ICANN85 Mumbai | CF - How it Works: Internationalized Domain Names
Tuesday, March 10, 2026 – 9:00 to 10:00 IST

CHAMPIKA WIJAYATUNGA

Hello, everyone, again. Welcome to How It Works - Internationalized Domain Names. My name is Champika Wijayatunga, and I am the Remote Participation Manager for this session. Please note that this session is being recorded and is governed by the ICANN Community Participant Code of Conduct, ICANN Expected Standards of Behavior, and the ICANN Community Anti-Harassment Policy.

Please observe the following guidelines to participate in this session. I will also post them in the chat for your reference. During this session, the questions will be read out loud if submitted within the Q&A pod. Interpretation for this session will include English, French, and Hindi. If you would like to speak during this session, please raise your hand. On-site participants will use a physical microphone to speak.

Only questions posted in the Q&A pod will be read aloud during the session as time permits. Please state your name for the record, the language you will speak, if speaking language other than English, and speak clearly at a moderate pace. Now, I will hand the floor over to Pitinan. Thank you.

PITINAN KOOARMORNPATANA Thank you, Champika, and welcome everyone to the session. So, my name is Pitinan Kooarmornpatana from the IDN team at ICANN. And today we will share with you a couple of points. First, we will go on the overview of what is IDNs or Internationalized Domain Names. And then we will also show you behind the scenes a little bit how to document all these rules into the technical of it. And then we will also invite a community member from Canada to share about their work on the First Nations languages.

And lastly, we have remote speakers from Indonesia to share their progress as well. So, we have quite a packed agenda, so perhaps feel free to paste your question, but we may take them at the end of the session. So, to begin with, to proceed how to enable the IDNs in local languages and scripts. So, this is when we talk about IDN, how it looks like.

So, you can see on the top, that's the Internationalized Domain Names, which is, this is for Devanagri script, parathpasha.parath. And this is, actually we have a slide posted in the sessions page as well. So, if you download the slides, you can copy and paste this link, and actually it works on the browser. So, that's on the top as the IDN form. Each one we called it the Unicode characters or so we also have the name U labels.

Behind the scene when it is converted into the protocol level, it is converted into the ASCII label which is start with X and dash dash as you see. So, for that one we call it A label. And how to enable this is come from a set of standards and we call it IDNA2008. So,

this is a standard developed by IETF, and this is a set of this standard track and informational track.

It talks about the definitions of all these names, the protocol, how to derive the Unicode code point categories to become the protocol property, and then also how to handle the script, which is maybe right to left, like the Arabic or Hebrew. So, just to give some examples, in one of the RFC, the number 5892, so this is about the property of the Unicode code point. So, by derive from the category of the letters, they have these four categories. First is the protocol valid, or we call PVALID, and this is generally the code point that possible to use in the DNS.

For example, this E with accent, in the Unicode code point is 00E9. The second type of property is the contextual rules required. So, this one is possible but you need to have some condition. It's not freely allowed. For example, the middle dot in Latin, sorry maybe it's a bit hard to see on the screen on the red, is the 00B7. So, this code point is only allowed to use if it comes in between the two Ls. This is to prevent the confusion when it uses the dot to separate the second level and top level. And there is also a type where it is disallowed.

For example, the sign and symbols, like the question marks here or the hash sign, because sometimes it can get confused by the protocol, whether this is the syntax or this is the label. And then the last type is the NS sign. So, in the Unicode code chart for each script, they block a certain number of code points for each script in

case there are some additional or some change. So, some of them may not be used.

So, those are the code points which are not being assigned. So, all this, however, in the standard itself is saying that if it's PVALID, it doesn't mean that it can be used in the DNS directly because there may be some variant definition or the rules beyond this property that needs to also be considered. And that is why we have to build in these rules called the label generation rules which we will show you in this session how it works.

So, the Label Generation Rules, to explain literally is how to use, what is the rules to form a labels in a particular script or languages. There are four parts of it. The metadata is explaining the rules. So, all this version, the date, language, and maybe you can add the description. Then we have three main sections, which is the code point repertoire, variant, and contextual rules. So, we go to each section one by one. But before going to that, there is also some design principles to consider.

So, we have this RFC 6912, which is the principle for the Unicode code point that can be included in the labels and used in the DNS. Quite a number of principles here, for example, longevity, like when you have new letters encoded in the Unicode, sometimes it's maybe if it's newly encoded in the later versions, the software or the devices may not catch up with this new standard.

