS records in a zone (e.g., NS, DNAME, CNAME, NAPTR) making the name entered resolvable in the DNS. (Note that delegating a name implies activating it, but activation does not necessarily imply delegation.)

These actions may result in a different user experience, as well as having an impact on the operations of ICANN, the TLD registry operator, and other stakeholders that are part of the Internet ecosystem. This section discusses the issues that arise in this area as a result of variant TLD labels.

5.3.1 Possible States for Variant Labels

The states associated with the actions described above are of the predictable forms: Blocked, Reserved, Allocated, Activated, Delegated, and Mirrored.

Since **Blocked** means that nobody may have the allocation, it follows that nobody could request such a state (for that would imply some sort of proto-allocation over the domain); therefore, Blocked is a pure matter of registry policy (in the context of this report, an action taken by the registry for the root zone, i.e. ICANN). Blocking could be the consequence of the combination of a registry policy and some state of the registry. For example, the rule for a code-point could be such that, if that code point is allocated, then some other code point must be blocked.

Because blocking is a matter of registry policy, the change of a label from blocked to any other state is either a consequence of a change in registry policy, or else a change in state in the registry such that the blocking condition (e.g., the allocation of a label with a block-generating code point) is removed.

[19] http://www.ietf.org/rfc/rfc1034.txt

A **Reserved** status results from the combination of policy and the requests of applicants. For example, suppose a registry policy permits (but does not require) a TLD label and its variant TLD label to be delegated. In the event the applicant chooses not to delegate the variant TLD label, registry policy may require that the variant TLD label be reserved to the applicant throughout the registration lifetime of the fundamental label. In accordance with an applicant request, the variant TLD label could move from being Reserved to being Allocated (and then anything that can be done with an allocated label would apply).

In principle, a change in registry policy (but not a change in registry state) could cause a label to move from Reserved to Blocked. If a change in registry state were to cause this, it would be an indication that the registry policy was inadequate in the first place, because it had not addressed potential conflicts well enough.

An **Allocated** status would result from any case where an application containing a variant TLD label is approved, and it isn't Blocked or Reserved. An allocated status could also result when a variant TLD label associated with a fundamental label is either also requested to be Activated, or required by registry policy to be Activated (see below). By request, the status of an Allocated label could be changed to Reserved. Allocated labels may normally be Activated, subject to usual registry policies, at the initiative of the registrant of that label (i.e., in the context of the root zone, usually the TLD operator).

An **Activated** status results from placing some sort of DNS record into the parent zone such that it is possible to perform a DNS lookup on the name and receive an answer. It is possible for the registry to require the simultaneous Activation of a group of names, denying all of them if any of them cannot be activated. **Delegation** and **Mirroring** are just species of Activation. A domain can be de-Activated by removing the relevant records from the DNS (for technical reasons, when there is a delegation the name often has to be removed from both the parent side and child side name servers in order that it stop resolving, but the details of how to ensure a name has stopped operating completely in the DNS are beyond the scope of the present report).

5.3.2 User Experience with Variant Labels

[Section pending]

5.3.3 ICANN Operations and Variant TLDs

In applying a variant management mechanism for the root, ICANN would incur a number of operational issues, discussed in this section.

5.3.3.1 Evaluation

The evaluation processes for gTLD applications are documented in the gTLD Applicant Guidebook.[20] IDN ccTLDs may currently be requested via the IDN ccTLD Fast Track process, in accordance with the procedures in the Final Implementation Plan.[21] The Fast Track was intended to enable the introduction of a limited number of non-contentious IDN ccTLDs, associated with the ISO 3166-1 two-letter codes, to meet near-term demand while the overall policy is being developed. The ccNSO is undertaking a policy development process[22] concerning the introduction of IDN ccTLDs.

These current processes do not allow for the delegation of IDN variant TLDs until such time as a variant management mechanism for the top level is in place. In the event that applications including variant TLDs could be submitted, the construction of variant sets could vary depending on the script or language involved. In some cases, there may be a "base label" with variant labels associated, while in other cases, a set of variant TLD labels would be essentially equivalent in status. Could a common format for characterizing a requested set of TLD labels be developed or would there need to be flexibility for each case to use a specific notation? Another issue would concern whether there was a maximum size to one of these sets, and what conditions might apply depending on the number of labels involved.

The impact of considering variant TLD labels ("strings") on evaluation processes depends significantly on the approach taken to a label generation policy, as discussed in section 4.1 above. If, at the time it is evaluating specific requests, ICANN can refer to an existing authoritative source that will produce a definitive result on whether two strings can be considered variant TLD labels, the impact would largely consist of new sub-processes to take the variant labels into account within existing evaluation steps.

If, however, such references are not pre-existing at the time ICANN is considering requests, then this work would need to occur in parallel, resulting in possibly extended timelines and, depending on the approach adopted, steps such as:

a. Review of documentation submitted by the applicant for validity.
b. Determining whether all variant labels listed in the application can be supported based on the documentation submitted.
c. Determining whether additional variant labels that were not included in the application should be included.

[20] http://www.icann.org/en/topics/new-gtlds/rfp-clean-19sep11-en.pdf

[21] http://www.icann.org/en/topics/idn/fast-track/idn-cctld-implementation-plan-16nov09-en.pdf

[22] http://ccnso.icann.org/policy/cctld-idn

In such a case, where evaluation processes need to account for a review and determination on the set of variant TLDs itself, issues to be addressed would include:

a. What would be the selection process and qualifications needed for individuals performing evaluation tasks? (The Chinese case study report, for example, has identified a need for linguistic experience and knowledge.)
b. What would be acceptable reference sources for evaluators to use in making judgments?
c. How would ICANN ensure that evaluators were unbiased in their decisions?
d. What precedential value would be set by evaluator decisions? Could each case continue to be assessed individually or would this create conflicts over time?

Similarly, the relevant costs for evaluation processes incurred by ICANN would vary depending on the availability of definitive sources and rules for establishing variant labels. A process involving ad hoc review and determinations on requests would add significantly both to the resources needed to execute such reviews and the risks to be anticipated in such a process. To the extent that additional costs are incurred by ICANN in connection with the evaluation of requests for variant TLD labels, an additional aspect of the set of evaluation issues concerns whether there should be additional fees instituted to cover any relevant costs.

These issues would seem to be consistent across scripts. While the sources and standards used by evaluators would vary according to the relevant script, the evaluation steps themselves would not differ.

