Procedure to Develop and Maintain the Label Generation Rules for the Root Zone in Respect of IDNA Labels

Version 2012-11-22b

Table of Contents

Abstract

This document provides a procedure for establishing some of the label generation rules for the root zone. The label generation rules (or what is sometimes, in a DNS protocol context, called "policy" as distinct from "protocol") govern the way a zone is operated. The procedure below provides a mechanism for creating and maintaining the rules with respect to IDN labels for the root. This mechanism can be used to determine which Unicode code points are permitted for use in U-labels in the root zone, what variants (if any) are possible to allocate in the root zone, and what variants (if any) are automatically blocked.

The procedure uses two classes of panel to make the determinations. The design is extremely conservative: the panels are arranged so that the natural and automatic answer is to refuse to include a code point or a variant rule. This is in keeping with principles the Internet Architecture Board has proposed for operation of zones, as well as ICANN's general responsibilities for the security and stability of the root. It is also designed to restrict additions to the root zone to those writing systems where there is a clear interest, possibly to the disadvantage of some language communities (particularly those that are endangered). At the same time, the procedure is intended to permit movement on some kinds of labels without waiting for every language community in the world to be ready.

A. Introduction, Motivations, and Background

A.1. Purpose of this section

This document defines procedures for creating and maintaining part of the label generation rules for the root zone. The resulting label generation rules will provide a consistent and predictable set of permissible code points for IDN TLDs and provide a way to determine whether there are variant labels (and if so, what they are). For the purposes of this document, the label generation rules contain four parts: the rules governing the permissibility of Unicode code points (the repertoire), any exchangeable code point variants that follow from those (the variant rules), the status of any resulting label, and a set of optional whole-label evaluation rules that determine whether the output of the previous three portions is still an acceptable label in the root zone. This document defines solely the *procedures*, and not the label generation rules themselves. This section presents some important background and some motivations for the procedure that is proposed later in the document.

A.2. Conventions and background

There is a bibliographic list in section F. We put references in square brackets, using what we hope is a meaningful short name for the item. When the same reference is used in running text, we use the short name without the square brackets. We refer to publications in the Request for Comments series by their RFC number, even if they are part of some other series (BCP, STD, and so on). Unicode documents are referred to following the Unicode convention, where the number sign is included in running text but not when used in a reference (so, "UTR#36", but "[UTR36]").

The following are prerequisites for understanding this document:

- "A Study of Issues Related to the Management of IDN Variant TLDs (Integrated Issues Report)" [IIR];
- IDNA2008 [RFC5890] [RFC5891] [RFC5892] [RFC5893] [RFC5894] [RFC5895];
- Unicode [Unicode62];
- "Principles for Unicode Code Point Inclusion in Labels in the DNS." [IABCP].

In addition, the terms defined in Appendix 2 of IIR are incorporated here by reference, and not reproduced. *Some of the terms defined in that Appendix 2 are used in a special or peculiar way*, and the text below is unlikely to be understood completely without having that terminology to hand. (We have not followed the capitalization convention of IIR, because some readers found it confusing, but the terms are otherwise the same.)

The draft sometimes includes discussion of things like "all of Unicode". That phrase is really an abbreviation for "all of the parts of Unicode that are somehow permitted under IDNA2008." Some parts of Unicode are already not permitted by IDNA2008. Anything that is not permitted by the protocol at all is automatically removed from consideration.

A.3. Label generation rules

Every zone on the Internet has, either implicitly or explicitly, a set of rules governing the labels allowed in that zone. Sometimes, these are implicit and trivial, such as, "I only permit labels that have something to do with my company," or, "We name all our hosts after moons of Jupiter." Sometimes, they are more involved: many TLDs have exclusion lists of labels that are not permitted, or other restrictions, such as allowable code points in a label. In the root zone today, two-character ASCII labels are either withheld or else allocated to qualified entities (on the basis of ISO 3166-1). We call all such rules the label generation rules, or LGR in short. (In this, we are following the terminology of IIR. These might better be called label determination or, at least in part, allocation rules. None of the terms are entirely satisfactory.)

This document provides a mechanism to generate some of those rules: the ones necessary for long-term operation of both IDNs and IDN variants in the root zone.

In parts of this document, we use "LGR" in a more informal sense, to refer to some subset of the total set of LGR. In addition, we sometimes use "LGR" to refer to *proposed* rules, which might be in contention and might not ultimately become part of the root LGR. The key thing these have in common is that they are all rules intended to govern which labels are permissible in a zone (in this case, the root zone).

A.3.1. Some starting premises

We start with the premise that it is beneficial for the Internet community to permit some labels conforming to IDNA2008 in the root zone. We acknowledge that not everyone agrees with this premise, but observe that the root zone already contains such labels, and conclude therefore that label generation rules are needed to govern such additions to the root zone. Moreover, the addition of U-labels and A-labels to the root is in keeping with DNS names being useful mnemonics: it is very hard to remember a name written using characters one does not normally use. The basic, positive good that ought to come from a new set of label generation rules is a basic repertoire of (assigned) Unicode code points that can be helpful in building usefully-mnemonic labels in the root zone. This does not mean that every word, name or string – or even most such strings – from a language will be eligible under the new label generation rules; only that useful mnemonics can be represented. There is no intent for the procedure and the rules it produces to support fully (or fail to support fully) any particular living language community.

The "Integrated Issues Report" [IIR] argues that, because the root zone is necessarily shared by everyone on the Internet, it is a special case, and needs a set of label generation rules that ensures minimal conflict, minimal risk to all users (independent of which language or script they are using), and minimal potential for incompatible change over time. These conclusions are in keeping with ICANN's commitments in respect of operational stability, reliability, resiliency, security, and global interoperability of the DNS

A.3.2. Variants

With the addition of U-labels and A-labels to the root zone, the question of IDN variants seems inevitably to follow. An IDN variant, as understood here, is an alternate code point (or sequence of code points) that could be substituted for a code point (or sequence of code points) in a candidate label to create a variant label that is considered the "same" in some measure by a given community of Internet users.

As noted in IIR (footnote 26, p 41), IDN tables (as described in RFC3743 and generalized in RFC4290) are implementations of label generation rules; they include two separable but linked components. These components are the code point repertoire for the zone (or zone repertoire), and the code point variant rules. Those two components are part of the label generation rules.

In addition, each variant label must be associated with a disposition that determines what can happen to a label at registration time, even if the disposition is usually implicit.

It is worth observing that variants are, at bottom, an attempt to make an exact-match system (the DNS) resemble natural language to some degree. This attempt comes not only with benefits (reduced user disappointment or confusion when it works), but also costs (including maintenance costs, difficulties arising from failure cases, and possibly the costs of unexpected traffic). See IIR for some discussion of these issues.

A.3.3. Characteristics of the Process Goals

The procedure outlined here is offered with certain goals.. Because of the interest in IDNA labels for the root, the intent of the process is to move as expeditiously as possible within the bounds of the safety and security for the root zone and in keeping with expert assessment.

Apart from the standard goals that ICANN has with any change to the root zone, the following four main characteristics apply:

Utility: Even though it is not a goal to support anything that can be written as a word, or name, in a particular language as a label, the mnemonics allowed by the label generation rules should have a certain utility to them. This would not be satisfied by very arbitrary restrictions of the repertoire or the code point variant rules.

Coverage: The coverage of the repertoire should be comprehensive, so as to not exclude a certain script community.

Non-arbitrariness: Provisions in the LGR should not be arbitrary, in the sense that security concerns are evaluated more tightly for one community over another.

Absence of bias: The procedures and their results should not be biased, in the sense that they only care about communities based on some criteria such as overall size, representation by a national government or similar factors.

These are characteristics of the *goals* for the development of the procedures, and not necessarily of the procedures themselves. It may be that no procedure meets every one of these goals separately, and that any given language community may be affected negatively in order to achieve a satisfactory result overall.

A.3.4. Precedents

Even where there are precedents based on existing TLDs, the procedure in this document establishes a new playing field. While existing labels will almost certainly have to be grandfathered even if they are in conflict with the label generation rules established by this procedure, that precedent and conflict is not a reason to invalidate any aspect of the new rules or this procedure.

A.3.5. Principles to constrain the label generation rules

IIR's argument is consistent with views expressed in an Internet Architecture Board Internet Draft [IABCP]. IABCP lays out several principles that are useful guides for (or perhaps constraints on) developing the label generation rules for any zone, including the root zone. The principles relevant to the root zone are paraphrased below, but the reader should consult IABCP for a full discussion. Note that, for the purposes of this document, a code point in the zone repertoire is necessarily an assigned code point.

The principles are not rules; they inform, but cannot substitute for the judgment of those making decisions in the label generation rules development process.; they are intended to be weighed in the context of the project goal, premises and characteristics outlined above.

Longevity Principle: A Code Point in the Zone Repertoire should have stable properties across multiple versions of Unicode.

For characters recently added to Unicode, there is a risk that as use of that character grows, refinements may need to be made in certain aspect of its recommended use. This principle intends to minimize that risk.

Usability Principle: A Code Point in the Zone Repertoire should not present recognition difficulties to the zone's intended user population and should not lend itself to malicious use.

Contrary to the generally understood meaning of this term, this principle addresses only the risk to usability of a code point due to abuse or recognition difficulties.

Andrew Sullivan 12-11-22 19:36
Comment [1]: Note: IABCB is probably about to ditch this term and this Pple, in favour of two new ones: Least Astonishment and Contextual Safety. Just a warning for now – change not pub'd yet

Inclusion Principle: The zone repertoire is built up by specific inclusion; the default status for any code point is that it is excluded.

Simplicity Principle: Overly complex rules are to be avoided, in favor of rules easily understood by users with only some background. In particular, in the root, rules should not require deep familiarity with a particular script or language.

An overall familiarity with basic concepts of Unicode, for example, would be acceptable background; rules based on such concepts, applied universally across scripts would not necessarily violate that principle.

Predictability Principle: People with reasonable knowledge of the topic should by and large reach the same conclusions about which code points should be included.

Stability Principle: Once a code point is permitted, it is almost impossible to stop permitting it: the act of permitting a code point cannot be undone. This is particularly true once a label containing this code point has been registered.

This principle addresses the risk introduced by the "sticky" nature of the label generation rules and expresses the risk it introduces. Choices that are least likely to be overturned by later review or future additions are preferable to those that may be overturned.

Letter Principle: Only Assigned Code Points normally used to write words should be permitted. Assigned Code Points normally used for both words and other purposes should not be permitted.

The letter principle intends to restrict the repertoire to the space needed for mnemonics, without relying on the Unicode letter property, which is drawn both too narrowly and too widely. The principle does not state that all letters must be permitted.

Conservatism Principle: Any doubt should be resolved in favor of exclusion of a code point rather than inclusion.

There is a sense in which the conservatism principle is an overriding principle. It requires that all decisions to include a code point or rule be based on strong agreement and high confidence.

All of these principles are not only applicable to the selection of repertoire, but, excepting perhaps the Letter Principle, are easily applied to code point variant rules as well; they are thus of prime relevance to the project's work.

There is a certain amount of tension between even similar principles. For example, reviewing a recently added code point would be at odds with the longevity principle but, by being able to evaluate it in the full context, might prevent the addition of a rule that would be in conflict if the code point was added in the future, thus satisfying the stability principle.

IABCP makes plain that it is talking about principles, and not algorithms. If there were an algorithm (or indeed any automatable system) for determining the label generation rules, it is reasonable to suppose the IAB would recommend using it. We must therefore conclude that, whatever the procedure to create and maintain the rules is, it will involve humans and judgment. Once created, the LGR will constitute the algorithm to be used to determine the eligibility of any label for inclusion in the root and to identify the variants of an IDN TLD label.

A.3.6. Evaluation Parameters

IIR outlined (pp 42-43) three independent parameters that could be used in evaluating the proposed process for label generation rules. They are comprehensiveness, expertise, and qualification. In the discussion arriving at the proposed process outlined below, we have identified a fourth parameter, centralization.

Section E, Evaluation of this Proposal Against the "Parameters" describes these four parameters in detail and uses them to evaluate the proposed process.

B. A development methodology

In this section, we outline a methodology for building and maintaining the root zone label generation rules relevant to IDNs in the root[1]. The label generation rules consist of both the zone repertoire and code point variant rules. The methodology recognizes that, while these two elements are logically separate, in practical terms any variants need to be considered at the same time as the repertoire.

We believe that core elements of the development methodology are also applicable to a later, perhaps less intensive maintenance phase, because of the basic similarity of the mechanics of both development and maintenance.

[1] Much of this document ignores the distinction, but strictly speaking the label generation rules developed under this procedure apply only to IDN labels. See discussion at the beginning of Section A.3.

B.1. Overview

A repertoire that satisfies the Longevity, Usability, Predictability, and Stability Principles cannot be built *ad hoc*. In order to conform to the Usability Principle, it will be necessary to bring to bear expertise in the writing systems appropriate to a range of code points. But, given that the user population for the root zone comprises everyone on the Internet and given the need to satisfy the Stability Principle, the final repertoire must have a unity that will not come from considering subsets of Unicode independently. We therefore envision a two-pass process that will allow the necessary reconciliation and integration.