So, even though the letter is already encoded in the Unicode, but maybe the keyboards still don't have that character. So, we

suggest that the code point that can be used in the domain name should be in the Unicode version for quite some times already to make sure it's stable. By the design, it should also be least astonishment.

For example, if you use a particular language or scripts and you never see that character used in your everyday life at all, somehow it appears on the domain name, maybe that creates some confusion and maybe is not understandable by the script community. It also has to be contextual safety. So, those code points where it needs some context to be able to use, those contexts will have to be in place.

Also, the conservatism principle, this is because in the domain name is the share zone, and basically people can use the resource of the name all together around the globe. So, there is also a principle where when you don't clear whether this code point should be used or not, it's always better to fall into the safer, more secure approach, which is more conservative. The next principle is the inclusion.

So, to decide what code point should be used, it actually should start from an empty set, and then one by one characters, you start to analyze all this. And if it's valid to be used, then you gradually add one by one. It shouldn't be from the starting point that you can include everything and then exclude the one that you don't need. So, it has to start from the empty set, and then gradually include.

So, all this, this is some of the principles that I like to point out. But just to jump to the last one, stability, because imagine if you allow something to be used on the domain name, later on it will be hard to take it out. So, before you allow it to be used, then all these principles have to be carefully analyzed. So, the first part of it, the code point repertoire, basically, we need to work with the community to answer what are the code point to be used for that particular script.

The designing goal, just to remind the domain names, basically, is the mnemonic system. And this is the mechanism for us, as in humans, to remember the name instead of remembering the IP addresses. So, all these designs have to still remain secure and stable. If the solution creates confusion, maybe this motivation cannot be met.

And also, because it's the name, it's not required to completely cover language or scripts, grammar or linguistic requirement. It doesn't have to be an actual word. For example, it can be abbreviations. And also, it strictly doesn't have to be a correct spelling, but it has to somehow be stable and predictable. So, let's go back a little bit on the ASCII domain name. So even in the ASCII, there is also some rules governed around it that we may not aware.

So, on the top level in the ASCII letters, you can only use the letter A to Z. This is because if you use the number on the top level as well, it's also possible that get confused by the IP addresses. So, by the standard on the top level, you can only use letter. But on the

second or third level, obviously you can use letter, digit and hyphen. So, there's already some rules in place. And this is the code chart for ASCII.

So, ASCII is basically the letters that you normally see on the keyboard. So, beyond the letters, you also see the signs, like mathematical sign and also some of these control keyboard, space, new line, and so on. So, if it's a top level, only the letters can be used, the one that highlight in green. But if it's the second level, then you can use the letters, the digits, and then the hyphen. So, those have to be short letters out.

Now, when we talk about Unicode, it's actually another body, Unicode Consortiums, who are encoding all these different script letters into the hexadecimal digits 4 or 5, and we called it the Unicode code point, and they produced this kind of chart. So, on the left, this is the example from Arabic chart, and on the right is a partial example from Ethiopian. So, those things right now, Unicode is at version 17.

It has already encoded 172 scripts. So, imagine if we start from the left, which is the ASCII space, it starts from basically we can call it one chart, one script. Only 63 code points are shortlisted to be used, and that already can create some confusion between perhaps I and L, o and zero, and so on. But now if we enabled it to be like the Unicode version, 172 scripts, and it's almost close to 300,000 characters.

So, we do need to shortlist this in the conservatism way, and so, we have to work with the community to identify how many code points are actually minimally required to cover the names in that script. And this is how we decide the code point. So, to work on this, we're starting from reviewing the code chart for that script, and then we need to answer all these questions, starting from the very basic whether this is protocol valid.

And then moving on, is it needed in the language or script? Is it suitable in the identifier? Sometimes it's used in everyday life, but it's not really used in the name, like repeating marks and so on. Is that Unicode encoding already stable? How accessible it is? For example, are these letters available on the input keyboard or not? And then we always have to go back to the RFC 6912. Is there any principle which is this code point might break if we add them in?

So, that's the code point part. And we will have some examples a little bit in the next section. But for now, let me move on to the second part, variant. So, another concept of IDN is, we also have some code points or the set of code points where the script community can consider them as the same. And let me show you some examples.

So, majorly there are two types. First is visually same, and generally we define them for the security purpose. So, if you see the text in blue, EPIC on the above left, and the text that look like EPIC on the right, if you see the code point behind it, it's starting from 0065, but the other one starting from 0435. So, basically,

those are two different scripts, sorry, two different, yes, two different scripts and two different labels.

