Label Generation Ruleset Specification
draft-ietf-lager-specification-00

Kim Davies Asmus Freytag

The quick summary

- A standardised approach to expressing registration rules for labels, known as “label generation rulesets”, or “LGRs”.
- Should allow all existing IDN tables, and known registry policies, to be reproduced in an objective machine readable format.
- Vision is anyone dealing with registry policies can implement an LGR runtime, and then not have to worry about hardcoding complex validation rules.
- Is not specific to IDNs!

Why? (Part 1)

- Current IDN implementors tend to use “IDN tables” to define which code points are allowed in domain labels.
- Many registries implementing contextual logic in their registry backends in a proprietary manner.
- Therefore, even for those that publish IDN tables, it is difficult to be sure you can replicate/reuse that logic.

Why? (Part 2)

- IDN tables are intended to be shared, e.g. “IDN repository” of registry IDN tables on IANA website
- Lacks consistent format and difficult to repurpose registry tables.
- RFC 3743 and RFC 4290 are not rich enough to express most registry policies.
- New gTLD Program has struggled with this too.
- Having a common format would greatly aid in re-use, validation for table format, etc.

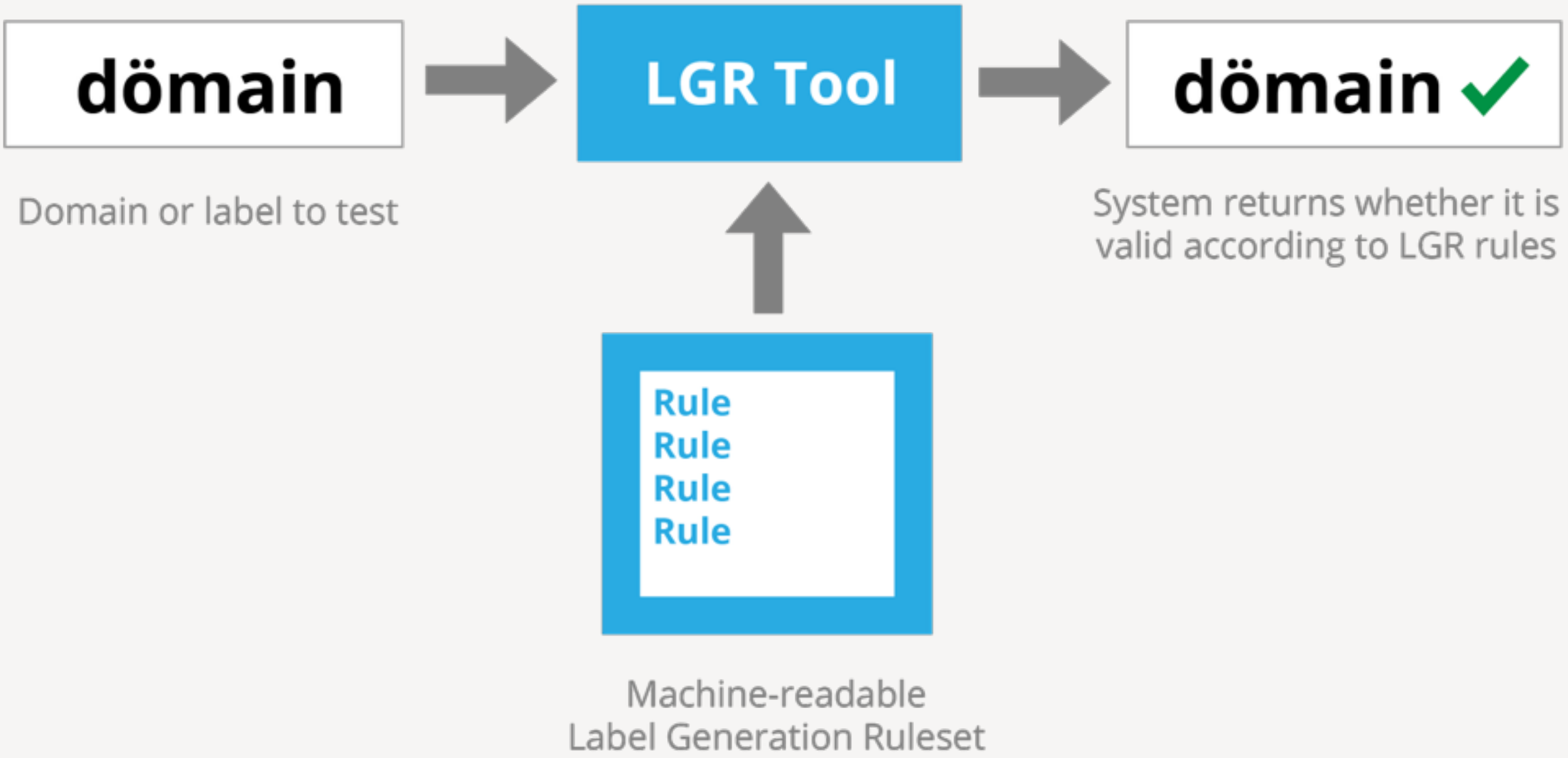
Why? (Part 3)