There are three basic elements to the final output of the procedure. The first is a list of code points that are permitted in the root zone under a particular named repertoire. The second is a list of variants, if any, for each such code point, regardless of the named repertoire(s) in which the code point appears. The third is , for each such variant relationship under a given named repertoire, the resulting disposition of labels under the variant rule. These elements are discussed in detail in the sections below.

As a general rule, in what follows we discuss only rules for code points. We note here that any place where a single code point is appropriate, a structured series of code points might be appropriate instead. A structured series of code points is one that, always and without any exception, may be bound together for the purposes of variant relationships. These structured series of code points are to variants what precomposed and decomposed code points are in Unicode (non-canonical) normalization forms. The point of including these structured series is to allow for certain variant relationships that might exist as a result of complex writing systems. Generation panels should seek to minimize such relationships wherever possible, and the integration panel is expected to scrutinize them very carefully.

B.1.1. How the output of this procedure is to be consumed

The operation of this procedure yields all the label generation rules for the root zone that apply to IDN labels. Any application for an IDN label in the root zone is evaluated using the label generation rules. This latter step is fully automatic, so the label generation rules that are the product of the procedure in this document must always yield a single answer for any (entire) candidate label (see section B.1.2.1 for more on this)

B.1.2. Output

The output of the process will be a unified set of label generation rules that specifies the following:

1. The complete list of code points permissible in U-labels in the root zone (the zone repertoire);
2. The complete code point variant rules (if any) for each code point. For any code point in the repertoire, this element of the label generation rules gives a complete list of all the other code points that are variants for the first..
3. The disposition of the labels resulting from the application of the rules in (2). For a given code point in the repertoire, this element specifies whether a resulting variant label is blocked or allocatable, depending on how the variant label was generated. The same code point under different tags (see next) might generate an allocatable variant label in one case, and a blocked variant label in another.
4. Assignment of one or more "tags" to each code point in the repertoire, and to each disposition of a variant in (3). For the same label, different dispositions may exist for the same variant label, as long as they do not share common tag values.
5. A final, optional, whole-label evaluation rule that determines whether the original, applied for candidate label as well as each of its variants is permitted in the root zone.

Different scripts require different treatment, so the label generation rules can generate labels and handle variants differently based on script. The mechanism to accomplish this differential treatment is the tag assigned in item (4); these are described further in section B.3.1. Note, however, the code point variant rules (in 2, above) are always exactly the same for any given code point. This way, code points are always related to one another in a predictable way. For the same reason, if a code point is associated with some whole-label evaluation rules, those whole-label evaluation rules must be the same no matter what the tag.

All code points sharing a specific tag will implicitly form a subset of the repertoire, not necessarily disjoint of all other such subsets. A given label must have only code points with the same tag. The disposition of any variant labels arising from those code points is also determined by the tag. Specific examples are found in the appendices.

B.1.2.1 Operating the resulting label generation rules

Because the output of the process is a set of label generation rules that are supposed to be applied as part of an automatic process, it is necessary that the label generation rules be operable by way of a set of deterministic functions. For clarity, we identify five logical operations, which could be combined into a single, larger function as a matter of implementation:

1. A function that operates on the LGR and takes a candidate TLD label and a tag as input and determines unambiguously whether the candidate label fits the repertoire subset associated with the tag.
2. A function that operates on the LGR and takes a candidate TLD label as input and determines unambiguously all the variants (if any) of that candidate label;
3. A function that operates on the LGR and takes a candidate TLD label as input and determines whether there is a match / conflict between it and any variant of any allocated TLD label, or of any currently requested TLD label;
4. A function that operates on the LGR and takes a candidate label and a tag as input and determines unambiguously which of the variants of that label must be blocked and which may be allocated in the root zone.
5. A function that operates on the LGR – specifically, the final whole-label rule portion – and takes a candidate label as input and determines unambiguously whether any given candidate label (including output of the foregoing steps) is permitted in the root zone.

If the Label Generation Rules created under the procedure put forth here can be expressed in terms of this set of functions, then they satisfy the necessary conditions for automatically applying the label generation rules for the root zone. If they cannot, then they are inadequate to the LGR purposes.

B.1.3. Two-pass process

The procedure for developing IDN TLD variant label generation rules consists of two passes. The first pass creates a set of label generation rules specific to a given script, writing system, language, or all of these; this task is carried out by Rule Generation Panels composed of people with deep experience or interest in the script, writing system or language used by some community of Internet users.

These panels will submit their proposals for the LGR to an ICANN Variant TLD Integration Panel consisting of independent experts in DNS, Unicode and scripts that has responsibility for the second pass. This second pass involves integrating the proposals into a single unified LGR for the root zone, taking into account the need for a secure, stable and reliable DNS root zone.

These two kinds of panels are assisted by advisors, who observe the activities of any or all active panels; who provide comment and advice on the technical matters of IDNA, Unicode, DNS, linguistics, or other matters; but who do not otherwise have a formal role in making a decision.

In what follows, we call the first kind of panel the “generation panel ”, and the second one the “integration panel.” Because there are many scripts and writing

systems, there will be multiple generation panels, but because there is only one root zone, there will only be a single integration panel.

The (proposed) LGRs generated by the generation panel are submitted to the integration panel for review, and, if approved, for integration into the LGR for the root zone.

It should be emphasized that generation panels generate *proposed* rules; there is no presupposition that any proposal will be accepted for integration. Any generation panel proposal for the LGR that cannot be accommodated by the integration panel will be returned to the originating panel for additional work. This dialogue and interaction will continue until the integration panel is satisfied that the proposal can be accommodated in the LGR. It is possible for the integration panel never to be satisfied.

Among the additional responsibilities of the integration panel is the definition of the maximal starting repertoire and the default whole label evaluation rules for the root zone. That provides a common starting point for the generation panels and eliminates unnecessary duplication of work.

The integration panel reviews the output of all available generation panels, taking that output as proposals for parts of the final set of all label generation rules. From time to time, the integration panel will release a version of LGR to ICANN and to the Internet community to enable the evaluation, processing and potential deployment of IDN TLD applications. The integration panel must produce, each time it completes a round of work, a single, complete, unified, and internally consistent set of rules – the unified LGR.

The LGR will be built up over time, beginning with the work of an initial set of panels that have ready expertise and experience to develop a proposal for the LGR for their scripts. Additional generation panels will be established over time and, as their proposals are completed and agreed, the integration panel will add those proposals to the unified LGR.

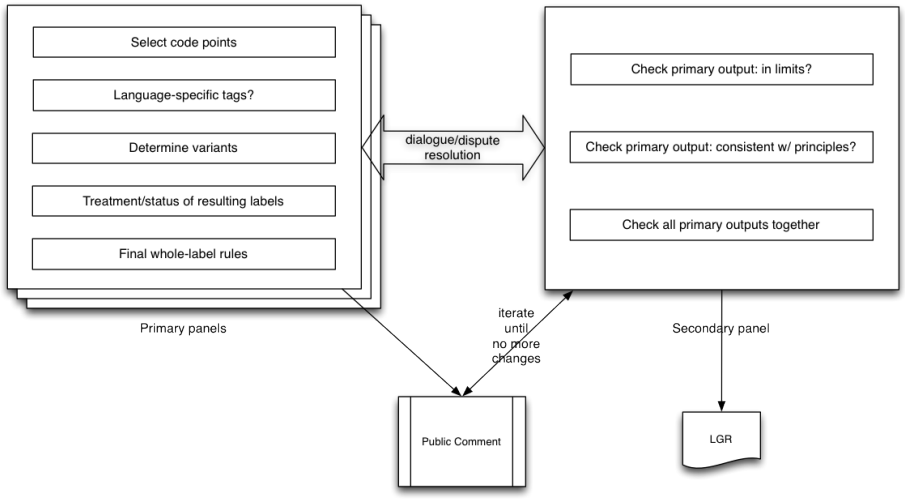
Based on the principles outlined above, the integration panel will take a conservative approach to the evaluation and integration of the LGR proposals from the generation panels, and to the release of the various versions of the unified LGR.

The two-panel approach depends on the idea that the two panels have different purposes. The generation panel is expected to be specialist. Its members or advisers need to have deep and wide expertise in the script(s) it is considering. While the integration panel also includes specialists in Unicode and writing systems, they are to be linguistic generalists: arguments that depend on the integration panel understanding particular points about the language or script in question will

normally be an indication that a generation panel proposal is insufficiently generic for use in the root zone. It is the task of the integration panel to minimize the risk to the root zone and Internet from any generation panel proposal and to ensure that these different proposals can be made into a consistent whole. In order to treat matters as generically as possible, the descriptions that follow do not make reference to specific examples. Neither does this section provide specific advice to panels. Some possible scenarios are discussed in detail in the appendices.

The diagram below is a high-level description of the process.

Asmusf 12-10-17 07:42
Comment [2]: Needs updating the diagram



B.2. Establishment, Composition,and Scope of Panels

B.2.1. Generation panel

Each generation panel works on a subset of Unicode relevant to one writing system[2] or a set of related writing systems. This work is broadly aligned along the script property of the Unicode code points in question, though it need not be restricted to a single script (see Section B.5.3).

B.2.1.1 Establishment

There are no formal criteria that automatically lead to establishing a generation panel. The actual request for use of a code point in the root zone (i.e. an containing

[2] http://unicode.org/glossary/#writing_system

that code point) is sufficient (but not necessary) to indicate a generation panel may be needed. However, other expressions of interest may be considered by ICANN as constituting grounds to establish a generation panel, not excluding explicit request to form a generation panel by members of a community of Internet users.

In response to such expressions of interest, or for any other reason (such as a suggestion by the integration panel), ICANN staff may establish a new generation panel. ICANN staff, in consultation with the users of the writing system or a set of related writing systems, and advised by the integration panel and other expert advisors, defines the initial scope of the generation panel and makes a public call for participation in the work of the panel. ICANN appoints a chair and may also appoint further members to the panel. With advice from the integration panel, ICANN staff will review whether the proposed generation panel, its composition and its scope meet the other requirements set forth here.

Generation panels may also be re-convened at some future time if necessitated by the need to maintain or update the LGR.

B.2.1.2 Scope

The scope of a generation panel is broadly aligned along the Unicode script property of the code points in question, though it need not always be restricted to a single script (see Section B.5.3.1). When generation panels are established, they are chartered to cover some section of the Unicode character repertoire, corresponding to the usual way of writing some language or group of languages. Normally, the section of the Unicode character repertoire corresponds to the Unicode script property of the code points, with a few inclusions of code points having the Inherited or Common property, as defined in [UAX24].

Not all scripts are eligible to form the scope of a generation panel. In particular, for the root zone, any language or script that is not in active, everyday use by a living community of speakers is barred from consideration, even if some parties ask that it be supported. This is in keeping with the Conservatism Principle, because it discourages more doubtful cases and only supports the real population of zone users (i.e. the population of Internet users).

B.2.1.3 Expertise

All generation panels should have significant expertise in the writing systems concerned, but need have neither overall expertise in all of Unicode, nor expertise in any other writing system. Panels may be made mostly of volunteers interested in that portion of the potential repertoire.

B.2.1.4 Diversity

Generation panels need to have some diversity of participation in order to be useful. They must have sufficient numbers of participants so as to ensure that the work of the panel is not all essentially that of a single person. In addition, participation should be diverse in sponsorship: a panel of 10 people all employed by the same organization is insufficiently diverse to qualify. At the same time, it should be borne in mind that the work of these panels is concerned with technical issues and involves linguistic expertise, and is not "representative" in nature.

If it is impossible to get sufficient numbers of participants with sufficient diversity and expertise, that is evidence that the code points in question and the rules associated with them are too specialized to be included in the root zone, and so the generation panel will not be established.

The diversity of expertise required depends to some degree on the nature of the repertoire to be considered by the panel. For certain simple scripts, those lacking a large repertoire or other complicating factors, a minimum panel size of three might be sufficient. For repertoires encompassing many languages or presenting technical or linguistic challenges, additional panel members or advisers would be needed.

B.2.1.5 Outreach

Because one of the goals of this procedure is not to be arbitrary, it is desirable to encourage participation from parties who might normally be outside the usual ICANN community of interests, or who might have special expertise in some script or its use with some language. Such outreach efforts should be organized *ad hoc* in ways appropriate to the script and relevant languages under consideration during the development and founding of a generation panel. It is not possible to specify outreach procedures in general, because different languages and scripts will demand different treatment. Measures may include public calls for participation, outreach via ICANN's GAC, and direct contact of known experts in the field. This outreach, however, is not intended to generate evidence of a significant community of interest where none exists. Outreach efforts must be proportionate not only to evidence of a real user community on the Internet, but also to the potential Conservatism Principle consequences of failing to consult some interested parties.