But for the human eyes, we cannot really tell the difference apart at all, it's the Latin and the Cyrillic. So, in this type of variants, we have to define them as the variant for the security purpose. And actually, the Latin script community and Cyrillic script community, they come together, work together, and define like, okay, 0065 and 0435 variant. There is also another type of variant, which is the define for the usability.

And this is mainly applicable for the Han script or the Chinese script and also the Arabic script. So, for Chinese script, you can see there's two labels. The second character of those labels are slightly different because it's the simplified Chinese and the traditional Chinese. These two versions have been used in the different regions of the world.

So, if you are a brand owner operating in mainland China but you want your brand to be able to access from the other places as well, you might want to have the two names active at the same time. And for that, the Chinese community defined that the last code point, variant of E-charter. And that effect is similarly happened for the Arabic as well.

So, defining variant also has some tasks that needs to be done to build the label generation rules. The goal for this is to make the mnemonic system minimize the confusion, and so we need to define them as the same within the script and also across the script.

And because we're going into the conservatism requirement, so most of the time we define it as the for the security issues, so we define it as a block variant, which means if one version already exists, the other version shouldn't exist anymore.

But for some minimal case, we want to have them active at the same time, and for that, we define it as allocatable variant. And this task is also not a trivial because sometimes you have to make some agreement within the community and also have work across the community to counter the conclusion. Then moving on, sorry, so to define the variant, you have to go through this question, is there any to a pair of code point, which is by the knowledge of the native script users, they already find it interchangeable or the same? And is there any alternative representations? Is there any variant in some certain context? And is there any tension of any of the other principles?

So, those are the variant part. And then let's move on to the last one, the rule parts. So, in some of the scripts, we may need some restrictions as you see in the RFC, because of some of the nature of the script or code point, you might need some context to make sure it's secure and stable. And for this, again, reminding is not really the side of the grammatical rules.

This is just to make sure that how to use the script is nothing unexpected by the user. Reduce the possibility to cause security issues, reduce the possibility to create the usability constraints.

And also in some cases, the rules are in place to reduce the allocatable variant labels. And let me show you some examples.

So, this is example for the rules that are in because we want to mitigate the risk. So, if you see this code point on the left and on the right, it looks very similar, almost homoglyphs. It's like vertical line with some head below. On the left, it's just one code point, 0E41. But on the right, is actually the two code points together. One is just one vertical line, 0E40.

So, in this case, if you see this out of the context, not side by side, and you saw it on the browser, you may think it's the same thing. So, in this case, the rules for TypeScript, we actually put something to disallow 0E40 to follow each other, just to make sure that such case cannot happen. There are also stability issues.

For example, some of the script, you might have these structural things that you have tone mark, it's like accent, which is basically one consonant should have only one tone marks because you cannot sing two notes at once. This is from Laos script. So, if you type two tone marks together, some of the system will just render it on top of each other, like on the left is on the Android phone. But some other system might render it next to the previous one, like on the iPhone.

So, with this case, it creates the instability and unpredictability and also makes users surprised because the fonts and everything is not prepared to render this. So, this kind of case in Loas script, there are also some rules to disallow the tone mark to follow another

tone mark. And the use of the LGR, so when we have these rules in place, actually because it's machine readable, it can be digested by the tool, and then when it's already pre-installed, we can input the labels and it can validate the labels and also generate the variants of that label.

And just quickly on my end of my section, we have the two projects which is involved in the LGR. One is the Root Zone LGR which is the rule for the top level. This one, we have been working with different script community we call Generation Panels, and then we have integrated them into one collection of Root Zone LGR, and right now we are at version 6, included 27 scripts already which support almost 400 languages. These are 11 years of work with 18 generation panels from 45 countries.

So, more than 10,000 plus hours of volunteer hours. These are the 27 scripts that we cover, and this is also the base of what are the scripts that can be applied for in this coming round as well. Another project we have is the reference at the second level, and this one is because if you operate a TLD and you want to offer a language or script under your TLD, you have to develop the label generation rule, sometimes it's called IDN table interchangeably, and those has to be reviewed by ICANN.

So, we developed the reference LGR to review those table in the stable manner. And these are the list of the reference LGR that we have. We have both languages base and script base. And you can see in the presentation later, I'll just move on. So, the collaboration

to develop all these reference LGR, we're actually working with different community based on the request basis.

