CoRE
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2017

P. van der Stok, Ed.
consultant
J. Jimenez
Ericsson

October 27, 2016

Mapping from LWM2M model to CoMI YANG model
draft-vanderstok-core-yang-lwm2m-00

Abstract

This document defines a set of rules to convert a LWM2M xml-based device specification to a YANG MODULE. The invocation of the server executing the converted YANG code makes use of CoMI. The mapping from the original LWM2M URI to the corresponding CoMI URI is presented.

Note

Discussion and suggestions for improvement are requested, and should be sent to roll@ietf.org.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Standardization organizations define interfaces hosted by processors to manipulate the connected equipment. Examples of such standardization organizations are BACnet, KNX, ZigBee, oBIX, OMA/ IPSO, and many others. These organizations plan to move to resource based interfaces. The data models proposed by these organizations are hierarchical models that can be specified in XML and describe classes with attributes and operations that can be instantiated to objects. An example is the OMA LWM2M (see [OMNA]) Object model, that standardizes eight numbered object types for device management. IPSO (see [IPSO]) expands those objects to handle applications. This document describes rules to translate xml specifications of the LWM2M/IPSO organizations to YANG [RFC7950], and the invocation of the YANG based server according to the CoAP Management Interface (CoMI) specification [I-D.vanderstok-core-comi].

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The following terms are defined in [RFC6241] and are not redefined here:

- client
- configuration data
- server
- state data

The following terms are defined in [RFC7950] and are not redefined here:

- data model
- data node

The terminology for describing YANG data models is found in [RFC7950].

1.1.1. Tree Diagrams

A simplified graphical representation of the data model is used in the YANG modules specified in this document. The meaning of the symbols in these diagrams is as follows:

> Brackets "[" and "]" enclose list keys.
>
> Abbreviations before data node names: "rw" means configuration data (read-write) and "ro" state data (read-only).
>
> Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
>
> Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
>
> Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Conversion rules LWM2M to YANG

LWM2M objects are typed, where each type is identified with a number. The object provides one or more instances which are numbered. An instance is composed of resources, also identified with numbers. An instance on a host can be accessed with the example URI: coap+lwm2m://example.com/object/instance/resource, where resource, instance and object are numbers, specified by the LWM2M specification.

When using YANG, the object identifiers, followed by the resource identifier, are YANG strings instead of numbers. The format of the instance identifier depends on the YANG model that is chosen to specify LWM2M objects.

For an automatic translation from the XML LWM2M specification to a YANG specification, the following rules apply for access, optional, units, and range specifications:

- The optional/mandatory aspect of the LWM2M resource is covered by the leaf’s mandatory "false/true" statement of YANG as specified in section 7.6.5 of [RFC7950]. The YANG statement "mandatory = TRUE" means that(given the right conditions) the leaf must exist.

- The R,W access aspects of a data item are translated using the YANG "config" statement as specified in section 7.21.1 of [RFC7950]. If "config" is "true", the Data nodes are part of configuration datastores, resulting in RW access. If "config" is "false", the data nodes are not part of configuration datastores, resulting in R access. YANG does not provide facilities to specify W access only.

- When the YANG RPC is specified, E access is meant. In section 7.14 of [RFC7950] RPCs are modelled for NETCONF using YANG input and output parameters. When input parameters are added, EW access is meant; when output parameters are added, ER access is meant and with both input and output parameters ERW access is meant.

- The YANG ACTION is specified in section 7.15 of [RFC7950]. Contrary to RPC, ACTION statement is associated with a data node. The data node has E access. Input leafs of the data node have W access, and output leafs have R access.

- To specify the range of a data resource the YANG range statement, specified in section 9.2.4 of [RFC7950], is used.

- YANG range can be used in a straightforward fashion for items of type integer. The range of decimal64, used for float, is less straightforward. The possible ranges are restricted by the fraction-digits which specifies the size of the fraction part of the float (see section 9.3.4. of [RFC7950].

- The YANG units statement, specified in section 7.3.3 of [RFC7950], is used to express the units.

- The attributes of the YANG leaf need to be presented in the order: "type", possibly qualified with "range", "units", "config", "mandatory", and finally "Description".