- Variant program for TLDs needs to use something to base its work on.
- Goal is to have a master “root LGR” which is the unison of various language and script specific rulesets.
- Therefore the ability re-use, adapt, merge from LGRs is a requirement.

The specification

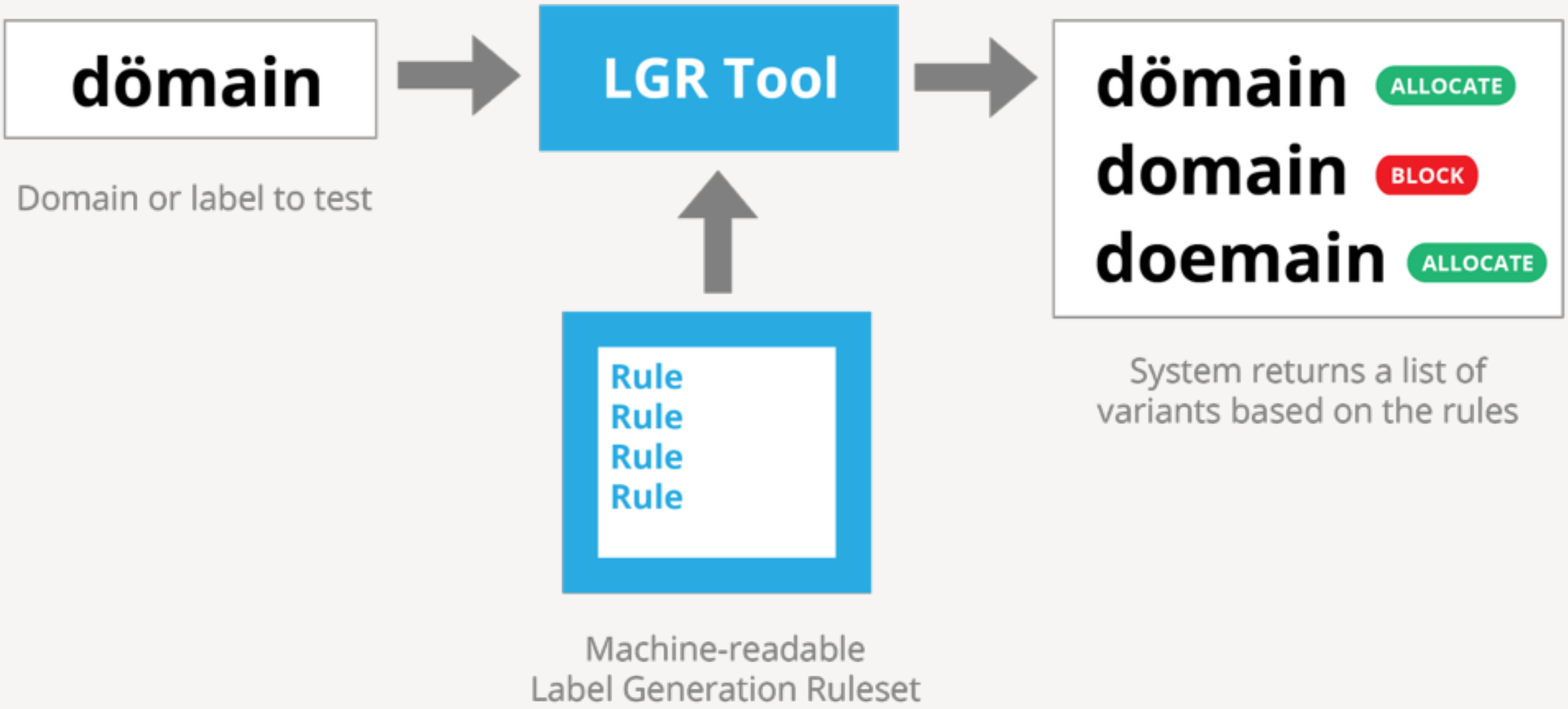
draft-davies-idntables

- XML based description of registry policies for “label generation”
- Allowable code points for domain registries, contextual rules, dispositions, etc.
- Co-authored by me and Asmus Freytag (Unicode Consortium); strong input from IDN variant project participants



