Post-Public Comment Scoping Exercise by the Vice-Chair of the Latin Diacritics PDP: Evaluating Consensus Over Character Choices and Its Limitations

Mark W. Datysgeld

Summary

Purpose

In this document, the Vice-Chair of the Latin Diacritics PDP outlines and provides additional context to the decisions made around what characters are to be considered within scope; which conversely excludes a number of characters. The goal is to explain (in what is hopefully and exhaustive manner) to the community and authors of relevant Public Comments why this was found to be the correct solution for this PDP during its discussion phase.

Consensus

Working Group

In the Initial Report, we defined the allowed set narrowly, in that the only difference between the ASCII string and the Latin-script diacritic string must be the presence of diacritical marks, and each such character must be decomposable into an ASCII character plus one or more combining diacritical marks under Unicode. The reasoning was straightforward: creating an objective and bounded scope. This carries the consequence of excluding widely used letters like æ, ŋ, þ, and ø because they do not decompose that way.

Vice-Chair's (Subjective) Understanding 1

The Working Group agrees that a broader model can be argued for on fairness and linguistic-inclusiveness grounds. However, the Working Group did not find that such a model could be incorporated into this PDP without: A) abandoning the clear and objective scope boundary that could be provided by the Unicode decomposition method; and B) materially expanding the policy problem beyond the chartered issue.

That would transform this PDP from a bounded mechanism addressing a defined policy gap into a broader and significantly more complex Latin-script equivalence exercise. The Working Group therefore chose the narrower model not because the broader one lacked appeal, but because the narrower model is the one that could be specified objectively, stress-tested coherently, and to put it bluntly: fairly.

Theorizing an alternative approach

Core reasoning

Were we to seek an alternative methodology, the benefits would be of having subjective completeness, serving more linguistic communities, and better aligning with user-facing assumptions when registering their domain names under a TLD. The downside is that we would stop having a mechanically testable scope and instead need a new policy layer for Latin-script equivalence, with unavoidable spillover into variants, contention, evaluation, and implementation.

The only realistic alternative to accomplish this would be to stop using Unicode decomposition as the scope boundary and instead use a new arbitrary, broader, Latin-script correspondence boundary. In other words: if a non-ASCII Latin-script label is found to be sufficiently tied to an ASCII label in user **perception** and/or orthographic **practice**, it could be considered for entering the same "simultaneous delegation" mechanism even when the relevant character is not decomposable into ASCII + combining mark. The eligibility test would then change.

Today the test is mechanical: decomposition plus RZ-LGR eligibility based on the definitions set out by Unicode; this follows in the tradition of the choice of ccTLDs being defined by the external ISO 3166-1 alpha-2 table, which leaves the scoping decision outside of the ICANN community's hands and allows it to focus on the policy alone.

Under a broader approach, we would define both the boundaries/testing and the policy. The test would have to become something along the lines of: a Latin-script non-ASCII label may be associated with an ASCII label where there is a recognized Latin-to-ASCII correspondence **of some sort** that results in the same end-user confusion or market-substitution concern that motivates the PDP.

The WG would then need to elect a mapping authority or develop a methodology to achieve that, replacing the current Unicode decomposition solution. Once characters like æ or ø are admitted, policy has to decide how they relate to ASCII strings on a per case basis. At that point, the policy is no longer about diacritics exception, but rather the need for a broader Latin-script equivalence framework.

This Latin-script equivalence framework was supposed or implied to come out of the prior RZ-LGR consensus, but such concerns are lightly touched upon there in the document's small variants definition. Changing the RZ-LGR has been **explicitly outside of the scope of the PDP** from its inception. What this means in practice is that a new solution of unknown nature would need to be developed or made to work within these new parameters **without altering the RZ-LGR**.