And the voluntary expertise that we need from the community is obviously the linguistic expert to develop the repertoire, the variant, and the rules. And also, we will request the community to provide a word list so we can do the further analysis and checks. And from ICANN's side, we will turn those identity descriptions into the rules and provide the technical LGR, and then ask the community to review.

Finally, when it's already reviewed by the community, ICANN will take it to the public comment for finalizing and then integrate into the existing reference LGR. So, if you are interested to enable your additional language of script, please contact us and we can take it from there.

So, now, I will hand it over to Akshat Joshi, who also used to work with us on the Root Zone LGR projects as a Neo-Brahmi Script Generation Panel Member, and he will take us through how to encode that into the technical part. Akshat, over to you.

AKSHAT JOSHI Thank you. This is Akshat for the record. So, we have seen how many language LGRs we have and how they actually work, what the purpose is. In this section is we will see how exactly an LGR looks, like how to actually make it, if you have any use case on your side,

how you can try to use the LGR for the purpose of trying out a language that you intend to go for or something like that.

So, there are four sections like Pitinan already has explained, metadata section. I'm just recapping it because in the next slide, we'll be able to see this in actual XML. One is the metadata section, the other one is a normative section which is code point to portray. Intermeshed with the code points are the variant information, and then in the end we have three types of rules that are context or label evaluation rules and the action rules.

These LGRs come in two forms. One is a fully machine-readable format that is XML, and the other one is HTML version of it, which you can use a LGR tool for generating out that particular version of it. Coming to the LGR tool, it's a tool provided by ICANN for developing and managing the LGR. Additionally, if you have some set of labels that you want to run through some pre-existing LGR or an LGR that you have made by yourself, you can use the tool as well.

And in case you're one of the registries, you want to check with some of the IDN tables that you have, whether they conform to a particular pre-existing LGR, you can use this tool as well. How to access the tool? You can go to lgrtool.icann.org, and you'll require an ICANN account for that, and you can freely start using the tool. There is a proper full user guide as well available for the tool at the link given on the slide.

Those who want to access the tool through their APIs, the code of the tool has also been open sourced, and you can use that version

as well. Okay, for the purpose of building an LGR in this session, we will take example of one of the fictional languages, A to G languages, H-A to G language. What are the characteristics of this language? The language has around 11 Latin letters and one sign.

Okay, that's what the community of the A to G language uses in their daily routine exchange. There are additional requirements by the language that there are two kinds of F characters present if you look at the code points. One F is normal, the other one is with the hook and community considers them to be the same. So, that makes a case for a variant configuration. And then there is an aloe vera click character that should never be at the beginning of any label. That's the requirement for this particular language.

Now, if we look at the Unicode code charts, all these characters come from basically the Latin script, but from three different codes charts. That one is the basic Latin that has all the letters from A to G. And then in the Latin 1 supplement, we have three characters, A with an acute, E with an acute, and then there is a degree sign present with a red square around it. In the next, we have Latin extended B -- just the previous slide, Pitinan. In the Latin extended B code chart, we have F with a hook and the I think that's a allow-willer click character.

Now there is a red square around the degree sign, and we'll see in the next slide why that is so. So, now that we just saw what are the set of questions that one needs to go through for inclusion of a code point into a repertoire. And for each of these characters, when we

go through each of these questions, all almost check out perfectly except the degree sign that it is not a PVALID character. So, that will not be able to be part of the LGR of this language.

Okay, going to the next part, that is the F character requirement, F and F with a hook. The community considers them to be the same, so that makes a case for us to create a variant between these two characters, and the disposition of this variant should be blocked because it is considered the same and community if there is one label made up of one using one F character, then this equivalent label with the other F should not be there in the system.

And then there is an additional whole-label evaluation rule requirement that the aloe vera click character should not be at the beginning of the label. So, let's see how it manifests in the form of a LGR. Okay, so on this slide, we see two versions of the LGR. On the left is the XML version, and on the right is -- XML is the machine-fully, machine-readable version, and HTML is a more or less human-friendly version that you can use various links to navigate through the LGR.

If you see look closely look at the XML, you can clearly see four sections that we talked about. First one in the meta tag, you see all the meta information regarding the LGR, then there is a data element which has code point info information and intermeshed with the code point information are the variant information. So, you will look at two places 0066, there is a mapping of 0192. And on the 0192, there is a mapping of 0066.