The issues associated with treatment of variant TLD labels are discussed in section 4.3 above. If a variant classification system is adopted, and rules are in place for how each type of variant label can be treated, the evaluation is limited to confirmation that the application does not conflict with these requirements. In the absence of such practices, the desired states for each of the labels would also need to be reviewed in evaluation. Issues in such a case, in addition to those mentioned above, would include whether a certain variant TLD label in a requested state would create either a) a security or stability issue, or b) a user confusion or user experience issue. State changes could require a specialized evaluation process in some cases.

Once the variant TLD labels in an application have been accepted through one of the means described above, the evaluation tasks for a TLD application that included variant TLD labels would proceed through the usual stages as relevant. This could include reviews for:

a. String similarity – This review determines whether an applied-for TLD is so similar to an existing TLD or other applied-for TLD that it creates a probability of user confusion. In the review steps developed by ICANN, this analysis is limited to visual similarity. The string similarity review takes place in the interest of avoiding user confusion through the delegation of many similar TLD strings. Issues to be considered in the case where variant TLD labels are part of an application include:

1. to what extent the set of variant TLD labels in the application would also need to be part of the string similarity review against existing TLDs or other applied-for TLDs.
2. whether this review would differ based on the requested status (e.g., Delegated, Reserved) for any of the variant TLD labels in the application.
3. the appropriate reference sources for identification of similar characters.

b. DNS stability – This review determines whether an applied-for TLD meets the technical string requirements. An issue to be considered in the case where variant TLD labels are part of an application is whether all variant TLD labels included, whether intended for active operation in the DNS or not, would need to pass this review.

c. Geographic names – The IDN ccTLD Fast Track process contains a meaningfulness requirement: string(s) must be a meaningful representation of the name of the corresponding country or territory. The gTLD evaluation process contains a review to determine whether evidence of requisite government approval is provided where required. In the case where variant TLD labels are part of an application, it would need to be determined whether such requirements would apply to every label contained in the application, including the variant TLD labels, or whether the requirements could be differentiated within the set.

d. Technical/Operational/Financial – These reviews take place in the gTLD evaluation process and test whether the applicant has the requisite technical, operational, and financial capability to operate a TLD registry. An issue to be considered here is whether there are additional reviews that ICANN should undertake concerning the variant management mechanism itself, i.e., to ensure that any guidelines or requirements concerning the applicant's operation and management of the variant TLDs are anticipated and understood by the applicant, and taken into account in its plans.

e. Registry services – This review takes place in the gTLD evaluation process and determines whether the registry services offered by the applicant might adversely affect DNS security or stability.

As an additional consideration, the gTLD evaluation process contains a mechanism for formal objection to be filed to an application on certain limited grounds. A formal objection by a party with standing will trigger a dispute resolution proceeding with an expert panel rendering a determination. Note that such an objection is to an application, not to a string. In the case of an application containing variant TLD labels, an issue to be addressed is whether the set of labels in the application would always be kept together for purpose of the objection and dispute resolution proceedings, or whether they could be split, so that only one of the labels within the set was at issue.

5.3.3.2 Management of Established Variant TLD Labels

As discussed above, one possible approach incorporates a common set of rules developed for the root zone, to determine the circumstances under which variant TLD labels may be categorized with a given status (cf. section 5.3.1 above). In the absence of such a ruleset, it will fall to ICANN to make these decisions on a case by case basis. This would for the most part take place in an evaluation phase, except in cases where requests were being considered to change the status for a particular variant label. In these cases, resources and standards for considering the requests would need to be developed, raising the issues of:

a) Who is/are the appropriate parties for making determinations on such requests, and

b) What standards should be used for considering such requests.

When variant TLD labels are assigned to particular states, it will additionally be ICANN's responsibility to maintain records concerning the names in these various states. The database of delegated top-level domains (i.e., the root zone database) is maintained and made publicly available via the IANA function. To the extent that variant TLD labels are "blocked," "reserved," "allocated," etc. lists of such labels would need to be maintained and updated, along with clearly documented procedures for the circumstances, if any, where states for these names may be changed.

5.3.3.3 Delegation of Variant TLDs

To the extent that requests for delegation of variant TLDs are approved, this would take place following the existing IANA delegation procedures. This would include maintaining and publishing registration data for new TLDs, and distribution of updated zone file data according to current procedures. There may need to be an updated record format or new fields in the database to account for the association of one or more TLDs as variant TLDs.

5.3.3.4 Contractual Provisioning

Where a request including variant TLD labels is approved, ICANN may enter into an agreement with the relevant registry operator. Certain frameworks are available for ccTLDs, while gTLDs are expected to enter into a standard registry agreement with ICANN. These mechanisms can continue to be used for cases where there are variant TLD labels. A set of issues to be resolved, however, concerns the requirements to be followed by operators of variant TLDs. These could include:

a. Whether there should be specific reporting requirements concerning the variant TLDs, and if so, which data points are desired for reporting.

b. Whether specialized technical requirements for the management of the variant TLDs are necessary to support the security and stability of the DNS, and if so, how these requirements are specified

c. Whether specialized policy requirements for the variant TLDs are necessary to support a good user experience, and if so, how these requirements are specified.

d. Whether specific service-level requirements for performance of the variant TLDs should be instituted, and if so, how these requirements are specified.

e. Whether the registry fee structure should be adjusted to take into account the existence of the variant TLDs, and if so, how this should occur.

5.3.3.5 Security and Stability of the DNS

Security considerations are relevant in making the determination of which code points are valid and should be used in top-level labels, as well as the criteria for considering requests for IDN variant TLD labels. Similarly, such considerations are relevant in the determination of rules for assigning variant labels to particular states, and for the management of IDN variant TLD labels. These have been discussed in the relevant sections above.

Additionally, there has been significant study, consultation, and analysis in connection with expansion of the root zone to include new top-level domains. Modeling, monitoring, and reporting will continue during, and after, the first application round of the New gTLD Program, so that root-scaling discussions can continue and the delegation rates can be managed. To the extent that variant TLDs are delegated, these would be incorporated in this modeling, monitoring and reporting.

Delegation of any new top-level domains is conditional on the continued absence of significant negative impact on the security and stability of the DNS and the root zone system (including the process for delegating TLDs in the root zone).

