OCF Kyiv – Technical WG – CR 2483 (Push Notification)

Legal Disclaimer

****** Change #1 (changed text) ******

7.8.2.1.2 “p” or Policy Parameter

The Policy Parameter defines various rules for correctly accessing a Resource referenced by a target URI. The Policy rules are configured by a set of key-value pairs as defined below.

The policy Parameter "p" is defined by:

“bm” key: The “bm” key corresponds to an integer value that is interpreted as an 8-bit bitmask. Each bit in the bitmask corresponds to a specific Policy rule. The following rules are specified for “bm”:

Bit Position	Policy rule	Comment
Bit 0 (the LSB)	discoverable	The discoverable rule defines whether the Link is to be included in the Resource discovery message via “/oic/res”. • If the Link is to be included in the Resource discovery message, then “p” shall include the “bm” key and set the discoverable bit to value 1. • If the Link is NOT to be included in the Resource discovery message, then “p” shall either include the “bm” key and set the discoverable bit to value 0 or omit the “bm” key entirely.
Bit 1 (2nd LSB)	observable	The observable rule defines whether the Resource referenced by the target URI supports the NOTIFY operation using an observe pattern (see section 11.4.1). With the self-link, i.e. the Link with "rel" value of "self", “/oic/res” can have a Link with the target URI of “/oic/res” and indicate itself observable. The "self" is defined by IETF RFC 4287 and registered in the IANA Registry for "rel" value defined at IANA Link Relations. • If the Resource supports the NOTIFY operation via the use of observe, then ”p” shall include the “bm” key and set the observable bit to value 1. • If the Resource does NOT support the NOTIFY operation via the use of observe, then “p” shall either include the “bm” key and set the observable bit to value 0 or omit the “bm” key entirely.
Bit 2 (3rd LSB)	pushable	The pushable bit defines whether the Resource referenced by the target URI supports the NOTIFY operation using a push pattern (see section 11.4.1). With the self-link, i.e. the Link with "rel" value of "self", “/oic/res” can have a Link with the target URI of “/oic/res” and indicates itself pushable. The "self" is defined by IETF RFC 4287 and registered

		in the IANA Registry for "rel" value defined at IANA Link Relations. • If the Resource supports the NOTIFY operation via the use of push, then "p" shall include the "bm" key and set the pushable bit to value 1. • If the Resource does NOT support the NOTIFY operation via the use of push, then "p" shall either include the "bm" key and set the pushable bit to value 0 or omit the "bm" key entirely.
Bits 32-7	--	Reserved for future use. All reserved bits in "bm" shall be set to value 0.

Note that if all the bits in "bm" are defined to value 0, then the "bm" key may be omitted entirely from "p" as an efficiency measure. However, if any bit is set to value 1, then "bm" shall be included in "p" and all the bits shall be defined appropriately.

****** Change #2 (new text) ******

11.4.3 Push Notification

11.4.3.1 Overview

A Server may be configured to provide a NOTIFICATION via a push mechanism rather than a pull mechanism (i.e. Observe, see Section 11.4.2). That is rather than a Client establishing one or more discrete Observe transactions with a Server, a Client may configure a Server such that an embedded Client within the Server pushes observable events to a defined destination.

The general principle is that a Server provides the representation of any Resource that is the subject of an UPDATE operation or an internal (to the Server) change in the represented Properties via the use of an embedded Client in an UPDATE operation to a pre-configured target (destination). The set of Resources for which information is pushed in this manner is defined by configuration that exists on the Server. ~~may be all of the Resources hosted by the Server or a specific sub-set of all Resources hosted by the Server.~~

11.4.4 Architectural Model

There are four logical elements that make up the architecture for push notifications. These being: the origin Server that hosts the Resources, an embedded Client within the origin Server that originates the push notifications ~~requests~~, a target Server that is the recipient of the ~~NOTIFICATION~~ push notifications ~~requests~~, and a Client that configures both the origin Server and (optionally) the target Server.

The Client providing configuration of the origin and target Servers may be a discrete Client or embedded in the target Server.

This architecture is illustrated in Figure XX.

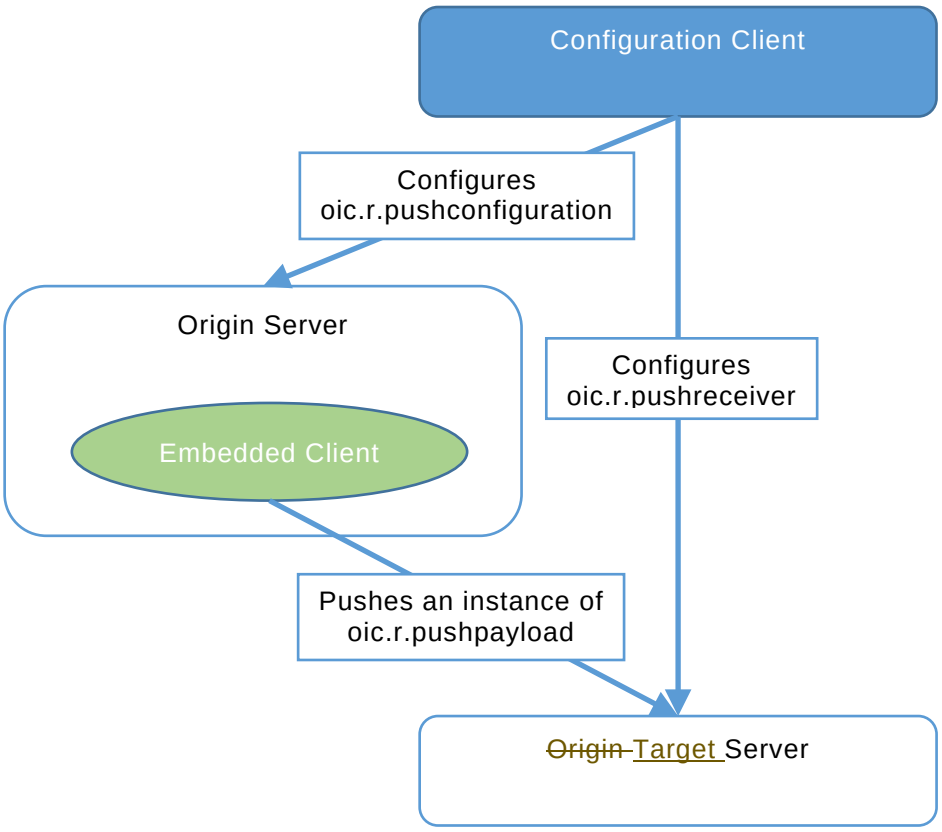

Figure XX. Push Architecture

As illustrated in Figure XX; a push relationship can be established by a 3rd party, that is by a Client that is not part of the originator of the notifications nor part of the recipient of the notifications. This further distinguishes the push model from an Observe model, for the latter the establishment of the Observe transaction is done by the Client that is the intended recipient of the notifications.

Figure ZZ is an example sequence diagram showing how the elements of the architecture may interact in realising push notifications.