So, this is, again, coming from the symmetry principle, that if one character is variant of the another, then another has to be the variant of the previous one as well. And in the last section, that is at the end of the data element you can see the rule section. We can see the encoding of the rule that is at the start of the word that gets applied on the 0192 character. Okay, going to the next slide.

How this one now that we have made this LGR. And let's see like one of the functions that we saw of the LGR tool was to be able to validate the label. So, now we are in that section where we are validating various labels that the language community might want to use. So, first one being an ABE label with A, B being the regular characters and E with an acute and it will check out perfectly fine.

It will be a valid character because all the code points are from the repertoire and there is no violation of any rule of any kind. So, this will turn out to be a valid character. The second label that is A with a diuresis and B, E same as the previous test level one. It will get invalidated but because the A with the diuresis character is not part of the code point repertoire of AG language, so it will be an invalid label for this language. Next slide.

Then coming to the variant, you remember we made F and F with a hook as the variants of each other. Now if we go for validating the label F A C E, where A is with an acute, it will be termed as valid label, but it will additionally generate an information saying that this label has a variant label available with it, that is F with a hook and the rest of the characters the same.

And the disposition of it will be blocked because we have mapped it as a blocked variant. If we go for the other label wherein the label now starting as in the label has F with a hook inside it, then the equivalent label which is F without hook will get blocked. So, just to elaborate slightly on this, it's not preference of one character over the other. It's just that within a label, whatever comes that will be available and other one will in a label will be blocked. Next slide.

Coming to the language rule. So, let's see how that works. So, you remember there was one character, aloe vera click character, which was not supposed to be as the as the first character of the label. So, in CA and that particular character in the end, that checks out perfectly, it's a valid label because it doesn't violate any rule. Test level 6 begins with that particular character and that's why it will be termed as invalid character, invalid label, sorry.

And then there will be a description of why this particular label is not a valid one. In the LGR tool, you can see that. Next slide. So, just to give an example of a similar thing in Devanagari LGR, again, there was an example in yesterday's session. The kitab word was taken. In this one, there is a deliberate mistake that is injected wherein the Matra sign that is there is given two times, which is supposed to be only once.

And that's why this label is getting termed as invalid. And there is a systematic reason coming out of the LGR tool for that. Then there is another interesting case of variants, which are called whole label variants, wherein the variants exist across the script. Now, this is

an example where the label on the left is a Devanagari label composed of all the characters which are entirely Devanagari, similar to the epic example that Pitinan gave.

And on the right is a Gurmukhi label. And to a normal Devanagari user, if I'll say, then they might say that this looks similar, but maybe with some artistic points. So, even they become whole-label variants. With this, I come to the end of my part. Hand it back over to Pitinan.

PITINAN KOOARMORNPATANA Thank you, Akshat. Then the next section, I would like to invite Louis to provide us the update on building the reference LGR by the working group.

LOUIS HOULE Thank you very much. So, for people who don't know me, my name is Louis Houle, and I'm co-chairing the UCAS working group. Today, I'm going to be very brief. First, I would like to just ask you to take a look at the map right now. You see in blue, these are dialects from Inuit that are used in Canada, Northern and East-North. In orange, you will see the Eastern Cree, and you will also see in green the Western Cree, and in pink and purple, you have Dakelh and Dene.

So those languages, you'll see at the next slide, well, it's syllabic. The syllabic has 31 languages, the syllabic script, and the Latin script in our environment has 80 languages. This means 101, and there's 191 languages spoken in Canada. So, this is a large chunk

of it. Next one, please. So, the scope of the work is to define in the context of ASCII and French SLD in Saint-Jean.

You see an example like francais.quebec, you have a citadel there. So, this is what we have right now. We have the First Nations are using a kind of equally English or French to have a domain name, but they don't have their own language. So, for instance you don't have naskapi.ca or .quebec or eastern tree or whatever name it would be, Kariki or Inuktitut or Inuit or whatever. So, this is what we're working on.

So, the syllabic and Latin working group steps. It's going to be documenting local language repertoire, we think, and validating the property value of the proposed LGR. The LGR that was existing and exploited right now is, we can say, outdated, not good, no more. Then, of course, we have to attest to contemporary use because there's no new words that appeared in their vocabulary that was not encoded so appropriately.

So, we've been defining in script variants, cross-script variants, the rules. We've been finalizing a proposal, and Madam right here has been very helpful on the XML, HTML production. We are ready to go to public comment and integration. This would be the next steps.