5.3.4 Registry / Registrar Operations

Should a variant management mechanism for the top level be adopted by ICANN, this would entail a number of operational issues for registry operators as well as for registrars operating for the relevant TLDs. Although not necessarily tied to ICANN's management of the root zone, these are issues that would be expected to be resolved the ICANN community. These issues are discussed in this section.

5.3.4.1 DNS Resolution

Depending on the variant management mechanism implemented, delegation of variant TLDs may mean the TLD operator is required to invest more resources in zone file generation and management of registrations in the variant TLDs. This could also propagate to secondary name services and therefore increase the cost of running the DNS services for the registry. The maximum investment is possible to describe by imagining that the variant management mechanism is implemented via mirroring without any sort of aliasing. In this case, the TLD

operator must in effect operate exactly as many zones as variants delegated, with a combinatorial number of labels in each such zone.

To the extent that an aliasing behavior is desired or implemented in TLD registries, this will have an effect on TLD registry operations. However, policies regarding DNS behavior could be difficult to enforce beyond the level in the DNS hierarchy at which the policy is defined. Specifically, a registry may choose to establish a policy wherein all possible variant labels will behave the same (return the same response in the DNS) at the TLD level of the DNS hierarchy. Although this can work in many cases at the TLD level, the DNS cannot enforce this policy on the delegated second-level domain names in the TLD. This can have a dramatic effect on the user experience.

5.3.4.2 Registration Process

The shared registration system (SRS) is a critical registry function enabling multiple registrars to provide domain name registration services in the TLD. This in many cases includes the EPP (Extensible Provisioning Protocol) interface between the registry and the registrars. Extensions to EPP may be required to enable registration of second-level domain names under the applied-for TLD and the variant TLDs. Without a standard implementation of such extensions, registrars may face complexities in interfacing with these registries implementing different extensions.

Although not a variant issue per se but a general IDN issue, indications of the relevant script(s) and language(s) for registered domain names may need to be incorporated by the TLD registry. As noted in the Devanagari case study report, it may be sufficient in some cases to tag a domain name with either its script or its language; however, a script may support a number of languages and in some cases, a language uses more than one script. The technical issue is that there is no uniform way to do this in the standard EPP protocol used.

This issue also affects registrars in two ways. To the extent there is no standard, a registrar will have to implement all EPP extensions that various registries may choose to specify to resolve this issue. For those ccTLDs that do not use EPP, registrars will have to implement whatever is required in order to support that ccTLD.

In addition, when registrars are present they are the interface to the registrant. Registrars that choose to support multiple scripts and languages will need to develop user interfaces that facilitate and simplify the identification of the script and language in use by a registrant, and permit the registrant to understand its choices with respect to the names it is actually contracting to operate when registering a name subject to variants.

Accordingly, it appears that a successful registration process in variant TLDs will require significant coordination, perhaps including an additional OT&E process, with registrars.

It will be the task of the registry operator to formulate policies on how domain names are managed in the variant TLDs. For example, policies could cover whether the same domain name

under the variant TLDs must be associated with the same registrant. Consideration would also need to be given to registry policies on expiration, deletion, and transfer of registered names. It will also be for the registry operator to determine the appropriate pricing models for such registration offerings. The relevant policies concerning registrations in the variant TLDs, as well as the relevant IDN tables or other reference documents used for domain name registration at the second or lower levels, should be made available to the public. A failure of consistency in these policies across registries could have disastrous effects on user expectations; see section 5.3.2 for more discussion.

Careful attention to registration policies for IDN variant TLDs is essential to minimize user confusion and opportunities for abusive registrations. It is expected that the IDN Implementation Guidelines will be followed in this regard. The Guidelines are a list of general standards for IDN registration policies and practices that are designed to minimize the risk of cybersquatting and consumer confusion, and respect the interests of local languages and character sets. Registries seeking to deploy IDNs under their agreements with ICANN have been authorized to do so on the basis of the Guidelines. The Guidelines are, as of this writing, silent on the practice of the management of variants.

5.3.4.3 Whois (Domain Name Registration Data Directory Service)

As discussed in several case study reports, there are two sets of issues related to IDN variant TLDs and Whois: the first set of Whois issues are caused by the introduction of IDNs in general, the second set of issues are caused specifically by variants. Both sets of issues need to be addressed to ensure a good and consistent user experience for querying domain name registration data.

Issues that IDN introduces to Whois services in general

As noted in several reports (Arabic, Chinese, Devanagari), the critical Whois issue facing the deployment of IDNs is the fact that the standard WHOIS protocol (as defined by RFC 3912) has not been internationalized, which means there is no standard way to indicate the character encoding in use.

The WHOIS protocol is a simple request and response transaction: a domain name is submitted to a server and output is returned. A consequence of the lack of internationalization is an increasing number of local, regional, and proprietary solutions that attempt to address the issue. This variability has a dramatically adverse effect on the user experience.

As the adoption of IDNs becomes more prevalent, Internet users will expect to be able to register domain names as well as registrant names and addresses in their native languages, using familiar scripts (character sets). This adoption is already well underway, increasing the priority to address this issue.

Issues that IDN variants introduce to Whois services specifically

Also noted in several reports (Arabic, Chinese, Devanagari), the introduction of variant TLDs causes a paradigm shift for Whois. Where currently there is a one-to-one lookup for a Whois record against a domain name, this might no longer be true in the case of variant TLDs. To illustrate this point further, consider the following domain label3.label2.label1., where each label also has p, q, and r variants respectively, thus the total number of variants for this domain is p*q*r, and each possible variant domain could have different statuses (e.g. reserved, allocated, delegated, blocked, etc.).

label31

…

label3p

label21

…

label2q

label11

…

label1r

The key issue here is to determine the correct response to a WHOIS query for a domain name lLabel3i.label2j.label1k. (request to information for label1k in the case of the root).

These issues require careful examination to determine to what extent data elements should be separated in the Whois database, and to what extent certain elements must always be subject to a joined relationship. Specific issues to be worked out include:

a. For a Whois query for a domain name with variant labels, should the variant labels be included in the response? What if the language or script of the variants cannot be understood or displayed by the user making the request? How could this be determined, since the WHOIS protocol does not have a mechanism to signal encoding?

b. If a U-label is reserved in the registry database but is not present in the DNS, should a Whois request for the domain name return a referral indicating the name is a variant of another name or return the response for the other name? or Should the response indicate that the name does not exist?

c. Is there a need for an additional query/service, which returns the Label Variant Set against a requested domain name? Should such a service also return the status of each label in the set?

d. Would the response against a blocked variant label be different from responses to labels with other status (reserved, allocated, etc.)?

e. Will the creation and expiration dates of the Variant Label Set be inherited from the fundamental label, as suggested? If yes, then if a variant label is either added or changes its state, how will this information be part of the Domain Name Registration Data? Would history be maintained and communicated for such changes?

