CSP CDS Interface Specification

Notification APIs

Version 1.0.2

MCP plc
Port Community Systems

CNSI

Notification APIs

1. Contents

Version 1.0.1 0

2. Version Control 3

3. Overview 3

4. CDS Notification Specifics 3

5. Examples 4

Example 1 – Routing Notifications per Badge 4

Example 2 - Routing Notifications by Type 5

Example 3 – Polling 6

6. Mailbox/Device ID Replacement 7

7. Partitions 7

8. Description of Notification APIs 8

Configure Topic Consumer 8

Get Topic Consumer 8

Send Heartbeat Notification 8

Get Notification Batch 8

Acknowledge Notification Batch 8

9. Get Notification Batch 8

Description 8

Endpoint 8

Success Response when Messages are Available 10

Success Response when No Notifications are Available 13

Error scenarios 13

Implementation Restrictions 14

10. Acknowledge a batch of notifications 15

Endpoint 15

Success Response 16

Error scenarios 16

11. Configure Topic Consumer 17

Endpoint 17

Success Response Messages Available 18

Error scenarios 18

12. Get Topic Consumer 19

Endpoint 19
Success Response 20
13. Send Heartbeat Notification 21
Endpoint 21
Success Response 21
Error scenarios 22
14. Diagrams 24
Notification Routing 25
Notification Service Receive Notification 26
Notification Push Mechanism 27
Notification Pull Mechanism 28

2. Version Control

Date	Version	Reason	Author
February 2020	1.0.0	Document released	MCP/CNS
March 2020	1.0.1	Add updates for PUSH notifications	MCP/CNS
June 2020	1.0.2	Clarify notification headers and add additional examples	MCP/CNS

3. Overview

Notifications will be used to notify of the outcome of asynchronous processing or to alert on events that have occurred in either the CSP's or HMRC's systems.

This includes:

- DMS Notifications produced directly from DMS
- Success and error responses raised by HMRC's MDTP platform.
- Inventory linking pre-check rejections
- Heartbeat notifications

In all these cases the event is routed to a topic based on the badge code associated with the event and by the type of event. Topics can be thought of as similar to the mailbox or Device ID currently in use by the SOAP web-services offered by CSPs for CHIEF. Alternatively, this can be thought of as a queue.

As with badges, topics are assigned by the CSP to a specific office location. No two office locations will be able to share the same topic. Message will only be routed to a topic if the badge code associated with the event is associated with the same office as the topic.

Notifications can be consumed from the notification service in one of two ways.

1) By a software package using HTTP requests to poll for message batches and sending acknowledgements of any batches received. We refer to this as the notification pull mechanism.
2) By a software package hosting a HTTPS web-service and configuring an endpoint consumer in the CSP system so that the CSP can POST http requests to the endpoint for consumption. We refer to this as the notification push mechanism.

Software Vendors can choose between implementing either or both of these two mechanisms. This will need to be taken into account, during the setup process for new customers in the CSP's systems.

4. CDS Notification Specifics

The notification service is being introduced by CSP as part of the custom declaration service (CDS) changes.

The predominant set of notifications that will occur in CDS are notification produced by IBM's DMS system. Notifications produced by the CSP's for inventory linking will also follow the DMS specification.

When creating a declaration, the submitter includes the LRN in the FunctionalReferenceID field (UCC 2/5). This value is unique, for each create declaration transaction against the individual submitters EORI. For CSPs the submitters EORI is determined by the EORI linked to the badge which is provided to the CSP's create customs declarations endpoint. Note the LRN value does not change in the case of making an amendment or cancellation requests for a declaration and you should use the value provided when creating a declaration. **All notifications returned by DMS or the inventory system will return the FunctionalReferenceID in the request body and this should be used to identify the declaration in the submitters software**.

5. Examples

Example 1 – Routing Notifications per Badge

Declaration Company has 3 badges issued by a CSP: DCA, DCB and DCC for their office in Dover.

They would like to keep all messaging for the 3 badges separate.

They request that the CSP set them up with 3 TOPICs for Dover one for each badge.

The CSP provides them with 3 topics: TOPIC1, TOPIC2 and TOPIC3. Routing rules are configured by the CSP to: route DCA badges to TOPIC1; DCB badges to TOPIC2; and DCC badges to TOPIC3.

The badge company software only support pull messaging. When the package is setup for users at Declarations Company's Dover office they will need user credentials for the software linked to Declaration Company's Dover office.

The software will need as part of the setup to use the above user credentials to provide the URL endpoint and Authorization header details that they would like to use for each of the 3 TOPICs given to Declaration Company above. [1] [2]

The software package uses the Configure Topic Consumer API to set the above details.

Someone working for Declaration Company submits a declaration through their software package to the CSP and provides the badge DCB this is set by their software package in the X-Badge-ID header of the HTTP request.

Initial validation checks are OK and the declaration is not inventory linked so can be submitted to HMRC.

The CSP sends the request to HMRC and receives a 204 No-Content response with a number of headers.

The CSP will supplement the headers returned in the HMRC response by adding the X-Badge-ID header set to DCB.

