OCF 2.3 – Device Management v2: software update – CTWG CR 2303

Legal Disclaimer

THIS IS A DRAFT SPECIFICATION DOCUMENT ONLY AND HAS NOT BEEN ADOPTED BY THE OPEN CONNECTIVITY FOUNDATION. THIS DRAFT DOCUMENT MAY NOT BE RELIED UPON FOR ANY PURPOSE OTHER THAN REVIEW OF THE CURRENT STATE OF THE DEVELOPMENT OF THIS DRAFT DOCUMENT. THE OPEN CONNECTIVITY FOUNDATION AND ITS MEMBERS RESERVE THE RIGHT WITHOUT NOTICE TO YOU TO CHANGE ANY OR ALL PORTIONS HEREOF, DELETE PORTIONS HEREOF, MAKE ADDITIONS HERETO, DISCARD THIS DRAFT DOCUMENT IN ITS ENTIRETY OR OTHERWISE MODIFY THIS DRAFT DOCUMENT AT ANY TIME. YOU SHOULD NOT AND MAY NOT RELY UPON THIS DRAFT DOCUMENT IN ANY WAY, INCLUDING BUT NOT LIMITED TO THE DEVELOPMENT OF ANY PRODUCTS OR SERVICES. IMPLEMENTATION OF THIS DRAFT DOCUMENT IS DONE AT YOUR OWN RISK AMEND AND IT IS NOT SUBJECT TO ANY LICENSING GRANTS OR COMMITMENTS UNDER THE OPEN CONNECTIVITY FOUNDATION INTELLECTUAL PROPERTY RIGHTS POLICY OR OTHERWISE. IN CONSIDERATION OF THE OPEN CONNECTIVITY FOUNDATION GRANTING YOU ACCESS TO THIS DRAFT DOCUMENT, YOU DO HEREBY WAIVE ANY AND ALL CLAIMS ASSOCIATED HEREWITH INCLUDING BUT NOT LIMITED TO THOSE CLAIMS DISCUSSED BELOW, AS WELL AS CLAIMS OF DETRIMENTAL RELIANCE.

****** change 1 ******

1 Software Update Resource

The Software Update Resource is used to control software updates of the Device.

In the Security Specification there is already a manual triggered software update mechanism available. The triggering of the Client (manual) software update is achieved via the security Resource Type "oic.r.pstat" by using the appropriate bits in the "tm" Property. The software update triggering results in updates of the "cm" Property in the "oic.r.pstat" Resource Type (see security spec XXX). The Software Update Resource adds additional features to the security specified mechanism, like:

- Specify the source to obtain the software package
- Time scheduled software update actions
- Status information, especially more info about various error situations

If the Device implements the Software Update Resource, it is required to implement the software update behaviour to actually update the software of the Device as indicated by the oic.r.pstat cm bits as defined in XXX(security spec). Also the security defined software update process shall use the data that is set on the Software Update Resource like the "purl" Property.

The Software Update Resource Type is "oic.r.softwareupdate" and is described in Table 1.

Table 1. Optional Software Update Resource

Predefined URI	Resource Type Title	Resource Type ID (“rt” value)	Interfaces	Description	Related Functional Interaction
NA	Software Update	"oic.r.softwareupdate"	"oic.if.rw" "oic.if.baseline"	The Resource exposes Properties to control and monitor the software update mechanism. The Properties exposed by Resource Type "oic.r.softwareupdate" are listed in Table 2.	Device Management

Table 2 defines the Properties of the "oic.r.softwareupdate" Resource Type.

Table 2. "oic.r.softwareupdate" Resource Type definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
new version	nv	string			R	no	New available Software version
Package url	purl	string	URL		RW	yes	Source of the software package, might be a HTTPS or CoAPs URL.
action	swupdateaction	string	enum, (see table XXX)		RW	yes	Scheduled action to do a software update.
state	swupdatestat	string	enum (see		R	yes	State of the software

	e		table XXX)				update
result	swupdateresult	integer			R	yes	Result of the software update, list of result codes.
lastupdate	lastupdate	string	date-time		R	no	Time of the last software update (in RFC3339 format), Initial set on date of manufacturing.
signage	signed	string	enum		R	no	Signage method of the software package, currently the only allowed value is “vendor”.
updatetime	updatetime	string	date-time		RW	yes	Scheduled time (in RFC3339 format) to do action which is specified in “swupdateaction” Property.

