What's the Problem?

1. Google for ”XML sensor data format”
2. Take the first search result
3. Go to the first example on the page

The CC128 message structure is slightly compressed compa

```
<msg>
  <src>CC128-v0.11</src> start of message
  <dsb>00089</dsb> source and software version
  <time>13:02:39</time> days since birth, ie days run
  <tmpr>18.7</tmpr> 24 hour clock time as displayed
  <sensor>1</sensor> temperature as displayed
  <id>01234</id> Appliance Number as displayed
  <type>1</type> radio ID received from the sensor
  <ch1> sensor Type, "1" = electricity
  <watts>00345</watts> sensor channel
  data and units
</ch1>
```
Text-Based vs. Uniform Identifiers

- Cannot make any use of the text identifiers beyond exact match
- Text identifiers do not have clearly defined scope or uniqueness properties
- Uniform, formally defined identifiers can be passed around more easily:
  - They are self-describing
  - Merging data from different sources easier
  - No coordination needed across types
- Conclusion 1: use URNs or URIs as identifiers
Identifier Types

- Semantics-based “sensor for the oven”
- Name-based “my_sensor_3”
- Location-based “coordinates X,Y”
- Address-based “http://[2001:db8::1]”
- Device ID-based (”mac=...” or ”serial=...”)

How do you configure this device to send a name or location?

Conclusion 2: Device IDs are attractive for many deployment cases – e.g., identifying specific devices in sensor data streams, storage servers and equipment inventory applications. Names are obviously needed too, but can exist at higher layers.
The Specification for "dev" URNs

urn:dev:mac:0024befffe804ff1  (my laptop's MAC address)

- Device identifiers based on EUI-48/64 MAC addresses
  - Similar to UUIDs but requires no real-time clocks, stable storage, and has easier process on the manufacturing side
- Device identifiers based on 64-bit 1-Wire addresses
- Device identifiers based on cryptographic identifiers – related to the security discussion from yesterday
- Extension rules for new types
SenML
draft-jennings-senml

Cullen Jennings
Zach Shelby
Jari Arkko
**Why?**

- Smart objects need common data format(s)
- JSON is an easy, relatively compact format
- Properly designed base format helps use a generic data container for typical smart object applications – no need to design a scheme just to represent temperature measurements
- Right design helps keep size down even on textual format
- JSON, XML, EXI mappings