So, next slide, please. So, the syllabic script languages. Inuktitut is covering seven sub-languages, dialects, or whatever you want to call them. It's been published in 2025 by the group. Syllabic Script LGR is then completed, and it's now even including Inuktitut, Cree,

Ojibway. So it's 18 sub-language set. Public comments are expected in March, April 2026. And for Dene and Dakelh LGR, well, it's expected a little bit later this year.

So, the last Latin script languages, the update Latin script is completed, including Nunavut and Labrador roman Inuktitut. We're talking here about Inuit languages. And I just wanted to tell you that we are arriving to Ontario. Ontario is completed. So this is, it's been completed yesterday. Remaining LGR syllabic script, there's one language and the Latin script, 48 languages. So, the work is not finished.

Next slide, please. I will take the ball that you threw and maybe give an example with what we've been working on. So, current implementation aims to mitigate the following risks. So, homoglyph, phishing, spoofing, trademark, and we could add a couple of things there.

The top-level security implementation is, we're talking here, of course, of second-level words, we're not talking about first-level extension in any way. Example of Latin scripts, you have le.quebec, LE, and in French, you have that possibility of all the diacritic signs. So, there is, of course, in any name that you will take, there's two possibilities. You can have a name that's going to be ASCII and the other one is going to be French with an accent. It can be some accent, it can be all of them, so the other ones that are still diacritic signs have to be blocked.

They can't be used, and they can't be combined. So, you need a tool that's going to be able to do that appropriately. The result is less room for abuse within the geoTLD scope, because, of course, we must say that some of the words that exist could have a diacritic that could be allowed and that you don't want to allow because it would throw confusion in the ring. So, of course, the public interest is secured DNS. Next one.

We are now in the same phase, the end of our run, and the result, we hope, is going to be a more open Internet. So, we're going to have a meeting in the next weeks, and we hope that the result of that work will mean that an internet for more people, an internet for the Inuit and the First Nation in Canada. I think that they don't have an inaccurate mode now. And as you know, well, the internet is for everyone. And as they say, one world and one internet. Thank you.

PITINAN KOOARMORNPATANA Thank you, Louis. Thank you for providing us the update. Then next, I would like to invite Gheafany from Indonesia, who will be speaking remotely to update on the reference of JFR Indonesian script. Ghea, over to you.

GHEAFANY WIDYATIKA PUTRI Thank you very much, Pitinan. Hi, everyone. I'm Ghea from PENDI, the organization that manages indonesians.id domain. So, today I'll share a quick community update on Indonesian scripts

especially on Balinese, Javanese and Pagan, and how we are helping these scripts become usable in domain names. Next slide, please. Let me start with a simple idea.

A domain name is your digital address. If your digital address can only be written in one script, many people feel the internet is not speaking the language. Indonesia has living local scripts, they are not museum artifacts, they are used in schools, cultural activities and communities. So, IDNs matter because they allow people to use their accounts online.

But here's the key, if we allow and see, think about it like a straight sign, if two signs look almost the same, people can get lost or worse, betrayed. So, our mission is to enable impulsions while keeping users protected. That's exactly where reference LGR work comes in.

Next slide, please. At a high level, a reference LGR answers three practical questions for a script as Pitinan has already explained before. The first one is which characters can be used and then which sequences are followed, what looks right and works in practice, and then which cases can be confusing. And therefore, they need safeguards including variants. The key point, this is not done by one organization alone, but it's community driven. So, our method is we organize FGDs with community stakeholders such as experts, practitioners, academics, and community representatives, and we capture real writing convention and how people will actually write what is acceptable and what is not.

And then we have something that really helps the process move. We call it a community champion. This is a trusted focal person and after that, a respected practitioner or academic who plays two roles. The first one is the mediator, and when the different viewpoints appear, and the second one is the main editor who keeps the document consistent across iteration, find itself, coordinates and documents outcomes and liaises with ICANN and ICANN review alignment with the LGR framework and guide the publication steps.

So, the workflow is a community practice and then documented rules and an iterative review, and then the last one is publication types. And actually, without the champion, you get 30 versions of my opinion. With the champion, you get one version of our consensus.

Next slide, please. Now, let's move to progress quickly by script. The first one is Balinese script. I can say that the Balinese is an important milestone for Indonesia because this is the first Indonesian local script to reach the milestone of an ICANN published second-level reference LGR in November 2024. And the SLD was launched with the governor of Bali on December 11, 2025.