The values of the "swupdatestate" Property are described in Table 3

Table 3 State definitions and state transitions of Software Update Resource

Description	Value of Property “swupdatestate”	equivalent "cm" bit values in pstat.	Transition allowed from state
Idle, waiting for updates	idle	Bit 64 = 0 Bit 128 = 0 Bit 256 = 0	nsa, svv, sva, upgrading
New Software Available (after checking for new software being available on the url indicated by purl). This step does not download the new software	nsa	Bit 64 = 0 Bit 128 = 0 Bit 256 = 1	idle, svv, sva, upgrading
Software Version validation(during downloading and checking the software integrity)	svv	Bit 64 = 0 Bit 128 = 0 Bit 256 = 1	idle, nsa, sva, upgrading
Software Version available(The software is downloaded and deemed to be valid))	sva	Bit 64 = 1 Bit 128 = 0 Bit 256 = 1	idle, nsa, svv, upgrading
Upgrading	upgrading	Bit 64 = 1 Bit 128 = 1 Bit 256 = 1	idle, nsa, svv, sva

The typical state transitions are described by Figure 1. The state transitions can be trigged manually or by a timed action. The manual state triggers (i.e. "tm" Property of "oic.r.pstat") are described in the security specification [XXX]. The timed state triggers are managed using the "swupdateaction" and "updatetime" Properties of the Software Update Resource to trigger software update actions at some future date and time. The action names for scheduled actions are listed in Table 4. When the “updatetime” for the timed action is in the past then the update shall not take place, it is implementation dependent if the UPDATE with an “updatetime” value in the past will give an error on the UPDATE.

Table 4 Value definitions for the Property "swupdateaction"

Description	Value of Property “swupdateaction”, for scheduled update actions.	Action take	equivalent pstat tm bits.
Nothing scheduled (not applicable)	idle	No action	
Initiate Software Availability Check	isac	Check on remote end point if a newer software version is available	tm bit 256
Initiate Software Version Validation	isvv	Downloads and verifies if the software version is valid.	tm bit 64
Initiate Secure Software Update	upgrade	Upgrades the software in the Device. It uses the downloaded and validated software package. If no validated software package is available on the Device, the device takes a necessary steps to obtain a validated software package, by downloading and verifying software of the external source.	tm bit 128

```
@startuml
[*] -down-> idle

idle -down-> new_version_check : pstat.tm@bit 256 = 1 (Initiate Software Availability
Check)

idle --> upgrading : pstat.tm@bit 128 = 1 (Initiate Secure Software Update)

idle : pstat.cm@bit256 = 0 (New Software Availability)
idle : pstat.cm@bit64 = 0 (Valid Software Available)
idle : pstat.cm@bit128 = 0 (upgrading))
idle : state = idle


new_version_check -up-> idle : pstat.cm@bit256 = 0 (no new version available)

new_version_check : device will only report
new_version_check : if new software is available
new_version_check: pstat.cm@bit256 = 0 (New Software Availability)
new_version_check: pstat.cm@bit64 = 0 (Valid Software Available)
new_version_check: pstat.cm@bit128 = 0 (upgrading)
new_version_check: state = idle


new_version_check ->  version_validation : pstat.cm@bit256 = 1 (new version available)
idle ->  version_validation : pstat.tm@bit 64 = 1 (Initiate Software Version Validation)

version_validation -> idle: pstat.cm@bit64 = 0 (no valid software available)
version_validation -> version_available: pstat.cm@bit64 = 1 (valid software available)
version_validation: pstat.cm@bit256 = 1 (New Software Availability)
version_validation: pstat.cm@bit64 = 0 (Valid Software Available)
version_validation: pstat.cm@bit128 = 0 (upgrading)
version_validation: state: nsa


version_available -> upgrading : Initiate Secure Software Update
version_available: device will transit automatically
version_available: when validation check is finished


version_available : pstat.cm@bit256 = 1 (New Software Availability)
```

```
version_available : pstat.cm@bit64 = 1 (Valid Software Available)
version_available : pstat.cm@bit128 = 0 (upgrading)
version_available : state : sva

upgrading : upgrading
upgrading : pstat.cm@bit256 = 1 (New Software Availability)
upgrading : pstat.cm@bit64 = 1 (Valid Software Available)
upgrading : pstat.cm@bit128 = 1 (upgrading)
upgrading : state = upgrading

upgrading -> idle: pstat.cm@bit128 = 0 (upgrading finished)
@enduml
```