B.2.1.6 Work plan

In order to avoid later doubt arising from insufficient diversity and expertise, at the time a generation panel is set up, it produces a short, rough work plan for the portions of Unicode that the panel expects to deal with. This includes the list of languages for each script that the panel will consider. The integration panel reviews that work plan and the résumés of the generation panel members. The integration

panel may make recommendations for additional participation, or additional or reduced scope of the generation panel's work. The recommendations are advice, not requirements, although they are an indication that the integration panel may later use its reservations about the participation in the generation panel as one of the reasons for the integration panel's doubts in any LGR proposal generated by that panel.

B.2.1.7 Sub-panels

Generation panels, particularly those concerned with scripts used for a wide variety of languages, might prefer to organize their work into sub-panels. Such additional organization of work is beyond the scope of this document, and we consider such a decision to be entirely a management decision on the part of the generation panels (so it is hereby neither forbidden nor encouraged). A generation panel that attempted to organize its work this way might find it desirable to echo the relationship between the generation panels and the integration panel in the relationship between the sub-panels and the generation panel. Subpanels may include participants from outside the generation panel itself (for instance, writing system experts with particular expertise for some languages; these participants all act formally as advisors.

B.2.1.8 Resolving disputes inside generation panels

A generation panel is expected to work on a set of code points, which might be used by several different languages. It is also expected to examine as many languages as practical in generating proposed rules pertaining to that set of code points. The review of a shared script by various language communities may result in disputes across languages. These disputes constrain the ability of a generation panel to propose rules regarding the script to the integration panel. In cases where the chair of the generation panel is unable to resolve a dispute, advisors can be called upon or assigned to assist in facilitating the dispute resolution process where appropriate. In keeping with the ability of any panel to call upon advisors as it sees fit (see Section B.2.3), the generation panel members can decide by consensus to call on any neutral third party that they trust to work as an advisor and facilitate the resolution of the dispute.

As a general rule, how a generation panel determines the output it will send to the integration panel is a management decision on the part of the generation panels, and is beyond the scope of this document. It is worth noting, however, that individual members of the generation panel who are dissatisfied with the recommendations sent to the integration panel will, like everyone else, have the opportunity to make their views known during the public comment period on the generation panel output. Since the integration panel is required to take such comments into account in its evaluation, as a practical matter the generation panel

will need to take into consideration strong and principled objections from any generation panel member. This does not mean that true unanimity in the generation panel is always required.

B.2.1.9 Initial and ongoing generation panels

A good guide for creating the initial set of generation panels would be to use the U-labels either that are already in the root zone or for which there is a pending application. The mere existence of an application (or even a delegation) using a code point is not sufficient reason to allow the code point in question; but it is positive evidence of some level of desire to use this code point, and by extension, to use the script to which this code point belongs.

Generation panels are chartered to work at "natural boundaries" for their task. Generation panels shall not be chartered in such a way as just to permit a small number of already requested Assigned Code Points. Instead, generation panels work on a writing system's alphabet, or all the code points from a single script where that script is used by several different languages, and so on. The exact boundaries of the scope of each panel cannot be stated in abstract (since cases are sensitive to the vagaries of writing systems). An initial list of the recommended generation panels is in section **Error! Reference source not found.**

Among the initial generation panels a special panel of experts may be chartered to deal with the "easy cases": single, well-defined scripts that are used for exactly one language. For a proposed scope for such a panel, see section C.2.1.6.

Once the procedure has produced the first root label generation rules, it is used for future iterations of the rules. New generation panels may be chartered whenever there is reasonable evidence of interest on the part of some language or writing system community, subject to the diversity requirements outlined in Section B.2.1.1.

In case there is sufficient interest in an as yet unconsidered (or excluded) writing that uses an already examined script, the generation panel for that script would be asked to reconvene. In establishing generation panels, broad outreach is to be greatly desired; the more information about languages—particularly but not only languages with large populations—available to the generation panels, the better they will be able to evaluate the repertoire relevant to their scripts in terms of inclusion and exclusion of specific characters on the base of established use in a modern writing system.

B.2.2. Integration panel

The second pass of the two-pass process is carried out by a unified expert panel. This panel reviews all available first-pass output and recommends a resulting set of

Andrew Sullivan 12-11-23 11:47
Deleted: C.1.1

Andrew Sullivan 12-11-23 11:47
Deleted: C.1.1.6

IDN label generation rules for the root zone. Because this integration panel reviews the composition and work plan of the generation panels at time of the latter's creation, the integration panel should be created first. The integration panel is also tasked with establishing the overall maximal repertoire and default whole label variant evaluation rules, which serve as starting point for the generation panels.

B.2.2.1 Composition

The integration panel should consist entirely of experts selected by ICANN on the basis of established expertise, whether consultants, staff or both. They are to be duly qualified against ethical conflicts, in order to minimize the possibility, or any appearance, that the process might be captured by interested parties. They are to be in a contractual relation with ICANN, so that ICANN may require, as part of its contract with the panelists, impartial and unbiased evaluation. The panel requires at least one expert in Unicode issues, at least one expert in IDNA and DNS issues (or one for each), and at least one expert in linguistics and writing systems (who could be the same as the first expert, but need not and often will not be).

The integration panel is a technical working panel, not a representative or deliberative body. It should be kept small in the interest of efficiency, but should not contain fewer then three members at any time.

No current or past member of any generation panel may be a member of the integration panel.

B.2.3. Advisors

Any panel, at its discretion, may call on advisors. Advisors have particular expertise in issues relevant to the questions at hand, and are there to provide observations given that expertise. Advisors may be paid consultants, ICANN staff, or volunteers.

Because of the specialized nature of their required expertise, we anticipate that ICANN would seek out qualified advisors to assist the generation panels in their work. The advisors have no formal role in the decisions of the panels, though their expert opinions may influence the panelists. Anyone taking an advisory rule is automatically excluded from voting membership in any primary or integration panel, while actively acting as advisers to any other panel. It seems likely that many ICANN staff, which will need to participate in the work of both the primary and integration panels, will fall into the category of "advisor".

B.3. Variant rule generation procedure

The basic job of the generation panel is to produce one or more proposed lists of code points to be included in the zone repertoire, and any associated code point

variant rules for those code points. Normally, a generation panel will produce one such list, but in some circumstances (such as where a single script is used as part of two different writing systems) it might need to provide more than one.

Dennis Jennings 12-11-5 17:12
Comment [3]: There needs to be a forward reference here to those circumstances

Andrew Sullivan 12-11-21 15:55
Comment [4]: Does the additional "such as" suffice?

Part of the work of the generation panel is to determine which of the relevant code points to include in the root zone repertoire, which resulting code point variant rules are necessary in order to provide useful mnemonics in the Root Zone, and which code points to exclude because they violate one or more of the Principles set forth in section A.3.5. In performing its task, the generation panel should examine as many languages as practical, especially those with larger populations. If variants are deemed necessary, the generation panel will need to provide the requisite dispositions for each variant. Finally, the panel needs to consider whether any additional whole-label evaluation rules need to be proposed.

The generation panel will provide its output in a single, complete format that is specified by the integration panel and adopted by the ICANN community, and which allows for efficient review and integration (possibly through the use of tools). The output must further satisfy the "logical" requirements stipulated in in section B.1.2, and be accompanied by detailed rationale and background for the decisions made in creating the various elements of the output.

B.3.1. Naming the rule subsets: tags

By necessity, the generation panels will consider a subset of Unicode., These subsets will not necessarily be fully disjoint, not least because some Unicode characters are shared between scripts and writing systems. At the same time, for linguistic reasons, it may be necessary to give different disposition to variant labels depending on script or language. For this reason, the output of the generation panel is associated with a descriptive identifier, called a tag. (In some cases, a generation panel might be responsible for the creation of output for more than one linguistic context, and each would get a separate tag.)

These tags are used at the time of application for a U-label in the root zone, in order to identify the relevant portion of the repertoire to consult. Every code point in the applied-for U-label must be included in the portion of the repertoire named by the tag, or else the application is invalid. The set of variant labels that arises from the applied-for U-label is calculated based on the code points in the application, but the resulting disposition of each such label is determined by the variant rules the tag names. As already noted, in most cases, repertoire subsets consist almost entirely of code points belonging to one script.

The requirement that all code points in a candidate U-Label fit into the repertoire corresponding to the tag submitted with the applications allows the process to

administer restrictions, such as forcing labels not to be made of code points with different script properties, unless that is specifically allowed under a given tag.

The following describes the detail of how to create tags based on script or language names.

B.3.1.1 Structure of tags

The "tags" are *language tags* as described in RFC 5646, with an extension to identify the tag as applying to the root DNS zone. They are composed of the following parts:

1. A language subtag, as specified in the language subtag registry maintained by IANA and created by RFC 5646;
2. A script subtag, using the appropriate ISO 15924 identifier;
3. A singleton subtag, to be assigned by IANA – in this document, it is styled TBD, but it will be a single lower-case ASCII letter when it is assigned;
4. The extension subtag "root", to indicate that this named set of label generation rules is for the root zone.

Because DNS labels are mnemonics, they are not necessarily in any given language. Instead, in the normal case, they are confined to a single script. For that reason the language subtag, used as the first of the four parts described above, is always "und", meaning "undetermined language". The second element in these tags is the script subtag, which is the ISO 15924 identifier appropriate to the script.

Note that ISO 15924 supports a script tag ("Japn") for the mixture of the Han, Hiragana and Katakana scripts used for the purpose of writing Japanese.

In cases where named repertoires overlap for any reason, the code point variant rules must generate the very same list of variants for each code point, independent of the nature of the tag or the linguistic environment of the applicant.

However, the operation of any given code point variant rule may not always produce a variant label with the same status, because the disposition for the same variant may be different for each tag.

It has not been possible to find a documented requirement that the LGR ever need to depend on the intended user population's language, and therefore this procedure does not envision a tag starting with a language subtag other than "und". The purpose of the tags is to identify the individual *script* repertoires as subsets of the repertoire finally accepted for the root. The fact that the tags happen to be defined as *language tags* in this document is entirely due to the convenience of using RFC 5646 for this purpose. Additional discussion of this matter is in Appendix F.

B.3.2. Unicode Script Mixing

The mechanism of using named repertoires with corresponding rule set allows in principle for the creation of mixed script repertoires; such repertoires would permit a single label that contains code points with different Unicode script properties. At the same time, it assures that any script mixing has to be explicitly allowed for, by creating a specific named repertoire that contains only the particular code points whose scripts may be mixed, and only for applications that use the corresponding tag. Thus, the procedure ensures the spirit of the Inclusion principle is applied – any mixing that is to be allowed has to be deliberately included via a named repertoire.

It is anticipated that, based on the Principles, script mixing would be normally be restricted by the integration panel, rather than allowed to be applied widely. Mixing of characters with the COMMON or INHERITED script property would probably have to be allowed in the appropriate contexts, as well as mixing of the Hiragana and Katakana syllabaries with Han for Japanese. Beyond that, we anticipate that such mixing would mostly be restricted, because of the systemic risks it presents. Detailed rules for what types of script mixing should be permissible are outside the scope of this document, which is properly concerned only with defining a suitable procedure within which that determination can be made.

B.3.3. Final whole-label evaluation rules

There are some sequences of code points that can be typed, but which are structurally ill-formed. This phenomenon is particularly important in complex writing systems. For such cases, the generation panel may propose a set of final whole-string evaluation rules. The primary purpose of such rules is to avoid labels that could give rise to corrupted or undefined display when rendered in a user agent. They are not intended to eliminate all possible "mistakes" that could be made under the rules for a given script or language.

Asmusf 12-10-17 07:42
Comment [5]: I believe it is necessary to limit the intent here.

Andrew Sullivan 12-11-21 15:59
Comment [6]: Why does this need scare quotes. Also, is the section not adequately limited in scope?

When the label generation rules are used, these whole-label evaluation rules are applied after all the other steps, to test all the resulting variant labels as well as the original, applied-for candidate label. If any label does not meet the tests in the whole-label evaluation rules, that label is automatically blocked notwithstanding any other result from the label generation rules.

A sensible place to start in building the final whole-label evaluation rules is the Unicode Standard Annex #29, "Unicode Text Segmentation" [UAX29]. UAX#29 develops the notion of extended grapheme clusters, and a useful generic whole-label evaluation rule might be a requirement that labels only ever be made of integral Extended Grapheme Clusters. A refinement of these rules might be required, depending on the script in question; such determinations should be made by the generation panel and confirmed by the integration panel.

With this approach, the whole-label evaluation rules resulting from this process will be based on a compatible extension of the extended grapheme clusters defined in UAX#29. They will be implemented by a refinement of the classification of code points carried out there, and the optional addition of rules governing the interaction of classes of code points in forming clusters. There will be only a single classification of code points for the Root Zone and a single set of such clustering rules.

Generation panels may propose refinements of classification for code points in any of the named repertoire subset and for the clustering rules governing them. In most cases, the integration panel can merge these in a straightforward way. Overly or unnecessarily complex refinements would be rejected, as they would violate the Simplicity or Predictability principles.