In that sense, variant interaction would also have to be revisited. The Initial Report currently walls the LD mechanism off from Variant TLD sets: no string in an ASCII/Latin-diacritic set may also be part of a Variant TLD set. In Annex B, the WG notes that allowing both regimes to operate together was rejected because it could create DNS security risks, complicate the work, and force reconsideration of EPDP-IDNs rules already adopted and under implementation.

In the broader model, contention and evaluation would become harder. Even under the narrow model currently adopted by the WG, some contention scenarios are out of scope and were determined to be handled by the New gTLD Program. A broader Latin-equivalence approach would significantly enlarge the number of such cases, as more strings would become eligible for same-entity treatment or quasi-equivalent treatment.

Vice-Chair's (Subjective) Understanding 2

Let's assume the real policy concern behind this work is to, and this is not a direct quote but rather an assumption: "evaluate pairs composed of one ASCII label and one visually/orthographically corresponding Unicode Latin label that can be operated by the same entity,". In that case, our Unicode decomposition rule is really only a proxy. It is a way to handle a much broader policy problem by targeting only one Unicode-defined subset of it.

Were we to attempt a new consensus, what would be the consequences?

In my impression, the greatest loss would be that of objectivity. The current model has a crisp answer to "why this one and not that one?", which is Unicode decomposition plus RZ-LGR eligibility. A broader approach loses that clean line and replaces it with ad-hoc policy judgment about correspondence, orthography, transliteration, and confusability. That is harder to defend procedurally and much harder to implement consistently.

A second automatic issue would be that of regime overlap. Under the current LGR, allocatable variant relationships already exist for dotless ı and sharp ß. A broader "all Latin" approach would inevitably raise the question whether the LD mechanism should absorb, coexist with, or defer to variant rules in these and future cases. Once that happens, we actively start redrawing the boundary between this PDP and the already established IDN variant regime.

A broader alternative model is fairer **in the abstract**, but we did not choose between two equally governable models. We chose between a bounded policy mechanism and an open-ended language-equivalence regime. The first can be specified, tested, and implemented. The second quickly stops being a "Latin diacritics" policy and becomes a much larger exercise in deciding which non-ASCII Latin characters should be treated as corresponding to which ASCII strings, by what standard, and with what interaction with existing variant rules.

The alternative was not rejected because it was irrational. It was rejected because it is not containable.

Dialoguing With Our Public Comments

The public Comments include criticism that the Unicode-decomposition rule is restrictive, that some non-decomposable Latin characters have obvious ASCII counterparts, and that a broader rule could be framed around Latin extensions in the RZ-LGR with obvious ASCII confusable counterparts.

This is fair and absolutely an interpretation that was taken into consideration. The question was never whether a broader model could be argued for. It was. The question was whether this PDP could produce a principled, objective, and implementable version of that broader model without turning itself into a different PDP.

We can admit that there are some non-decomposable Latin characters where there is an "obvious" or policy-relevant ASCII counterpart, likely using a broader Latin-extension or confusability logic rather than Unicode decomposition. That is exactly the kind of alternative articulated in the Public Comment: include Latin extensions in the RZ-LGR with an obvious confusable ASCII counterpart, and potentially even characters like AE/OE despite the fact that the ASCII counterpart would be multi-character.

The current adopted model uses an external, stable, non-ad hoc criterion. The broader model presented to us has no equally clean stopping rule. Terms like "obvious confusable ASCII counterpart" are policy-significant but methodologically soft. They invite exactly the sort of line-drawing fights that a narrow PDP tries to avoid: which characters count as "obvious," whether one-to-one and one-to-many mappings should both be allowed, how orthographic practice should be weighed against visual similarity, and who gets to decide edge cases.

The NCSG's comment illustrates this slippage the most clearly by moving from single-character counterparts such as cases to AE/OE cases where the ASCII counterpart is two characters, which is definitely outside of the scope that the GNSO Council permitted this PDP to work under. This is inexecutable within our constraints, regardless of the group's understanding, and this would call for the opening of a completely new effort at the Council level.