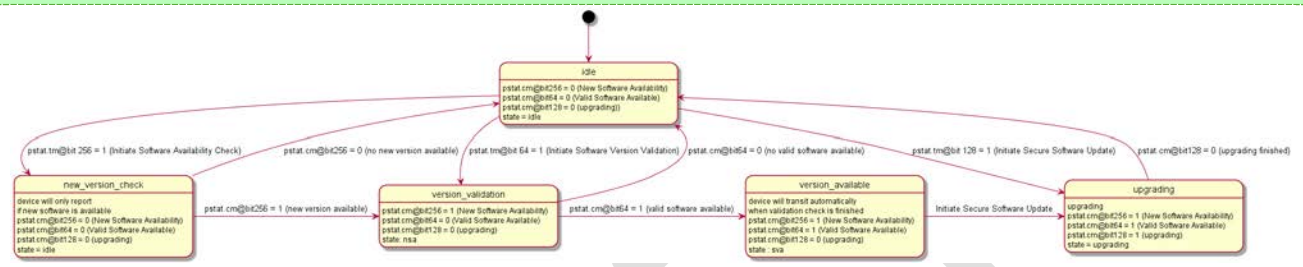

Figure 1. Typical state transitioning diagram for software update

The Property "purl" indicates the URL to obtain the software package from. This URL shall be a fully qualified URL. If the value is an empty string (“”) then the Device will use the built in vendor defined URL (see security specification). If a built in URL is not implemented, setting the "purl" value on empty string will result in error 6 as defined in Table 5.

Table 5 list of codes of the "swupdateresult" Property.

Description	code
Idle.	0
Success, everything went well.	1
Not enough RAM.	2
Not enough Flash.	3
Connection lost.	4
Software validation failure.	5
Invalid URL to receive the software package.	6
Unsupported protocol for download URL.	7
Firmware update failed.	8

Software transport error codes. HTTP result codes when accessing the URL to download the software package.	400-600

Figure [XXX2] depicts a typical update scenario. This scenario is using the observability of pstat, so that the client is informed on the changes of the cm bit value to track the progress.

```
@startuml
participant client
participant pstat
participant softwareupgrade as firmware

box "Resources on the same Device"
participant pstat
participant firmware
end box


participant "external server" as external


== observe device ==

client -> firmware: observe state
firmware --> client : state: idle

client -> pstat: observe cm status
pstat --> client : cm: bits zero

== software upgrade initialisation ==

client -> firmware : set purl (external server)
firmware -> client: ok

== trigger software upgrade ==

client -> pstat : Initiate Secure Software Update
pstat -> firmware: Initiate Secure Software Update

== new software availiblity check ==

firmware -> external: check if new software is available (purl, version)
external -> firmware : new software available
hnote over pstat,firmware
New Software Available (on external server)
end note
firmware --> pstat: state: nsa
firmware --> client: state: nsa
pstat --> client: cm New Software Available

== software download and validation ==
pstat --> pstat : tm: Initiate Software Version Validation\n (internal)
pstat --> client : cm: Software Version Validation
pstat -> firmware : Initiate Software Version Validation
firmware --> client : state: svv

firmware -> external : retrieve package
hnote over firmware
downloading software
end note
external -> firmware : package complete

hnote over firmware
software downloaded
end note
```

```
hnote over pstat,firmware
software validation: ok
end note

firmware --> pstat : state: sva
firmware --> client : state: sva

pstat --> client : cm: valid version available

== upgrade image ==

pstat --> pstat : Initiate Secure Software Update
pstat -> firmware : Initiate Secure Software Update
pstat --> client : cm: upgrading
firmware --> client : state: upgrading

hnote over pstat,firmware
firmware upgraded
end note

pstat --> pstat : tm triggers cm (bits zero)
pstat --> client : cm: idle

firmware --> client : state: idle (finished, version= new version)

@enduml
```

