Workgroup: Network Working Group
Internet-Draft:
draft-bormann-cbor-cddl-freezer-13
Published: 28 February 2024
Intended Status: Informational
Expires: 31 August 2024
Authors: C. Bormann
Universität Bremen TZI

A feature freezer for the Concise Data Definition Language (CDDL)

Abstract

In defining the Concise Data Definition Language (CDDL), some features have turned up that would be nice to have. In the interest of completing this specification in a timely manner, the present document was started to collect nice-to-have features that did not make it into the first RFC for CDDL, RFC 8610, or the specifications exercising its extension points, such as RFC 9165.

Significant parts of this draft have now moved over to the CDDL 2.0 project, described in draft-bormann-cbor-cddl-2-draft. The remaining items in this draft are not directly related to the CDDL 2.0 effort.

Table of Contents

1. Introduction

In defining the Concise Data Definition Language (CDDL), some features have turned up that would be nice to have. In the interest of completing this specification in a timely manner, the present document was started to collect nice-to-have features that did not make it into the first RFC for CDDL [RFC8610], or the specifications exercising its extension points, such as [RFC9165].

Significant parts of this draft have now moved over to [I-D.bormann-cbor-cddl-2-draft], which in turn references [I-D.ietf-cbor-update-8610-grammar], [I-D.ietf-cbor-cddl-more-control], [I-D.ietf-cbor-cddl-modules]. The remaining items in Sections 3 to 4 of this draft are not directly related to the CDDL 2.0 effort. Section 5 might turn into a part of CDDL 2.5.

The remaining sections are not proposing to change CDDL, but are ancillary developments: Section 6 is more interesting for the ecosystem of tools around CDDL. Section 7 examines extending the area of application of CDDL beyond CBOR and JSON.

There is always a danger for a document like this to become a shopping list; the intention is to develop this document further based on the rapidly growing real-world experience with the first CDDL standard. Some sections are labeled WONTFIX, reflecting an assumption that the specific extension objective will not be addressed or will be addressed in a different way.

2. Base language features

2.1. Cuts

Section 3.5.4 of [RFC8610] alludes to a new language feature, *cuts*, and defines it in a fashion that is rather focused on a single application in the context of maps and generating better diagnostic information about them.

The present document is expected to grow a more complete definition of cuts, with the expectation that it will be upwards-compatible to the existing one in [RFC8610], before this possibly becomes a mainline language feature in a future version of CDDL.

3. Literal syntax

Literal syntax is also discussed in Appendix A.1 of [I-D.bormann-cbor-cddl-2-draft], which might provide another approach to Section 3.1. This appendix is in turn based on ideas in [I-D.ietf-cbor-edn-literals].

3.1. Regular Expression Literals (WONTFIX)

Regular expressions currently are notated as strings in CDDL, with all the string escaping rules applied once. It might be convenient to have a more conventional literal format for regular expressions, possibly also providing a place to add modifiers such as /i. This might also imply text .regexp ..., which with the proposal in Section 4.1 then raises the question of how to indicate the regular expression flavor.

(With the support for ABNF in [RFC9165], the need for this is reduced. Also, the proliferation of regular expression flavors is hard to address with a single syntax.)

4. Controls

Controls are the main extension point of the CDDL language. It is relatively painless to add controls to CDDL; this mechanism has been exercised in [RFC9090] for SDNV [RFC6256] and ASN.1 OID related byte strings, and in [RFC9165] for more generally applicable controls, including an interface to ABNF [RFC5234] [RFC7405]. A more recent collection of additions that is ready for standardization is specified in [I-D.ietf-cbor-cddl-more-control].

Several further candidates have been identified that aren't quite ready for adoption, of which a few shall be listed here.

4.1. Control operator .pcre

There are many variants of regular expression languages. Section 3.8.3 of [RFC8610] defines the .regexp control, which is based on XSD [XSD2] regular expressions. As discussed in that section, the most desirable form of regular expressions in many cases is the family called "Perl-Compatible Regular Expressions" ([PCRE]); however, no formally stable definition of PCRE is available at this time for normatively referencing it from an RFC.

The present document defines the control operator .pcre, which is similar to .regexp, but uses PCRE2 regular expressions. More specifically, a .pcre control indicates that the text string given as a target needs to match the PCRE regular expression given as a value in the control type, where that regular expression is anchored on both sides. (If anchoring is not desired for a side, .* needs to be inserted there.)

