FDT, networking, IETF

40 years of Formal Description Techniques in open systems

Carsten Bormann, 2020-05-15
Some slides stolen from https://u.nu/cddl-tutorial

Carsten Bormann

Universität Bremen TZI

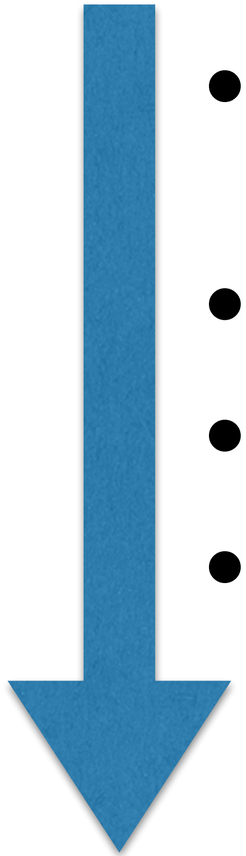
IETF ROHC, 6LoWPAN, CoRE WG

IRTF T2T RG

http://slides.cabo.space

RFC 2429	RFC 2509	RFC 2686	RFC 2687	RFC 2689	RFC 3095	RFC 3189
RFC 3190	RFC 3241	RFC 3320	RFC 3485	RFC 3544	RFC 3819	RFC 3940
RFC 3941	RFC 4629	RFC 5049	RFC 5401	RFC 5740	RFC 5856	RFC 5857
RFC 5858	RFC 6469	RFC 6606	RFC 6775	RFC 7049	RFC 7228	RFC 7252
RFC 7400	RFC 7959	RFC 8132	RFC 8138	RFC 8307	RFC 8323	RFC 8428
RFC 8610	RFC 8710	RFC 8742	RFC 8746	working	on	it…

Bringing the Internet to new applications



- "Application X will **never** run on the Internet"
- …
- …
- "How do we turn off the remaining parts of X that **still** aren't on the Internet"?

How I (used to) use FDT

- 1984: Studienarbeit (~ Bachelor Thesis):
  Compiling time point selection expressions (TPSE) into state logic,
  part of RSPL_Z, a protocol specification language
- 1982–, on and off: CLPT ➔ **SGML** (➔ XML); DSSSL (➔ XSL/XSLT)
- 1983–: X.409 ➔ **ASN.1** scars; ASN.1 PER helped kill **H.323** mid-1990s
- 2002–2007: helped with RFC 4997,
  Formal Notation for RObust Header Compression (**ROHC-FN**)
- 2011: Dagstuhl 11042: Improper use of FDT helped kill OSI
- 2013: RFC 7049 CBOR ➔ 2014–2019 RFC 8610 **CDDL** (see next slides)
- 2015: **T2TRG** ➔ 2016 IOTSI ➔ 2017 WISHI ➔ 2019 OneDM ➔ **SDF**

Data Definition Language

- JSON: “Goto” format for structured data interchange, largely replacing XML
- (Compare: ASCII ➔ UTF-8: “Goto” format for text interchange)
- “Let's use JSON as our interface format”
  ~ “Let's use ASCII as our programming language”
- Generic data model: All that can be said in JSON
- Specific data model: All that should be said in JSON for this application
- Do this (“data definition”) in English, or in a machine-readable way?

JSON and CBOR

- JSON: RFC 8259, **text-based** format for structured data interchange
- Generic data model:
  - Atomic data: false, true, null; numbers (decimal), text strings
  - Containers: arrays, maps (“objects”)
- CBOR: RFC 7049, **binary** format for structured data interchange
- Generic data model: like JSON, plus:
  - Byte strings (binary data); more well-defined number system
  - Extension mechanism (“tags”), with batteries included (e.g., date/time)
- “Diagnostic notation” (RFC 7049, 8610) — for literature, not for interchange

CDDL (RFC 8610)

Concise data definition language

- Background: Since 1977, text-based formats in IETF are described in BNF, specifically ABNF (“Augmented BNF”, RFC 5234)
  - Generative grammar notation, basics are familiar to most computer people

```
day-name = "Mon" / "Tue" / "Wed" / "Thu" / "Fri" / "Sat" / "Sun"
```

  - ABNF has slight idiosyncrasies, familiar to many IETF people
- Idea 2014/2015: Let’s do an ABNF-like grammar for structured data ➔ CDDL
  - Based on Bert Greevenbosch’s early CDDL proposal + ABNF concepts
  - Learning from XML: W3C Schema ➔ Relax-NG ➔ Relax-NG Compact
  - Standard published in 2019 as RFC 8610
  - Active work on extensions continues in IETF CBOR WG