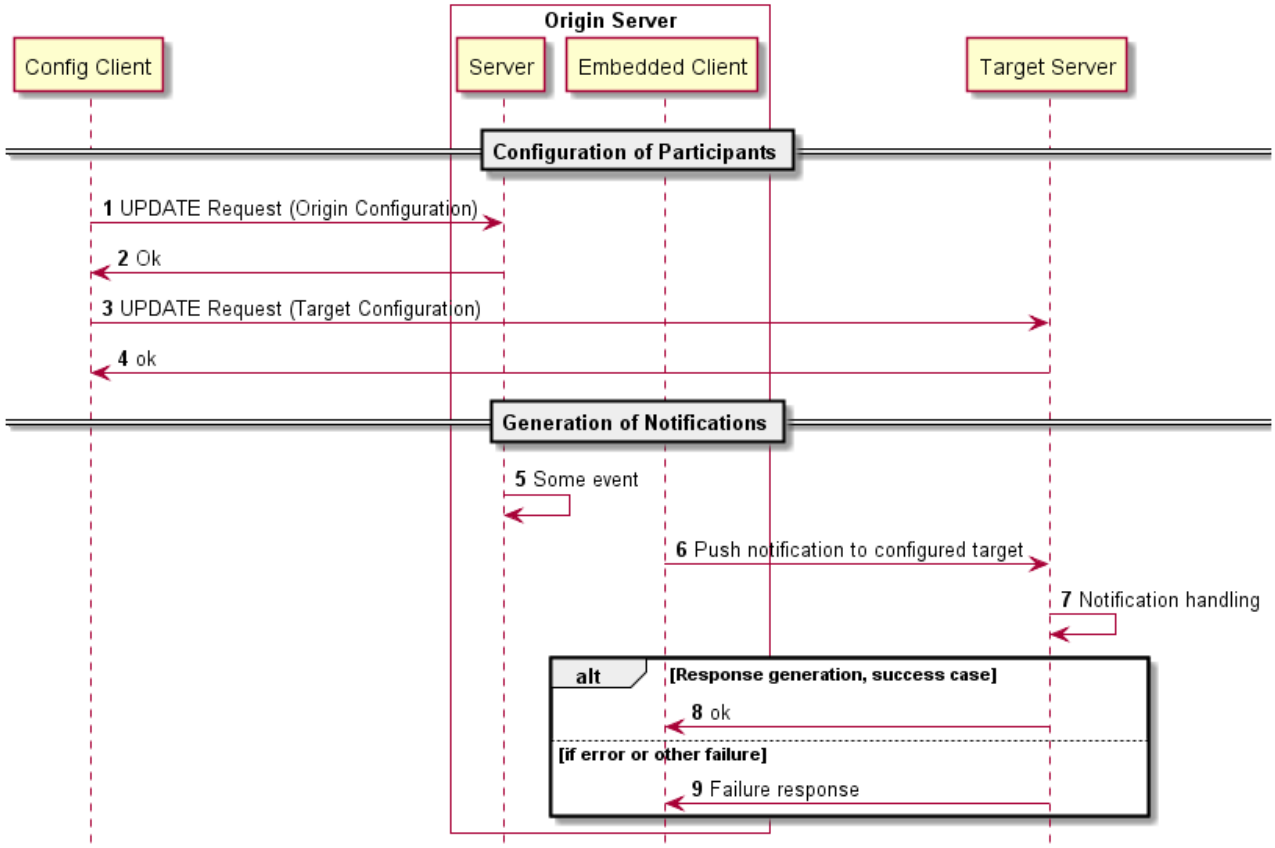

Figure XX. Example Push Sequence

Table XX provides additional details for the steps captured in Figure XX.

Table ZZ. Example Push Sequence Details

Step	Description
1	The Client configuring the Server that will originate the push notifications via an embedded Client provisions the instance of "oic.r.pushconfiguration" with notification selectors(s); each notification selector identifies a set of candidate Resources from which information may be pushed. The destination for the pushed information is established by composing a Push Proxy with a specific notification selector(s).
2	The Server provides a success path response
3	The Client configuring the Server that will be the recipient of the push notifications provisions the instance of "oic.r.pushreceiver" with the URLs at which notifications may be received and if there are limitations, the Resource Types that may be supported.
4	The Server provides a success path response
5	Some event occurs on the origin server that results in a state change in a Resource identified by one or more notification selectors
6	The embedded Client within the origin Server generates an UPDATE request to destination identified by the composed Push Proxy for the notification selector. The payload for the UPDATE request is an instance of "oic.r.pushpayload"
7	The target Server handles the received push notification in accordance with its own internal application logic
8	If no errors, the target Server provides a success path response
9	If errors occur, the target Server provides a non-success path response

11.4.5 Origin Server Configuration

11.4.5.1 Overview

A Server that ~~wishes to~~ allow~~s~~ for the configuration of one or more target Servers for push notifications shall expose an instance of "oic.r.pushconfiguration", this is a Collection containing instances of a notification selector ("oic.r.notificationselector"). Composed with the instance of a notification selector is a Push Proxy ("oic.r.pushproxy"); the Push Proxy provide~~s~~~~ing~~ the target for the information that is pushed as a result of the application of the notification selector. ~~A notification selector is not required to have a Push Proxy composed with it.~~

Please see Figure XX for a pictorial representation of "oic.r.pushconfiguration":

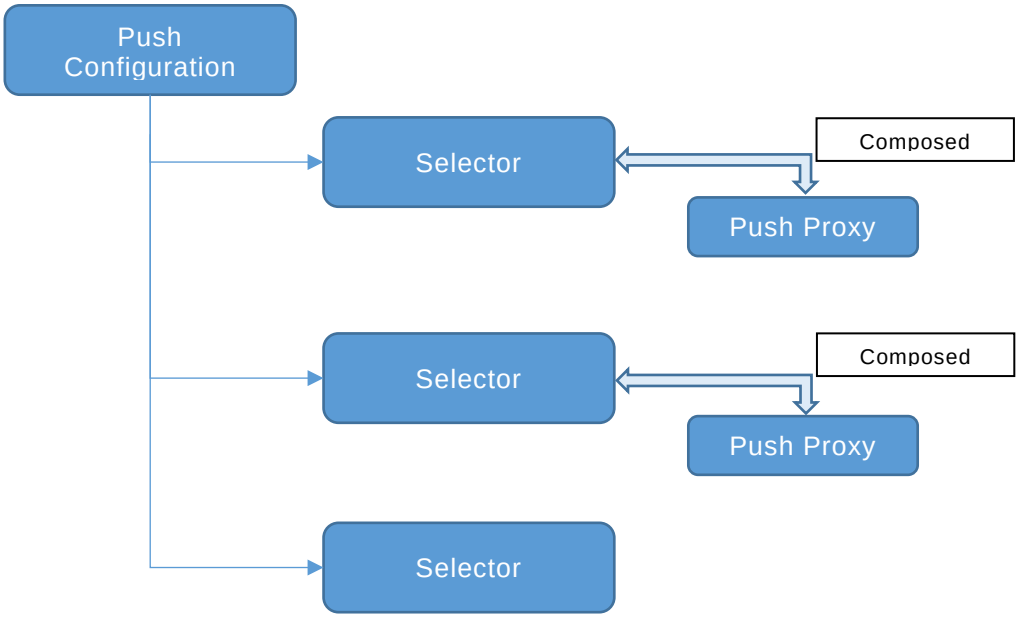

Figure XX. Example Pictorial Push Configuration Collection

The Push Proxy (see Section 11.4.5.3 ~~XXX~~) defines the target of the pushed information, the instance of "oic.r.notificationselector" provides Properties that allow for selection of specific Resources to be pushed. Within an instance of "oic.r.notificationselector" are optional Properties "phref", "prt", and "pif" which are applied in a similar manner to a query parameter in a URI (see clause 7.10), and when applied, only Resources that are identified as a result of application of the Properties may be pushed. When multiple Properties are present a logical "and" operation shall ~~is~~be applied between them. ~~If the instance of "oic.r.notificationselector" is an empty object (i.e. "{}") then all Resources exposed by the Server may be pushed.~~

The set of Resources that may be pushed is established when the notification selector is configured.~~,~~ The exception to this are~~but~~ any Resources that have been ~~are~~ dynamically created on the Server ~~dynamically~~via use of the "oic.if.create" OCF Interface; these shall not ~~may not~~ be pushed even though there ~~are~~may be ~~any~~ notification selectors that match them. ~~because it is barely difficult to detect state change of the dynamically created Resources in terms of implementation perspective. thus any Resources that may be created on the Server after the establishment of push notifications shall not be included in a push payload. A Client may trigger re-evaluation of the pushed Resource set by re-configuring the Push Proxy (see clause XXX).~~

NOTE: In all cases, only those Resources that are marked as "pushable" in the Policy Link Parameter in “/oic/res” may be pushed.

11.4.5.2 Push Proxy Resource Type

The Push Proxy Resource Type is defined in Table XX.

Table XX. Resource Types for Push Proxy

Example URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/exampleResourceURI	Push Proxy	"oic.r.pushproxy"	"oic.if.rw", "oic.if.baseline"	Adds Push Proxy functionality to a Source Resource	Notifications

The Push Proxy Resource Type is composed with a Notification Selector Resource Type, this enables notifications to be pushed to a defined target Resource when the Resources identified by applying the Notification Selector are updated, or when their state changes. Therefore an instance of "oic.r.pushproxy" shall always be composed with an instance of "oic.r.notificationselector".

The Properties of the Push Proxy Resource Type are defined in Table 2.

Table 2. Push Proxy Resource Property definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Push Target URI	pushtarget	string	URI		RW	yes	Points to the target of the UPDATE operation sent as a notifcation
Source Resource Type	sourcert	array of string			RW	yes	Always set to "oic.r.pushpayload"
State	state	string enumeration	One of ["waitingforprovisioning","waitingforupdate","waitingforresponse", " waitingforupdatemitigation", " waitingforresponsemitigation", "error","timeout"]		RW	yes (RETRIEVE), no (UPDATE)	Current state of the Push Proxy

The "Push Target URI" Property, "pushtarget", contains a pointer to the target Resource, that is the Resource to which the notification shall be sent. The URI reference may be to either a local resource name, for example "output1", or a URI on the local device, for example "/scenes/athome", or a full URI including scheme and authority components, for example "ocf://850faa5d-ccaf-4293-9452-f4fcab2e2c39/scenes/atwork". Note that when the URI uses the "ocf" scheme then the URI shall be resolved to a transport protocol URI as defined in the OCF Core Specification before use in an UPDATE operation. The "pushtarget" may also be an empty string (""); in which case there is no associated target Resource and so no notifications may be sent; this occurs when a Push Proxy is pre-configured with other Resources on a Device.

The "Source Resource Type" Property, "sourcert", shall be set to "oic.r.pushpayload". Future use cases may be defined that add to the set of possible Resource Types, for the purposes of this document, the Property shall only contain "oic.r.pushpayload".