5.3.4.4 Data Escrow

As noted by the Chinese case study team, a specific data format has been specified in the gTLD Applicant Guidebook for registries to submit the registration data to the data escrow service provider. The data escrow format currently supports variants; however modifications may be needed in light of the issues detailed in this report to support variant TLDs.

5.3.4.5 Rights Protection Mechanisms

The use of variant TLD labels (and variant domain names in these TLDs) may have an impact on existing rights protection mechanisms such as the UDRP. The UDRP is a policy for resolving disputes arising from alleged abusive registrations of domain names (for example, cybersquatting), allowing expedited administrative proceedings that a trademark rights holder initiates by filing a complaint with an approved dispute resolution service provider. In the case where domain names exist in variant TLDs, a critical issue to be resolved concerns whether all names in a variant set are considered together in the event of a UDRP proceeding, or whether and under what circumstances there could be separate consideration and determinations on some names in the set. Where certain names have different statuses, this could also have an impact on such a case. The applicable fees in UDRP proceedings might also be adjusted to take into account the existence of registrations in variant TLDs.

A policy whereby variant names are kept together in all UDRP cases may force registrants into dispute resolution proceedings as a result of registry variant practice (such as automated generation of a variant label set). However, a practice of splitting and making separate determinations within the set runs counter to the interest in maintaining a straightforward and predictable process, particularly since the existence of variant labels suggests that the contents of the set are equivalent in some way.

Note also that a UDRP complainant will present trademark or rights data that does not necessarily follow the same variant generation policies that are used for the root zone or by the registry. A complainant must prove that the domain name in question is identical or confusingly similar to a trademark or service mark in which the complainant has rights. Although a panel might make use of existing variant policy references in considering a case, it is more likely that a panel would refer to standards of trademark law for determining what is “identical or confusingly similar,” and such standards would be an important consideration relating to issues in this area.

A determination in a UDRP case may result in cancellation or transfer of a domain name, or no action if the complainant does not prevail. Issues to be resolved in this context concern whether a determination would also need to apply to the entire variant set, or whether could there be a split decision concerning the treatment of names (e.g., a decision that of the set A, B, and C, only labels A and B infringe the complainant's rights while label C does not). There are wide implications here given that a legal determination could trump the variant policy and possibly conflict with accepted reference sources on which labels are considered variant labels.

In addition, the existence of variant TLDs may have an impact on new rights protection mechanisms being instituted as part of the New gTLD Program. New gTLD registries are required to introduce certain rights protection mechanisms during their startup phases. These include a sunrise period and a trademark claims service to provide notice to a potential registrant where a domain name matches a trademark that has been recorded in the Trademark Clearinghouse. These services use a specified definition of identical match. To the extent that registrations take place in variant TLDs according to the registry policy, these rights protection processes should take into account the existence of variant labels. Consideration would need to be given to the definition of identical match and the possible incorporation of variants into the criteria for triggering a Sunrise or Trademark Claims notice.

The Uniform Rapid Suspension (URS) system is a complement to the UDRP, to be used when suspension of a domain name is the desired outcome. Accordingly, the issues discussed above also apply here.

The (Trademark) Post-Delegate Dispute Resolution Policy (PDDRP) addresses infringing uses of the TLD post-delegation. However, outcomes under this policy would take the form of remedial measures to ensure against future infringing registrations, suspension of new domain name registrations, or, in extraordinary circumstances, termination of the Registry Agreement. Since domain name registrants are not a party to the action, a recommended remedy would typically not take the form of deleting, transferring, or suspending domain name registrations. In the case of variant TLDs, the issues to be considered here would concern whether the penalties could apply to just one of the variant TLDs, or would necessarily apply to all.

5.3.4.6 Security and Stability Considerations for TLD Registries/Registrars and Service Providers

Security and stability considerations should also be relevant to registry operators, registrars and data escrow agents or other service providers. As noted in the Devanagari case study report, a suggestion for evaluating variant policies and their implementation is to log, review, and analyze DNS query traffic. Specifically, the behavior of applications and services, and sometimes the users that use them, can be inferred from traffic patterns found in sequences of DNS queries and responses. For example, registries could review DNS traffic of the TLD for queries of non-existent domains (i.e., in DNS terms reviewing the NXDOMAIN responses). An analysis of these transactions may indicate that language tables are incomplete or that variant usage is not as expected. Conversely, an analysis of the queries that indicated that certain type of variant is not

being queried (while the fundamental label or other type of variant is) could indicate a superfluous variant category.

Providing a consistent, uniform, and non-surprising (i.e., user expected) experience to the user is an essential component of stability. DNS transaction logs could provide some insight into user expectations and thus some ability to confirm that the needs of a user community are being met.

Some TLD registries may wish to consider partnerships with second-level domain holders to continue the analysis at lower levels in the DNS hierarchy.

As is clear from this section, there are a number of complex issues in the operational area that registries, registrars, and other providers should be aware of to facilitate successful operation of variant TLDs. As noted by the Latin case study team and others, the impact of variant TLDs on registries and registrars may be highly dependent upon differing implementation methods, and any proposed implementation will require broad stakeholder participation to ensure that registries and registrars provide stable, secure, consistent, and unambiguous DNS operations. This includes the greatest possible clarity in communication and understanding of variant TLDs, to limit IDN end user, registrant, registrar, and registry confusion.

Areas of application behavior, resolution and registration services, WHOIS service, and business logic all need to be examined in order to determine if these objectives are achievable.

6 Other Related Issues

A number of the case study reports described issues related to the set of variant issues, including, for example, normalization considerations, URL display, keyboard and font related issues. These issues relate to the general usability of IDN labels, and are key considerations in producing a good user experience. As they do not fall entirely into the category of variant issues, they are not discussed in depth in this report. However, a list of such issues has been compiled and is listed in <Appendix [REF]> to provide a reference for work to proceed on these issues.