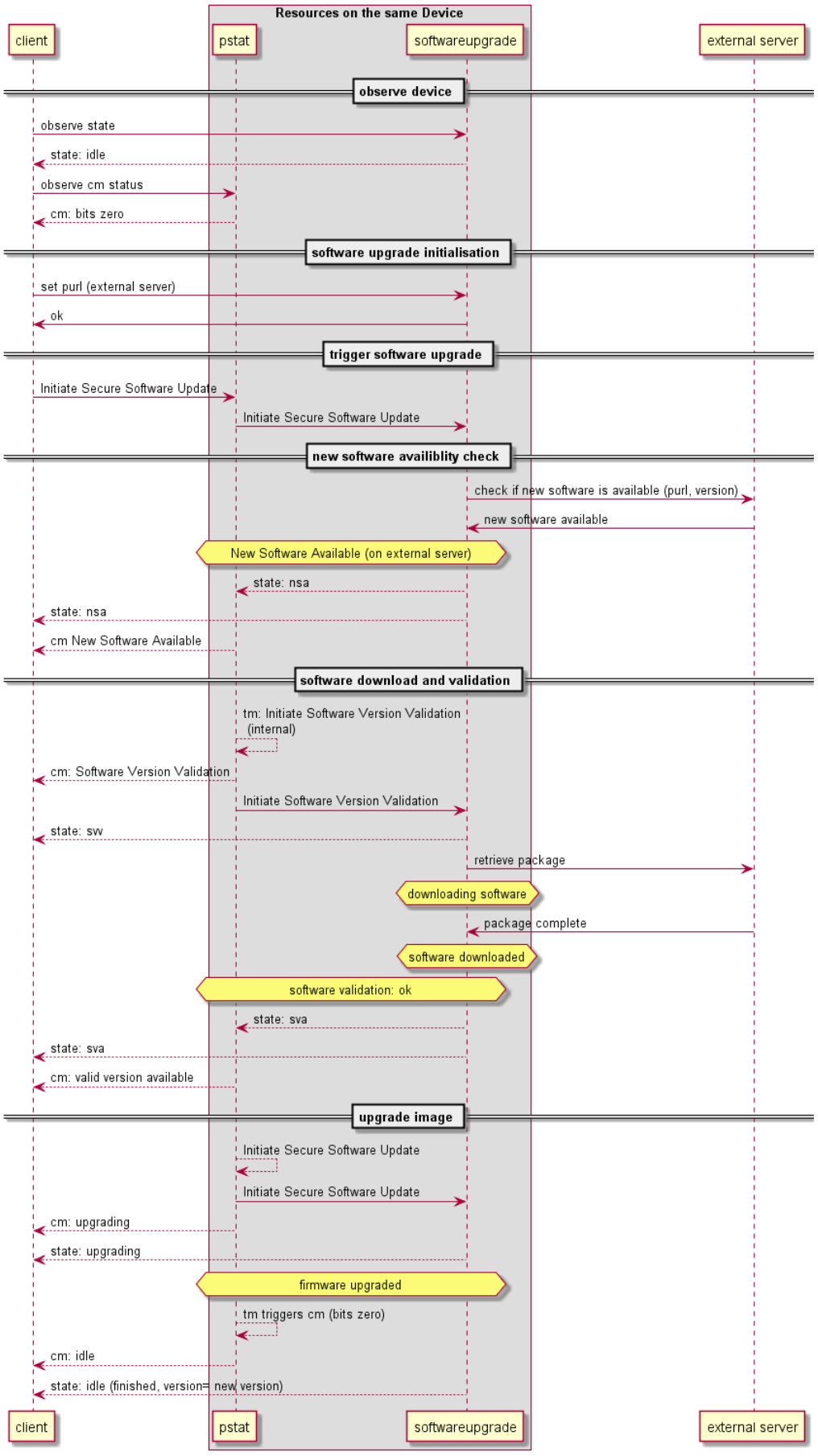

Figure 2. Typical sequence for none scheduled upgrading software

A.1 software update

A.1.1 Introduction

The resource to do scheduled update

A.1.2 Wellknown URI

/softwareupdateResURI

A.1.3 Resource type

The resource type (rt) is defined as: ['oic.r.softwareupdate'].

A.1.4 Swagger 2.0 definition

```
{
  "swagger": "2.0",
  "info": {
    "title": "software update",
    "version": "v1-20181210",
    "license": {
      "name": "copyright 2016-2017 Open Connectivity Foundation, Inc. All rights reserved.",
      "x-description": "Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:\n        1.
Redistributions of source code must retain the above copyright notice, this list of conditions and the
following disclaimer.\n        2.  Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the documentation and/or other
materials provided with the distribution.\n\n        THIS SOFTWARE IS PROVIDED BY THE Open
Connectivity Foundation, INC. \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR
WARRANTIES OF NON-INFRINGEMENT, ARE DISCLAIMED.\n        IN NO EVENT SHALL THE Open Connectivity
Foundation, INC. OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,
OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)\n        HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.\n"
    }
  },
  "schemes": ["http"],
  "consumes": ["application/json"],
  "produces": ["application/json"],
  "paths": {
    "/softwareupdateResURI" : {
      "get": {
        "description": "The resource to do scheduled update",
        "parameters": [
          {"$ref": "#/parameters/interface-all"}
        ],
        "responses": {
            "200": {
              "description" : "",
              "x-example":
                {
                  "rt": ["oic.r.softwareupdate"],
                  "nv" : "my version",
                  "purl": "https://myvendor/myexampleurl",
                  "swupdateaction": "idle",
                  "swupdatestate" : "idle",
                  "swupdateresult" : 0,
                  "lastupdate" : "2015-01-09T14:30Z",
                  "signed" : "vendor",
                  "updatetime" : "2015-01-09T14:30Z"
                }
                ,
              "schema": { "$ref": "#/definitions/swupdate" }
            }
```

```
        }
      },
      "post": {
        "description": "Start/Stop collecting and reset the networking monitor resource\n",
        "parameters": [
          {"$ref": "#/parameters/interface-rw"},
          {
            "name": "body",
            "in": "body",
            "required": true,
            "schema": { "$ref": "#/definitions/swupdate-update" },
            "x-example":
              {
                  "purl":  "https://myvendor/newversion",
                  "swupdateaction": "upgrade",
                  "updatetime" : "2030-01-09T14:30Z"
              }
          }
        ],
        "responses": {
            "200": {
              "description" : "",
              "x-example":
                {
                  "rt":   ["oic.r.softwareupdate"],
                  "nv" : "my new version",
                  "purl":  "https://myvendor/myexampleurl",
                  "swupdateaction": "upgrade",
                  "swupdatestate" : "idle",
                  "swupdateresult" : 0,
                  "lastupdate" : "2015-01-09T14:30Z",
                  "signed" : "vendor",
                  "updatetime" : "2030-01-09T14:30Z"
                },
              "schema": { "$ref": "#/definitions/swupdate" }
            }
        }
      }
    }
  },
  "parameters": {