In cases of conflict between different whole-label evaluation rules proposed by the generation panels, it is up to the integration panel to work with the affected generation panels to achieve a resolution. If no agreement can be reached, the proposed refinements will not be added.

B.3.4. Tasks in rule generation

B.3.4.1 Defining the Repertoire

The generation panel's starting point is the repertoire of Unicode characters needed for the writing systems in question, already reduced based on IDNA2008 and the principles in IABCP as well as reduced by code points defined as restricted for identifiers, as specified in Table 1 of UTS#39.

This starting repertoire consists of all the assigned code points likely to be used and is usually made up of code points with the same Unicode script property, plus relevant code points with the Inherited or Common properties. This is the maximal repertoire. The choice of script dictates the value of the tag associated with the repertoire.

The panel's second task will be to select from the maximal repertoire a unified set of code points covering the alphabet repertoires of the languages used by or known to the panelists, since these will obviously be of interest to them. Typically there is a base set shared by most languages, together with local extensions for specific languages.

The third task will be to look at those code points from the maximal repertoire that are not indicated in the second task. The panel must exclude any code points used only for archaic or historical purposes (for example in medieval manuscripts). It must further exclude all code points used as special phonetic or other notational characters unless they are in current use in a natural orthography. (For example,

there is considerable overlap between the International Phonetic Alphabet and orthographies in Africa and the Americas.)

To complete the third task, the panel shall next consider each of the remaining code points and establish whether other writing systems can be identified authoritatively as containing these code points, making them eligible for inclusion into the repertoire. Any character that cannot be authoritatively established as being used for everyday writing in a living language will be excluded. For this determination the panel may rely on outside expert knowledge; but it is the responsibility of a generation panel to come to a definite conclusion whether it is both safe and useful to select a particular code point as candidate for the repertoire. In case of uncertainty or doubt about the expertise or the authoritativeness of the information available, the panel must exclude the character as candidate.

B.3.4.2 Creating the Variant Rules

Having proceeded through these tasks and determined a repertoire, the panel considers each included code point in turn, and determines whether there are any code point variant rules for the code point. The variant rules from the generation panel fall into three categories:

1. Code point substitutions, 1:1;
2. Code point substitutions 1:many or many:1;
3. Code point substitutions of one series of code points for another series of code points, but only in cases where the writing system needs it due to features of that writing system. This category explicitly excludes any case where the many:many relationship cannot be treated as completely determined.

Any rule that depends on context will require very strong evidence that it is in fact required to write any useful mnemonics for some language users; the generation panel shall proceed on the presumption that a code point or variant that requires context rules is likely to violate the Simplicity, Conservatism, and Usability principles. In any case, code points permitted by IDNA2008 under the CONTEXTO and CONTEXTJ rules are automatically excluded.

Part of the specification of the code point variant rules involves specifying the resulting disposition of variant code points, in order to determine whether a code point substitution results in an allocatable or blocked variant. It is possible that the only difference between two tagged portions of the repertoire involving some or all of the same code points will be the resulting disposition of the code points. See section B.3.1 for more discussion.

As already noted, for each combination of code point and variant, a different disposition may be provided; otherwise the dispositions are equal for all variations of each code point. When evaluating a label, the disposition of a variant label is the disposition of the least permissive variant contained within. In order of increasing permissiveness, the dispositions are "blocked" and "allocatable." For example, a variant label with at least one code point variant with the disposition "blocked" is therefore "blocked".

Andrew Sullivan 12-10-25 17:39
Comment [7]: I haven't a clue what this sentence means. Can we delete it?

B.3.4.3 Whole label evaluation

After generating the variant label rules, the generation panel will define any final whole-label evaluation rules necessary for the named repertoire.

As discussed in B.3.3, these rules are required to be representable by a refinement of the general mechanism in UAX#29, tailored for this purpose by the integration panel as a starting point for all panels. Therefore, any such additional final whole label evaluation rules would need to be expressed as further refinement of the character classification and interaction rules in the scheme of UAX#29 and transmitted in that form as part of the proposal submitted to the integration panel for integration.

B.3.4.4 Submitting proposals

The generation panel, when it has completed its work, sends its proposals to the integration panel. At the same time, the generation panel's proposals are posted for public comment using the prevailing ICANN public comment procedures of the day. This permits those who did not participate in the generation panel in question to make their views known to the integration panel.

B.4. Label generation rule integration procedure

The integration panel reviews the output of every generation panel from which a proposal is available at the time the integration panel begins review. The integration panel evaluates each proposal. The integration panel first confirms that the proposal stays within the maximal repertoire defined as the starting point by the integration panel, and that it conforms to the other requirements for output set forth in this document.

It then evaluates the proposal for consistency with the Principles and for the risk it presents, in the context of the entire starting repertoire of Unicode code points. Proposals that do not meet the principles or create unacceptable systemic risk are

rejected. A generation panel's proposal may be to permanently exclude a code point, which would factor in this evaluation.

The integration panel further ensures all proposals together form an internally coherent set of label generation rules, and rejects any proposals that are in conflict. It is at this stage that the integration panel reviews the whole-label evaluation rules, both in the context of the given generation panel proposal, as well as in the context of all such proposals. Based on the Simplicity and Predictability Principles, the integration panel ensures that structurally comparable writing systems have consistent whole-label evaluation rules.

Finally, it reviews whether the proposals collectively meet any other requirements (symmetry etc.) set forth in this document. Proposals that fail this review will be rejected.

A rejection of a proposal must be accompanied by a justification citing the specific issues with the proposal that gave rise to the rejection. The justification must include an accounting of which (if any) of the Principles the proposal violates according to the integration panel, as well as every other justification the integration panel has for its decision. Every rejection by the integration panel must be available on the Internet; a web-accessible archive of the decisions is sufficient to meet this requirement.

Because the integration panel is required to make its determination unanimously, a single objector in disagreement with the rest of the integration panel will prevent the integration panel from accepting a generation panel's proposal. In case of such disagreement, the integration panel's reasons for rejection should include an indication that the panel is divided, it must include the reasoning behind the objection, and should include the details of the disagreement within the panel if there.

Unlike the generation panels, the integration panel will consider possible interactions with Unicode characters outside the proposed set of rules and, if necessary, reject the generation panel's proposal based on such issues. The integration panel includes in its evaluation possible sets of rules that are not yet proposed, as might happen when a writing system did not initially attract a generation panel but might be expected to attract one in future. Decisions by the integration panel are required to be unanimous; any proposal that does not attract unanimous acceptance is automatically not accepted. The integration panel must take into account any public comments submitted in response to the posting of the generation panel's output.

While there may be a "first mover advantage" to the establishment of generation panels, the order in which the integration panel would evaluate proposals would be

based on the order in which they are received, but nevertheless constrained by the need to evaluate all proposals in context of each other, and compared to the rest of the basic constrained repertoire.

B.4.1. Transitivity and symmetry of rules

In order to meet the Simplicity Principle, code point variant rules need to be symmetric and transitive. That is, if the code point or series of code points V_1 has a variant rule that produces the code point or series of code points V_2, then in the label generation rules V_2 also has a variant rule that produces V_1. Further, if V_2 has a variant rule that produces the code point or series of code points V_3, then V_1 must also have a variant rule that produces V_3. This requirement may on occasion produce labels that would violate the linguistic criteria for being considered true variants, and may also result in the generation of extra blocked variants that lead to the exclusion of other possibly useful labels. It is nevertheless appropriate in the root zone, where the goal is not to maximize the number of possible labels but to minimize the confusion possible in a shared environment supporting heterogeneous linguistic communities.

The integration panel review verifies that these conditions are met not only within a given generation panel proposal, but across the entire root zone. Where necessary, the integration panel creates additional variant rules to make the entire set transitive and symmetric. The disposition for any such additional variants would be "blocked". If such additional variant rules would fall entirely *within* a named repertoire, the corresponding generation panel proposal is rejected. (The generation panel would then reissue the proposal with the required rules added, but with dispositions as chosen by the generation panel).

Note that symmetry and transitivity *does not* mean that every variant in a named repertoire must also appear as a candidate code point in that named repertoire: if V_1 is (say) a Japanese character and it includes V_2 as a variant, that does not mean that V_2 need be part of the Japanese repertoire. It merely means that if V_2 is part of *any* named repertoire, it must also generate V_1 as a variant. The variant rules themselves are symmetric and transitive; the resulting dispositions need not be.

B.4.2. Conflicts with the generation panel

In case the integration panel rejects a proposal by a generation panel, the panels involved must negotiate agreement. The panels must reach consensus on any label generation rule for it to be included. This is true even for the resulting treatment, such that the integration panel might agree that the code point in question should be included in the zone repertoire, but that the resulting treatment of other code points

should be different (e.g. blocked instead of allocatable). In any case where the panels cannot agree, the result is always to reject the assigned code point in the zone repertoire, or to reject the code point variant rule. This is in keeping with the Conservatism Principle. If the generation panel is unwilling to make that modification, the entire generation panel's proposal is rejected.

If the integration panel and a generation panel disagree about an element of a proposal, they are considered in conflict, and the element of that proposal is considered in dispute. The panels are expected to find a way to resolve such conflicts and to avoid them in the first place by informal discussions in advance of final submission of a proposal.

B.4.3. Communication between panels

The panels may discuss points of disagreement (or probable disagreement) at any time, as formally or informally as they see fit, provided that all such communications are treated as being publicly available. A useful mechanism might be a mailing list for the integration panel, with a public archive. Interaction between the secondary and generation panels should be as formal as necessary for productive work, but need not be more formal. Our intention is that the final label generation rules be clearly the product of collaboration among diverse communities (and members of those communities), rather than being the product of the integration panel alone. Any decisions and recommendations by the various panels are public.

Panels for related or structurally similar scripts are encouraged to communicate or cooperate in the interest of arriving at a more consistent treatment of repertoires and variants for the root zone.

B.4.4. Decision of the integration panel is atomic

The integration panel's evaluation of a given generation panel's output is atomic: either it accepts a proposal completely, or it rejects it completely. That is, the integration panel is free to reject a generation panel's recommendations, but it is not free to amend the details of the generation panel's proposal. It may, however, return a generation panel's proposal with a suggestion about what would change the opinion of the integration panel. Most importantly, the integration panel is required to provide detailed reasoning for its rejection in every case. The integration panel's decision is public; it cannot be appealed, but the generation panel is free to submit the proposal again (altered or unaltered).

B.4.5. Label generation rule recommendations and public comment

The integration panel creates a set of recommended label generation rules that includes the union of all the approved proposals from the generation panels.

When the integration panel has created such a set, it is posted for public comment using the prevailing ICANN procedures. If any of the proposals from a generation panel are under dispute, they are to be excluded from public comment. Instead, the recommendations must include a note indicating the outstanding dispute and the measures being undertaken to resolve it.

During the public comment period, a conflict may be resolved, or the integration panel may approve additional generation panel proposals. It is important to allow these to proceed without undue delays. An announcement will be made that the dispute is resolved or additions to the recommendations are now pending, and a supplement detailing the specific changes or additions is published according to mechanisms then available in the public comment process. (As of this writing, the only such mechanism appears to be to add the proposed supplement to the public comments themselves.) The public comment period is allowed to proceed as originally scheduled, but a new public comment is required on the entire, unified final proposal.

At the end of the public comment period, the integration panel receives and reviews the public comment. If as a result it makes alterations (including those resulting from changes it initiated during public comment), the output is treated as a new integration panel output. When the integration panel makes no more alterations due to public comment, the resulting label generation rules become the new label generation rules for the root zone.

Where a conflict was not resolved during the public comment period, the affected rules are not part of the recommendations.

B.5. Starting points for the panels

B.5.1. Panels start with the latest version of Unicode

It is possible that, at the time the work begins, there will be available a version of Unicode that has not yet been evaluated for use with IDNA2008. Panels should start with the latest version of Unicode anyway. This is consistent with the Longevity Principle: the ultimate label generation rules should be stable for the new version of Unicode too, and the properties of any assigned code point must be stable as compared to the previous Unicode version, or else the code point in question violates the Stability Principle. Assigned code points new to Unicode, however, and those that have had altered properties in the latest version, will be left out of the repertoire in any case, both because of the Usability and Conservatism Principles and because they do not yet appear in RFC5892

In assessing the stability of a character's identity and usage, a recent change in its Unicode character properties can be an indicator of lack of stability. However, Unicode defines over one hundred character properties. Some are normative, some

informative and some are provisional. A change in some property assignments, such as Script_Extensions, would normally be less of an indicator about a change in a character's identity or use, but rather the result of details of its usage having become better known over time. It is therefore incumbent on the panels to use judgment in evaluating the stability of characters.