7 Concluding Sections

[TBD]

Appendix 1 – Terminology

Purpose of this Document

This updated Definitions document has been developed by the ICANN IDN Variant Issues Project (VIP) team to provide a starting point for the set of definitions of the terms to be used in the final issues report. This document is prepared for discussion, and it is expected that its content will change on the basis of discussion prior to, during, and after the Dakar ICANN meeting in October, 2011. In particular, the team expects that discussion will reveal where further synthesis is possible or desirable (or both), and that the terms in need of definition will be determined in part by the shape the final report takes.

This document uses definitions from many documents that have been developed outside ICANN. The primary documents used are:

- Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework (RFC 5890)
- Terminology Used in Internationalization in the IETF (RFC 6365)
- The Unicode standard including the standard annexes
- Each of the six case study team reports (available at: http://www.icann.org/en/announcements/announcement-4-03oct11-en.htm).

Many of the terms in the script-specific sections are copied directly from their respective reports, and contain cross-references and discussion that makes sense only in the context of those reports. They are included here for convenience, and copied verbatim in order to avoid the accidental introduction of any errors in meaning.

Methodology

In this section, we describe our proposed methodology for synthesizing these definitions.

1. First, we separate terms into two categories: general terms and case study specific terms. General terms refer to terms that are generic and applicable to all cases (e.g. U-label, A-label). Case study specific terms refer to those terms that are used only in a specific case study.
2. For general terms, if it is defined in RFC 5890, RFC 6365, the Unicode Standard, we copy the relevant definitions from those RFCs and simply refer to them.
3. For case study specific terms, we copy directly from the definition sections of the case study report, if any.

4. Where terms are defined differently by different teams, the terms have been harmonized and, if need be, expanded upon to weld them into a single term useful for the unified report.

It is important to note that these terms are sometimes inspired by, but do not apply to, the specific script team reports; if those reports needed special terminology, they used it, as well as the common set of definitions originally proffered for their use. Some terms will not be useful for every script.

Format of the Definitions in this Document

In the body of this document, the source for the definition is shown in angle brackets, such as "<Unicode>", "<RFC6365>". Many definitions are shown as "<IDNVIP>", which means that the definitions were crafted originally for this document. For case study specific teams, they are shown as <Arabic|Chinese|Cyrillic|Devanagari|Greek|Latin-VIP>. Editorial notes, particularly about problematic definitions, are in square brackets ([]).

For some terms, there are commentary and examples after the definitions. In those cases, the part before the angle brackets is the definition that comes from the original source, and the part after the angle brackets is commentary that is not a definition (such as an example or further exposition).

Generic Terms

Unit (Character/Letter/Code Point…….)

Abstract Character: A unit of information used for the organization, control, or representation of textual data. <Unicode Standard, section 3.4, D7>

Assigned Code Point: A mapping from an Abstract Character to a particular Code Point in the code space. <Unicode Standard, section 2.4>

Code Point: A value in the Unicode code space. The meaning here is restricted to meaning D10 in the Unicode Standard, section 3.4. <Unicode Standard, section 3.4>

Label Valid Character: An Abstract Character which might be used to form a label; it may be any sort, including letters, digits, diacritics, and so on. These are Abstract Characters, not Unicode Code Points.

Valid Code Point: In a Language Variant Table, the list of code points that is permitted to be registered for that language. Any other code points, or any string containing them, will be rejected…. The Valid Code Point list appears as the first column of the Language Variant Table. <RFC 3743, section 2.1.12> The term Valid Code Point is deprecated in this report, because it can be confusing depending on the context. Use "Label Valid Code Point" instead.

Protocol Permitted Code Point: A code point that has either the property PROTOCOL VALID, or has the property CONTEXT RULE REQUIRED, as defined in RFC 5892. <IDNVIP> Note that not every Protocol Permitted Code Point may be used in any given zone: zone administration policy, the satisfaction of protocol-required context rules, and zone-specific context rules may all disallow code points to be registered that are otherwise Protocol Permitted.

Label Valid Code Point: The subset of Protocol Permitted Code Points listed in the Label Generation Policy of a zone, and which may be used to form a label within that zone. <IDNVIP> Note that a Label Valid Code Point may not always be valid in a given label: context rules could still rule out the Code Point at that position in a string.

Root Zone Label Valid Code Point: The subset of Label Valid Code Points that may be used to form a label inside the root zone.

Table, Repertoire, …

Language Character Repertoire: A set of Code Points identified by some identifier (such as a tag for identifying language as defined in RFC 5646). The definition of the Language Character Repertoire is ideally performed in a way appropriate to some community of language users, and might colloquially be understood as "the characters used to write a language". In most cases, most of the Code Points in a Language Character Repertoire will come from the same Script Table. <IDNVIP>

Language Variant Table: The key mechanisms of [RFC 3743] utilize a three-column table, called a Language Variant Table, for each language permitted to be registered in the zone. Those columns are known, respectively, as "Valid Code Point," "Preferred Variant," and "Character Variant," and are defined separately …. <RFC 3743> Language Variant Tables are one type of Label Generation Policy. The original RFC 3743 definition has been expanded upon considerably in various deployed systems, and not every deployed table has exactly three columns. In order to reduce confusion, this document will use Language Variant Table only to refer to such tables as conform to the specification in RFC 3743.

Script Table: A Script Table is a table of Unicode Code Points all having the same script property value. <Unicode Standard Annex #24>

Labels, TLDs, Names, Strings, ….

A-label: An ASCII-Compatible Encoding form of an IDNA-valid string. <RFC 5890> The full definition for A-label is in RFC 5890, together with the definition of U-label and some ancillary definitions. The reader is urged strongly to see that document. A-labels must be complete labels: IDNA is defined for labels, not for parts of them and not for complete domain names. By definition, every A-label begins with the IDNA ACE prefix, "xn--", followed by a string that is a valid output of the Punycode algorithm (RFC 3492) and hence a maximum of 59 ASCII characters in length. The prefix and string together must conform to all requirements for a label that can be stored in the DNS including conformance to the rules for LDH labels. Apart from all the other requirements, a string can only be an A-label if it can be decoded into a U-label

using the Punycode algorithm, which U-label can be decoded back into the same original string using the same algorithm.

