YANG Physical Entity Model

draft-entitydt-netmod-entity-00

IETF 94

Martin Björklund
Andy Bierman
Jie Dong
Dan Romascanu

Background

There is a need for a YANG model of physical hardware.

The ENTITY-MIB is one of the successful MIBs – leverage that.

ENTITY-MIB, ENTITY-SENSOR-MIB, ENTITY-STATE-MIB

The design team was created to define an initial model.

Overview

The ENTITY-MIB has two parts – physical (hardware) and logical (a single agent manages multiple contexts).

This work focuses on physical hardware only.

```
+--ro entity-state
|  +--ro last-change?              yang:date-and-time
|  +--ro physical-entity* [name]
|     +--ro name                   string
|     +--ro class?                 identityref
|     ...
+--rw entity {entity-config}?
   +--rw physical-entity* [name]
      +--rw name                   string
      +--rw asset-id?              string
      ...
```

Physical Hardware Type

The MIB has two objects, one generic, IANA-maintained *class* (enum with ~15 classes), and a vendor-specific *type* (AutonomousType).

In the YANG model we use a single leaf that is an identity.

IANA should maintain a YANG module with identities corresponding to the ~15 classes (*chassis, container, fan, battery, sensor, …*)

Vendors use them as-is, or derive from them:

```
identity acme-cpu-fan {
  base ianaent:fan;
}
```

Physical Hardware Attributes 1(2)

Flat list of physical entities, with support for containment.

A bunch of attributes – the WG needs to decide which attributes to keep, remove, or add.

Physical Hardware Attributes 2(2)

```
+--ro physical-entity* [name]
    +--ro name                string
    +--ro class               identityref
    +--ro physical-index?     int32 {entity-mib}?
    +--ro description?        string
    +--ro contained-in*       -> ../../physical-entity/name
    +--ro contains-child*     -> ../../physical-entity/name
    +--ro parent-rel-pos?     int32
    +--ro hardware-rev?       string
    +--ro firmware-rev?       string
    +--ro software-rev?       string
    +--ro serial-num?         string
    +--ro mfg-name?           string
    +--ro model-name?         string
    +--ro alias?              string
    +--ro asset-id?           string
    +--ro is-fru?             boolean
    +--ro mfg-date?           yang:date-and-time
    +--ro uri*                inet:uri
    +--ro uuid?               yang:uuid
```

Sensor and State 1(2)

There are additional MIBs, ENTITY-SENSOR-MIB and ENTITY-STATE-MIB.

```
container sensor-data {
  when 'derived-from-or-self(../class,
                             "iana-entity", "sensor")' {
  if-feature sensor-data;
```

```
   +--ro physical-entity* [name]
      +--ro sensor-data {entity-sensor}?
         +--ro data-type?              entity-sensor-data-type
         +--ro data-scale?             entity-sensor-data-scale
         +--ro precision?              entity-sensor-precision
         +--ro value?                  entity-sensor-value
         +--ro oper-status?            entity-sensor-status
         +--ro sensor-units-display?   string
         +--ro value-timestamp?        yang:date-and-time
         +--ro value-update-rate?      uint32
```

Sensor and State 2(2)

ENTITY-STATE_MIB adds generic states to components

```
container state {
  if-feature entity-state;
```

```
ro physical-entity* [name]
+--ro state {entity-state}?
   +--ro state-last-changed?  yang:date-and-time
   +--ro admin-state?         locked | unlocked | shutting-down
   +--ro oper-state?          enabled | disabled | testing
   +--ro usage-state?         idle | active | busy
   +--ro alarm-status?        critical | ... | warning
   +--ro standby-status?      hot | cold | providing-service
```

Configurable nodes

The MIB supports writable objects that are supposed to persist across reboots. They are modeled as config data in the YANG model:

```
+--rw entity {entity-config}?
   +--rw physical-entity* [name]
      +--rw name             string
      +--rw serial-num?      string
      +--rw alias?           string
      +--rw asset-id?        string
      +--rw uri*             inet:uri
      +--rw admin-state?     entity-admin-state {entity-state}?
```

Next Steps

Adopt as a WG document?

Need review.