- In the presented YANG specification the LWM2M resources are specified as leafs of a YANG list.

YANG lists may contain key leafs which uniquely identify an instance in a list. By specifying a key leaf (for example called "instance") that contains the list instance number, the YANG list instance can be uniquely referenced by the instance number. Accordingly, OMA objects are modelled as YANG lists. The value of the "instance" leaf in the list is equal to the instance number of the OMA object. The numbering of the instances does not need to be consecutive. The OMA resources are the other leafs of the YANG list.

Choices need to be made how to represent the numbered object ID, and resource ID as YANG identifiers. YANG identifiers are strings and cannot be represented by numbers.

The YANG identifier strings need to be mapped to numbered identifiers. The appendices show 3 ways to represent the LWM2M device ID and resource ID in the YANG specification.

- In Appendix A, Yang Identifiers are modelled as strings that start with string "ID" followed by the identifier number (see module humidityID).

- In Appendix B, Yang Identifiers of objects and resources are modelled as strings that are equivalent to the OMA object- and resource- name (see module humidityNM).

- In Appendix C, the OMA device is modelled as a YANG container composed of an identifier and a list of instances. The list is composed of an instance number and a set of resource containers. The resource container is composed of the pair (attribute identifier number, IPSO resource specification)(see module humidityLF).

Below the tree diagrams (see Section 1.1.1 for an explanation of the syntax) of the three valid YANG modules are shown.

```
   module: ietf-yang-humidityID
   +--ro ID3304* [instance]
   +--ro instance                            uint16
   +--ro ID5700                              decimal64
   +--ro ID5701?                             string
   +--ro ID5601?                             decimal64
   +--ro ID5602?                             decimal64
   +--ro ID5603?                             decimal64
   +--ro ID5604?                             decimal64
   +---x ID5605


   module: ietf-yang-humidityNM
   +--ro IPSO-humidity* [instance]
   +--ro instance                             uint16
   +--ro Sensor_Value                         decimal64
   +--ro Units?                               string
   +--ro Min_Measured_Value?                  decimal64
   +--ro Max_Measured_Value?                  decimal64
   +--ro Min_Range_Value?                     decimal64
   +--ro Max_Range_Value?                     decimal64
   +---x Reset_Min_and_Max_measured_values

   module: ietf-yang-humidityLF
   +--rw IPSO-humidity
   +--ro identifier    uint16
   +--ro resources* [instance]
   +--ro instance                             uint16
   +--ro Sensor_Value
   |  +--ro identifier?                       uint16
   |  +--ro content                           decimal64
   +--ro Units
   |  +--ro identifier?                       uint16
   |  +--ro content?                          string
   +--ro Min_Measured_Value
   |  +--ro identifier?                       uint16
   |  +--ro content?                          decimal64
   +--ro Max_Measured_Value
   |  +--ro identifier?                       uint16
   |  +--ro content?                          decimal64
   +--ro Min_Range_Value
   |  +--ro identifier?                       uint16
   |  +--ro content?                          decimal64
   +--ro Max_Range_Value
   |  +--ro identifier?                       uint16
   |  +--ro content?                          decimal64
   +--ro Reset_Min_and_Max_measured_values
   +--ro identifier?                          uint16
   +---x reset
```

Module humidityLF of Appendix C is the most complex one and is not recommended. Module humidityID of Appendix A works but is a bit forced approach and lacks the resource name. Module humidityNM of Appendix B is the most natural approach where the YANG identifiers are equal to the device (type) and resource names.

CoMI [I-D.vanderstok-core-comi] uses a conversion from names to numbers to reduce the request URI size, and the payload of the server requests and answers. The LWM2M organization specifies both the names and the numbers of the devices and resources. The number of the resource is not unique and for the CoMI identifier the resource number needs to be prefixed by the device number to be unique.

3. URI convention

The invocation URI of a LWM2M resource looks like:

```
coap+lwm2M://example.com/object/instance/resource
```

In this section it is assumed that the YANG mapping of the module humidityNM of Appendix B is used.

When YANG is used, the LWM2M resource invocation can follow the RESTCONF convention using http, or the CoMI convention using CoAP.

