CoRE: Problem Details, CURIE

June 8th (Wednesday), 14:00-15:30 UTC

draft-ietf-core-problem-details
Disposition of first IETF LC comments

draft-ietf-core-href, draft-ietf-cbor-packed
CURIEs and CRIs

branch `ad-review` (merged)

Issue #24: (e) be more explicit about diagnostic notation
Issue #25: (e) explain made-up values in example
Issue #26: (e) add more detail to self-reference

Issue #21: Meaning of problem title

The title (key -1): A short, human-readable summary of the problem shape. It SHOULD NOT change from occurrence to occurrence of the same problem shape.

- Remnant of RFC 7807, where the title was a description of the problem type.
- concept "stable across instances" copied from RFC 7807
  - made sense when title was basically a human readable version of type
- 7807 problem type now subsumed by problem shape

- ➔ "A short, human-readable summary of the problem shape."
- + "SHOULD not try to summarize the information given with the problem details"?
  - i.e., the summary might include that the account does not have enough money, but not how much money it has and how much would be needed.

Issue #22: Untagged text strings for human readable text

Both "title" and "detail" can use either an unadorned CBOR text string (text) or a language-tagged text string (tag38); see Appendix A for the definition of the latter.

— This is always human-readable text, so what would unadorned mean?

— ➔ (1) make "en" the default?
  — making "foo" a shorthand for 38(["en", "foo", false])
— ➔ (2) make the default depend on context?
— ➔ (3) no unadorned text allowed?

Issue #23: More examples in Section 3.1 (SPDe)

"It would be nice with an example in Section 3.1 as well."

- Problem with examples that show fictional registered values:
  - people start using value from the example
  - → threshold of usefulness before one includes an example that is not actually registered
- Five "examples" are the defined Standard Problem details entries

- **→ wontfix**

PR #27: "ignore-unknown" issues (in progress)

OPSDIR review (Joel Jaeggli):
"ignore-unknown" only addresses consumer behavior
What if item with unknown entries is stored/forwarded?

- Add: RECOMMENDED to retain unknown for store/forward
- Exceptions to this SHOULD:
  - storing/forwarding in different format (conversion needed)
  - filtering forwarder (avoiding undesired disclosures)
    - if filter doesn't know what it is, can't forward

Next steps with draft-ietf-core-problem-details

- telechat date: ??? (2022-06-16???)
- work needed:
  - some reviews will be JIT (2022-06-10): GENART, ARTART, I18NDIR
  - process AD DISCUSS/COMMENT positions
- could be approved ~ end of June

CURIE and CRI

CBOR-packed vs. CURIEs in CRI

Without function tags:

— Easy to do prefix:
`coaps://coap.me` ➔ `[-2, ["coap", "me"]]`
Usage: `coaps://coap.me/foo/bar` ➔ `` `225([["foo","bar"]]) ``

— Harder to do prefix:
`coaps://coap.me/foo/` ➔ `[-2, ["coap", "me"], ["foo"]]`
`coaps://coap.me/foo/bar` ➔ `` `225([[???,"bar"]]) ``

CURIE function tag

— Use in CBOR packed argument table to indicate CURIE processing instead of simple concatenation

```
coaps://coap.me/foo/ ➔
CURIE1([-2, ["coap", "me"], ["foo"]])
...
coaps://coap.me/foo/bar ➔
225([0, ["bar"]])
```

— Problem: CURIE semantics are based on URI syntax

weird CURIE cases

```
'ht' + 'tp://coap.me/foo/
bar'
'http://coa' + 'p.me/foo/bar'
'http://coap.me' + '/foo/bar'
'http://coap.me/' + 'foo/
bar'
'http://coap.me/foo' +
'#bar'
'http://coap.me/foo#' +
'bar'
```

It is always possible to express as CRI:

— left hand side

— right hand side

without knowing the other?

function tag processing semantics

Most general:

— take inputs and convert back to URI
— concatenate
— convert back to CRI

what is the benefit of using CRIs then?

Most useful:

— operate semantically on CRI + CRI reference
— find spot where rhs latches into lhs
— convert seemingly path rhs to hostname, fragment ID?

'http://coap.me/foo#' + 'bar'
'http://coa' + 'p.me/foo/bar'

subsetting CURIE?

Is there a CURIE subset where all this does make sense?

➔ develop corpus of CURIEs to look at
derive meaningful subset from those

What to do if CURIE is outside subset?

don't despair just yet

There are lots of "sane" CURIEs.

```
"namespace": {
  "foo": "https://example.com/"
}

...: { "sdfRef": "foo:#/sdfData/temperatureData" }
```

Both sane and with obvious CRIs:
LHS `https://example.com`
RHS `#/sdfData/temperatureData`

is there a better CURIE?

CURIE solves a widely appreciated problem

Solution is lexical (~ URI)
• lexical solutions always cause problems

Can there be a solution that is structural (~ CRI)?
• Can this be backported (made understood) to URI space?

Can structural solution be our subset of lexical CURIEs?