SDF Extension for Non-Affordance Information

draft-ietf-asdf-sdf-nonaffordance-02

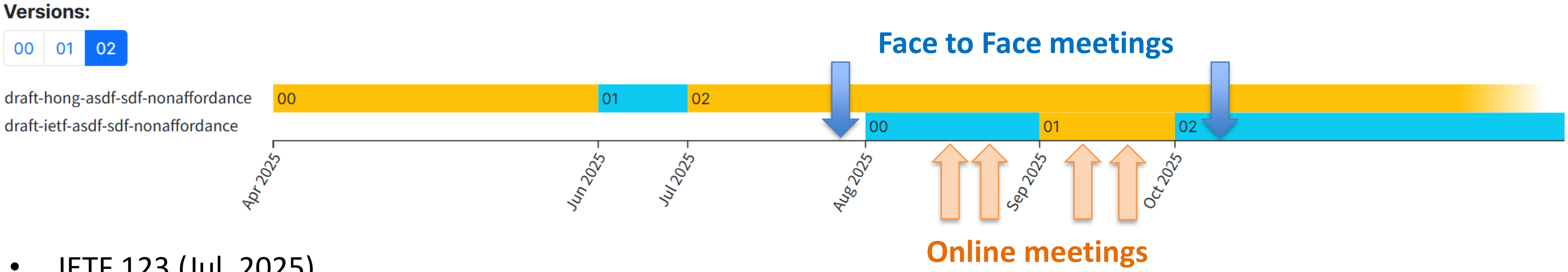
– IETF 124 ASDF WG –

Jungha Hong and Hyunjeong Lee

ETRI

Nov. 05, 2025

History of the document



- IETF 123 (Jul. 2025)
  - Call for WG adoption
  - Discussed changing the class name from *sdfNonaffordance* to ***sdfContext***
- draft-ietf-asdf-sdf-nonaffordance-00 (Aug. 2025)
  - **Adopted by WG** after IETF 123 meeting
- draft-ietf-asdf-sdf-nonaffordance-01 (Sep. 2025)
  - **Changed from *sdfNonaffordance* to *sdfContext***
  - Added the qualities of sdfContext
- draft-ietf-asdf-sdf-nonaffordance-02 (Oct. 2025)
  - Enhanced use cases

New class, sdfContex

- sdfContext is a peer to sdfProperty, sdfAction, and sdfEvent
- It is used to **represent non-affordance information** when authoring the SDF model
- Definitions under sdfContext are **read-only** in any generated interface
- Examples include location, manufacturer details, calibration parameters, installation attributes, and static identifiers

```
{
  "sdfObject": {
    "deviceWithContext": {
      "sdfProperty": {
        "temperature": {
          "type": "number",
          "unit": "°C"
        }
      },
      "sdfContext": {
        "serialNumber": {
          "type": "string",
          "description": "Unique factory-assigned identifier"
        },
        "installationLocation": {
          "type": "string",
          "description": "Location within the building where the
device is installed"
        },
        "calibrationOffset": {
          "type": "number",
          "unit": "°C",
          "description": "Offset applied to raw measurements for
calibration"
        }
      }
    }
  }
}
```

Figure 1: SDF Object containing a Property and a Context entry

Formulating a litmus test

- Litmus test on when to use sdfContex and when to use sdfProperty: *under development at the design team*

- Also working on litmus test on
  - ContextSnapshot vs. Proofshot
  - identityManifest vs. Construction

- Plans to add the litmus test and guidelines in the Appendix

Common message format – under discussion

Context Snapshot

Original Example

```
{
  "thingId": "envSensor:abc123",
  "timestamp": "2025-07-01T12:00:00Z",
  "contextSnapshot": {
    "installationInfo": {
      "floor": 3,
      "mountType": "ceiling",
      "indoorOutdoor": "indoor"
    }
  }
}
```

```
{
  "namespace": {
    "models": "https://example.com/models",
    "boats": "https://example.com/boats"
  },
  "defaultNamespace": "boats",
  "sdfInstanceOf": "models:#/boatWithHeater/sdfObject/envSensor",
  "sdfInstance": {
    "sdfContext": {
      "timestamp": "2025-07-01T12:00:00Z", // Should the timestamp h
      "thingId": "envSensor:abc123",
      "installationInfo": {
        "floor": 3,
        "mountType": "ceiling",
        "indoorOutdoor": "indoor"
      }
    }
  },
```

- Plans to add the syntax of sdfContext using CDDL in the Appendix

contextSnapshot vs. Proofshot

- ContextSnapshot
  - Descriptive identity & context (No live sensor values)
  - “Who am I, where am I?”

- Proofshot
  - Real-time operational status (Dynamic values)
  - “What is my current state?”

contextPatch vs. Deltas

- contextPatch
  - Partial update of non-affordance metadata: “What changed in my environment/context”
  - Performs an update
  - Only changed keys since last contextSnapshot
  - Efficient: no need to resend entire snapshot
  - Focus on contextual metadata (location, mounting, classification)

- Deltas
  - Partial update of instance operational state: “What changed in my operational state”
  - Reports the result of a state change
  - Delta compared to previous Proofshot
  - Uses algorithms (e.g., JSON Merge Patch) to apply changes
  - Focus on dynamic values (sensor state, property updates)

identityManifest vs. Construction

- identityManifest
  - Declare immutable identity and certification metadata
  - “Who I permanently am”
  - One-time or rare update
  - Immutable identity: manufacturer, model, firmware, certifications
  - Supports device registry and compliance
- Construction
  - Purpose: Instantiate a device/twin in the system with required parameters
  - “How I am created in the system”
  - Creates a new instance of the sensor
  - Provides initial parameters
  - Links to model definition

Q&A

Thank you