When using RESTCONF (see section 3.5.3 of [I-D.ietf-netconf-restconf] the invocation of object with instance = number will look like:

```
http://example.com/object/instance=number/resource
```

In the case of CoMI the object/resource numbers are used, and not the names, to reduce the payload size. The instance is specified in a query parameter. Consequently, the LWM2M resource on a server executing a YANG specification, is accessed according to the CoMI specification with:

```
coap://example.com/identifier?k=number
```

The identifier is a composition of the object number and the resource number. Assume that n is smallest number for which 10**(n+1)/resource >= 1. The value of the identifier = (object*10**n)+resource.

Assume that the IPSO-humidity/Sensor_Value 3304/5700 numbers are composed to the numeric identifier 33045700. According to [RFC4648], the identifier is represented in base64 which leads to B-DzE. The URI for the CoMI invocation of instance 0 of IPSO-humidity/ Sensor_value will look like:

coap://example.com/B-DzE?k=0

For LWM2M objects with only one instance, the k=0 can be omitted.

4. observation and notification

An LWM2M server uses "observe" to receive notification from the server. This remains unchanged with YANG servers and CoMI.

5. Payload format

The payload of the request and the response follows the payload format specified for CoMI. The content format is CBOR [RFC7049]. The YANG objects are returned as maps containing (identifier, value) pairs. Where the identifier is the numeric identifier discussed in Section 3. and the value is of the type specified by the YANG specification of the server. The CBOR encoding of the YANG types is specified in [I-D.ietf-core-yang-cbor].

6. YANG extensions to LWM2M

By adding keys leafs to a list object, YANG allows additionally the selection of instances by the contents of the key leafs.

The FETCH method of CoAP makes it possible to request multiple resource instances in one request.

The notification statement of YANG encourages a more flexible specification of notifications.

7. Security considerations

To be filled in

8. Acknowledgements

We are grateful to

9. Changelog

NO changes from nothing to version 00

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <http://www.rfc-editor.org/info/rfc7950>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <http://www.rfc-editor.org/info/rfc4648>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

[I-D.vanderstok-core-comi]
Stok, P. and A. Bierman, "CoAP Management Interface", draft-vanderstok-core-comi-09 (work in progress), March 2016.

[I-D.ietf-core-yang-cbor]
Veillette, M., Pelov, A., Somaraju, A., Turner, R., and A. Minaburo, "CBOR Encoding of Data Modeled with YANG", draft-ietf-core-yang-cbor-02 (work in progress), July 2016.

10.2. Informative References

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-17 (work in progress), September 2016.

[OMNA] "Open Mobile Naming Authority (OMNA)", Web http://http://technical.openmobilealliance.org/Technical/technical-information/omna.

[IPSO] "IP for Smart Objects (IPSO)",
Web http://ipso-alliance.github.io/pub/.