The Property "state" contains the current or desired state of the Push Proxy. When provided in a RETRIEVE response the value shall be one of defined enumeration values in Table 3. When used in an UPDATE operation the only value that shall be accepted by a Device receiving the UPDATE is "waitingforupdate", other values shall result in the UPDATE operation being rejected. Also, an

UPDATE that sets "state" is only valid when received by a Push Proxy in a state of "timeout" or "error"; reception in any other state shall result in the operation being rejected.

11.4.5.3 Push proxy state machine

The Device that exposes a Push Proxy Resource Type realizes an individual instance of a state machine per Push Proxy.

Table 3 describes the states of which the state machine is composed.

Table 3. Push Proxy States

State Friendly Name	State enumeration	State Acronym	Description
Waiting for Provisioning	waitingforprovisioning	WFP	Proxy exists but the "pushtarget" Property is an empty string (i.e. not populated).
Waiting for Update	waitingforupdate	WFU	Proxy has been configured and pending Resource state change.
Waiting for Response	waitingforresponse	WFR	Resource state has changed, UPDATE sent to the push target.
Waiting for Update Mitigation	waitingforupdatemitigation	WFUM	UPDATE has been sent to the push target and error situation occurs (transport layer timer expiration or error response) while Proxy has an error mitigation algorithm to handle this situation.
Waiting for Response Mitigation	waitingforresponsemitigation	WFRM	Resource state has changed. UPDATE sent to the push target by the error mitigation algorithm.
Error Response Received	error	ERR	Non-success path response received for the UPDATE operation.
Timeout Condition Detected	timeout	TOUT	Transport layer timeout detected waiting for a response to the UPDATE operation.

The following provides more detail with respect to the behaviour of the push proxy in each defined state:

– **"waiting for provisioning" (WFP)** state shall be entered when the "pushtarget" Property of a Resource that includes the Push Proxy Resource Type is an empty string (i.e. ""). This may be an initial state in the case where a Push Proxy is pre-configured on a Device, or a state that is entered following configuration of the Property by a Client. In the WFP state no action shall be taken on a state change. An existing Push Proxy not initially in this state may only enter this state from the "waiting for update" (WFU) state.

– **"waiting for update" (WFU)** state shall be entered after successful creation of a Resource that includes a Push Proxy Resource Type, and also after the successful completion of an UPDATE operation triggered by a state change of the source Resource. When the source Resource is successfully updated or a state change occurs, an UPDATE request shall be constructed and sent

to the target Resource, and the "waiting for response" (WFR) state is entered. An UPDATE request to the source Resource may be acknowledged before the triggered UPDATE request to the target Resource is sent.

– **"waiting for response" (WFR)** state shall be entered when an UPDATE request is sent to a target Resource. If no response is received before any applicable transport layer timers expire, then the "timeout" (TOUT) state is entered. If an error response is received, then the "error" (ERR) state is entered. If a non-error response is received, then the "waiting for update" (WFU) state is entered.

– **"waiting for update mitigation" (WFUM)** state shalle be entered from "waiting for response" (WFR) state only if there is an error mitigation algorithm to handle error situations (transport layer timer expiration or error response) when the error situation occurs. The algorithm starts when the Push Proxy enters this state. In this state the Push Proxy may try to send another UPDATE request and enter "waiting for response mitigation" (WFRM) state. The Push Proxy enters this state from "waiting for response mitigation" (WFRM) state when the error situation continues to occur.

– **"waiting for response mitigation" (WFRM)** state shall be entered from "waiting for update mitigation" (WFUM) state after an UPDATE request has been sent to a target Resource. If any error situation (transport layer timer expiration or error response) occurs again, then "waiting for update mitigation" (WFUM) state shall be entered to continue error mitigation algorithm. If a non-error response is received, then the error mitigation algorithm is stopped and the "waiting for update" (WFU) state shall be entered, or if the error mitigation algorithm fails in the end, the proper error state (TOUT or ERR) is entered.

– **"timeout condition detected" (TOUT)** state shall be entered when any applicable transport layer response timer expires. A Client may force a transition from the "timeout condition detected" (TOUT) state to the "waiting for update" (WFU) state by sending an UPDATE request to the Push Proxy Resource that sets the "state" Property to "waiting for update" (WFU). Otherwise, the Push Proxy remains in this state until the Push Proxy composition is deleted. .

– **"error response received" (ERR)** state shall be entered when an error response is received from the target Resource in response to an UPDATE request. A Client may force a transition from the "error response received" (ERR) state to the "waiting for update" (WFU) state by sending an UPDATE request to the Push Proxy Resource that sets the "state" Property to "waiting for update" (WFU). Otherwise, the Push Proxy remains in this state until the Push Proxy composition is deleted.

Figure 1 shows the Push Proxy operational state machine.

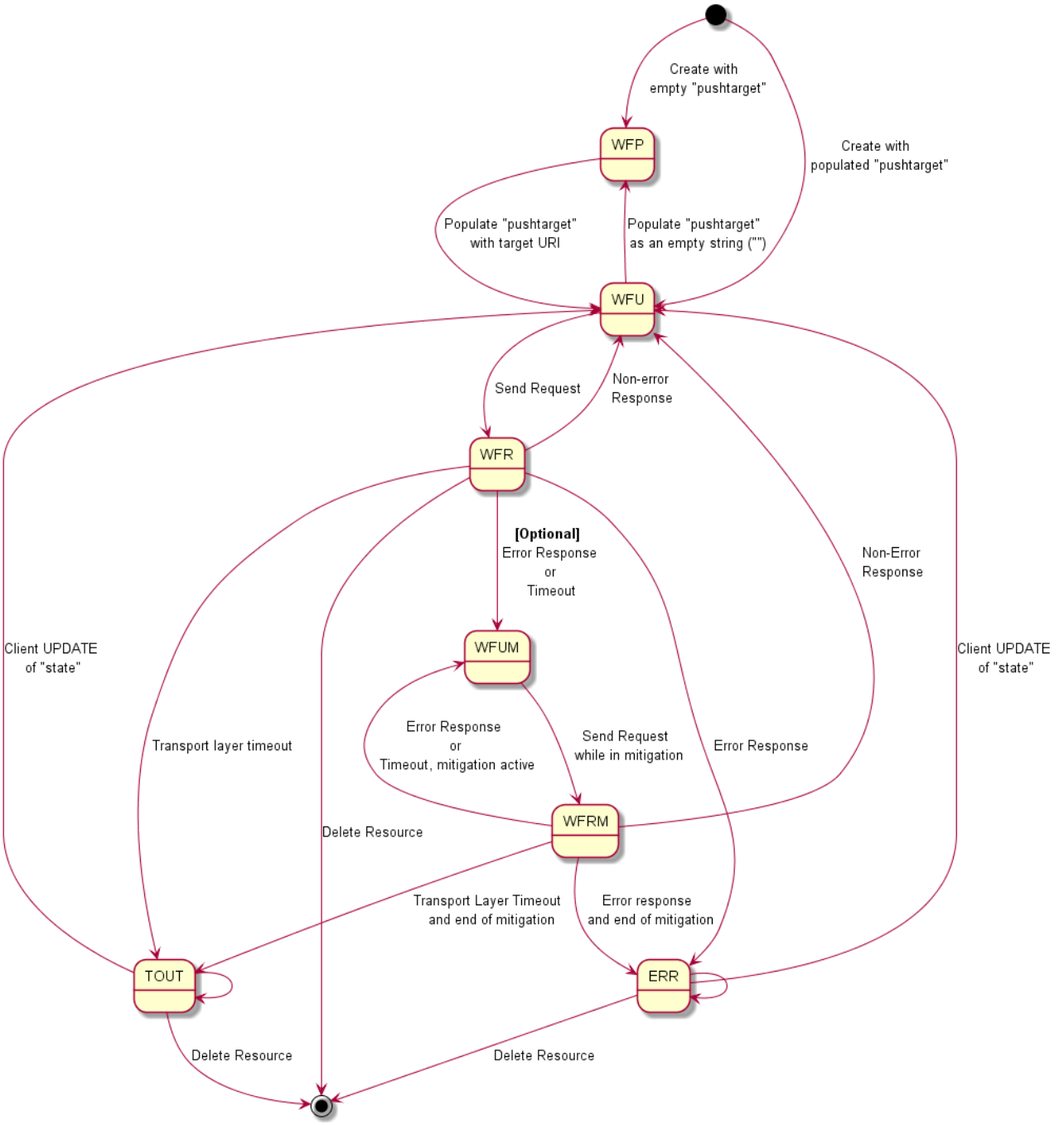

Figure 1. Push Proxy Operational State Machine

The Push Proxy is initialized when the multi-value "rt" Resource that is composed with the Push Proxy Resource Type is created. This may be done by a Client using the "oic.if.create" OCF Interface or may be already composed with other exposed Resources as part of the Device implementation. With respect to the Client creation case, please see the example in Figure 2. In all cases, a configured Push Proxy is in the "wait for update" state.