And the second one is Javanese. For Javanese, we also have strong momentum. ICANN visited Yogyakarta, especially Pitinan last year, to observe real usage directly with local stakeholders, and the Javanese reference LGR has been already submitted. It's currently in finalization with ICANN toward publication leading to the public

comment stage. I think this is a good example why community engagement matters. We are not building rules in isolation and we are aligning them with how the script is actually used.

And the last one is Pegon script. For Pegon, the reference algebra has been submitted and we are awaiting ICANN feedback. Pegon has active use across communities, so the next step is to incorporate review feedback and iterate with stakeholders to keep the rules more practical and safer. Next slide, please.

Here is a small showcase of how we balance security and insurability using a simple example. Okay, dominants are often read quickly. Either check message on a small screen or on printed materials, so visual confusion is a real issue. In Javanese script, some digits can resemble letters. So, our working group took a conservative approach. We exclude digits, but ASCII digits and Javanese digits, to reduce confusion and potential misuse. One illustration, as you can see, the Javanese digit 1 can look similar to the Javanese letter go. If users can reliably tell the difference, it creates such mistakes, misdeliver, or impersonation.

Now, does this reduce usability? Potentially, yes. So we provide a usability friendly background. Write number as words in labels. So, if you want to create a domain like ICANN1, you can write it as ICANONE, such as like that. And this is how we balance. We started with a way to save baseline to users, and then we kept labels practical by aligning with real writing practices, and revised over

time as we learned from community feedback and actual use. Next slide, please.

To close, what's next? For Javanese, we complete realization with Toa publication, and public comment. For the Pegon, we can iterate based on ICANN feedback. And the Balinese, we continue strengthening adoption with practical guidance and outreach. And overall, actually, we want a sustainability process so more Indonesian scripts can follow when the community is ready, such as Batak, Sundanese or Buginese. I think that's all. Thank you, Pitinan. Back to you.

PITINAN KOOARMORNPATANA: Thank you, Ghea. And I think that bring us to the end of our presenters. So, now, I would like to open the floor for any question or comment. If you are speaking, please state your name and affiliation for the record. Thank you.

DIPAK PARMAR: My name is Dipak Parmar for a record. I have a basic question on how the end user are they typing a domain name or they are doing by voice, because it is a huge difference of input level if I am saying something, say my English will have Indianized English and other country will be European, so same later, what we are disputing about two way of a code, how it will be coming as a input, because our mother tongue is always somewhere in a tongue. So, if it's a

voice-based input for a domain, then we need to really relook everything.

LOUIS HOULE

ICANN speak a little bit from our experience with the First Nation and Inuit languages. We have people on the working group who are specialists of keyboards, and they've been helping us to clean all the problems that would still exist in the current time with some of the keyboards or use of some characters that are obsolete or not current, not in use anymore. So, those persons, I'm not talking about only one person, but we're talking about the authority of keyboards in the world, so they don't make only UCAS keyboards.

DIPAK PARMAR

Just a little follow-up on that. Our experts are expert, but end users are not expert.

PITINAN KOOARMORNPATANA

Maybe I add a little bit to your question on whether the voice as an input, how would it be? So, this project because is domain names, basically, currently by the procedure of how to develop this go by the visual base. So, after, it's actually input is the input either from the keyboard or when you speak is also eventually turned into some text.

So, at the moment, the rule is based on the visual and in some of the community, maybe when it's clear rules on what is the variant semantically, it could be defined like that as well. But obviously,

we'll not expand to the scope of voice or accent at the moment, but thank you for the input. That is something on the input level which is maybe not covered by the LGR. Thank you.

MOHAMMAD ZULQAR NAYEN: Hi, my name is Zulqar. I'm with ISOC Ontario. I have a quick question for Louis. So, the work that you've been doing with Indigenous Canadian syllabus for internet, I'm just curious, like, are the Indigenous community, are they driving the demand for this? Or is it something you're building, your team is building now, and they will later join? What is the situation?

LOUIS HOULE: Both. It depends on the language. It depends on the region. Because our intent was very simple and limited. We just wanted to be able to offer, with the registry .quebec, the languages that were spoken in Québec. So, we started with Inuit. Among the seven dialects, there's only two spoken in Québec.

If I take the other languages, again, you have Cree, [00:53:21 – inaudible], period. But the problem is that in Canada, as you know, the First Nation, as you saw on my first slide, you see circles, and this is Quebec. The circle for Inuit, it's more than Quebec.