Similarly, .es2018re could be defined for ECMAscript 2018 regular expressions with anchors added.

See also [RFC9485], which could be specifically called out via .iregexp (even though .regexp as per Section 3.8.3 of [RFC8610] would also have the same semantics, except for a wider range of regexps).

4.2. Endianness in .bits

How useful would it be to have another variant of .bits that counts bits like in RFC box notation? (Or at least per-byte? 32-bit words don't always perfectly mesh with byte strings.)

4.3. .bitfield control

Provide a way to specify bitfields in byte strings and uints to a higher level of detail than is possible with .bits. Strawman:

```
Field = uint .bitfield Fieldbits

Fieldbits = [
  flag1: [1, bool],
  val: [4, Vals],
  flag2: [1, bool],
]

Vals = &(A: 0, B: 1, C: 2, D: 3)
```

Note that the group within the controlling array can have choices, enabling the whole power of a context-free grammar (but not much more).

5. Co-occurrence Constraints

While there are no co-occurrence constraints in CDDL, many actual use cases can be addressed by using the fact that a group is a grammar:

```
postal = {
  ( street: text,
    housenumber: text) //
  ( pobox: text .regexp "[0-9]+" )
}
```

However, constraints that are not just structural/tree-based but are predicates combining parts of the structure cannot be expressed:

```
session = {
  timeout: uint,
}

other-session = {
  timeout: uint  .lt [somehow refer to session.timeout],
}
```

As a minimum, this requires the ability to reach over to other parts of the tree in a control. Compare JSON Pointer [RFC6901] and JSON Relative Pointer [I-D.handrews-relative-json-pointer], as well as Stefan Gössner's jsonpath, a JSON analogue of XPath that has recently been standardized [RFC9535].

More generally, something akin to what Schematron is to Relax-NG may be needed.

6. Alternative Representations

For CDDL, alternative representations e.g. in JSON (and thus in YAML) could be defined, similar to the way YANG defines an XML-based serialization called YIN in Section 11 of [RFC6020]. One proposal for such a syntax is provided by the cddlc tool [cddlc], which is reproduced below. This could be written up in more detail and agreed upon. (Since cddlc version 0.1.8, the "mem"-labeled array includes information about the presence of a cut, see Section 3.5.4 of [RFC8610].)

```
cddlj = ["cddl", +rule]
rule = ["=" / "/=" / "//=", namep, type]
namep = ["name", id] / ["gen", id, +id]
id = text .regexp "[A-Za-z@_$](([-.])*[A-Za-z0-9@_$])*"
op = ".." / "..." /
  text .regexp "\\.[A-Za-z@_$](([-.])*[A-Za-z0-9@_$])*"
namea = ["name", id] / ["gen", id, +type]
type = value / namea / ["op", op, type, type] /
  ["map", group] / ["ary", group] / ["tcho", 2*type] /
  ["unwrap", namea] / ["enum", group / namea] /
  ["prim", ?((6, type/uint, ?type) // (0..7, ?uint))]
group = ["mem", bool, null/type, type] /
  ["rep", uint, uint/false, group] /
  ["seq", 2*group] / ["gcho", 2*group]
value = ["number"/"text"/"bytes", text]
```

The "prim"-labeled array includes support for non-literal tag numbers (Section 3.2 of [I-D.ietf-cbor-update-8610-grammar]).

More recently, a variant of this format has been used for easier processing. It collects rules in a map (JSON object) and binds generic parameters to argument positions. This variant will be described in a further revision of this document.

7. Other target formats

CDDL has originally been designed to describe CBOR and JSON data. One format of interest is comma-separated values, CSV [RFC4180]. [I-D.bormann-cbor-cddl-csv] is a draft for using CDDL models with CSV.

8. IANA Considerations

This document makes no requests of IANA.

9. Security considerations

The security considerations of [RFC8610] apply.

10. References

10.1. Normative References

[RFC8610]
Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[RFC9165] Bormann, C., "Additional Control Operators for the Concise Data Definition Language (CDDL)", RFC 9165, DOI 10.17487/RFC9165, December 2021, <https://www.rfc-editor.org/rfc/rfc9165>.

10.2. Informative References

[cddlc] "CDDL conversion utilities", n.d., <https://github.com/cabo/cddlc>.