Figure 2. Creating a Push Proxy Resource

```
Method = CREATE
URI =
/dynamic/somecollection?if=oic.if.create&rt=oic.wk.col
Payload =

{
 "rt": ["oic.r.notificationselector", "oic.r.pushproxy"],
 "if": ["oic.if.baseline", "oic.if.a", "oic.if.rw"],
 "p": {"bm":3},
 "rep": {
  "value": false,
  "pushtarget": " ocf://850faa5d-ccaf-4293-9452-
f4fcab2e2c39/power/switch",
  "sourcert": ["oic.r.pushpayload"]
 }
}

Response = CREATED
Payload =
{
 "href": "02367721",
 "ins": 02367721,
 "rt": ["oic.r.notificationselector", "oic.r.pushproxy"],
 "if": ["oic.if.baseline", "oic.if.a", "oic.if.rw"],
 "p": {"bm":3},
 "rep": {
  "value": false,
  "pushtarget": "ocf://850faa5d-ccaf-4293-9452-
f4fcab2e2c39/power/switch",
  "sourcert": ["oic.r.pushpayload"]
 }
}
```

When a notification is triggered due to a change in the set of Resources identified by the Notification Selector, an UPDATE request is sent to the target Resource, and the Push Proxy enters the "wait for response" state. When a non-error response is received, the Push Proxy returns to the "wait for update" state.

11.4.5.5 Push Proxy Life Cycle

Figure 4 shows a life cycle example of a Push Proxy

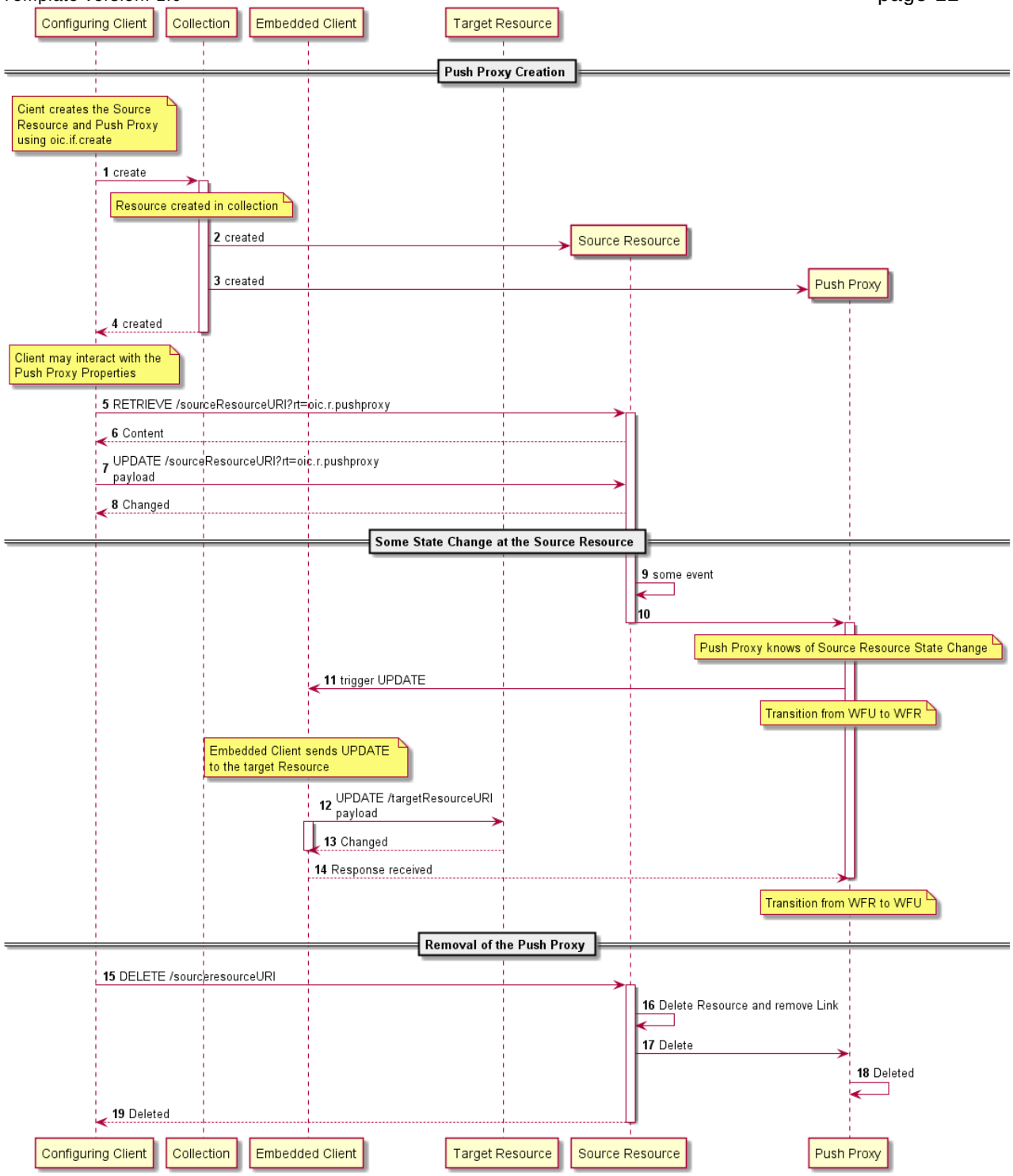

Figure 4. Push Proxy Life Cycle Example

11.4.5.6 Security Considerations

– All requirements with regard to the behaviour of a Push Proxy as defined in the OCF Security Specification shall be met.

~~11.4.5.5~~11.4.5.7 Push Configuration and Notification Selector Resource Types

The Push Configuration and Notification Selector Resource Types are as defined in Table XX.

Table XX. Optional Push Notification Core Resources for Server Configuration

Example URI	Resource Type Title	Resource Type ID ("rt" value)	Interfaces	Description	Related Functional Interaction
"/example/notificationselectorURI"	Notification Selector	"oic.r.notificat~~e~~ionselector"	"oic.if.rw", "oic.if.baseline"	The Resource through which the ~~Device~~ Server is~~can be~~ configured with a selector for push notifications. The Properties exposed by the Resource are listed in Table YY.	Notifications
"/example/pushconfigurationURI	Push Configuration	"oic.r.pushconfiguration"	"oic.if.ll", "oic.if.create", ~~"oic.if.delete",~~ "oic.if.baseline"	A specialization of a Collection that contains only instances of "oic.r.notificationselector" with composed Push Proxy.	Notifications

Table YY defines the details for the "oic.r.notificationselector" Resource Type.

Table YY. "oic.r.notificationselector" Resource Type definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Pushed Resource URI	phref	string			RW	no	URI of a Resource to be pushed
Pushed Resource Type	prt	array	array of strings		RW	no	Resource type(s) of Resource(s) to be pushed
Pushed Interface	pif	array	array of strings		RW	no	OCF Interface(s) of Resource(s) to be pushed

The Push Configuration ("oic.r.pushconfiguration") Resource Type defines no Properties additional to those defined for all instances of a Collection in Table 9. However, the Push Configuration does impose restrictions of the values that shall be populated in the "rt" and "rts" Properties. These are described in Table ZZ below.

Table ZZ. "oic.r.pushconfiguration" Resource Type definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Links	links	Array	See Table 9		RW	yes	See Table 9
Resource Type	rt	array	["oic.r.pushconfiguration"]		R	yes	See Table 4.
Resource Types	rts	array	["oic.r.notificationselector","oic.r.pushproxy"]		R	yes	See Table 9

~~11.4.5.6~~11.4.5.8 Push Configuration Collection Manipulation

Instances of a notification selector with an optionally composed Push Proxy may be created using the "oic.if.create" OCF Interface defined in Section XXX.

A Server may expose an instance of "oic.r.pushconfiguration" with a pre-configured notification selector and composed Push Proxy if the target is known via other means; in this case a Client may be able to update the Push Proxy target via the "oic.if.rw" OCF Interface exposed by the composed Resource.

~~11.4.5.7~~11.4.5.9 Notification Selector Population

The notification selector Resource is an object that contains Properties that may be applied as a selection filter to the set of possible Resources to be pushed. If the Resource is an empty object (i.e. "{}") ~~all Resources that are marked as "pushable" in "/oic/res" may be pushed~~then nothing shall be pushed. As such, a selector does not necessarily correspond to a single Resource on the origin Server, it may resolve to multiple Resources, and in the absence of a selector, resolution results in ~~all~~ no Resources ~~that are marked as "pushable"~~.

The Properties within the notification selector align with the set of Link Parameters that may exist for a Resource, thus matching semantically the actions of a query parameter.