And if you take any of the languages that we've been treating, the concept of Quebec, Ontario, Maritimes, British Columbia, whatever, you can't work on that plan like that because it won't apply. So, actually, we kind of just discovered something that was

obvious to many, but we were naive. If you start it, you want to finish it.

MOHAMMAD ZULQAR NAYEN Thank you.

PITINAN KOOARMORNPATANA Yes, please.

LANCE HINDS Thank you, Chair. Lance Hinds, Chair of LACRALO, and in my other hat, an IT NGO, actually looking to see how much we can use technologies to advance language preservation and cultural preservation in Guyana, in the Caribbean. Regarding the First Nation consultations, I'm curious about what happens on the ground, because, for example, we don't have that skill set.

So, what we've had to do because of lack of documentation sometimes is actually crawl through the language to identify the diacritics in this instance. So, you run into clashes. So, for example, you would have like what looks like a right single quotation mark, as opposed to a modifier letter, apostrophe, or what is actually a glottal stop. And I'm curious in your experience, where with that -- So, the rule says that you should go back to the community, and they're the ones who are going to decide which is the best one.

Now, what ends up happening is that if you say, well, ICANN's preferred form for linguistic apostrophe, somebody says, well, go

with the best practice and go use that. But then somebody may object on the end and say, well, you can't do that on your own. So, I'm just wondering with the experience that you have with when you run into clashes, how you deal with it.

PITINAN KOOARMORNPATANA Thank you. And it's a very valid question. And hopefully we have a good example of that. So, if we go back to the very basics on how to enable all this, the first step is to make sure that all these labels are IDNA2008 compliance.

So if we go back to that, and many of these single quotations or the exclamation in some languages, which is used as a click sound or nasal sound, is actually in the disallow property, because it's also used in other syntax in the protocol. So that's the state of the status at the moment. Some of the character is really not in the PVALID property. But that's also something I think Louis can share of what he found and what he's going to do.

LOUIS HOULE Yes, and we've been playing with PVALID so many times during that exercise. I just want to add that as far as I'm concerned, I'm not a specialist in any of those languages. My specialty is in French. But to tell you, we have people on that group who can manage to answer those issues. We have people on the group.

Bill, for instance, published in 2025, the Anastasia P. Dictionary. So, we presume that he knows what he's talking about, and we've

been systematically relating with the community for any of the issues, questions that we would have on any of the PVALID or not valid.

All of the questions about the use, is that taught at school? Is that in use commonly? And we're not the ones who decided of A or B. The community did, actually. At the end of the day, this is what happened.

PITINAN KOOARMORNPATANA Thank you, Lance. Thank you for the question. And we have one hand raised. Bill, would you like to unmute your mic?

BILL JOURIS This is Bill Jouris for the record. Part of the problem we see, I think, stems from the fact that people in the IDN project, and even more, the people in the IETF who wrote the standards for all of this stuff, none of them were native speakers of a language which includes clicks or glottal stops, and therefore, they were pretty cavalier in deciding that they just would ignore all of those.

The nominal reason is they look like punctuation marks or something, but the reality is that could have been worked around except that the people writing the rules didn't think it was significant. So, that's why the apostrophe as a glottal stop, for example, is disallowed. The people who wrote the rules just didn't care about that. That may be doing them a disservice, but that's certainly how it looks from where I sit. Thank you.

PITINAN KOOARMORNPATANA Thank you, Bill, and thank you for the comment. So, just to add on that, actually, there are also more things to not be allowed as well. For example, in some of the script, they might need zero with non-joiner to actually use in the everyday language to combine the two letters. But because it's invisible, it's not allowed in the top-level domain, for example.

Because this is also the shared resource, that's why by the design, we based on the standard that developed by IETF. And I guess the motivation is to keep it secured. Obviously, if there are demand to use this or the workaround, we would like to start some conversation and see how we can find a solution together. But thank you for the input, Bill. I appreciate your input.

CHAMPIKA WIJAYATUNGA Maybe I think we have reached the time, and there are no questions in the Q&A pod as well. So, I think we can close the session. Thank you very much for all the speakers and everyone who participated in this session. Thank you, and we can stop the recording. Thank you.

PITINAN KOOARMORNPATANA Thank you, everyone. Thank you to all the speakers. Thank you for joining us.

[END OF TRANSCRIPTION]