Our Charter was narrow from the outset: a single issue, under limited circumstances, concerning simultaneous delegation of a base ASCII gTLD and a Latin-script diacritic version that are not variants. The report and charter both frame the effort that way. The WG has consensus that broader issues are intentionally left to future PDPs so this one can remain narrower. This what was very directly and precisely asked of us by the GNSO community.

Even a narrower adjacent expansion, namely allowing overlap between ASCII/Latin-diacritic sets and Variant TLD sets, was recognized by the WG as philosophically attractive but operationally dangerous due to security risks, significantly complicating and delaying the work, and requiring revisiting recommendations aligned with EPDP-IDNs, likely affecting rules already adopted by the Board and under implementation for the Next Round.

If that much trouble is triggered by that one adjacent expansion, a broader character-scope expansion is difficult to portray as being within our scope. If even one nearby expansion destabilizes scope, timing, and policy alignment, a much broader character-theory expansion was never going to be a minor amendment, but an entirely different PDP or PDP phase instead.

It would require, at minimum, a new inclusion methodology beyond Unicode decomposition; a new rule for one-to-one versus one-to-many ASCII correspondences; a new treatment of overlap with RZ-LGR and Variant TLD logic; new stress-testing across edge cases; and likely re-checking consistency with EPDP-IDNs and SubPro-derived assumptions.

To be clear and precise: we did not decline the broader model because we failed to notice it. We declined it because adopting it with due responsibility would have required a materially different Charter, a much longer horizon, and a much larger policy design exercise than was allowed and asked of us.

Technical discussion

In the chosen Unicode model, the elegance is that “diacritic” is not a cultural or linguistic judgment. It is a Unicode-structural one: canonical decomposition to ASCII base letter plus combining mark(s). Unicode normalization explicitly distinguishes canonical equivalence from compatibility equivalence; NFC/NFD preserve compatibility composites, while NFKC/NFKD fold them. On the other hand, the “ae and so forth” family does not neatly sit in the same bucket as ordinary accented letters.

Relevant reading:

1. “UAX #15” (Sections 1.1 and 1.2) explains the exact difference between canonical and compatibility equivalence, and why those are not the same thing. It also provides a direct example of a compatibility mapping “dž → dž”, which is useful because it shows that Unicode does sometimes model multi-character correspondences explicitly; just not uniformly across all Latin letters people may intuitively group together.
2. “RFC 5892” shows that several of the characters that fall outside of the scope of our chosen combinatorial methodology are PVALID in IDNA, including SMALL LIGATURE OE

and O WITH STROKE. This clarifies that the problem at hand is not “these are forbidden on the internet”, but rather “what policy theory links them to ASCII labels for same-entity treatment?”

If we are doing an honest technical evaluation, there are actually **3 buckets to consider**:

Bucket A: Ordinary diacritic letters

Examples: á, ç, ň, ō
Base ASCII letter + combining mark(s), our current model. UAX #15 explains this as canonical equivalence.

Bucket B: Compatibility composites and ligature-like cases

Examples: dž, fi.

Unicode treats these through compatibility, not canonical equivalence, and NFC/NFD do not collapse them away.

Bucket C: Letters that resemble “ASCII + something,” but which are not just diacritics

Examples: æ, œ, ø, ß, þ, ð, ŋ

Attractive on grounds of fairness but methodologically dangerous. What makes Bucket C difficult is that **the intuitive ASCII counterpart is not one stable thing across the whole set**:

- **æ, œ**: invite one-to-many correspondences in ae, oe
- **ø**: invites one-to-one-looking correspondence to “o”, but semantically it is not just “o plus a detachable diacritic”.
- **ß**: invites “ss”.
- **þ**: invites “th”.
- **ð**: can invite “d” or “dh”.
- **ŋ**: invites “ng”.
- **dotless ı**: is not “i with missing dot” in the same policy sense as an accent mark.