Figure X shows an example notification selector that limits the Resources to be pushed to those that have the Resource Type that is listed. The result of the use of this selector is that all Resources on the origin Server, with an "rt" of "oic.r.sensor.carbonmonoxide", that are marked as "pushable", may have push notifications generated to the configured target Server whenever there is a state change in those Resources.

```
{"prt":["oic.r.sensor.carbonmonoxide"]}
```

Figure X. notification selector example for the given "prt"

Figure Y shows an example notification selector that limits the Resources to be pushed to the Resource at the Resource URI that is listed. The result of the use of this selector is that the Resource on the origin Server identified with an "href" of "/myDevice/mySelectedResource", if it is marked as "pushable", may have push notifications generated to the configured target Server whenever there is a state change in that Resource.

```
{"phref":"/myDevice/mySelectedResource"}
```

Figure Y. notification selector for the given "phref"

~~11.4.5.8~~11.4.5.10 Notification Selector Operational Considerations

~~11.4.5.8.1 Selected Resource is a Collection~~

~~If one (or more) of the Resources matching the notification selector is a Collection then a push notification shall only be generated for events on the Collection when the "pushqif" Property of "oic.r.pushproxy" contains an OCF Interface via which a Collection can be observed (e.g. "oic.if.ll", "oic.if.b").~~

~~If one (or more) of the Resources matching the notification selector is an Atomic Measurement then a push notification shall only be generated for events on the Atomic Measurement when the "pushqif" Property of "oic.r.pushproxy" contains an OCF Interface via which an Atomic Measurement can be observed (e.g. "oic.if.b").~~

~~11.4.5.8.2~~11.4.5.10.1 No Resources Match the Selector

If no Resources match the provisioned notification selector, then the set of Resources to be pushed is effectively an empty set and no notifications shall be~~are~~ generated by the Server. ~~The composed Push Proxy shall be~~is set to a state of "waiting for provisioning".

If a Resource that was selected ~~set~~ via application of the notification selector is deleted, leaving the set of Resources to be pushed as an empty set, no notifications are generated by the Server. ~~The composed Push Proxy is set to a state of "waiting for provisioning".~~

~~11.4.5.9~~11.4.5.11 Push Proxy Population Considerations

Associated with each Resource linked from the Push Configuration there shall be a composed Push Proxy, and the following requirements apply to the Push Proxy Resource in such cases.

For each Push Proxy, the "~~href~~pushtarget" Property is populated with the target URI for the push. The target provided by the Push Proxy shall not be a multicast address. If a Client attempts to configure such an address the Server shall reject the request with a response code indicating bad request.

For each Push Proxy, if a Client attempts to configure the "sourcert" Property to any value other than "oic.r.pushpayload", then the Server shall reject the request with a response code indicating bad request.

Figure Z provides an example of a notification selector Resource composed with a Push Proxy that results the sending of notifications to the target identified by the "pushtarget" Property.

```
{
"rt": ["oic.r.notificationselector","oic.r.pushproxy"],
"phref":"/myDevice/mySelectedResource",
"pushtarget": "~~coaps~~ocf://myTarget/myTargetURI",
"sourcert": "oic.r.pushpayload",
~~"pushqif": "oic.if.rw"~~"state": "waitingforprovisioning"
}
```

Figure Z. Example composed notificationselector and pushproxy

11.4.6 Target Server Configuration

All push notifications contain a payload of type "oic.r.pushpayload" (see clause 11.4.8). Within the payload ~~are is~~shall be the complete representation~~s~~ (i.e. all Properties including Common Properties) of the Resource at the origin Server ~~appropriate~~ which experienced~~to~~ the state change that generated the notification. Should the target Server have limitations with respect to the Resource Types that can be received as part of an "oic.r.pushpayload" representation that the Server may optionally be explicitly configured to define which Resource Types it is capable of receiving. This is realized by exposing an instance of "oic.r.pushreceiver" which is an object array, each object containing the URI to which information may be pushed and an instance of the “rts” Property (see section 7.8.2.4) listing the Resource Types that can be received within a pushed payload representation. If the “rts” Property is an empty array then there is no restriction in the Resource Types than can be received in an "oic.r.pushpayload" representation. A target Server shall expose an instance of "oic.r.pushreceiver" Resource.

The Push Receiver Resource Type is as defined in Table ZZ.

Table ZZ. Optional Push Receiver Core Resources for Target Server Configuration

Example URI	Resource Type Title	Resource Type ID (“rt” value)	Interfaces	Description	Related Functional Interaction
“/example/pushreceiverURI”	Push Receiver	“oic.r.pushreceiver”	“oic.if.rw”, ”oic.if.baseline”	The Resource through which a Device can be configured as a target for push notifications. The Properties exposed by the Resource are listed in Table BB.	Notifications

Table BB defines the details for the “oic.r.pushreceiver” Resource Type.

Table BB. “oic.r.pushreceiver” Resource Type definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Push Receivers	receivers	array of object			RW	~~no~~yes	Resource set that can be received by the push target

Each object within the "receivers" array is made up of two Properties, these are defined in Table CC.

Table CC. “receivers” object definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Receiver URI	receiveruri	URI			RW	yes	URI at which push notifications may be received
Resource Types	rts	array of string			RW	yes	Resource Type values that may be received at the "receiveruri"

When the configuration Client sends an UPDATE to the target Server it may use a "receiveruri" query parameter, otherwise the configuration Client should provide the whole list of the "receivers" object array which shall replace existing "receivers" Property. When the push target receives an UPDATE with a "receiveruri" query parameter, it shall only update receiver object which matches the "receiveruri" query parameter. If there is no receiver object which matches the "receiveruri" query parameter, a new receiver object shall be created.

When the configuration Client sends a DELETE to the target Server it may use a "receiveruri" query parameter, otherwise the configuration Client shall remove whole receiver object array. When the push target receives a DELETE with a "receiveruri" query parameter, it shall only remove the receiver object which matches the "receiveruri" query parameter.

An example of an instance of "oic.r.pushreceiver" is provided in Figure CC.

```
"receivers": [
   {"receiveruri": "/mylocaltargeturiforthermostats”,
    "rts": ["oic.r.temperature","oic.r.humidity"]
   },
   {"receiveruri": "/mylocaltargeturifordontcare”,
    "rts": []
   }
]
```

Figure CC. example push receiver configuration

11.4.7 Target Server Behaviour

If all "rt" values in the pushpayload are part of the configured "rts" Property against the target URI, or if the "rts" Property is empty, then the push target shall handle the operation as an ~~regular~~ UPDATE against the target URI with the caveat that the entirety of the received pushpayload is handled as a writeable entity.

If a push target receives an UPDATE operation containing a pushpayload that it is not configured to handle (i.e. an “rt” in the pushpayload is not part of the configured “rts” against the target URI) then the push target shall respond with a response of "Forbidden".

If a push target receives an UPDATE operation containing a pushpayload that it is configured to handle (i.e. all “rt” values in the pushpayload are part of the configured “rts” against the target URI) but it is unable to parse a provided Resource Representation (contents of the “rep” Property) then the push target shall respond with a response of "Bad Request".

A Server may choose to make the "receiveruri" discoverable or non-discoverable. However, if a RETRIEVE to the "receiveruri" would return an empty payload (i.e. nothing has been pushed to it) then the Resource shall be non-discoverable. ~~If nothing has been pushed to the specific target URI ("receiveruri") when a Client tries to read the specific target URI, the target Server shall respond with a response of "Not Found".~~

~~If all ""rt"" values in the pushpayload are part of the configured ""rts"" Propery against the target URI or if the "rts" Property is empty, and origin Server has proper right to update target Resource designated by target URI ("receiveruri"), the contents of the target URI shall be replaced with pushpayload.~~

11.4.8 Notification Payload

Anytime there is a change in the state of the Resources that are subject to a push notification as defined by the application of a specific notification selector, the Server shall send~~s~~ an UPDATE operation to the target defined in the "~~href~~pushtarget" Property of the Push Proxy that is composed with the notification selector. The representation provided in the UPDATE operation shall be an instance of an "oir.r.pushpayload" Resource. The Push Payload Resource is defined in Table AA; this is an array of representations where for each representation the Resource also provides additional meta-information to enable the target Server to understand the contents of the representation itself (e.g. the Resource Type, the OCF Interface that has been applied). An array is used as more than one Resource may be providing state information at any one time.

Table AA. Push Payload Resource

Example URI	Resource Type Title	Resource Type ID (“rt” value)	Interfaces	Description	Related Functional Interaction
“/example/pushpayloadURI”	Push Notification Payload	“oic.r.pushpayload”	“oic.if.r”, ”oic.if.baseline”	The Resource through which pushed notification information is provided to the push target The Resource exposes an array of JSON objects, the Properties exposed by the objects are listed in Table YY.	Notifications

Table YY defines the details for the JSON object that is carried as an array item within the “oic.r.pushpayload” Resource Type.

Table YY. “oic.r.pushpayload” array entry definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Context URI	anchor	uri			R	yes	Context URI of the Resource being pushed
URI	href	string			R	no	Server URI of the Resource being pushed
Resource Type	rt	array			R	yes	Resource Type of the Resource being pushed
Interface	if	array			R	yes	OCF Interface(s) that ~~has~~ are ~~been applied to the~~

							~~representation in "rep"~~valid for the Resource being pushed
Representation	rep	object			R	yes	Resource representation

Note that the “href”, if included, is always a relative URI to the root of the source Device (i.e. the Server that is originating the push request).

The "anchor" Property contains an OCF URI with the authority component set to the <deviceID> of the Device hosting the source Resource.