U-label: An IDNA-valid string of Unicode Code Points, in Normalization Form C (NFC) and including at least one non-ASCII character, expressed in a standard Unicode Encoding Form (such as UTF-8). <RFC 5890> The full definition for U-label is in RFC 5890, together with the definition of A-label and some ancillary definitions. The reader is urged strongly to see that document. A candidate string, to be a U-label, is subject to the constraints about permitted characters that are specified in Section 4.2 of RFC 5891 and the rules in Sections 2 and 3 of RFC 5892, and the Bidi constraints in RFC 5893 if it contains any character from scripts that are written right to left. Apart from all other requirements, a string can only be a U-label if it can be decoded into an A-label using the Punycode algorithm, which A-label can be decoded back into the same original string using the same algorithm.

Domain: A domain is identified by a domain name, and consists of that part of the domain name space that is at or below the domain name which specifies the domain. <RFC 1035> The Domain Name System (DNS) name space is a tree structure, with each node and leaf on the tree corresponding to a resource set. These nodes are identified by their names, and the portion that is so named (including everything underneath) is called a “domain”. For example, the domain “example.com.” is inside the “com.” domain, which is inside the zero-length root domain (“.”). The names “two.one.example.com” and “one.example.com” are both in the “example.com” domain, with “two.one.example.com” also being inside “one.example.com”. Not every domain is a Zone.

Zone: A division of the data in the DNS, defined by the boundaries of all the cuts in the database relative to its domain. <RFC 1034, section 4.2> Note that the foregoing text is not taken verbatim from RFC 1034. In particular, RFC 1034 describes the zones only after all the cuts are made in the database; but for the practical purposes of identifying the zones for a given domain name, it is enough to know where all the parent-side and child-side zone cuts are for that name. The cuts are identified by SOA (Start of Authority) Resource Records. For a complete understanding of zones, zone cuts, classes, and domains, the reader is directed to RFC 1034.

Fundamental TLD: The Fundamental Label form of a Variant TLD Set. <IDNVIP>

Fundamental Label: The U-label used as the basis for producing the Variant Label Set. <IDNVIP> In many cases, this will be the label received as a request for allocation by the registry for the zone. In cases where the Label Generation Policy is implemented using a Language Variant Table, the Fundamental Label must be constructed entirely from Valid Code Points (and might not be the label received in a request for allocation).

Policy-Valid Label: A U-label or A-label which is valid under the Label Generation Policy for the zone. <IDNVIP> For practical purposes, the test of validity under the policy will likely be performed on the U-label form.

Internationalized Domain Label (IDL): The term “Internationalized Domain Label” or “IDL” will be used instead of the more general term “IDN” or its equivalents. This is the string of characters of the domain name being applied for and has been validated as suitable for inclusion in the DNS zone file. In the case of an IDN TLD, the IDL is simply the string of characters of the TLD being applied for and has passed the evaluation.

Variant Label: A U-label that results when a Fundamental Label is processed using the Label Generation Policy. <IDNVIP> For the purposes of implementation, this operation needs to be reversible; if the resulting Fundamental Label is ambiguous (that is, more than one Fundamental Label results), the handling of such a result needs to be covered by the Label Generation Policy.

Activated Variant Label: A Variant Label that is activated by a registry.

Allocated Variant Label: A Variant Label that is allocated by a registry.

Blocked Variant Label: A Variant Label that is blocked by a registry (to avoid a conflict) and is not allowed to be allocated.

Reserved Variant Label: A Variant Label that is reserved by a registry without allocation but which may be allocated on request.

Reserved Name: A name set aside for a potential allocation to a particular registrant (or TLD registry in the case of TLDs in the root). The name is not allocated, but could be if/when certain conditions are met.

Sets, Subsets, ….

Variant Character Set : The set of code points consisting of a Valid Code Point and all of its variants.

Variant Label Set : A set of U-labels consisting of one Fundamental Label and its zero or more Variant Labels.

IDL Package: An “IDL Package” is a collection of IDLs as determined by the guidelines in RFC 3743. All labels in the package are "reserved", meaning they cannot be registered by anyone other than the holder of the Package. These reserved IDLs may be "activated", meaning they are actually entered into a zone file as a "Zone Variant". The IDL Package also contains the language tag. The IDL and its variant labels form a single, atomic unit, however, not all labels in the package are active.

Activated Variant Label Subset: The subset of Variant Label Set that is activated, or alternatively, the set containing the Fundamental Label and all its Activated Variants.

Allocated Variant Label Subset: The subset of Variant Label Set that is allocated, or alternatively, the set containing the Fundamental Label and all its Allocated Variants.

Reserved Variant Label Subset: The subset of Variant Label Set that is reserved, or alternatively, the set containing all the Reserved Variants of a Fundamental Label (and the Fundamental Label, if it is not activated).

Blocked Variant Label Subset: The subset of Variant Label Set that is blocked, or alternatively, the set containing all the Blocked Variants of a Fundamental Label.

Policies

Label Generation Policy : A formal specification that can be used to formulate or validate a label and determine whether two labels can be considered distinct for allocation. If two labels are not considered distinct for allocation, as per the policy, they are referred to as variants of each other. Variants are symmetric in Arabic script.

TLD Label Generation Policy: A formal specification that can be used to formulate or validate a TLD label and determine whether two TLD labels can be considered distinct for allocation.

Arabic Script Label Generation Policy: Policy specified to generate labels for Arabic script. For Arabic script, this would include at least a list of Protocol Valid Code Points allowed in forming labels, their Character Variants, additional Label formation constraints/rules and meta information (e.g. including script, owner, version, date, etc.).

Arabic Script TLD Label Generation Policy: Policy specified to generate TLD labels for Arabic script.

Domain Name Blocking Policy: Refers to a policy that has effect of certain domain names in a TLD registry becoming unavailable for allocation (for example, due to implementation of variant-related policies).

Actions/Processes

Allocation: In a DNS context, the first step on the way to Delegation. A registry (the parent side) is managing a zone. The registry makes an administrative association between a string and some entity that requests the string, making the string a (candidate) label inside the zone, and a candidate for delegation. Allocation does not affect the DNS itself at all. <IDNVIP>

Delegation: In a DNS context, the act of entering parent-side NS (nameserver) records in a zone, thereby creating a subordinate namespace with its own SOA (start of authority) record. See RFC 1034 for detailed discussion of how the DNS name space is broken up into zones. <IDNVIP>

Activation: The process of making a domain name resolvable.

Reservation: In Arabic Script IDN variants context, this is the process of having an unallocated variant label which relates to a Fundamental label that is allocated.

Blocking: In Arabic Script IDN variants context, this is the process of having a variant label not allowed for allocation to anyone as long as its Fundamental label is allocated.