Appendix A. YANG identifiers as IDnumbers

Yang Identifiers are modelled as string that starts with ID followed by the identifier number. The device object is modelled as a list that contains multiple instances.

```
   <CODE BEGINS> file "ietf-humidityID@2016-07-25.yang"
           module ietf-humidityID{

           yang-version 1.1;  // needed for action

           namespace
           "urn:ietf:params:xml:ns:yang:ietf-humidityID";

           prefix humid;

           organization
           "IPSO";

           contact
           "WG Web:   http://tools.ietf.org/wg/core/
           WG List:  mailto:core@ietf.org

           WG Chair: Carsten Bormann
           mailto:cabo@tzi.org

           WG Chair: Jaime Jimenez
           mailto:jaime.jimenez@ericsson.com

           Editor:   Peter van der Stok
           mailto:consultancy@vanderstok.org

           Editor:   Jaime Jimenez
           mailto:jaime.jimenez@ericsson.com";

           description
   "This module contains information about the operation of the
   IPSO LWM2M humidity sensor with ID 3304.

   Copyright (c) 2016 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
           (http://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

           revision "2016-07-25" {
           description "Initial revision.";
           reference
   "I-D:draft-vanderstok-core-yang-lwm2m: YANG language applied
   to the LWM2M IPSO humidity sensor specification";
           }

           list ID3304 {
           key instance;
           config  false;     // should be same for key leaf
           description
           "IPSO humidity: The humidity sensor is composed of
            a set of instances";
           leaf instance {
           type uint16{
           range "0..1";   // only one instance zero (0)
           }
           config false;    // R access
           mandatory "true";
           description
           "the number of the humidity sensor instance";
           }
           leaf ID5700 {
           type decimal64{    // YANG has no float
           fraction-digits 2;
           range "10.0 .. 66.6";}
           config false;    // R access
           mandatory "true";
           description
           "Sensor Value: Last or Current Measured Value
              from the Sensor";
           }
           leaf ID5701 {
           type string;
           units "Defined by 'Units' resource";
           config false;    // R access
           description
           "Units: Measurement unit definition
               e.g. 'Cel' for temperature in Celsius";
           }
           leaf ID5601 {
```

```
           type decimal64{    // YANG has no float
           fraction-digits 2;
           range "10.0 .. 66.6";}
           units "Defined by ’Units’ resource";
           config false;   // R access
           description
           "Min Measured Value: The minimum value measured
             by the sensor since power ON or reset";
           }
           leaf ID5602 {
           type decimal64{    // YANG has no float
           fraction-digits 2;
           range "10.0 .. 66.6";}
           units "Defined by ’Units’ resource";
           config false;    // R access
           description
           "Max Measured Value: The maximum value measured
              by the sensor since power ON or reset";
           }
           leaf ID5603 {
           type decimal64{    // YANG has no float
           fraction-digits 2;
           range "10.0 .. 66.6";}
           units "Defined by ’Units’ resource";
           config false;    // R access
           description
           "Min Range Value: The minimum value that
              can be measured by the sensor";
           }
           leaf ID5604 {
           type decimal64{    // YANG has no float
           fraction-digits 2;
           range "10.0 .. 66.6";}
           units "Defined by ’Units’ resource";
           config false;    // R access
           description
           "Max Range Value: The maximum value that
              can be measured by the sensor";
           }
           action ID5605 {
           //E access: this is an RPC
           //    without input and output parameters
           description
           "Reset Min and Max measured values: Reset the
              Min and Max measured values to current value";
           }
           }  // list ID3304
           } // module ietf-yang-humidity
```

```
          <CODE ENDS>
```