    "interface-rw" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
      "enum" : ["oic.if.rw"]
    },
    "interface-all" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
      "enum" : ["oic.if.rw", "oic.if.baseline"]
    }
  },
  "definitions": {
    "swupdate" : {
      "properties": {
        "rt" : {
          "items": {
            "enum": [
              "oic.r.softwareupdate"
            ],
            "type": "string"
          },
          "minItems": 1,
          "maxItems": 1,
          "type": "array",
          "readOnly": true,
          "uniqueItems": true
        },
        "if": {
```

```
          "description": "The interface set supported by this resource",
          "items": {
            "enum": [
              "oic.if.baseline",
              "oic.if.rw"
            ],
            "type": "string"
          },
          "minItems": 2,
          "maxItems": 2,
          "type": "array",
          "readOnly": true,
          "uniqueItems": true
        },
        "n": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
schema.json#/definitions/n"
        },
        "id": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
schema.json#/definitions/id"
        },
        "nv" : {
          "description": "New available Software version",
          "maxLength": 64,
          "type": "string",
          "readOnly" : true
        },
        "purl" : {
          "description": "Source of the software package, might be a HTTPS or CoAPs URL",
          "maxLength": 64,
          "type": "string",
          "format": "uri"
        },
        "swupdateaction" : {
          "description": "Scheduled action to do a software update",
          "maxLength": 64,
          "type": "string",
          "enum": [
              "idle",
              "isac",
              "isvv",
              "upgrade"
            ]
        },
        "swupdatestate" : {
          "description": "State of the software update",
          "readOnly": true,
          "type": "string",
          "enum": [
              "idle",
              "nsa",
              "svv",
              "sva",
              "upgrading"
            ]
        },
        "swupdateresult" : {
          "description": "Result of the software update, list of result codes",
          "readOnly": true,
          "type": "integer"
        },
        "lastupdate" : {
          "description": "Time of the last software update (in RFC3339 format), Initial set on date of
manufacturing",
          "readOnly": true,
          "maxLength": 64,
          "type": "string",
          "format": "date-time"
```

```
        },
        "signed" : {
          "description": "Signage method of the software package, currently the only allowed value is
'vendor'.",
          "readOnly": true,
          "type": "string",
          "enum": [
              "vendor"
            ]
        },
        "updatetime" : {
          "description": "Scheduled time (in RFC3339 format) to do action which is specified in
'swupdateaction' Property.",
          "maxLength": 64,
          "type": "string",
          "format": "date-time"
        }
      },
      "required": ["purl", "swupdateaction", "swupdatestate", "swupdateresult", "updatetime"]
    },
    "swupdate-update" : {
      "properties": {
        "purl" : {
          "$ref": "#/definitions/swupdate/purl"
        },
        "swupdateaction" : {
          "$ref": "#/definitions/swupdate/swupdateaction"
        },
        "updatetime" : {
          "$ref": "#/definitions/swupdate/updatetime"
        }
      },
      "required": ["purl", "swupdateaction", "updatetime"]
    }
  }
}
```