Name Aliasing: The abstract concept of two or more domain names “behaving as one” by Policy or technical means. This concept has still unresolved issues, definitional, technical and political. Currently this concept is technically served by the use of CNAME and DNAME records in a zone file, allowing for

the aliasing of a domain name or a DNS tree to another. With the introduction of the concept of variants, arguments have been raised for the necessity of the presentation of a mechanism that would allow the users to keep in sync domain name trees but without the limitations of DNAME and CNAME. Experts have discussed the use of a CNAME+DNAME kind of record but without concluding on the advantages and disadvantages such a record would present. Issues with the use of DNSSEC, and the inability to address what "behaving as one domain name" stands for in the real world have stalled the discussions on this issue.

Bundling Domain Names: The registry policy of registering certain domain names as a set, depending on select characteristics (e.g. homograph domain names). In some cases, bundling is used for name aliasing purposes.

Domain Name Bundling: Registration technique that makes multiple domain names share all registration parameters (such as creation/expiration date, associated name servers etc.) except the domain name itself. Changes to any of these registration parameters should normally take effect on all the domain names in a bundle.

Variants Definitions

Arabic Script Character Variant: A Label Valid Character which is replaceable with another Label Valid Character within a label, as defined by a Label Generation Policy. The relationship is symmetric in Arabic script.

Character Variant: In a Language Variant Table, a "Character Variant" is an entry on the second list of code points corresponding to each Valid Code Point and providing possible substitutions for it. Unlike the Preferred Variants, substitutions based on Character Variants are normally reserved but not actually registered (or "activated"). Character Variants appear in column 3 of the Language Variant Table. The term "Code Point Variant" is used interchangeably with this term.

Preferred Variant: In a Language Variant Table, a "Preferred Variant" is an entry on the list of code points corresponding to each Valid Code Point and providing possible substitutions for it. These substitutions are "preferred" in the sense that the variant labels generated using them are normally registered in the zone file, or "activated." The Preferred Code Points appear in column two of the Language Variant Table. "Preferred Code Point" is used interchangeably with this term.

Zone Variant: A "Zone Variant" is either a Preferred or Character Variant Label that is actually to be entered (registered) into the DNS, that is, into the zone file for the relevant zone. Zone Variants are also referred to as Zone Variant Labels, active labels, or Activated Labels.

Alternate Names: Two names are alternates of one another just in case, for a namespace starting with one, the namespace starting with the other is isomorphic to the first, subject to the usual DNS loose consistency strictures. In the current DNS, there are 2 different techniques for this. The first is aliasing: CNAME, DNAME, and other such techniques redirect a name or a tree, effectively substituting one label for another during DNS lookup. The second is by using provisioning constraints, such that an underlying

provisioning system always effects a change in all of the alternate names whenever that change is effected in one of the alternates. A fuller discussion of this topic is included for information in Appendix B.

Aliased name: A domain name that has been aliased with one or more names under the concept of Name Aliasing. The technical solution that is currently available for aliasing a domain name to another is the use of a CNAME or a DNAME record in a zone file, essentially mapping a domain name or a DNS tree to another. Voices for other technical solutions have been raised over the last few years but without any results.

Homograph: A word that shares the same written form with a word of the same or different script but may have different meanings. The same could apply for any string of letters, even if not a word. The characters that make this possible between scripts are mentioned as homograph characters in the document (e.g. the Latin A and the Greek A).

Homophone: Two different words that are pronounced the same. The same could apply for any string of letters, even if not a word.

[The term "homograph" has a well-established definition as being a property of words within a language. It does not extend across language/script boundaries. Code points in different language/script that are represented using the same glyph are "homoglyphs" and, as this term is language/script agnostic, it can also be applied to two elements of the same language/script. Two strings that appear to be identical but in fact contain one or more homoglyphs are "homoglyphic strings". If they are not exactly the same, it is correct to speak of their homoglyphic distance. This in turn provides a potential metric for similarity.]

Combining Marks: A commonly used synonym for combining character; a character with the General Category of Combining Mark (M). [source: Unicode]

Composite-character variants: Abstract Characters that do not have a single assigned code point assigned, but can be represented by multiple code points.

Font: A collection of glyphs used for the visual depiction of character data. A font is often associated with a set of parameters (for example, size, posture, weight, and serifness), which, when set to particular values, generate a collection of imagable glyphs. <UNICODE>

Glyph: (1) An abstract form that represents one or more glyph images. (2) A synonym for glyph image. In displaying Unicode character data, one or more glyphs may be selected to depict a particular character. These glyphs are selected by a rendering engine during composition and layout processing. <UNICODE>

Glyph Image: The actual concrete image of a glyph representation having been rasterized or otherwise imaged onto some display surface. <UNICODE>

Writing style: Conventions of writing the same script in different styles. Different communities using the script may find text in different writing styles difficult to read and possibly unintelligible. For example, the Perso-Arabic Nastaliq writing style and the Arabic Naskh writing style both use the Arabic script but have very different renderings and are not mutually comprehensible. Writing styles may have significant impact on internationalization; for example, the Nastaliq writing style requires significantly more line height than Naskh writing style. <RFC6365>

IDNA Symmetry Constraint: A-label/U-label transformation must be symmetric: an A-label A1 must be capable of being produced by conversion from a U-label U1, and that U-label U1 must be capable of being produced by conversion from A-label A1. <RFC 5890>

Language Tag: Language tags, as defined in RFC 5646, are used to help identify languages, whether spoken, written, signed, or otherwise signaled, for the purpose of communication. This includes constructed and artificial languages but excludes languages not intended primarily for human communication, such as programming languages.

Case Specific Terms

Arabic

Arabic Letter: are used to write Arabic script based languages and used to write words.

Non-Joining Characters: Those characters that do not connect to letters before or after them; e.g. U+0621 LETTER HAMZA, U+0674 HIGH HAMZA, and U+200C ZWNJ.

Right-Joining Characters: Those characters that connect to the letter before them; e.g. all letters based on ALEF, REH, DAL, and WAW, and a few other letters.

Dual-Joining Characters: Those characters that connect to the letters before and after them; e.g. all other Arabic letters than those listed above.