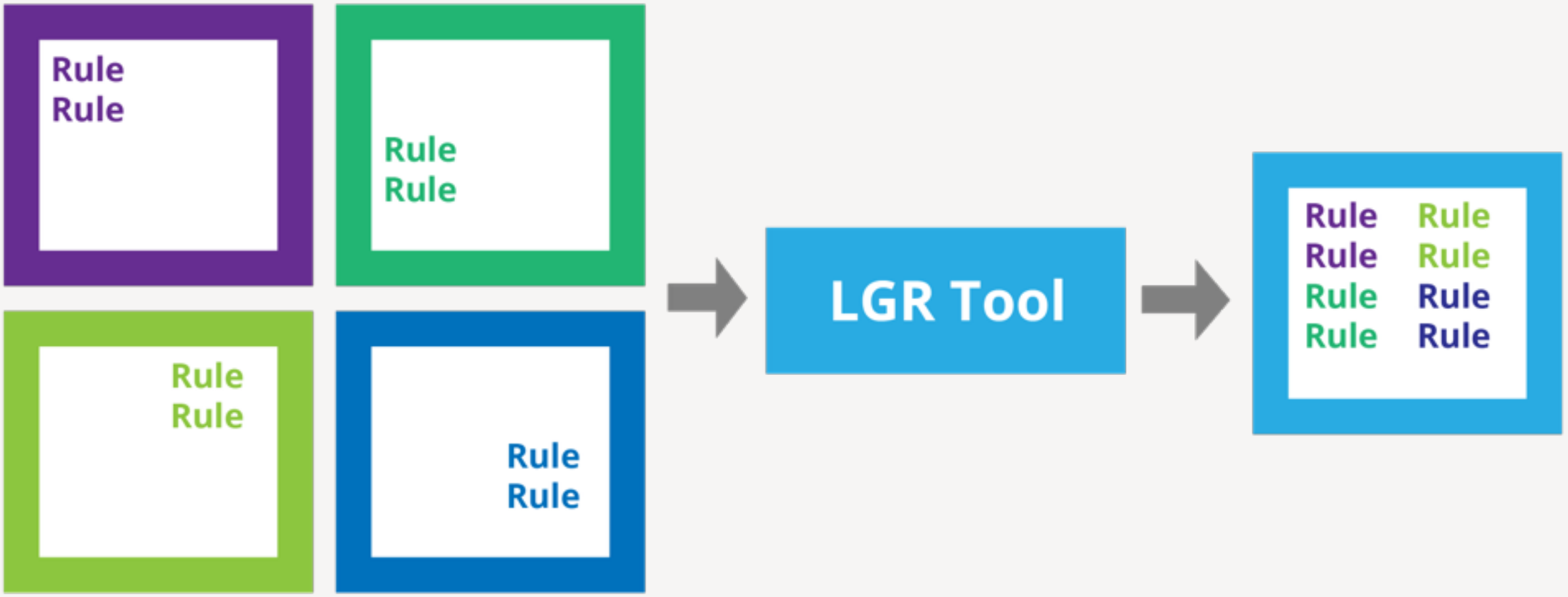
Simple validation checking

Provide label, respond whether that label accords with the LGR's rules.