A.1.5 Property definition

Table 6 The properties definitions of the resource with type 'rt' = softwareupdateResURI

Property name	Value type	Mandatory	Access mode	Description
rt	array: see schema	No	Read Only	
if	array: see schema	No	Read Only	The interface set supported by this resource
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
nv	string	No	Read Only	New available Software version
purl	string	Yes	Read Write	Source of the software package, might be a HTTPS or CoAPs URL
swupdateaction	string	Yes	Read Write	Scheduled action to do a

				software update
swupdatestate	string	Yes	Read Only	State of the software update
swupdateresult	integer	Yes	Read Only	Result of the software update, list of result codes
lastupdate	string	No	Read Only	Time of the last software update (in RFC3339 format), Initial set on date of manufacturing
signed	string	No	Read Only	Signage method of the software package, currently the only allowed value is 'vendor'.
updatetime	string	Yes	Read Write	Scheduled time (in RFC3339 format) to do action which is specified in 'swupdateaction' Property.
purl	multiple types: see schema	Yes	Read Write	
swupdateaction	multiple types: see schema	Yes	Read Write	
updatetime	multiple types: see schema	Yes	Read Write	

A.1.6 CRUDN behaviour

Table 7 The CRUDN operations of the resource with type 'rt' = softwareupdateResURI

Resource	Create	Read	Update	Delete	Notify
/softwareupdateResURI		get	post		observe