How RFC 7071 would have looked like in CDDL

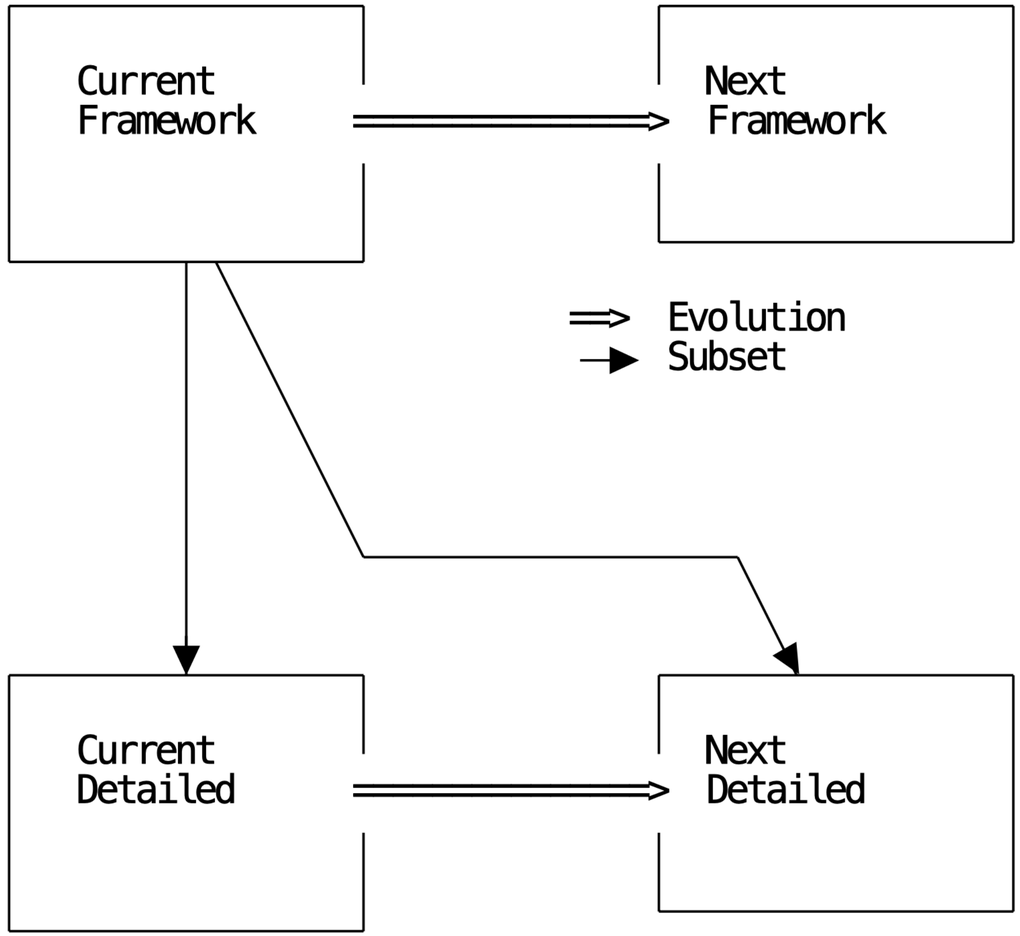
Much of the technical content of RFC on one slide

```
reputation-object = {                    ; This is a map (JSON object)
  application: text                      ;  text string (vs. binary)
  reputons: [* reputon]                  ;  Array of 0-∞ reputons
}

reputon = {                              ; Another map (JSON object)
  rater: text
  assertion: text
  rated: text
  rating: float
  ? confidence: float                    ; "?" == optional
  ? normal-rating: float
  ? sample-size: uint                    ; "uint" == unsigned integer
  ? generated: uint
  ? expires: uint
  * text => any                          ; 0-∞, express extensibility
}
```

The case against validation

- Data formats are **interfaces** that **evolve**
- Nailing down the syntax of an interface hinders that evolution
- In OneDM, we are starting to understand we need multiple syntaxes at any time:
  - ➔ current and next
  - ➘ loose (anticipating evolution) and strict
- CDDL has `.within` as a first step for ➘



» FDT use is highly vulnerable to what we call “process confabulation” in software engineering; constant vigilance against that is required to come up with realistic usages. «

Yours truly, 2017-10-28 on ietf@ietf.org

What we can get from an data description FDT?

- **Validation**
  - Is this what the protocol allows?
  - Generative (ABNF, Relax-NG, CDDL) vs. predicate-based vs. hybrid
- **Augmentation → Annotation → Transformation**
  - Augmentation: Adding information in the data model (e.g., defaults, PSVI)
  - Annotation: Adding information beyond the data model (e.g., semantics)
  - Transformation: Transforming from one data model to another