To ensure that all generation panels start off with characters that meet certain minimal requirements for consideration as part of this process, the integration panel will initiate the process by defining the set of code points that it considers to fulfill certain minimal criteria to be eligible as part of the root repertoire, pending further review during the two-stage process. The final repertoire is then expected to be a subset of this initial set. Generation panels and the integration panel may refuse to include any of these code points, and because of the Inclusion principle, only those code points that some generation panel requests out of this overall repertoire would actually be eligible for addition to the final repertoire.

B.5.2. Relationship to existing IDN tables

A brief survey of the set of IDN tables registered with IANA establishes that some of them are registered for scripts that would not satisfy the criteria for support in the root zone laid out in this document. While each table relates to a code point repertoire for some zone, their existence alone is inadequate rationale for including such repertoire in the root. This is because of the Usability Principle, and the fact that the root zone's user population is the entire Internet population. Assumptions about user populations that might be appropriate for particular zones (especially those of ccTLDs) can be mistaken in the context of every writing system or language on the planet.

Andrew Sullivan 12-10-25 18:15
Comment [8]: I don't understand the question.

At the same time, the existing repertoires may be useful starting points in two ways. First, if an assigned code point is available in any repertoire, that may constitute supportive, but not conclusive evidence of a need for that assigned code point. Second, if an assigned code point is available in several repertoires, it may be evidence of need for that assigned code point, or evidence of contentious use of the assigned code point, or both.

Generation panels in particular would be expected to use the relevant tables as a contributing source of information in forming conclusions about the repertoire.

B.5.3. Relationship to Unicode properties

B.5.3.1 Script and Script_Extension

Every assigned Unicode code point has exactly one script property (see [UAX24]), and it may be tempting to use that property directly to restrict the assigned code points available to be considered by any generation panel, and to regulate what

other assigned code points are permitted in any variant rule. Unfortunately, such an approach suffers from four defects:

1. Many (perhaps most) languages use assigned code points from more than one script, particularly if the Inherited and Common scripts are considered independently.
2. Conversely, if Common and Inherited are simply included in every other script, the category is too broad. There are, for example, assigned code points in the Common script that are not used with Latin, but U+002D HYPHEN MINUS is in the Common script.
3. Many languages use assigned code points from the Latin script, particularly in a computing context, even though they are normally written using a different script (the mixed use of Latin together with another script is currently not anticipated by this procedure, however)
4. Even if a generation panel is restricted to a single script, the integration panel may need to consider cross-script cases (e.g. U+0061 LATIN SMALL LETTER A vs. U+03B1 GREEK SMALL LETTER ALPHA vs. U+0430 CYRILLIC SMALL LETTER A). While resolving string-confusability issues is beyond the scope of this project, the integration panel will need to take into consideration the consequences of the label generation rules for the Usability and Conservatism Principles.

Accordingly, it is not possible *a priori* to limit a generation panel to only one script property, and it is even less desirable so to limit the integration panel.

The recent versions of Unicode contain the Script_Extensions property. One of the things it is intended to do is to narrow the perhaps overly broad Common and Inherited scripts properties. It appears that Script_Extensions will be a useful tool with which to restrict the scope of work for generation panels.

B.5.3.2 Maximal set of code points

As outlined in Section B.3.4.1, before any generation panel starts work, the maximal set of code points that it may consider must be determeined. The code points in this set will need to meet the following conditions:

- Assigned in the latest version of the Unicode Standard
- Explicitly included by IDNA2008 as PVALID
- Not restricted for identifiers in Table 1 of UTS#39
- Not used for writing an excluded script

Excluded scripts are those that, according to the integration panel, do not have a living language community. That pre-empts code points from those scripts from being included in the zone repertoire. In making its decision, the integration panel

must consider evidence of actual speakers and writers of a language as part of its evidence. Scripts that are excluded on the basis of no living language community do not get considered when establishing generation panels, and therefore do not get reviewed. This does not mean that the integration panel has to be unduly conservative in ruling out a script. In fact, the Conservatism Principle demands that any script for which there is doubt on whether it used for a living language community ought to be excluded.

Writing systems used by very small numbers of people ("endangered languages") do not meet this test. Writing systems used by now extinct languages (e.g. Linear B) are excluded. See section C.2.1.1 for our recommendations of initially excluded scripts.

Andrew Sullivan 12-11-23 11:47
Deleted: C.1.1.1

Note, there are cases, even in historically recent times, where a language community has chosen to switch to a different script. Therefore, any exclusion of a script from consideration cannot be guaranteed to be permanent. The integration panel must be able to respond to the emergence of new user communities or new writing systems for new communities and determine whether allowing additional scripts is warranted and safe at a future time.

In section 3.1, Unicode Technical Standard#39 "Unicode Security Mechanisms" [UTS39] includes a mechanism for evaluating Assigned Code Points to determine whether they are appropriate for use in identifiers. This determination is based in part on whether a code point is part of a script not used for writing a living language, or a script that is of limited use, or otherwise not yet widely used, as defined in UAX#31 "Unicode Identifier and Pattern Syntax", Tables 4 through 7 [UAX31]. A listing of the code points thus restricted can be found at http://www.unicode.org/Public/security/latest/xidmodifications.txt.

As result of evaluation using that mechanism, some of the scripts identified in UAX#31 might turn out to be eligible for the root after all. This would be an area for judgment by the integration panel.

Generation panels must not include in their proposed repertoires any assigned code point that is not included in the maximal repertoire defined by the integration panel.

B.5.4. Distinguishing among states resulting from variants

IIR explores at length the difference between variants that are intended to go into the zone, and variants that, because of the existence of some other label, must not go into the zone. See Section 5 of IIR. We accept those states as a foundation for the present work.

The integration panel may deliver a rule that has one of two results: potential allocation ("allocatable") or blocking ("blocked").

B.5.4.1 Potential allocation

A potential allocation rule says that once the variant label is generated, that variant label may be allocated to the applicant for the original label. The allocated label is then subject to any activation restrictions that might be appropriate under prevailing ICANN policies or agreements (or both); but in principle, any label that is allocated may be delegated, according to the wishes of the party to whom the label is allocated.

B.5.4.2 Blocking

A blocking rule says that a particular label must not be allocated to anyone under any circumstances. We may distinguish between two types of blocking. The first is simply a consequence of the non-inclusion of an assigned code point in the zone repertoire. By definition, any U-label that contains a code point not in the zone repertoire is blocked. This could change if a subsequent repertoire expands to include the formerly excluded code point. The second type of blocking is an explicit decision to block the resulting label. A change to such a rule would require study, could only be undertaken case by case, and would suggest serious problems with this procedure in light of the Conservatism Principle.

B.5.5. Default whole label evaluation rules

The integration panel will specify the whole label evaluation rules that, by default, apply to the entire root zone repertoire. The starting point for the integration panel will be the rule that all labels must consist of an integral number of Extended Grapheme Clusters as defined in UAX#29. See section B.3.4.3.

Whatever the integration panel adopts will become the starting point for the generation panels. They may propose additional refinements.

Andrew Sullivan 12-11-22 21:49
Comment [9]: I'm concerned that I don't fully get how the whole label evaluation rules work.

B.6. Panels, Conservatism, and the Limits of Knowledge

In all matters the integration panel's judgment is to be governed by the Conservatism and Stability Principles. If the integration panel delivers a repertoire while there is still work to be done on other parts of Unicode, as is inevitable, we expect subsequent iterations of the repertoire. We expect those iterations to increase the size of the repertoire, *and not* to remove any code point or change any code point variant rules to reflect the new inclusions.

If an iteration of the process causes a subsequent repertoire to remove a code point that was in an earlier repertoire or to change an existing variant rule, all operation of the procedure must halt. A review of the process must ensue to determine whether it is effective at following the principles outlined in Section A.3, and whether it is possible (and how) to add additional checks to the procedure to avoid recurrence of similar failures. Because it is impossible to state in advance what the failure might be (since if we knew, we could write rules to avoid it), the ICANN Board will determine the nature and scope of the review, and will appoint the reviewers. If such halts are called frequently, that is a reason to believe that this procedure does not work, in which case a new procedure will be needed. One effect of the overarching Conservatism Principle should be that these events will not happen frequently; but given the procedure's reliance on human judgment, it may be necessary to tolerate errors from time to time.

B.6.1. Risks of the procedure

The foregoing may lead the reader to conclude that, for the procedure outlined here to work, the integration panel must have not only an extraordinarily high degree of intellectual discipline, but must also be very nearly omniscient and infallible. As a result, the procedure places ICANN in the position of having to defend determinations about applications for root labels; and moreover, of judging whether an applicant's claims about the relationships among both labels and code points are correct. That may cause ICANN to be embroiled directly in disputes with applicants.

Such a system may be at odds with recent directions in ICANN procedures, which have generally been to avoid having ICANN participate directly in disputes and instead have external bodies adjudicate those disputes instead. A sketch of a suggestion for an alternative approach that relies on third-party dispute resolution is in Appendix H.

C. Other considerations about the procedure

C.1. How early can we have some label generation rules?

Asmusf 12-10-17 07:42

Comment [10]: This should be bumped up one level in the outline – it's a discussion about the proposed procedure, like sections C and D – the exception might be B.6.2 which is either outside the process or an integral self-perpetuating part of it.

Some communities have grappled with variant issues already, and they may believe they are "ready to go". Some have argued that it would be unfair to make those communities wait until everyone else in the world is ready. There is some merit to this position, since if we took seriously the requirement to wait until *everyone* is ready, we might never be able to act: we would have to wait until we were sure that the encoding of every possible writing system was complete.

The integration panel may deliver a repertoire before waiting for all generation panels to complete, *provided that* it has strong reason to believe that there will be no overlap between the Unicode code point range it is delivering, and the work of an

existing (or likely prospective) generation panel. There are two cases where this appears to be likely.

The first is for zone repertoires restricted to one script, which script is unrelated to any other script, and is used for just one writing system – what we might call an "isolated" script. A candidate zone repertoire of this first case may contain only assigned code points used for one language (or set of closely related languages), and never used for anything else.

The second (perhaps more common) case is for zone repertoires that may be built from more than one script, but where a single generation panel that has, in the opinion of the integration panel, considered the issues for all the potential users of the included assigned code points. For the second case, the integration panel must be all but certain that there are no possible uses of the assigned code points included in the rules that have not been considered by the generation panel. For practical purposes, this will restrict "early ruling" to code points that are used in only a few writing systems.

C.2. Suggested initial cases

This section should be read as advice or a suggestion, but not as normative. In what follows, we discuss some particular cases where we think panels should be created that span more than one writing system, suggest the initial list of panels, and suggest some scripts that should be excluded from the start. In our view, some panels with a broader mandate are needed to arrive at recommendations that satisfy the Usability and Stability principles in particular. However, whether such panels can be created depends on cooperation by the relevant linguistic communities.

C.2.1.1 Initial exclusions

Some scripts with characters that are PVALID under IDNA2008 have been identified as of primarily or exclusively historical use, meaning that at present we judge it to be unlikely that they would attract a meaningful audience in terms of a user community in the context of the root. This likely warrants exclusion of code points having those script properties from being included in the root zone repertoire.

Examples of scripts identified as likely belonging to this class are Avestan, Brahmi, Carian, Coptic, Cuneiform, Cypriot, Deseret, Egyptian Hieroglyphs, Glagolitic, Gothic, Imperial Aramaic, Inscriptional Pahlavi, Kharoshthi, Linear B, Lycian, Lydian, Mandaic, Meroitic Cursive, Meroitic Hieroglyphs, Ogham, Old Italic, Old Persian, Old South Arabian, Old Turkic, Parthian, Phags-pa, Phoenician, Runic, Samaritan, Shavian, Tagalog, and Ugaritic.

Other scripts currently under ballot for inclusion in the Unicode standard, and which would in due course most likely be members of this class, are Caucasian Albanian, Duployan, Elbasan, Khudawadi, Linear A, Mahajani, Manichaean, Modi, Nabataean, Old Hungarian, Old North Arabian, Old Permic, Palmyrene, Pau Cin Hai, Psalter Pahlavi, Tangut, and Tirhuta.

The integration panel should use UAX#31 as its starting point to determine the list of scripts that are excluded under this provision.

C.2.1.2 Han and related

We believe it would be a good decision to convene a single generation panel to treat Han and, at the same time, writing systems used in conjunction with the Han script. In effect, this would constitute a "CJK" panel. This is in keeping with the Chinese Variant Issues Program report [ChineseVIP].

C.2.1.3 Cyrillic, Greek, and Latin

Because of the shared history of Latin, Greek, and Cyrillic scripts, it seems prudent that the integration panel be required to have complete input from generation panels for each in order to make any determination.

There have been suggestions that every script may need to mix with at least part of Latin. This request needs careful examination, but appears to us to be a risky initial assumption.

It is, however, unrealistic to expect that the generation panels will be able to consider every writing system that is based on Latin or Cyrillic. These scripts are used in too wide a variety of languages to make this feasible. This leads to the problem that any repertoire proposed will in all likelihood have to be only a partial repertoire, or else be in danger of including code points for languages for which the particulars of usage are too poorly understood to allow inclusion.