An example of an instance of "oic.r.pushpayload" is provided in Figure DD.

```
[{
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/mysensor",
    "rt": ["oic.r.sensor.carbonmonoxide"],
    "if": ["oic.if.s"],
    "rep": {
      "value": true
    }
}]
```

Figure DD. example pushpayload content

11.4.9 Observability of Notification Selectors

Notification selector does not support Observation because all notification selectors are dynamically created and deleted ones. ~~A Client may Observe an instance of a . In such cases the payload provided in the response to the Observe defined in Section 11.4.10. Observe responses shall be sent using the same logic that is applied to the selector in the push case; that is whenever any of the Resources that are identified by the application of the selector exhibit a change in state.~~

11.4.10 Common requirement for origin Server and target Server

Origin Server and target Server shall expose "oic.d.push" as a value of "rt" of "/oic/d" to show that they can push or receive push notifications.

~~11.4.10 Example of Use~~

~~See Annex D~~

****** Change #2 (changed text) ******

1 A.8 Push Configuration Resources

A.8.1 Introduction

A specialization of a Collection that contains only instances of "oic.r.notificationselector" composed with "oic.r.pushproxy". Each instance of them includes filtering parameters for the Resources to be pushed and Target Resource to which updated Resource of origin Server will be pushed.

A.8.2 Well-known URI

None

A.8.3 Resource type

The Resource Type is defined as: "oic.r.pushconfiguration".

A.8.4 OpenAPI 2.0 definition

```
{
  "swagger": "2.0",
  "info": {
    "title": "Push Configuration",
    "version": "2022-06-15",
    "license": {
      "name": "OCF Data Model License",
      "url":
"https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LICE
NSE.md",
      "x-copyright": "Copyright 2019 Open Connectivity Foundation, Inc. All rights reserved."
    },
    "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
  },
  "schemes": [
    "http"
  ],
  "consumes": [
    "application/json"
  ],
  "produces": [
    "application/json"
  ],
  "paths": {
    "/PushConfigurationResURI" : {
      "get": {
        "description": "Collection of oic.r.notificationselector with associated push proxies.\nAllows
a Server to be configured with one or more Push Notification destinations.\n",
        "parameters": [
          {"$ref": "#/parameters/interface-ll"}
        ],
        "responses": {
          "200": {
            "description" : "",
            "x-example": [
              {"href": "/pushconfig/1", "rt": ["oic.r.notificationselector","oic.r.pushproxy"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[fe80::b1d6]:1122"}]},
              {"href": "/pushconfig/2", "rt": ["oic.r.notificationselector", "oic.r.pushproxy"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[fe80::b1d6]:1122"}]}
            ],
            "schema": { "$ref": "#/definitions/links-in-response" }
          }
        }
      }
    },
    "/PushConfigurationResURI?if=oic.if.baseline" : {
      "get": {
        "description": "Collection of oic.r.notificationselector and associated push proxies.\nAllows
a Server to be configured with one or more Push Notification destinations.\n",
        "parameters": [
```

```
          {"$ref": "#/parameters/interface-baseline"}
        ],
        "responses": {
          "200": {
            "description" : "",
            "x-example": {
              "rt": ["oic.r.pushconfiguration"],
              "if": ["oic.if.ll","oic.if.create","oic.if.baseline"],
              "rts": ["oic.r.notificationselector", "oic.r.pushproxy"],
              "links": [
                {"href": "/pushconfig/1", "rt": ["oic.r.notificationselector","oic.r.pushproxy"],
"if": ["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[fe80::b1d6]:1122"}]},
                {"href": "/pushconfig/2", "rt": ["oic.r.notificationselector", "oic.r.pushproxy"],
"if": ["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[fe80::b1d6]:1122"}]}
              ]
            },
            "schema": { "$ref": "#/definitions/get-baseline-response" }
          }
        }
      }
    },
    "/PushConfigurationResURI?if=oic.if.create" : {
      "post": {
        "description": "Collection of oic.r.notificationselector and associated push proxies.\nAllows
a Server to be configured with one or more Push Notification destinations.\n",
        "parameters": [
          {
            "$ref": "#/parameters/interface-create"
          },
          {
            "$ref": "#/parameters/body-create"
          }
        ],
        "responses": {
          "201": {
            "description" : "new link and corresponding target Resource are created",
            "x-example": {
              "href": "/pushconfig/1",
              "rt": ["oic.r.notificationselector", "oic.r.pushproxy"],
              "if": ["oic.if.baseline", "oic.if.rw"],
              "ins": 4213291245,
              "p": { "bm": 3 },
              "rep": {
                "rt": ["oic.r.notificationselector", "oic.r.pushproxy"],
                "if": ["oic.if.rw", "oic.if.baseline"],
                "phref": "/myAirquality",
                "prt": [
                  "oic.r.airquality"
                ],
                "pushtarget" : "coaps://[2001::200]:49355/pushed-resource-airquality",
                "sourcert": [
                  "oic.r.pushpayload"
                ],
                "state": "waitingforupdate"
              }
            },
            "schema": { "$ref": "#/definitions/post-create-response" }
          }
        }
      }
    }
  },
  "parameters": {
    "interface-baseline" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
      "enum" : ["oic.if.baseline"]
    },
    "interface-ll" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
```

```
      "enum" : ["oic.if.ll"]
    },
    "interface-create" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
      "enum" : ["oic.if.create"]
    },
    "body-create" : {
      "in" : "body",
      "name" : "notificationselector-pushproxy",
      "required" : true,
      "schema" : {
        "$ref" : "#/definitions/post-create-request"
      },
      "x-example" : {
        "rt": [ "oic.r.notificationselector", "oic.r.pushproxy" ],
        "if": [ "oic.if.rw", "oic.if.baseline" ],
        "rep": {
          "phref": "/myFilterResURI",
          "prt": [
            "oic.r.airquality"
          ],
          "pushtarget" : "coaps://[2001::200]:49355/pushed-resource-filter",
          "sourcert": [
            "oic.r.pushpayload"
          ]
        }
      }
    }
  },
  "definitions": {
    "oic.oic-link": {
      "type": "object",
      "properties": {
        "if": {
          "description": "The OCF Interface set supported by the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": [
              "oic.if.baseline",
              "oic.if.rw"
            ]
          }
        },
        "rt": {
          "description": "Resource Type of the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "enum": [
              "oic.r.notificationselector",
              "oic.r.pushproxy"
            ],
            "type": "string",
            "maxLength": 64
          }
        },
        "anchor": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/anchor"
        },
        "di": {
```

```
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/di"
        },
        "eps": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/eps"
        },
        "href": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/href"
        },
        "ins": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/ins"
        },
        "p": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/p"
        },
        "rel": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/rel_array"
        },
        "title": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/title"
        },
        "type": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/type"
        }
      },
      "required": [
        "href",
        "rt",
        "if"
      ]
    },
    "links-in-response": {
      "type": "array",
      "items": {
        "$ref": "#/definitions/oic.oic-link"
      }
    },
    "get-baseline-response" : {
      "type" : "object",
      "properties": {
        "rt": {
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": ["oic.r.pushconfiguration"]
          }
        },
        "if": {
          "description": "The OCF Interface set supported by this Resource",
          "type": "array",
          "minItems": 1,
          "readOnly": true,
          "items": {
            "type": "string",
            "maxLength": 64,
```

```
            "enum": [
              "oic.if.ll",
              "oic.if.create",
              "oic.if.baseline"
            ]
          }
        },
        "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"
        },
        "rts": {
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": [
              "oic.r.notificationselector",
              "oic.r.pushproxy"
            ]
          }
        },
        "links": {
          "$ref": "#/definitions/links-in-response"
        }
      },
      "required": [
        "rt",
        "if",
        "links"
      ]
    },
    "rep-in-request": {
      "type": "object",
      "properties": {
        "phref": {
          "description": "URI of a Resource to be pushed",
          "type": "string",
          "maxLength": 256
        },
        "prt": {
          "description": "Resource type(s) of Resource(s) to be pushed",
          "type": "array",
          "uniqueItems": true,
          "items": {
            "type": "string",
            "maxLength": 64
          }
        },
        "pif": {
          "description": "OCF Interface(s) of Resource(s) to be pushed",
          "type": "array",
          "uniqueItems": true,
          "items": {
            "type": "string",
            "maxLength": 64
          }
        },
        "pushtarget": {
          "description": "Points to the target of the UPDATE operation sent as a notifcation",
          "type": "string",
          "maxLength": 256
        },
        "sourcert": {
          "description" : "Always set to oic.r.pushpayload",
```

```
          "type" : "array",
          "uniqueItems" : true,
          "items" : {
            "type": "string",
            "maxLength": 64,
            "enum" : [
              "oic.r.pushpayload"
            ]
          }
        },
        "state": {
          "description": "Current state of the Push Proxy",
          "type": "string",
          "enum": [
            "waitingforprovisioning",
            "waitingforupdate",
            "waitingforresponse",
            "waitingforupdatemitigation",
            "waitingforresponsemitigation",
            "error",
            "timeout"
          ]
        }
      },
      "required": [
        "pushtarget",
        "sourcert"
      ]
    },
    "rep-in-response": {
      "type": "object",
      "properties": {
        "rt": {
          "description": "Resource Type of the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "enum": [
              "oic.r.notificationselector",
              "oic.r.pushproxy"
            ],
            "type": "string",
            "maxLength": 64
          }
        },
        "if": {
          "description": "The OCF Interface set supported by the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": [
              "oic.if.rw",
              "oic.if.baseline"
            ]
          }
        },
        "phref": {
          "description": "URI of a Resource to be pushed",
          "type": "string",
          "maxLength": 256
        },
        "prt": {
          "description": "Resource type(s) of Resource(s) to be pushed",
          "type": "array",
          "uniqueItems": true,
          "items": {
            "type": "string",
```

```
            "maxLength": 64
          }
        },
        "pif": {
          "description": "OCF Interface(s) of Resource(s) to be pushed",
          "type": "array",
          "uniqueItems": true,
          "items": {
            "type": "string",
            "maxLength": 64
          }
        },
        "pushtarget": {
          "description": "Points to the target of the UPDATE operation sent as a notifcation",
          "type": "string",
          "maxLength": 256
        },
        "sourcert": {
          "description" : "Always set to oic.r.pushpayload",
          "type" : "array",
          "uniqueItems" : true,
          "items" : {
            "type": "string",
            "maxLength": 64,
            "enum" : [
              "oic.r.pushpayload"
            ]
          }
        },
        "state": {
          "description": "Current state of the Push Proxy",
          "type": "string",
          "enum": [
            "waitingforprovisioning",
            "waitingforupdate",
            "waitingforresponse",
            "waitingforupdatemitigation",
            "waitingforresponsemitigation",
            "error",
            "timeout"
          ]
        }
      },
      "required": [
        "rt",
        "if",
        "pushtarget",
        "sourcert",
        "state"
      ]
    },
    "post-create-request": {
      "type" : "object",
      "properties": {
        "rt": {
          "description": "Resource Type of the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "enum": [
              "oic.r.notificationselector",
              "oic.r.pushproxy"
            ],
            "type": "string",
            "maxLength": 64
          }
        },
        "if": {
          "description": "The OCF Interface set supported by the target Resource",
          "type": "array",
          "minItems": 1,
```

```
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": [
              "oic.if.rw",
              "oic.if.baseline"
            ]
          }
        },
        "rep": {
          "$ref": "#/definitions/rep-in-request"
        }
      },
      "required": [
        "rt",
        "if",
        "rep"
      ]
    },
    "post-create-response": {
      "type" : "object",
      "properties": {
        "href": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/href"
        },
        "rt": {
          "description": "Resource Type of the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "enum": [
              "oic.r.notificationselector",
              "oic.r.pushproxy"
            ],
            "type": "string",
            "maxLength": 64
          }
        },
        "if": {
          "description": "The OCF Interface set supported by the target Resource",
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": [
              "oic.if.rw",
              "oic.if.baseline"
            ]
          }
        },
        "ins": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/ins"
        },
        "p": {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/p"
        },
        "rep": {
          "$ref": "#/definitions/rep-in-response"
        }
      },
```

```
      "required": [
        "href",
        "rt",
        "if",
        "ins",
        "rep"
      ]
    }
  }
}
```