Appendix B. YANG identifiers as resource names

Yang Identifiers are modelled as strings that represent the resource name. The device object is modelled as a list with multiple instances.

```
   <CODE BEGINS> file "ietf-humidityNM@2016-07-25.yang"

   module ietf-humidityNM{

   yang-version 1.1;  // needed for action

   namespace
               "urn:ietf:params:xml:ns:yang:ietf-humidityNM";

   prefix humid;

               organization
               "IPSO";

               contact
               "WG Web:   http://tools.ietf.org/wg/core/
               WG List:  mailto:core@ietf.org

               WG Chair: Carsten Bormann
               mailto:cabo@tzi.org

               WG Chair: Jaime Jimenez
               mailto:jaime.jimenez@ericsson.com

               Editor:   Peter van der Stok
               mailto:consultancy@vanderstok.org

               Editor:   Jaime Jimenez
               mailto:jaime.jimenez@ericsson.com";

               description
     "This module contains information about the
   operation of the IPSO LWM2M humidity sensor with ID 3304.

   Copyright (c) 2016 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust’s Legal Provisions
   Relating to IETF Documents
         (http://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

               revision "2016-07-25" {
               description "Initial revision.";
               reference
               "I-D:draft-vanderstok-core-yang-lwm2m:
   YANG language applied to the LWM2M IPSO humidity sensor
   specification";
               }


               list IPSO-humidity {
               key instance;
               config   false;   //  should be same as key leaf
               description
               "3304: The humidity sensor is composed of
                 a set of instances";
               leaf instance {
               type uint16{
               range "0..1";   // only one instance zero (0)
               }
               config false;    // R access
               mandatory "true";
               description
               "the number of the humidity sensor instance";
               }
               leaf Sensor_Value  {
               type decimal64{    // YANG has no float
               fraction-digits 2;
               range "10.0 .. 66.6";}
               units "Defined by ’Units’ resource";
               config false;    // R access
               mandatory "true";
               description
               "5700: Last or Current Measured Value
                  from the Sensor";
               }
               leaf Units {
               type string;
               units "Defined by ’Units’ resource";
               config false;    // R access
               description
```

```
"5701: Measurement unit definition
    e.g. 'Cel' for temperature in Celsius";
}
leaf Min_Measured_Value {
type decimal64{    // YANG has no float
fraction-digits 2;
range "10.0 .. 66.6";}
units "Defined by 'Units' resource";
config false;    // R access
description
"5601: The minimum value measured by
   the sensor since power ON or reset";
}
leaf Max_Measured_Value {
type decimal64{    // YANG has no float
fraction-digits 2;
range "10.0 .. 66.6";}
units "Defined by 'Units' resource";
config false;    // R access
description
"5602: The maximum value measured
    by the sensor since power ON or reset";
}
leaf Min_Range_Value {
type decimal64{    // YANG has no float
fraction-digits 2;
range "10.0 .. 66.6";}
units "Defined by 'Units' resource";
config false;    // R access
description
"5603: The minimum value that can be measured
   by the sensor";
}
leaf Max_Range_Value{
type decimal64{    // YANG has no float
fraction-digits 2;
range "10.0 .. 66.6";}
units "Defined by 'Units' resource";
config false;    // R access
description
"5604: The maximum value that can be measured
   by the sensor";
}
action Reset_Min_and_Max_measured_values {
// E access: this is an RPC
// without input and output parameter
description
"5605: Reset the Min and Max measured values
```

```
                   to current value";
                }  // rpc
                }  // list ID3304
                } // module ietf-yang-humidity

                <CODE ENDS>
```