The integration panel in that case will have to make the determination whether it is acceptable or too risky to move forward with a particular proposed partial repertoire, based not only on whether it is comprehensive enough to serve as an initial repertoire for that script, but also on whether it is perhaps too inclusive given the available knowledge.

C.2.1.4 Brahmi-derived scripts

The variant issues project developed a report only on Devanāgarī [DevanagariVIP]. The resulting report, however, hinted at issues common to Devanāgarī and other Brahmi-derived scripts. At the same time, a large number of scripts that are derived

from Brahmi bear little modern relation to one another. Therefore, it seems prudent that the integration panel and other relevant experts be consulted during the initiation of generation panels for any Brahmi-derived scripts at the same time, in order to ensure that generation panels that might impinge on one another for the integration panel are all constituted at once. In addition, it seems that it would be wise to deal with related Brahmi-derived scripts be handled by a single generation panel as far as possible.

C.2.1.5 Arabic

The Arabic issues report [ArabicVIP] made plain that the Arabic VIP team did not have adequate expertise to cover all the different uses of Arabic when reporting. For the purposes of the generation panel, it will be extremely important to address those gaps when proposing the Arabic portion of the zone repertoire and associated code point variant rules.

If these gaps cannot be closed, the integration panel will have to make the determination whether it is acceptable to or too risky to move forward with a particular proposed partial repertoire. The issue here is similar, but by no means the same as, that in Section C.2.1.3,

Andrew Sullivan 12-11-23 11:47
Deleted: C.1.1.3

C.2.1.6 The "easy cases"

We have identified a certain number of scripts, with small repertoires and used by one or by only a few languages, as "easy cases": at present we judge them to be relatively unproblematic, and they may not need to have very large generation panels or protracted discussion. (This does not mean that they would not require a generation panel to do the processing; it is just recognized that the discussions about their repertoire are unlikely to be complex or problematic.)

The scripts identified as most likely belonging to this class are Armenian, Georgian, and Thaana.

D. How the procedure aligns with the Principles

In general, the panels' deliberations are to be guided by the principles listed in Section A.3.5. Below we include some specific remarks on the ways the procedure achieves this.

D.1. Longevity Principle

Assuming the panels are doing their work, the Longevity Principle should be enforced by both the generation and integration panels. The panels are supposed to begin using the latest version of Unicode, but also to take into consideration the

stability of Unicode character properties. If the panels both fail to behave in this way, then there is a risk either that code points will be permitted for allocation in the root zone that do not work with multiple versions of Unicode, or that code point substitution rules will be adopted that work well in peculiar contexts, but that will work poorly in other (perhaps future) contexts.

D.2. Usability Principle

The Usability Principle aims at ensuring that the allocated code points included in the zone repertoire are useful as elements in unique identifiers. To the extent that a code point is confusing to the user population – either by accident or else by way of malicious use – use of the code point fails to adhere to the Usability Principle in that context.

The integration panel, especially, is responsible to ensure adherence to the Usability Principle. It is explicitly charged with considering the entire user population, which is everyone on the Internet.

D.3. Inclusion Principle

The procedure is an example of the Inclusion Principle in action, since every rule or code point is excluded until reviewed and then explicitly included.

D.4. Simplicity Principle

Part of the point of having the integration panel is that it performs a check of the Simplicity Principle. The integration panel cannot possibly include experts in every language and script, but the members must have general knowledge of Unicode, IDNA, DNS, or all of the above. If any member of the integration panel cannot understand the rationale for inclusion of some rule, then that member will not support the rule, and it will not proceed. This is the purpose of the unanimity requirement for the integration panel.

D.5. Predictability Principle

The proposal follows the Predictability Principle in much the same way it follows the Simplicity Principle: if the integration panel does not immediately agree with the recommendations of the generation panel, or if members of the integration panel disagree with each other, that is a good reason to suppose that the rule in question is not really predictable.

D.6. Stability Principle

Especially in the case of the root zone, the Stability Principle is less a matter of guidance and more a statement of fact. The proposed procedure attempts to minimize the possibility that any label generation rule will be permitted for the root zone without that rule having been considered as carefully as possible for any negative consequences. If there is a failure such that the integration panel determines that a previously active rule needs to be removed, this procedure requires that the procedure itself be subject to review.

D.7. Letter Principle

The panels are required to follow the Letter Principle in deliberations.

D.8. Conservatism Principle

The proposal is consistent with the Conservatism Principle in two ways. First and most important, because the integration panel is supposed to reject anything it does not positively think is safe, the Conservatism Principle is built in to the integration panel's criteria. Second, in the event of disagreement between the generation and integration panels, the proposed rule that is the subject of the disagreement is automatically excluded from the root label generation rules.

E. Evaluation of this Proposal Against the "Parameters"

Section A.3.6 introduced the four independent parameters that can be used in evaluating the proposed process for label generation rules.

- Comprehensiveness
- Expertise
- Qualification
- Centralization

The remainder of this section will describe these parameters in detail and use them to evaluate the proposed process.

For each parameter, we will examine their possible extreme values and consider their consequences on the IABCP principles outlined in Section A.3.5.

After that brief analysis, we will use these parameters to evaluate the proposed procedure for creating and maintaining the label generation rules

E.1. Overview of the Parameters

E.1.1. Comprehensiveness

The comprehensiveness parameter describes the extent to which all of Unicode is being considered. The maximal setting corresponds to the requirement that every code point in Unicode be evaluated before proceeding. The minimal setting would require review only of code points actually requested for allocation in the root zone.

Because Unicode changes from version to version, it is actually impossible to consider "all of Unicode": a future release will introduce new code point assignments that may be permitted under the IDNA2008 specification. Those code points will by definition not have been considered by a panel considering an earlier Unicode version. While it would be possible in principle to consider every code point in some version of Unicode, under the Inclusion Principle any code point not explicitly included would be excluded. And, since unassigned Unicode code points are DISALLOWED under IDNA2008, when the panels are considering the older version of the Unicode character repertoire they are, by definition, not including code points that will be assigned in a later version of Unicode.

It is not enough simply to investigate code points. A comprehensive analysis requires one also to take into account the way each language or writing system uses its repertoire of code points.

Consequences of requiring maximal comprehensiveness

The consequence of requiring maximal comprehensiveness is mostly procedural: considering every code point would take a very long time, and might never complete. This is not a risk in terms of the IABCP Principles, but ICANN would face considerable pressure to do something in the meantime, and if it did that would almost certainly violate the Conservatism Principle. In addition, it is possible that, when investigating all of Unicode, those performing the investigation will be tempted to rule in favor of including a code point or associated rule they do not, or do not fully, understand. This would violate the Conservatism and Inclusion Principles.

It would be possible to reduce the above risks by considering a smaller subset of Unicode – that is, by requiring less than maximal comprehensiveness. The consequence of that would be the risk of later additions.

Consequences of accepting minimal comprehensiveness

Minimal comprehensiveness introduces a high risk of subsequent violations of the Stability and Usability Principles. If evaluations are made only for code points as they are requested, then a later request could introduce factors that were not under

consideration in an earlier evaluation. Because a later request is likely to include code points not previously requested, later evaluations will have to expand the repertoire. If those code points were not considered previously, then there is some risk that there will be changes to the rules (and by definition, there will be changes to the repertoire). Those changes could include new rules that introduce a conflict with older rules, making a formerly acceptable code point into one that is unacceptable.

That would be a violation of the Stability principle.

E.1.2. Expertise

The Expertise parameter reflects who is involved in establishing the repertoire and rules. IIR includes under this description both the question of the degree of expertise and the degree of centralization in the development in the rules; in the present case, we are distinguishing between these parameters; see section E.1.4 for discussion of centralization. Requiring maximal expertise would place the entire burden for development on experts in the subject area. Experts would be needed in all the relevant topics, including at least Unicode, software internationalization, IDNA, and DNS; for all of these, expertise in both protocols and operations is required. A minimal requirement, in contrast, would accept a rules definition from anybody, regardless of their knowledge of the script or protocols in question.

Consequences of requiring maximal expertise

It is hard to see how requiring maximal expertise could result in any violation of the Principles, but such a setting could be practically unsustainable. Because of the limited pool of experts, requiring maximal expertise would also automatically result in an increase in centralization.

Because a panel of experts would have to undertake all development itself, there is the potential that it could take a long time (this is also a risk with respect to centralization). Moreover, if those desiring to register IDNs in the root zone are not included in the development of the rules, then there is a considerable risk that they will object to the experts' judgment when it is rendered. If a language purist were to be part of the panel, it is possible that such a participant could effectively prevent an outcome he or she did not like, because of the unanimity requirements.

Consequences of accepting minimal expertise

Since accepting the minimum would require no expertise at all for the establishment of rules, there is every reason to suppose that any resulting rules would violate the Conservatism Principle whenever that principle does not yield a result someone

wants. It would also likely produce divergent or inconsistent rules, thereby violating the Simplicity and Predictability principles.

E.1.3. Qualification

The Qualification parameter reflects the extent to which code points can be restricted *a priori* for inclusion, or the rules about them determined at least partly according to properties of those code points. (It might also be called the Automaticity parameter; here we follow the IIR and use the name "Qualification".) A maximal setting for this parameter would result from using something like the Unicode script property, and establishing rules that only permit labels made of code points all with the same script property. To prevent confusion, also, code points with the script property Common or Inherited would be automatically disqualified. A minimal setting of this parameter would correspond to using no property of code points in evaluation, relying instead on arbitrary combinations within the bounds of IDNA2008.

Consequences of requiring maximal qualification

Because the Unicode script property does not map perfectly to any writing system – especially in the context of the DNS – strictly qualifying characters is likely to be very surprising for at least for some classes of user, not least because of the exclusion of Common and Inherited code points. Moreover, a single writing system as used by one language may contain rules inconsistent with other languages using the same writing system; this violates Usability, Simplicity, and Predictability. Maximal qualification, therefore, seems at once too broad and too narrow.

Consequences of accepting minimal qualification

Applying minimal or no qualification runs the risk of violating the Usability Principle, particularly in respect of abuses. In addition, the minimal setting seems likely to permit violations of the Letter Principle, though it is hard to see what formal property could be used to ensure that the Letter Principle will be followed without also running into the problems outlined in IABCP (particularly Section 2).

E.1.4. Centralization

The centralization parameter describes the extent to which rules are made by a single group of people. A maximal setting would mean that all rules are proposed, considered, and set by a central committee. Accepting a minimal setting would allow rules defined by anybody at all. The case of minimal centralization would require some, possibly centralized, mechanism for conflict resolution.

Consequences of maximal centralization

This may not violate any of the Principles, but might be politically unacceptable because of an appearance that it is not sensitive to community concerns. The required expertise is also so diverse that it is unlikely that it can be collected effectively in a single body.

Consequences of accepting minimal centralization

A free-for-all approach runs considerable risk of violating the Usability, Conservatism, Simplicity, and Stability Principles, and might violate the Predictability Principle as well.

The first issue with minimal centralization is that different types of users will almost certainly submit rules for different ranges of Unicode code points. There is every reason to believe that different people will treat the same code point in different ways. If the different rules are not reconciled, then the total set of rules will likely violate the Usability and Simplicity Principles, as well as the Predictability Principle. In any case, such unreconciled rules violate Conservatism and Predictability, because the total set of rules might not be internally consistent. If the rules are reconciled, then Conservatism is violated, since a later addition of new rules is likely to violate the Stability Principle.

It seems plain that, even with minimal centralization, the rules still need to be reconciled. By definition, at the minimal setting there is no group of experts to reconcile the rules, so the only mechanisms for reconciliation are first come, first served; or a secondary rule that, in the event of conflict, both conflicting rules are removed from the rule set. The first come, first served mechanism is in clear violation of the Usability, Predictability, and Simplicity Principles: it does not address the entire root zone user population, and the only way to make the rule predictable and easily understood is to know the order in which requests arrived. On the other hand, the second mechanism, of denying both conflicting rules, provides a simple method for someone to prevent all IDN labels in the root zone, by submitting one rule designed to conflict with any other rule that is submitted.

E.2. This procedure and the various parameters

The boundaries outlined above suggest that a successful procedure for generating label generation rules will correspond to a setting of the four parameters that minimizes the risks of complete paralysis, while at the same time limiting the number of the labels that are permitted, within the goal of allowing for useful mnemonics. The overarching Conservatism Principle suggests that allowing the smallest number of possible labels is desirable; and, given the user population of the root zone (i.e. everyone who ever uses the Internet), the Usability Principle, with its

focus on positive recognition and limits on possible misuse of labels leads to the same conclusion. The Longevity, Stability, Simplicity, and Letter Principles all militate in the same direction.

E.2.1. Comprehensiveness

This process recognizes that it is effectively impossible to review even a single version of the Unicode Standard in a comprehensive manner. The reason goes beyond the sheer number of code points – itself a daunting problem – and extends to the need to review the use of these code points by each and every contemporary writing system. For many of these writing systems, it is effectively impossible to access the required information. Moreover, the Unicode Standard itself has not completed its task of supporting all such writing systems, which is part of why it is being updated regularly as additional information becomes available.