Join-Causing Characters: Those characters that connect to the letters before and after them, but do not change shape themselves; i.e. only U+200D ZWJ and U+0640 TATWEEL. With respect to those categories, Arabic Script Letters could be defined as follows:

Non-Joining Letters: The group of non-joining characters which are letters (by Unicode's definition); i.e. U+0621 LETTER HAMZA and U+0674 HIGH HAMZA.

Joining Letter: The union of Right-Joining Letters and Dual-Joining Letters which cursively join with letters following them.

Right-Joining Letters: The group of right-joining characters which are letters; i.e. all letters based on ALEF, REH, DAL, and WAW, and a few other letters.

Dual-Joining Letters: The group of dual-joining characters which are letters; i.e. all other Arabic letters.

Arabic-Indic Digits: Forms of decimal digits commonly used along with Arabic script and comprised of two sets of digits. The set <U+0660-9> is commonly used in Arabic-speaking world, while the set <U+06F0-9>, often referred to as Eastern Arabic-Indic, is used in Iran and Pakistan. Although European digits (1, 2, 3,…) derive historically from these forms, they are visually distinct and are coded separately. (Arabic-Indic digits are sometimes called Indic numerals; however, this nomenclature leads to confusion with the digits currently used with the scripts of India.)

Arabic Digits: The term "Arabic digits" may mean either the digits in the Arabic script (see above and Arabic-Indic digits) or the ordinary ASCII digits. When the term "Arabic digits" is used in Unicode specifications, it means Arabic-Indic digits.

Arabic Ligature: A single glyph representing a combination of one or more Arabic Letters. This means that the isolated form of a character can be considered a ligature. It's worth noting that this is different from the Unicode use of the term.

Form of a Letter - Arabic Script: A Letter in Arabic Script can occur in up to four different forms within a ligature. These include the following:

Isolated form:

It is the standalone form of a Letter, i.e. when the letter does not join with any other letter, forming a single letter.

Initial form:

It is the form of a right-joining-letter when it occurs in the beginning of a ligature, joined with at least one more letter after it, to form a ligature.

Medial form:

It is the form of a right-joining-letter when it occurs in the middle of a ligature, joined with at least one letter on either side, to form a ligature.

Final form:

It is the form of a joining-letter when it occurs at the end of a ligature, joined with at least one more character before it, to form a ligature.

Chinese

Han Script Variant: Characters with different visual forms but with the same pronunciations and with the same meanings as the corresponding official forms in the given language contexts. For more details please refer to the section 5 of this report.

Chinese Official Form: In different country/region, the government specifies “official forms” for a set of general use Hanzi. In Mainland China, they are called normalized Hanzi (规范字 U+89C4 U+8303 U+5B57), and in Taiwan, they are called orthographic Hanzi (正體字 U+6B63 U+9AD4 U+5B57).

CJK Characters: “CJK characters” are characters commonly used in the Chinese, Japanese, or Korean languages, including but not limited to those defined in the Unicode Standard as ASCII (U+0020 to U+007F), Han ideographs (U+3400 to U+9FAF and U+20000 to U+2A6DF and U+2A700 to U+2B73F and U+2B740 to U+2B81F), Bopomofo (U+3100 to U+312F and U+31A0 to U+31BF), Kana (U+3040 to U+30FF), Jamo (U+1100 to 11FF and U+3130 to U+318F), Hangul (U+AC00 to U+D7AF and U+3130 to U+318F), Kangxi Radicals(U+2F00 to U+2FDF), CJK Radicals Supplement (U+2E80 to U+2EFF), and the respective compatibility forms.

CJK Unified Ideograph: An ideograph is a graphic symbol that represents an idea. Chinese Hanzi, Japanese Kanji and Korean Hanja are often referred to as ideographs. Since 1990, tens of thousands of Chinese Hanzi, Japanese Kanji and Korean Hanja have been merged into CJK Unified Ideographs and their Extension in ISO/IEC 10646 and Unicode. In this document, if not otherwise specified, the term “ideograph” means a CJK Unified Ideograph.

Cyrillic

Greek

Greeklish (Source: Wikipedia, http://en.wikipedia.org/wiki/Greeklish, text in Italics): a portmanteau of the words Greek and English, also known as Grenglish, Latinoellinika/Λατινοελληνικά or ASCII Greek, is the Greek language written using the Latin alphabet. Unlike standardized systems of Romanization of Greek, as used internationally for purposes such as rendering Greek proper names or place names, or for bibliographic purposes, the term Greeklish mainly refers to informal, ad-hoc practices of writing Greek text in environments where the use of the Greek alphabet is technically impossible or cumbersome, especially in electronic media. Greeklish was commonly used on the Internet when Greek people communicate by forum, e-mail, IRC, instant messaging and occasionally on SMS, mainly because older operating systems didn't have the ability to write in Greek, or in a Unicode form like UTF-8. Nowadays most Greek-related content appears in native Greek.

Tonos: Greek accent mark, acute accent (Greek Tonos, U+0384)

Dialytika (diaeresis): Greek accent mark (appears on the letters “ϊ” (e.g. Greek small letter iota with dialytika, U+03CA) and “ü” (e.g. Greek small letter upsilon with dialytika, U+03CB) to show that a pair of vowel letters is pronounced separately, rather than as a diphthong – see http://en.wikipedia.org/wiki/Diphthong). It can also be combined with tonos over the same letters, Greek small letter iota with dialytika and tonos, U+0390 and Greek small letter upsilon with dialytika and tonos, U+03B0.

Katharevousa: (Greek: Καθαρεύουσα, [kaθaˈrevusa], lit. "puristic [language]"), is a form of the Greek language conceived in the early 19th century as a compromise between Ancient Greek and the Modern

Greek of the time, with a vocabulary largely based on ancient forms, but a much-simplified grammar. Originally, it was widely used both for literary and official purposes, though seldom in daily language. In the 20th century, it was increasingly used for official and formal purposes, until Dimotiki became the official language of Greece in 1976 (Source: Wikipedia, http://en.wikipedia.org/wiki/Katharevousa).

Dimotiki: (Greek: δημοτική [γλώσσα] [ðimotiˈci], "[language] of the people") is the modern vernacular form of the Greek language. The term has been in use since 1818. Demotiki refers particularly to the form of the language that evolved naturally from ancient Greek, in opposition to the artificially archaic Katharevousa, which was the official standard until 1976. The two complemented each other in a typical example of diglossia until the resolution of the Greek language question in favour of Demotiki (Source: Wikipedia, http://en.wikipedia.org/wiki/Dimotiki).