A.9 Composition Resource of Notification Selector and Push Proxy

A.9.1 Introduction

Each instance of composition Resource of Notification Selector and Push Proxy includes filtering parameters for the Resources to be pushed and Target Resource to which updated Resource of origin Server will be pushed.

A.9.2 Well-known URI

None

A.9.3 Resource type

The Resource Type is defined as: ["oic.r.pushconfiguration", "oic.r.pushproxy"]

A.9.4 OpenAPI 2.0 definition

```
{
  "swagger": "2.0",
  "info": {
    "title": "Notification Selector-Push Proxy",
    "version": "2022-06-15",
    "license": {
      "name": "OCF Data Model License",
      "url":
"https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LICE
NSE.md",
      "x-copyright": "Copyright 2019 Open Connectivity Foundation, Inc. All rights reserved."
    },
    "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
  },
  "schemes": [
    "http"
  ],
  "consumes": [
    "application/json"
  ],
  "produces": [
    "application/json"
  ],
  "paths": {
    "/NotificationSelectorPushproxyResURI?if=oic.if.rw" : {
      "get": {
        "description": "Resource that defines the selector for Push Notifications",
        "parameters": [
          {"$ref": "#/parameters/interface-rw"}
        ],
        "responses": {
          "200": {
            "description" : "",
            "x-example": {
              "phref": "/myAirquality",
              "prt": [
                "oic.r.airquality"
              ],
              "pushtarget" : "coaps://[2001::200]:49355/pushed-resource-airquality",
              "sourcert": [
                "oic.r.pushpayload"
              ],
```

```
              "state": "waitingforupdate"
            },
            "schema": { "$ref": "#/definitions/get-nspp-rw-response" }
          }
        }
      },
      "post": {
        "description": "Updates the current notification selector information.\n",
        "parameters": [
          { "$ref": "#/parameters/interface-rw" },
          { "$ref": "#/parameters/body-update" }
        ],
        "responses": {
          "204": {
            "description" : "the notification selector-push proxy is updated successfully\n"
          }
        }
      }
    },
    "/NotificationSelectorPushproxyResURI?if=oic.if.baseline" : {
      "get": {
        "description": "Resource that defines the selector for Push Notifications",
        "parameters": [
          {"$ref": "#/parameters/interface-baseline"}
        ],
        "responses": {
          "200": {
            "description" : "",
            "x-example": {
              "rt": ["oic.r.notificationselector", "oic.r.pushproxy"],
              "if": ["oic.if.rw", "oic.if.baseline"],
              "phref": "/myAirquality",
              "prt": [
                "oic.r.airquality"
              ],
              "pushtarget" : "coaps://[2001::200]:49355/pushed-resource-airquality",
              "sourcert": [
                "oic.r.pushpayload"
              ],
              "state": "waitingforupdate"
            },
            "schema": { "$ref": "#/definitions/get-nspp-baseline-response" }
          }
        }
      }
    }
  },
  "parameters": {
    "interface-rw" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
      "enum" : ["oic.if.rw"]
    },
    "interface-baseline" : {
      "in" : "query",
      "name" : "if",
      "type" : "string",
      "enum" : ["oic.if.baseline"]
    },
    "body-update": {
      "name": "notificationselector-pushproxy",
      "in": "body",
      "required": true,
      "schema": { "$ref": "#/definitions/post-nspp-rw-request" },
      "x-example": {
        "phref": "/myFilterResURI",
        "pushtarget" : "coaps://[2001::200]:49355/pushed-resource-filter",
        "sourcert": [
          "oic.r.pushpayload"
        ]
      }
    }
```

```
  },
  "definitions": {
    "get-nspp-baseline-response" : {
                "type": "object",
      "properties": {
        "rt": {
          "description": "Resource Type of the Resource",
          "items": {
            "enum": ["oic.r.notificationselector", "oic.r.pushproxy"],
            "type": "string",
            "maxLength": 64
          },
          "minItems": 1,
          "uniqueItems": true,
          "readOnly": true,
          "type": "array"
        },
        "if": {
          "description": "The interface set supported by this resource",
          "items": {
            "enum": [
              "oic.if.rw",
              "oic.if.baseline"
            ],
            "type": "string",
            "maxLength": 64
          },
          "minItems": 1,
          "readOnly": true,
          "uniqueItems": true,
          "type": "array"
        },
        "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"
        },
        "phref" : {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/href"
        },
        "prt": {
          "description": "Resource Type(s) of the Resource(s) to be pushed",
          "type": "array",
          "items" : {
            "type" : "string",
            "maxLength": 64
          },
          "minItems" : 1
        },
        "pif": {
          "description": "The OCF Interface(s) of the Resource(s) to be pushed",
          "type": "array",
          "items": {
            "type" : "string",
            "enum" : ["oic.if.baseline", "oic.if.ll", "oic.if.b", "oic.if.lb", "oic.if.rw",
"oic.if.r", "oic.if.a", "oic.if.s" ]
          },
          "minItems": 1
        },
        "pushtarget": {
          "description": "Points to the target of the UPDATE operation sent as a notification",
          "type": "string",
          "maxLength": 256
        },
        "sourcert": {
          "description" : "Always set to oic.r.pushpayload",
```

```
          "type" : "array",
          "uniqueItems" : true,
          "items" : {
            "type": "string",
            "maxLength": 64,
            "enum" : [
              "oic.r.pushpayload"
            ]
          }
        },
        "state": {
          "description": "Current state of the Push Proxy",
          "type": "string",
          "enum": [
            "waitingforprovisioning",
            "waitingforupdate",
            "waitingforresponse",
            "waitingforupdatemitigation",
            "waitingforresponsemitigation",
            "error",
            "timeout"
          ]
        }
      },
      "required": [
        "rt",
        "if",
        "pushtarget",
        "sourcert",
        "state"
      ]
    },
    "get-nspp-rw-response" : {
      "type": "object",
      "properties": {
        "phref" : {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/href"
        },
        "prt": {
          "description": "Resource Type(s) of the Resource(s) to be pushed",
          "type": "array",
          "items" : {
            "type" : "string",
            "maxLength": 64
          },
          "minItems" : 1
        },
        "pif": {
          "description": "The OCF Interface(s) of the Resource(s) to be pushed",
          "type": "array",
          "items": {
            "type" : "string",
            "enum" : ["oic.if.baseline", "oic.if.ll", "oic.if.b", "oic.if.lb", "oic.if.rw",
"oic.if.r", "oic.if.a", "oic.if.s" ]
          },
          "minItems": 1
        },
        "pushtarget": {
          "description": "Points to the target of the UPDATE operation sent as a notification",
          "type": "string",
          "maxLength": 256
        },
        "sourcert": {
          "description" : "Always set to oic.r.pushpayload",
          "type" : "array",
          "uniqueItems" : true,
          "items" : {
            "type": "string",
            "maxLength": 64,
            "enum" : [
              "oic.r.pushpayload"
```

```
            ]
          }
        },
        "state": {
          "description": "Current state of the Push Proxy",
          "type": "string",
          "enum": [
            "waitingforprovisioning",
            "waitingforupdate",
            "waitingforresponse",
            "waitingforupdatemitigation",
            "waitingforresponsemitigation",
            "error",
            "timeout"
          ]
        }
      },
      "required": [
        "pushtarget",
        "sourcert",
        "state"
      ]
    },
    "post-nspp-rw-request" : {
      "type": "object",
      "properties": {
        "phref" : {
          "$ref":
"https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
schema.json#/definitions/href"
        },
        "prt": {
          "description": "Resource Type(s) of the Resource(s) to be pushed",
          "type": "array",
          "items" : {
            "type" : "string",
            "maxLength": 64
          },
          "minItems" : 1
        },
        "pif": {
          "description": "The OCF Interface(s) of the Resource(s) to be pushed",
          "type": "array",
          "items": {
            "type" : "string",
            "enum" : ["oic.if.baseline", "oic.if.ll", "oic.if.b", "oic.if.lb", "oic.if.rw",
"oic.if.r", "oic.if.a", "oic.if.s" ]
          },
          "minItems": 1
        },
        "pushtarget": {
          "description": "Points to the target of the UPDATE operation sent as a notification",
          "type": "string",
          "maxLength": 256
        },
        "sourcert": {
          "description" : "Always set to oic.r.pushpayload",
          "type" : "array",
          "uniqueItems" : true,
          "items" : {
            "type": "string",
            "maxLength": 64,
            "enum" : [
              "oic.r.pushpayload"
            ]
          }
        },
        "state": {
          "description": "Current state of the Push Proxy",
          "type": "string",
          "enum": [
            "waitingforprovisioning",
            "waitingforupdate",
```

```
            "waitingforresponse",
            "waitingforupdatemitigation",
            "waitingforresponsemitigation",
            "error",
            "timeout"
          ]
        }
      },
      "required": [
        "pushtarget",
        "sourcert"
      ]
    }
  }
}
```