Appendix C. YANG identifiers as additional leaf

The device object is modelled as a container composed of an identifier and a list of instances. The list instance is composed of an instance number and a set of resource containers. The resource container is composed of the pair (attribute identifier number, IPSO resource specification).

```
     <CODE BEGINS> file "ietf-humidityLF@2016-07-25.yang"

     module ietf-humidityLF{

     yang-version 1.1;  // needed for rpc

     namespace
       "urn:ietf:params:xml:ns:yang:ietf-humidityLF";

     prefix humid;

                    organization
                    "IPSO";

                    contact
                    "WG Web:   http://tools.ietf.org/wg/core/
                    WG List:  mailto:core@ietf.org

                    WG Chair: Carsten Bormann
                    mailto:cabo@tzi.org

                    WG Chair: Jaime Jimenez
                    mailto:jaime.jimenez@ericsson.com

                    Editor:   Peter van der Stok
                    mailto:consultancy@vanderstok.org

                    Editor:   Jaime Jimenez
                    mailto:jaime.jimenez@ericsson.com";
```

```
                   description
                   "This module contains information about the
   operation of the IPSO LWM2M humidity sensor with ID 3304.

   Copyright (c) 2016 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust’s Legal Provisions
   Relating to IETF Documents
   (http://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX;
   see the RFC itself for full legal notices.";

                   revision "2016-07-25" {
                   description "Initial revision.";
                   reference
                   "I-D:draft-vanderstok-core-yang-lwm2m:
   YANG language applied to the LWM2M IPSO humidity sensor
   specification";
                   }

                   container IPSO-humidity{
                   description
                   "Device separated in identifier and list";
                   leaf identifier{
                   type uint16;  // fixed to 3304
                   config false;
                   mandatory "true";
                   description
                   "the LWM2M identification number of the device";
                   }
                   list resources {
                   key instance;
                   config  false;   // should be same as key leaf
                   description
                   "3304: The humidity sensor is composed of
                      a set of instances";
                   leaf instance {
                   type uint16{
                   range "0..1";   // only one instance zero (0)
                   }
                   config false;    // R access
                   mandatory "true";
                   description
                   "the number of the humidity sensor instance";
                   }  // instance number
                   container Sensor_Value  {
                   description
                   "Resource separated in identifier and content";
                   leaf identifier{
                   type uint16;   // fixed to 5700
                   config false;  // R access
                   description
                   "identifier should contain the value 5700";
                   }
                   leaf content{
                   type decimal64{    // YANG has no float
                   fraction-digits 2;
                   range "10.0 .. 66.6";}
                   units "Defined by ’Units’ resource";
                   config false;    // R access
                   mandatory "true";
                   description
                   "5700: Last or Current Measured Value
                     from the Sensor";
                   }  // content
                   }  // container
                   container Units {
                   description
                   "Resource separated in identifier and content";
                   leaf identifier{
                   type uint16;   // fixed to 5701
                   config false;  // R access
                   description
                   "identifier should contain the value 5701";
                   }
                   leaf content{
                   type string;
                   units "Defined by ’Units’ resource";
                   config false;    // R access
                   description
                   "5701: Measurement unit definition
                     e.g. ’Cel’ for temperature in Celsius";
                   }  // content
                   }  // container
                   container Min_Measured_Value {
                   description
                   "Resource separated in identifier and content";
                   leaf identifier{
                   type uint16;   // fixed to 5601
                   config false;  // R access
                   description
                   "identifier should contain the value 5601";
                   }  // identifier
                   leaf content{
                   type decimal64{    // YANG has no float
                   fraction-digits 2;
                   range "10.0 .. 66.6";}
                   units "Defined by ’Units’ resource";
                   config false;    // R access
                   description
                   "5601: The minimum value measured by the
                     sensor since power ON or reset";
                   }  // content
                   }
                   container Max_Measured_Value {
                   description
                   "Resource separated in identifier and content";
                   leaf identifier{
                   type uint16;   // fixed to 5602
                   config false;  // R access
                   description
                   "identifier should contain the value 5602";
                   }
                   leaf content{
                   type decimal64{    // YANG has no float
                   fraction-digits 2;
                   range "10.0 .. 66.6";}
                   units "Defined by ’Units’ resource";
                   config false;    // R access
                   description
                   "5602: The maximum value measured by
                     the sensor since power ON or reset";
                   }  // content
                   }  // container
                   container Min_Range_Value {
                   description
                   "Resource separated in identifier and content";
                   leaf identifier{
                   type uint16;   // fixed to 5603
                   config false;  // R access
                   description
                   "identifier should contain the value 5603";
                   }  // identifier
                   leaf content{
                   type decimal64{    // YANG has no float
                   fraction-digits 2;
                   range "10.0 .. 66.6";}
                   units "Defined by ’Units’ resource";
                   config false;    // R access
                   description
                   "5603: The minimum value that can be measured
                      by the sensor";
                   }  // content
                   }  // container
                   container Max_Range_Value{
                   description
                   "Resource separated in identifier and content";
                   leaf identifier{
                   type uint16;   // fixed to 5604
                   config false;  // R access
                   description
                   "identifier should contain the value 5604";
                   }  // identifier
                   leaf content{
                   type decimal64{    // YANG has no float
                   fraction-digits 2;
                   range "10.0 .. 66.6";}
                   units "Defined by 'Units' resource";
                   config false;    // R access
                   description
                   "5604: The maximum value that can be measured
                     by the sensor";
                   }  // content
                   }
                   container Reset_Min_and_Max_measured_values {
                   description
                   "Resource separated in identifier and action";
                   leaf identifier{
                   type uint16;   // fixed to 5605
                   config false;  // R access
                   description
                   "identifier should contain the value 5605";
                   }
                   action reset{
   // E access: this is an RPC without input and output parameters
                   description
                   "5605: Reset the Min and Max measured values to
                    current value";
                   } // action  reset
                   } // container Reset_min_and_max
                   }  // list resources
                   }  // container IPSO-humidity (3304)
                   } // module ietf-yang-humidity

                   <CODE ENDS>
```

Authors' Addresses

Peter van der Stok (editor)
consultant

Phone: +31-492474673 (Netherlands), +33-966015248 (France)
Email: consultancy@vanderstok.org
URI: www.vanderstok.org

Jaime Jimenez
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Phone: +358-442992827(Finland)
Email: jaime.jimenez@ericsson.com