By admitting “ae and so forth,” it is not merely broadening character scope. There is a change in the unit of analysis. This broader model pushes us toward at least one of these:

- Character-to-character correspondences based on visual or orthographic similarity.
- Character-to-string correspondences, such as æ ↔ ae or ß ↔ ss.

- Language-specific transliteration conventions.
- Some mixed standard(s) drawing from all three.

Unicode test

Unicode data behavior test in Python:

- æ, œ, ø, ß, ð, þ, ŋ, ı do not decompose under NFD or NFKD
- ij has a compatibility decomposition to ij
- dž has a compatibility decomposition to d + ž, and NFKD yields dž

Case Study: æ

Acceptance of equivalence between “æ” and “ae” is useful because it exposes hidden escalation quite clearly.

Immediate questions:

- Is the policy allowing single-character ↔ multi-character equivalence?
- If yes, is that limited to historical ligatures, or also things cases such as ß ↔ ss and þ ↔ th?
- Is the basis visual confusability, orthographic convention, transliteration practice, keyboard fallback, or market substitution?
- Is the rule language-neutral or language-dependent?
- If language-neutral, why those mappings and not others?
- If language-dependent, who adjudicates and maintains them?

“ae” is not just an extra character. It is the gateway to string correspondence as policy.

The moment we accept æ on the basis that it corresponds to “ae”, we have crossed from diacritic handling into ASCII expansion logic. From that point onward, the policy no longer rests on canonical character structure but on a correspondence methodology that must choose **in an arbitrary manner** among ligature history, transliteration, orthography, and user perception.

Comparison table of key characters

Character or group	Unicode type	ASCII + combining?	Compatibility decomposition?	Expected ASCII fallback	Why this destabilizes a broader scope
á / ç / ñ / ō	Ordinary diacritic letters	Yes	N/A	a / c / n / o	These are the clean cases. They fit a structural rule based on canonical decomposition.
æ	Independent Latin letter, not handled like a diacritic	No	No	ae	This forces a shift from character-to-character diacritic handling to character-to-string correspondence. Once æ→ae is allowed, the policy is no longer just about diacritics.
œ	Ligature-named Latin letter, but not canonically decomposable for normalization purposes	No	No	oe	Same as æ.
ø	Distinct Latin letter with stroke	No	No	o	This is dangerous because it looks visually close to an ASCII base letter, but Unicode normalization does not treat it as a decomposable diacritic form. Admitting it means the criterion becomes perceived correspondence, not Unicode structure.

ł	Distinct Latin letter with stroke	No	No	l	Same as ø.
þ	Distinct Latin letter	No	No	th	Same pattern as ß, but even more obviously dependent on transliteration convention rather than diacritic logic.
ð	Distinct Latin letter	No	No	d or dh	This exposes a second problem: even the fallback is contestable. Once the ASCII counterpart itself is arguable, the inclusion rule becomes methodologically weak.
ŋ	Distinct Latin letter	No	No	ng	Another character-to-string case. It widens the debate from visual similarity into phonetic or orthographic correspondence.
ij	Compatibility composite / ligature case	No	Yes	ij	Compatibility equivalence is weaker than canonical equivalence, and Unicode warns that NFKC/NFKD should not be applied blindly because they erase distinctions that may matter semantically or contextually. (Unicode)

dž	Compatibility composite with mixed result	No	Yes	dz	This is even messier than ij: Unicode gives a compatibility decomposition, but not to plain ASCII only: it yields d + ž, and under full compatibility decomposition that becomes d + z + combining caron. That shows how quickly a broader “simple” approach turns into edge-case handling.
ə	Distinct Latin letter (schwa)	No	No	e or a, depending on who argues	Useful as a control case. It shows that once you move beyond decomposition, the category can keep expanding into letters that are Latin-script but have no disciplined ASCII-correspondence rule.