A.10 Push Receiver Resource

A.10.1 Introduction

The Resource through which a Device can be configured as a target Server for push notifications.

A.10.2 Well-known URI

None

A.10.3 Resource type

The Resource Type is defined as: "oic.r.pushreceiver"

A.10.4 OpenAPI 2.0 definition

```
{
  "swagger": "2.0",
  "info": {
    "title": "Push Receiver",
    "version": "2022-06-15",
    "license": {
      "name": "OCF Data Model License",
      "url":
"https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LICE
NSE.md",
      "x-copyright": "Copyright 2019 Open Connectivity Foundation, Inc. All rights reserved."
    },
    "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
  },
  "schemes": ["http"],
  "consumes": ["application/json"],
  "produces": ["application/json"],
  "paths": {
    "/PushReceiverResURI": {
      "get": {
        "description": "Resource that defines the receiver for Push Notifications",
        "parameters": [
          {
            "$ref": "#/parameters/interface-rw"
          }
        ],
        "responses": {
          "200": {
            "description": "",
            "x-example": {
              "receivers": [
                {
                  "receiveruri": "/mylocaltargeturiforthermostats",
                  "rts": ["oic.r.temperature", "oic.r.humidity"]
                },
                {
                  "receiveruri": "/mylocaltargeturifordontcare",
                  "rts": []
                }
              ]
```

```
          },
          "schema": { "$ref": "#/definitions/get-rw-response" }
        }
      }
    },
    "post": {
      "description": "replace whole push receiver objects\n",
      "parameters": [
        { "$ref": "#/parameters/interface-rw" },
        { "$ref": "#/parameters/body-receivers-update" }
      ],
      "responses": {
        "204": {
          "description": "whole receiver objects are replaced successfully\n"
        }
      }
    },
    "delete": {
      "description": "delete whole push receiver objects\n",
      "parameters": [
        {
          "$ref": "#/parameters/interface-rw"
        }
      ],
      "responses": {
        "204": {
          "description": "whole receiver objects are removed successfully\n"
        }
      }
    }
  },
  "/PushReceiverResURI?receiveruri=<uri>": {
    "post": {
      "description": "Updates a push receiver object which has `receiveruri`\n",
      "parameters": [
        { "$ref": "#/parameters/interface-rw" },
        { "$ref": "#/parameters/receiveruri" },
        { "$ref": "#/parameters/body-receiver-update" }
      ],
      "responses": {
        "204": {
          "description": "a receiver object is updated successfully\n"
        }
      }
    },
    "delete": {
      "description": "delete a push receiver object which has `receiveruri`\n",
      "parameters": [
        { "$ref": "#/parameters/interface-rw" },
        { "$ref": "#/parameters/receiveruri" }
      ],
      "responses": {
        "204": {
          "description": "a receiver object was removed successfully\n"
        }
      }
    }
  },
  "/PushReceiverResURI?if=oic.if.baseline": {
    "get": {
      "description": "Resource that defines the receiver for Push Notifications",
      "parameters": [
        { "$ref": "#/parameters/interface-baseline" }
      ],
      "responses": {
        "200": {
          "description": "",
          "x-example": {
            "rt": ["oic.r.pushreceiver"],
            "if": ["oic.if.rw", "oic.if.baseline"],
            "receivers": [
              {
                "receiveruri": "/mylocaltargeturiforthermostats",
```

```
                "rts": ["oic.r.temperature", "oic.r.humidity"]
              },
              {
                "receiveruri": "/mylocaltargeturifordontcare",
                "rts": []
              }
            ]
          },
          "schema": { "$ref": "#/definitions/get-baseline-response" }
        }
      }
    }
  }
},
"parameters": {
  "interface-baseline": {
    "in": "query",
    "name": "if",
    "type": "string",
    "enum": ["oic.if.baseline"]
  },
  "interface-rw": {
    "in": "query",
    "name": "if",
    "type": "string",
    "enum": ["oic.if.rw"]
  },
  "receiveruri": {
    "in": "query",
    "name": "receiveruri",
    "type": "string"
  },
  "body-receiver-update": {
    "in": "body",
    "name": "receiver",
    "required": true,
    "schema": {
      "$ref": "#/definitions/receiver"
    },
    "x-example": {
      "receiveruri": "/mylocaltargeturifordontcare",
      "rts": []
    }
  },
  "body-receivers-update": {
    "in": "body",
    "name": "receivers",
    "required": true,
    "schema": {
      "$ref": "#/definitions/post-rw-request"
    },
    "x-example": {
      "receivers": [
        {
          "receiveruri": "/mylocaltargeturifordontcare",
          "rts": []
        },
        {
          "receiveruri": "/mylocaltargeturifordontcare-2",
          "rts": []
        }
      ]
    }
  }
},
"definitions": {
  "receiver": {
    "description": "a definition of URIs at which push payloads may be received",
    "type": "object",
    "properties": {
      "receiveruri": {
        "format": "uri",
        "type": "string"
```

```
        },
        "rts": {
          "description": "The list of allowable Resource Types for this instance of a push receiver",
          "type": "array",
          "items": {
            "type": "string",
            "maxLength": 64
          },
          "minItems": 0
        }
      },
      "required": ["receiveruri", "rts"]
    },
    "receivers": {
      "description": "Definitions of URIs at which push payloads may be received",
      "type": "array",
      "items": {
        "$ref": "#/definitions/receiver"
      },
      "minItems": 0
    },
    "get-rw-response": {
      "type": "object",
      "properties": {
        "receivers": {
          "$ref": "#/definitions/receivers"
        }
      },
      "required": ["receivers"]
    },
    "get-baseline-response": {
      "type": "object",
      "properties": {
        "rt": {
          "type": "array",
          "minItems": 1,
          "uniqueItems": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": ["oic.r.pushreceiver"]
          }
        },
        "if": {
          "description": "The OCF Interface set supported by this Resource",
          "type": "array",
          "minItems": 1,
          "readOnly": true,
          "items": {
            "type": "string",
            "maxLength": 64,
            "enum": ["oic.if.rw", "oic.if.baseline"]
          }
        },
        "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"
        },
        "receivers": {
          "$ref": "#/definitions/receivers"
        }
      },
      "required": [
        "rt", "if", "receivers"
      ]
    },
    "post-rw-request": {
```

```
      "type": "object",
      "properties": {
        "receivers": {
          "$ref": "#/definitions/receivers"
        }
      },
      "required": ["receivers"]
    }
  }
}
```