[I-D.bormann-cbor-cddl-2-draft]
Bormann, C., "CDDL 2.0 and beyond -- a draft plan", Work in Progress, Internet-Draft, draft-bormann-cbor-cddl-2-draft-04, 27 February 2024, <https://datatracker.ietf.org/doc/html/draft-bormann-cbor-cddl-2-draft-04>.

[I-D.bormann-cbor-cddl-csv] Bormann, C. and H. Birkholz, "Using CDDL for CSVs", Work in Progress, Internet-Draft, draft-bormann-cbor-cddl-csv-04, 24 December 2023, <https://datatracker.ietf.org/doc/html/draft-bormann-cbor-cddl-csv-04>.

[I-D.handrews-relative-json-pointer] Luff, G. and H. Andrews, "Relative JSON Pointers", Work in Progress, Internet-Draft, draft-handrews-relative-json-pointer-02, 18 September 2019, <https://datatracker.ietf.org/doc/html/draft-handrews-relative-json-pointer-02>.

[I-D.ietf-cbor-cddl-modules]
Bormann, C., "CDDL Module Structure", Work in Progress, Internet-Draft, draft-ietf-cbor-cddl-modules-01, 18 December 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-cddl-modules-01>.

[I-D.ietf-cbor-cddl-more-control]
Bormann, C., "More Control Operators for CDDL", Work in Progress, Internet-Draft, draft-ietf-cbor-cddl-more-control-03, 26 February 2024, <https://

datatracker.ietf.org/doc/html/draft-ietf-cbor-cddl-more-control-03>.

[I-D.ietf-cbor-edn-literals]
Bormann, C., "CBOR Extended Diagnostic Notation (EDN): Application-Oriented Literals, ABNF, and Media Type", Work in Progress, Internet-Draft, draft-ietf-cbor-edn-literals-08, 1 February 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-edn-literals-08>.

[I-D.ietf-cbor-update-8610-grammar]
Bormann, C., "Updates to the CDDL grammar of RFC 8610", Work in Progress, Internet-Draft, draft-ietf-cbor-update-8610-grammar-03, 29 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-update-8610-grammar-03>.

[PCRE] "Perl-compatible Regular Expressions (revised API: PCRE2)", n.d., <http://pcre.org/current/doc/html/>.

[RFC4180] Shafranovich, Y., "Common Format and MIME Type for Comma-Separated Values (CSV) Files", RFC 4180, DOI 10.17487/RFC4180, October 2005, <https://www.rfc-editor.org/rfc/rfc4180>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/rfc/rfc6020>.

[RFC6256] Eddy, W. and E. Davies, "Using Self-Delimiting Numeric Values in Protocols", RFC 6256, DOI 10.17487/RFC6256, May 2011, <https://www.rfc-editor.org/rfc/rfc6256>.

[RFC6901] Bryan, P., Ed., Zyp, K., and M. Nottingham, Ed., "JavaScript Object Notation (JSON) Pointer", RFC 6901, DOI 10.17487/RFC6901, April 2013, <https://www.rfc-editor.org/rfc/rfc6901>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/rfc/rfc7405>.

[RFC9090] Bormann, C., "Concise Binary Object Representation (CBOR) Tags for Object Identifiers", RFC 9090, DOI 10.17487/RFC9090, July 2021, <https://www.rfc-editor.org/rfc/rfc9090>.

[RFC9485] Bormann, C. and T. Bray, "I-Regexp: An Interoperable Regular Expression Format", RFC 9485, DOI 10.17487/RFC9485, October 2023, <https://www.rfc-editor.org/rfc/rfc9485>.

[RFC9535] Gössner, S., Ed., Normington, G., Ed., and C. Bormann, Ed., "JSONPath: Query Expressions for JSON", RFC 9535, DOI 10.17487/RFC9535, February 2024, <https://www.rfc-editor.org/rfc/rfc9535>.

[XSD2] Malhotra, A., Ed. and P. V. Biron, Ed., "XML Schema Part 2: Datatypes Second Edition", W3C REC REC-xmlschema-2-20041028, W3C REC-xmlschema-2-20041028, 28 October 2004, <https://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>.

Acknowledgements

Before [RFC8610] was finally published, many people have asked for CDDL to be completed, soon. These are usually also the people who have brought up observations that led to the proposals discussed here. Sean Leonard has campaigned for a regexp literal syntax.

Author's Address

Carsten Bormann
Universität Bremen TZI
Postfach 330440
D-28359 Bremen
Germany

Phone: +49-421-218-63921
Email: cabo@tzi.org