The proposal recognizes the essential additive nature of the process and the requirement to allow for a similar additive nature in terms both of repertoire and variant rules. At the same time, the initial phase of the work should attempt to deal with the repertoires and writing system for at least the major scripts and user communities.

The process does not require a comprehensive consideration of everything in Unicode, because it relies on the establishment of generation panels to tackle subsets of Unicode. Those generation panels are based on communities of interest, and we presume that some portions of Unicode will not be interesting. Indeed, some portions of Unicode are simply marked as not eligible, on the grounds that there is no living language community that uses those ranges of Assigned Code Points.

As a further constraint, the integration panel is charged with ensuring that the final set of label generation rules takes into account not only then-current, but likely future uses of Unicode in the root zone; this includes effects from code points or rules not actively in use or under consideration, but that might be requested in future.

We may say, then, that the process offers a moderate level of comprehensiveness. It attempts to offer progress – particularly for those scripts where the issues are well understood – while yet constraining the label generation rules so that the Stability and Usability Principles are followed.

One clear risk that arises from the arrangement is the possibility that the integration panel will miss some future possibility, and agree to some portion of label generation rules that cuts off a future portion of the repertoire or otherwise cause difficulty in the future. Because the procedure depends on human judgment, there

is no way in general to prevent such an outcome, although the design of the integration panel (and its dependence on expertise; see Section E.2.2) is intended to minimize the risk as far as possible. See also the discussion in Section B.6.1.

E.2.2. Expertise

Because of the two-panel structure, the procedure relies upon a high degree of expertise without confining itself only to experts. Initial development is undertaken by generation panels that are neither necessarily expert in any particular area, nor in the whole of Unicode. The integration panel, on the other hand, has a responsibility for total review and is ultimately responsible for the label generation rules as deployed in the root zone.

The procedure recognizes that a shared resource, like the root zone, requires cross-script expertise, but that each script and writing system will bring its own issues. Instead of requiring a single expertise level, the proposal provides for different levels and types of expertise at each level of panel.

The expertise parameter for the generation panels could be said to be at a medium or moderate level, while for the integration panel it must be at a high value.

This approach permits timely progress and ensures that opinions other than those of the experts are taken into consideration during development. At the same time, requiring a final ruling by independent, impartial (i.e. disinterested) experts increases the probability of following the Conservatism Principle.

E.2.3. Qualification

The procedure opts for pre-defining qualification largely in the negative sense. The first qualification is a consequence of IDNA2008, which implicitly excludes many code points (that is, everything starts off excluded, and a code point gets added only by explicit inclusion). The second pre-defining qualification is not to permit any code points and scripts that are not used for everyday writing. This includes dead scripts, as well as specific code points used for dead languages, and for specialized uses such as phonetics, and the like. Finally, the integration panel will only permit in the zone repertoire those code points normally used to write words. There is no single Unicode property that covers any of these restrictions, although the repertoire for historic scripts can be derived from the script property.

The procedure does not impose a formal link to any Unicode property for purposes of qualification, in an attempt to permit the generation panels to select those parts of Unicode that they believe to be the best fit for their needs. Overall, this corresponds to an intermediate value for the Qualification parameter. Having whittled down the eligible possible characters, the precise Unicode properties or

other information to be used for the qualification of characters is left to the generation panels to sort out, with the important limitation that code points not suited for use in identifiers are not allowed.

E.2.4. Centralization

Given the political and technical realities of needing a single, internally consistent set of rules to govern the root zone, the procedure attempts to strike a balance between wide participation and deep expertise of individual scripts and languages, and the control and consistency that may come from having a central authority.

The proposed process uses two types of panels: the first type of panel – the generation panels – are to allow broader community input for each part of the repertoire, and the second type of panel – the unique integration panel – is to provide a centralized body of experts that are able to resolve conflicts and to represent the needs of the root zone as a whole. The proposal as a whole is characterized by a moderate level of centralization, while allowing for highly centralized reconciliation of the generation panels outputs.

The requirement for consensus between the primary and integration panels is intended to ensure that the reconciliation process will give results in line with the Conservatism principle.

F. References

[ArabicVIP] Hussain, S, *et al.* "Internationalized Domain Names Variant Issues Project Arabic Case Study Team Issues Report". (Marina del Rey, California: ICANN, October 2011). http://archive.icann.org/en/topics/new-gtlds/arabic-vip-issues-report-07oct11-en.pdf.

[ChineseVIP] Lee, X. *et al.*, "Report on Chinese Variants in Internationalized Top-Level Domains". (Marina del Rey, California: ICANN, October 2011). http://archive.icann.org/en/topics/new-gtlds/chinese-vip-issues-report-03oct11-en.pdf.

[DevanagariVIP] Govind, *et al,,* "Devanāgarī VIP Team Issues Report". (Marina del Rey, California: ICANN, October 2011). http://archive.icann.org/en/topics/new-gtlds/devanagari-vip-issues-report-03oct11-en.pdf

[LatinVIP] Frakes, J, *et al.,* "Considerations in the use of the Latin script in variant internationalized top-level domains: Final report of the ICANN VIP Study Group for the Latin script". (Marina del Rey, California: ICANN, October 2011). http://archive.icann.org/en/topics/new-gtlds/latin-vip-issues-report-07oct11-en.pdf.

[IABCP] Sullivan, A., Thaler, D. , Klensin, J. , and O. Kolkman, "Principles for Unicode Code Point Inclusion in Labels in the DNS." draft-iab-dns-zone-codepoint-pples-00.txt. Work in progress. Available at http://wiki.tools.ietf.org/html/draft-iab-dns-zone-codepoint-pples-00. Visited 2012-09-21.

[IIR] Internet Corporation for Assigned Names and Numbers, "The IDN Variant Issues Project: A Study of Issues Related to Management of IDN Variant TLDs (Integrated Issues Report)." (Marina del Rey, California: ICANN, February, 2012). http://www.icann.org/en/topics/idn/idn-vip-integrated-issues-final-clean-20feb12-en.pdf.

[ISO15924] *Codes for the representation of names of scripts*, ISO 15924:2004. Available from http://www.unicode.org/iso15924/. Visited 2012-09-21.

[RFC3743] Konishi, K., Huang, K., Qian, H., and Y. Ko, "Joint Engineering Team (JET) Guidelines for Internationalized Domain Names (IDN) Registration and Administration for Chinese, Japanese, and Korean", RFC 3743, April 2004.

[RFC4290] Klensin, J., "Suggested Practices for Registration of Internationalized Domain Names (IDN)", RFC 4290, December 2005.

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

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

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

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

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

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

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

[UAX24] UAX #24: *Unicode Script Property.* An integral part of The Unicode Standard. Most recent version available from http://www.unicode.org/reports/tr24/. Visited 2012-09-21.

[UAX29] UAX #29: *Unicode Text Segmentation.* An integral part of The Unicode Standard. Most recent version available from http://www.unicode.org/reports/tr29/. Visited 2012-09-21.

[UAX31] UAX #31: *Unicode Identifier and Pattern Syntax.* An integral part of The Unicode Standard. Most recent version available from http://www.unicode.org/reports/tr31/. Visited 2012-09-21.

[Unicode62] The Unicode Consortium. The Unicode Standard, Version 6.2.0, defined by: "The Unicode Standard, Version 6.2.0", (Mountain View, CA: The Unicode Consortium, 2012. ISBN 978-1-936213-07-8). http://www.unicode.org/versions/Unicode6.2.0/.

[UTS39] UTS#39: *Unicode Security Mechanisms*. Available from http://www.unicode.org/reports/tr39/. Visited 2012-09-21.

Asmusf 12-10-17 07:42

Comment [11]: The "An integral part of The Unicode Standard." Follows Unicode's recommendation vis a vis citation. See http://www.unicode.org/versions/#Unicode_Standard_Annexes
I think listing the editors is a mistake for "evergreen" documents, as they may change, so I'm not adding that.

Appendix A Example of a functioning label generation rules with variants

Let us suppose that a Latin panel, going against the advice in the Latin variant issues report [LatinVIP], determined that it needed to produce variants for Latin characters. The tag in this case is "und-latn-TBD-root". This is an imaginary example, is probably wrong in the details, and is not intended to guide any future primary or integration panel in any deliberation.

Suppose that the generation panel determines that, to be useful, the variant relationship in Latin is from a single "base" character to every "decorated" character in the script. In effect, every character in the ASCII alphabet would have a variant with every possible combination of that character and any of diacritic used with that character.

So, for instance, the character U+0073 (s) LATIN SMALL LETTER S might be in the repertoire. Its code point substitution rules might include U+015B (ś) LATIN SMALL LETTER S WITH ACUTE, U+015D (ŝ) LATIN SMALL LETTER S WITH CIRCUMFLEX, U+0161 (š) LATIN SMALL LETTER S WITH CARON, and so on. Similarly, the character U+0064 (e) LATIN SMALL LETTER E might be in the repertoire. Its code point substitution rules might include U+00E8 (è) LATIN SMALL LETTER E WITH GRAVE, U+00EB (ë) LATIN SMALL LETTER E WITH DIAERESIS, and U+1EBB (ẻ) LATIN SMALL LETTER E WITH HOOK ABOVE.

Asmusf 12-11-6 09:44
Comment [12]: Following the convention in the Unicode Standard, let's move the (s) between code point and name.

Thus, when someone applied for the label "test", the label generation rules would also generate variants: teśt, teŝt, tešt, tėst, tẻšt …

In this particular case, there seem to be two possible directions for the disposition of the resulting labels. In principle, it seems unlikely that any of the code points would necessarily lead to a blocked label, so every one of these variant labels would be allocatable. Whether any of them was delegated would be a separate matter to be determined under prevailing ICANN policy at the time. The Conservatism Principle, however, suggests minimization of the number of variants. So, perhaps the Latin label variant disposition rules should say that, if any one of the set of variant labels is delegated, then all the others must be blocked. In that case, the applied for label "test" would be allocated, and all the other variants blocked so that nobody else could successfully apply for them.

Importantly, for reasons of symmetry, if the application were for "teśt", the list of variants would be the same. In case all variants were to be blocked, it would cause the (non-IDN) label "test" to be blocked. This illustrates the way that the IDN repertoire for the zone has implications for traditional LDH-labels (and conversely).

An important question for the panels is whether there would need to be variant rules across scripts to ensure blocking. For instance, the (nonsense) Cyrillic label тєѕт (CYRILLIC SMALL LETTER TE U+0442, CYRILLIC SMALL LETTER IE U+0435, CYRILLIC SMALL LETTER DZE U+0455, CYRILLIC SMALL LETTER TE U+0442) seems a plausible candidate to be blocked in the event the label "test" is allocated, and it seems quite likely that all the three Cyrillic code points in question will be included in the repertoire. On the other hand, "тєѕт" does not seem to be a useful label, and it seems unlikely that anyone is going to spend the money necessary for a root zone delegation as a phishing strategy. The most conservative approach, however, might be to create such rules in an effort to minimize the potential for such problems.

Appendix B Examples of structurally invalid strings

There are strings that are structurally invalid, but that are possible to type. For instance, it is possible to put together a series of combining marks that would be IDNA2008 PVALID but that should not be permitted as labels. The character U+0300 COMBINING GRAVE ACCENT is PVALID. So, it is possible to construct a PVALID string U+0300 U+0300 U+0300 U+0065 (it might look like this: ̀̀̀e). Such a string is not structurally valid, however: it is a string that begins with three combining marks.

The purpose of the final whole-label evaluation, where it is in place, is to prevent cases like these from being possible labels in the root zone.

Appendix C Examples of characters not suitable for labels

The following present some examples of characters that would not be expected to satisfy the conditions for being included in the code point repertoire for their script, in this case Latin.

1) Examples of historic use characters *within* a script, e.g. Latin characters specifically.

- U+01BF ƿ LATIN LETTER WYNN used in Old English manuscripts
- U+021D ȝ LATIN SMALL LETTER YOGH used in Middle English and some early modern Scots
- U+A729 ꜩ LATIN SMALL LETTER TZ used in 16th-century Mayan texts

2) Examples of phonetic use only characters, again for Latin

- U+1D1D ᵝ LATIN SMALL LETTER SIDEWAYS U used in Uralic phonetic transcription
- U+1D6F ᵯ LATIN SMALL LETTER M WITH MIDDLE TILDE used for velarized [m]
- U+1D7A ᵺ LATIN SMALL LETTER TH WITH STRIKETHROUGH used in American dictionaries

3) Examples of special use Latin characters not needed for mnemonics

- U+0195 ƕ LATIN SMALL LETTER HV used for transcription of Gothic
- U+A723 ꜣ LATIN SMALL LETTER EGYPTOLOGICAL ALEF
- U+A75B ꝛ LATIN SMALL LETTER R ROTUNDA used in early printing.