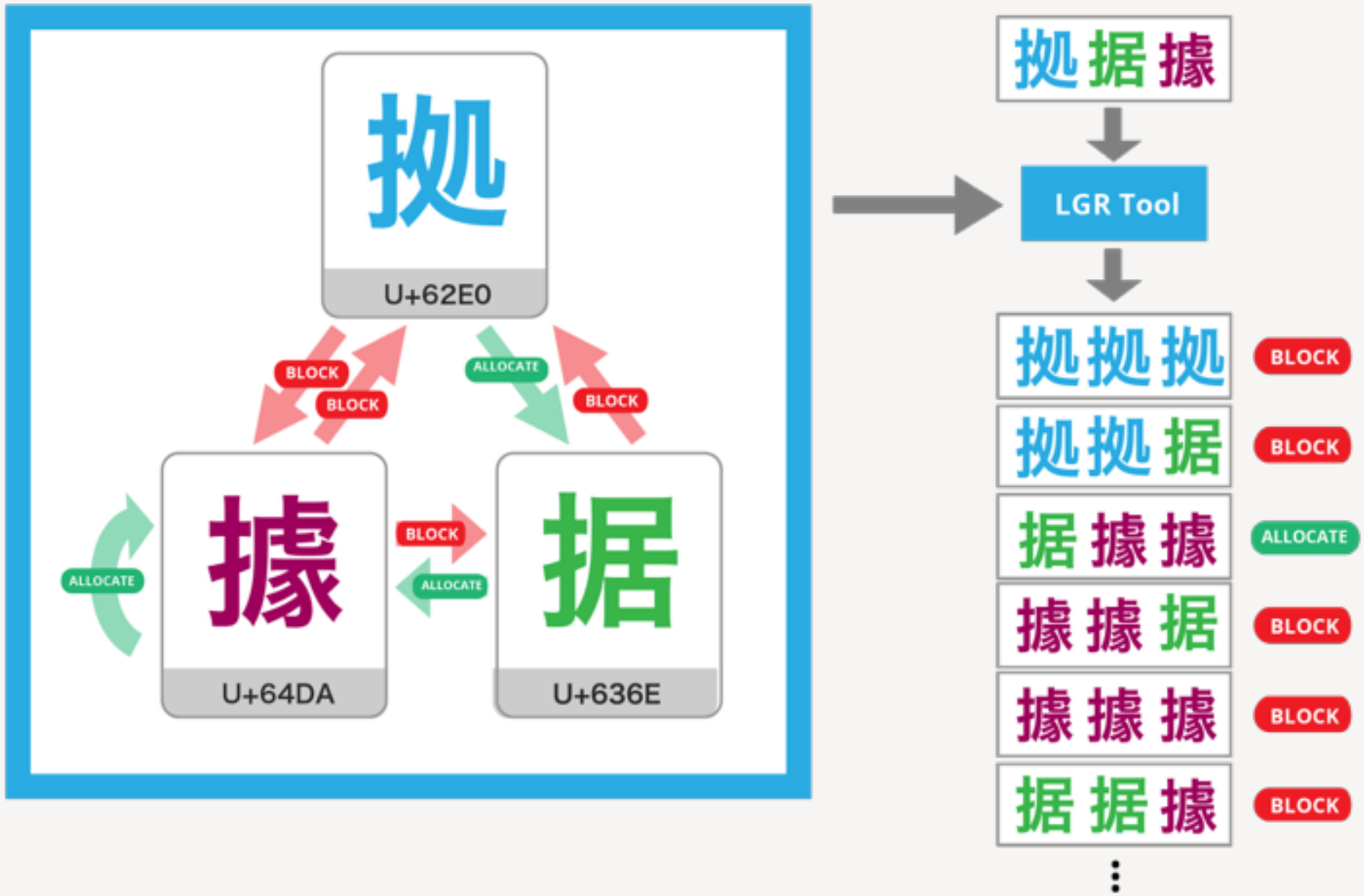
Variant label generation

Take an input label, generate permutations along with actions to take.



LGRs can be merged

LGRs can be diffed



LGRs can represent complex interdependencies

LGRs have…

- Code point lists, with tagging classes and dispositions
- Variants for specific codepoints, variants are 0..n codepoints. Variants can be conditional by meeting certain tests.
- Whole label evaluation rules, allowing approaches based on regex like concepts.
- Ways to leverage all the Unicode properties, to diminish the need to be derivative.
- Metadata in standard format
- A clear schema so table validity can be automatically checked.

```
<?xml version="1.0" encoding="utf-8"?>
<lgr xmlns="urn:ietf:params:xml:ns:lgr-1.0">
<data>
   <char cp="002D" comment="HYPHEN (-)" />
   <range first-cp="0030" last-cp="0039" comment="0-9" />
   <range first-cp="0061" last-cp="007A" comment="Latin small letter A-Z" />
</data>
</lgr>
```

HelloWorld.lgr

Minimal LGR to permit standard LDH labels

```
<data>
   <char cp="200D" when=“joiner” />
</data>
<rules>
   <class name=“virama” property=“ccc:9” />
   <rule name=“joiner”>
       <look-behind>
           <class by-ref=“virama” />
       </look-behind>
       <anchor />
   </rule>
</rules>
```

Contextual rules; Derived properties; Regular Expressions

This allows zero-width joiner (U+200D) when following a virama (implements IDNA context rule)

Current status

- Is the standard format for ICANN's IDN variant project.
  - Community "generation panels" are developing their language/script rules using the draft format. "Integration panel" will merge these inputs into a common Root LGR.
- Know of some working on implementations, and others planned
- All known issues either in IETF issue tracker or discussed recently on lager mailing list

Thank you.

- kim.davies@icann.org
- asmus@unicode.org