These characters, while indubitably associated with a script that sees widespread modern use, are themselves not part of that usage, and it would be expected that the generation panel would filter them out.

Asmusf 12-11-6 09:40
Comment [13]: Images needed

Appendix D An Arabic Script Example

The Arabic script provides two versions for the letter Kaf. The Arabic-language form of the Kaf is encoded at U+0643 (ك) ARABIC LETTTER KAF and the Persian form at U+06A9 (ک) ARABIC LETTER KEHEH. Both code points are in the same script, Arabic.

A tabular representation of this portion of the label generation rules might look like this:

Code point	Allocatable variant	Blocked variant	Tag
U+0643	U+06A9	-	und-arab-TBD-root
U+06A9	U+0643	-	und-arab-TBD-root

Alternatively, the rule might be that one is permitted only one Kaf. In that case, a tabular representation of this portion of the label generation rules would look like this:

Code point	Allocatable variant	Blocked variant	Tag
U+0643	-	U+06A9	und-arab-TBD-root
U+06A9	-	U+0643	und-arab-TBD-root

Asmusf 12-11-6 09:40
Comment [14]: I fail to see what these tables and formalisms provide beyond the text in the suggested rewrite – if we were discussing rules based on language, that might be different.

Andrew Sullivan 12-11-22 23:30
Comment [15]: The point of the tables is actually to make this more consistent with other appendices.

Appendix E An Example of Chinese and Japanese

Chinese and Japanese share some characters. Moreover, both Chinese and Japanese have, at different times and in different ways, altered the use of those characters. Chinese readers and writers have different expectations for the relationship between the altered characters than Japanese readers and writers. Chinese and Japanese have different script identifiers in ISO 15924. As a result, when there are code points shared between the two scripts, they will (at least sometimes) have different variants and dispositions.

An example may help to demonstrate this. Consider the following characters: CJK UNIFIED IDEOGRAPH-6200, U+6200 (戀); CJK UNIFIED IDEOGRAPH-604B, U+604B (恋); CJK UNIFIED IDEOGRAPH-7231, U+7231 (爱); and CJK UNIFIED IDEOGRAPH-611B, U+611B (愛). In Chinese, the characters are used thus:

An example may help to demonstrate this. Consider the following CJK unified ideograph characters[3]: U+6200 (戀), U+604B (恋), U+7231 (爱) and U+611B (愛). In Chinese, the characters are used thus:

Simplified Chinese	Traditional Chinese
恋 U+604B	戀 U+6200
爱 U+7231	愛 U+611B

In Japanese, however, the characters are used differently:

Current form	Old form
恋 U+604B	戀 U+6200
愛 U+611B	

Given the variant relationship observed between simplified and traditional characters for Chinese, there is a (symmetric) allocatable variant relationship between U+604B and U+6200, and another between U+611B and U+7231. In contrast, Japanese readers and writers do not consider that there is a strict variant relationship between any current form and old form characters. For Japanese,

[3] In the Unicode Standard, the full character name for a CJK Unified Ideograph is determined from the character code, and therefore somewhat redundant. For example, the name for U+6200 (戀) would be CJK UNIFIED IDEOGRAPH-6200.

Asmusf 12-11-6 09:40
Comment [16]: The convention used in the Unicode Standard is U+XXXX (x) NAME OF CHARACTER. I think we should change to that convention here as well. (I've not made that edit, because change tracking would make it unreadable).

Asmusf 12-11-6 09:50
Comment [17]: Alternative:

I dropped the full character names, they look very official, but for CJK ideographs they are rather uninformative when given in conjunction with the code point.

If you feel this information is useful, then I suggest a footnote as sketched here.

Andrew Sullivan 12-11-22 23:31
Comment [18]: By dropping these, we violate the notational conventions we set up at the beginning of the document. Consistency and small mind, perhaps, but it makes the introductory text more complicated because we have to explain that we don't do it all the time.

Asmusf 12-11-6 09:50
Comment [19]: As Dennis has pointed out.

therefore, U+604B blocks the use of U+6200 (and conversely), and U+611B has no relationship to U+7231.

As a result, we might expect that the generation panels will produce different results. The relevant subset of the label generation rules for Chinese coming from the generation panel (und-hani-TBD-root) might be represented this way:

Code point	Allocatable variant	Blocked variant	Tag
恋 U+604B	戀 U+6200	-	und-hani-TBD-root
戀 U+6200	恋 U+604B	-	und-hani-TBD-root
愛 U+611B	爱 U+7231	-	und-hani-TBD-root
爱 U+7231	愛 U+611B	-	und-hani-TBD-root

The generation panel for Japanese (und-jpan-TBD-root) might come up with a different set of rules:

Code point	Allocatable variant	Blocked variant	Tag
恋 U+604B	-	戀 U+6200	und-jpan-TBD-root
戀 U+6200	-	恋 U+604B	und-jpan-TBD-root
愛 U+611B	-	-	und-jpan-TBD-root

Now, remember that the integration panel is required to integrate the proposals into a single set of label generation rules, and that it is allowed to add blocking rules to any tagged repertoire in order to achieve this goal[4]. In this example, one additional blocking rule is needed and the unified rules for these code points will be as follows:

Code point	Allocatable variant	Blocked variant	Tag
恋 U+604B	戀 U+6200	-	und-hani-TBD-root
恋 U+604B	-	戀 U+6200	und-jpan-TBD-root
戀 U+6200	恋 U+604B	-	und-hani-TBD-root
戀 U+6200	-	恋 U+604B	und-jpan-TBD-root
愛 U+611B	爱 U+7231	-	und-hani-TBD-root
愛 U+611B	-	爱 U+7231	und-jpan-TBD-root
爱 U+7231	愛 U+611B	-	und-hani-TBD-root

As a consequence of these rules, a root label that used the code point U+604B would block a label with U+6200 in the same position when applied for using the und-jpan-

Asmusf 12-11-6 09:40
Comment [20]: Better in a footnote, I think

Asmusf 12-11-6 09:40
Comment [21]: Less is more

Andrew Sullivan 12-11-22 23:32
Comment [22]: Removing the note about duplicates will cause someone to say that this is unimplementable because they don't understand database theory. We should state it somewhere.

[4] In case where the requirement to add a blocking rule stems from the fact that the rules for a specific tagged repertoire was incomplete, the integration panel would reject that repertoire, until it is made complete, rather than simply add the missing rules. This is not the case here, as the missing rule is a cross-repertoire one.

TBD-root tag, but it would produce an allocatable label were it applied for under the und-hani-TBD-root tag.

Asmusf 12-11-6 09:40
Comment [23]: Copy edit

Applying for U+611B using the und-jpan-TBD-root blocks the use of U+7231 in the same location in any label, no matter which tag it is applied under. This is so, even though U+7231 is not a character in Japanese at all. and does not appear in the tagged repertoire und-jpan-TBD-root. Because it is not part of that repertoire, it cannot be used in any label applied for with the und-jpan-TBD-root tag.

Appendix F Motivation for using language tags

This procedure depends on language tags, but requires that the language always be "und". This is in keeping with the fact that the root zone is *always* a zone for every language. The fact that the tags, structurally, happen to be language tags is an artifact of the choice of standard for the construction of the tags. It might have been simpler simply to use ISO 15924 script codes, and leave it at that.

Other zones, however, may be more sensitive to language, and it is useful to have a tag format that can always be used everywhere. (This is also the point of including the zone name in the tag.) In such use, it might be necessary to use tags with a different language subtag than "und". In such a case, there might also be more than one tag for the same script. The different tags would have different repertoires, and they might result in different dispositions for the resulting variant labels.

It appears that such an eventuality will never be appropriate for the root zone, so we have not here offered such a mechanism.

Appendix G Examples of differences with the Applicant Guidebook

The Applicant Guidebook for the new gTLD process ("AGB") includes a set of rules governing IDNA labels. They're contained in section 2.2.1.3.1, Part II. The procedure in this document will result in new rules that completely replace those in the AGB.

The AGB does not permit allocation of any variant labels. Rules resulting from the procedure in this document may permit allocation of variant labels in some cases.

The AGB requires every code point in an application to have the same Unicode script property, except for cases where the established orthographies and conventions say otherwise. The difficulties with this rule are outlined in Section B.5.3.1 of the present document.

The AGB also permits only code points with Unicode General Categories Ll, Lo, Lm, Mn, or Mc. This appears to be an attempt to yield something like the Letter Principle, but it may not be quite enough. Below are some examples of consequences of the AGB rules, and ways that the procedure outlined above might yield different results.

Many of the cases below will appear to be issues that are resolved by nobody being willing to pay the cost of registering a new TLD using the code points in question. But the new procedure is intended to be useful for both ccTLD and gTLD cases, and to be able to generate rules automatically for the indefinite future (without thereby involving the Board in controversies).

1. Ancient writing

The AGB rules do nothing to prevent things like U+13000, EGYPTIAN HIEROGLYPH A001 (𓀀), from being used. The hieroglyphs are generally letters (with General Category Lo), generally PVALID, and they have the script property Egyptian_Hieroglyphs. They are, however, unusable except to specialists, and there is no reason to suppose that a new TLD ought to be created using hieroglyphs.

The same thing is true of other obsolete writing systems. The procedure in this document provides a mechanism that rejects those characters from the outset.

2. Marks

The inclusion in the AGB of General Categories Mn and Mc is an attempt to include code points that are sometimes used to write words; for instance, virtually all vowels in Devanagari are in those General Categories. General Category Mn, however, contains a large number of marks that are PVALID but that should probably not be used indiscriminately. For instance, U+0300, COMBINING GRAVE ACCENT (◌̀) is a grave accent that combines with the previous code point (the example there, for instance, is a grave over a space). Because the AGB rules do not constrain how the code points may be put together, it is possible to put these marks together in ways that are structural nonsense (for instance, by combining a grave with another grave: ◌̀◌̀, or by putting a Devanagari vowel in a place where it does not belong). To be fair, not all such nonsense will be possible in valid U-labels (because of Unicode normalization rules), but the procedure in this document provides a way to make rules that further restrict structural nonsense.

Andrew Sullivan 12-11-21 14:01

Comment [24]: This is written this way because there are also implicit vowels, as I understand it, and therefore it's not "all vowels". But it still seems the wrong way to say it. Suggestions?

3. Difficult characters

There are some code points that present no difficulties in one context, but that present a significant problem in others. The most obvious example of these is U+02BC, MODIFIER LETTER APOSTROPHE (ʼ). This code point is a letter (General Category Lm). It is an important letter in many Polynesian languages, and is also used in Ukrainian. Although it is part of the Common script, its use is sufficiently common in some languages that it would likely meet the test of "well-established orthographies" to permit script mixing. The character is PVALID.

Nevertheless, the code point is troublingly similar to U+0027, APOSTROPHE ('). Given the practicalities of keyboards, many people frequently end up typing U+0027 where they believe they are typing U+02BC.

There is in fact no easy answer to this kind of problem, but the AGB rules do not permit treating characters one by one in the way that the procedure outlined in this document does. Depending on the interpretation of the AGB, either U+02BC is simply prohibited (which seems like an arbitrary restriction) or else it is allowed (in which case, there is a significant possibility for trouble across language communities). The procedure in this document offers a way to bring to bear linguistic and technical expertise on the topic, and also presents a way by which the repertoire can gradually expand so that, even if U+02BC is not included at first, subsequent iterations might determine (in light of experience) that it could be.

4. Variants

The AGB provides no mechanism whatever for the automatic identification of variant code point relationships. It instead relies on the applicant to identify

variants, partly based on a table to be provided by the applicant. Such variants are not to be allocated (or delegated), but are set aside for future evaluation.

The basic approach, where applicants submit different tables depending on what they are asking for, is incoherent for the root zone. The root zone is one zone, and in the end there can be exactly one list of acceptable code points in the zone and exactly one list of acceptable variants in that zone (though the resulting labels might have different dispositions, depending on how the application came. These issues are explored at length in the rest of this document). Moreover, there is no real procedure in the AGB for coping with the case where different applicants generate different variants based on the same code points.

The procedure in this document provides a mechanism to reconcile the various competing claims, and permits the allocation (and presumably subsequent delegation) of variant labels in circumstances where adjudged both safe and warranted.

Appendix H An alternative proposal

During the development of this procedure, some people objected to it on the manifold grounds that it was at once too complex, and that it put ICANN right in the middle of every dispute with the expert panels or in the role of having to defend at length rules made by experts. The effect is a complicated procedure that does not actually prevent the nastiest disputes, and which also requires the integration panel to be effectively infallible.

Those objections were accompanied by an alternative procedure. We include our understanding of the procedure here for completeness.

Under the alternative proposal, any applicant may submit any U-label. As long as the label meets the protocol requirements of IDNA, the label is presumably allowed. In addition, any applicant may submit any list of variant labels. The variant relationship is initially presumed to be an accurate and useful one.

Any party at all is allowed to object to the proposed new label, on any grounds. These disputes will be resolved by external dispute-resolution professionals, much the way trademark or other such disputes are resolved.