This notification is then sent to the routing service.

The routing service see this is a MDTP response and has badge DCB which has a route to TOPIC2.

The message is sent to TOPIC2 to be persisted for returning to the Declaration Company.

---

[1] The software package can use the same URL for all TOPICs – If doing so the software house will need to ensure only the intended recipient receives the notification.

[2] This step could be done using a HTTP client such as postman rather than built into the software package.

As the topic is not configured for push based mechanism the CSPs will wait for the software house to poll.

In the meantime, DMS has processed the requests and produced a DMSACC Notification to the CSP the headers of this notification include the X-Badge-ID set to DCB as that was the badge used to submit the original declaration.

The routing service see this is a DMS notification and has badge DCB which has a route to TOPIC2.

The message is sent to TOPIC2 to be persisted for returning to the Declaration Company.

As the topic is not configured for push based mechanism the CSPs will wait for the software house to poll.

Example 2 - Routing Notifications by Type

Declaration Company software house wants the MDTP response kept separate from the DMS and inventory linking notifications. Declaration Company do not want to differentiate between badge codes.

Declaration Company will request that the CSP sets up two topics with MDTP requests going to one topic and other notifications going to the second topic.

The CSP provides them with two topics: TOPIC_MDTP and TOPIC_DMS with routes configured to send MDTP response for any of their 3 badges to TOPIC_MDTP and inventory linking pre-check request and DMS notification for their 3 badges to TOPIC_DMS.

As part of the software package setup the two topics are configured for push based messaging.

A declaration is submitted to the CSP using badge DCB.

It pass initial validation and does not need to be inventory pre-checked.

The MDTP platform responds to the CSP with a 204 No-Content response.

The CSP sends this to its own routing service.

The routing service see this is a MDTP response and has badge DCB which has a route of TOPIC_MDTP.

The message is sent to TOPIC_MDTP to be persisted for returning to the Declaration Company.

As the software package has configured a push based consumer for TOPIC_MDTP the CSPs polling mechanism is monitoring for changes on that topic.

It sees the new message and makes a POST request to the configured endpoint with all the headers from the original MDTP response the X-Badge-ID header and the endpoints Authorization header. As the response was a MDTP 204 No-Content response, the body of this request will be empty.

The software package returns a 200 OK response. As this is in the 2XX range the CSP updates TOPIC_MDTP to show that the message was sent and checks to see if more messages are ready to be sent.

In the meantime DMS has processed the requests and produced a DMSACC Notification to the CSP the headers of this notification include the X-Badge-ID set to DCB as that was the badge used to submit the original declaration.

The routing service see this is a DMS notification and has badge DCB which this time has a route of TOPIC_DMS.

The message is sent to TOPIC_DMS to be persisted for returning to the Declaration Company.

As the software package has configured a push based consumer for TOPIC_DMS the CSPs polling mechanism is monitoring for changes on that topic.

It sees the new message and makes a POST request to the configured endpoint with all the headers from the original DMS Notification headers and the endpoints Authorization header. The message body will contain the message body of the DMS notification.

Example 3 – Polling

Declaration Company cannot stand up a web-service within their organisation for receiving notification. They therefore purchase a software product that supports polling for notifications.

They request from the CSP a single TOPIC to route all their notifications to.

The CSP provides them with TOPIC4 and configures routing for all 3 of their badges to that same topic.

The software package is polling the CSPs notification batch API checking for messages. As no notifications are currently available the CSP will return a 204 No-Content response. The software package will wait 30 seconds before trying to poll again.

A declaration is submitted to the CSP using badge DCB.

It pass initial validation and does not need to be inventory pre-checked.

The MDTP platform responds to the CSP with a 204 No-Content response.

The CSP sends this to its own routing service.

The routing service see this is a MDTP response and has badge DCB which has a route of TOPIC4.

The message is sent to TOPIC4 to be persisted for returning to the Declaration Company.

As the topic is not configured for push based mechanism the CSPs will wait for the software house to poll.

In the meantime DMS has processed the requests and produced a DMSACC Notification to the CSP the headers of this notification include the X-Badge-ID set to DCB as that was the badge used to submit the original declaration.

The routing service see this is a DMS notification and has badge DCB which this time has a route of TOPIC4.

The message is sent to TOPIC4 to be persisted for returning to the Declaration Company.

The software package polls again for notifications. This time the CSP sees that TOPIC4 has two messages pending. It creates a batch response containing both messages.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<notifications topic="TOPIC4" count="2">
    <notification id="5231b533-ba17-4787-98a3-f2df37de2aD">
        <queuedDateTime>2018-02-05T12:05:06.658Z</queuedDateTime>
```

```
        <headers>
            <header name="X-Badge-ID" value="ABC"/>
            <header name="X-CSP-ID" value="ABC-1234567890123"/>
            <header name="X-Notification-Type" value="API"/>
            <header name="Content-Type" value="application/xml"/>
            <header name="ConversationID"
                    value="00001101-0000-1000-8000-00805f9b34fb"/>
        </headers>
        <body>PE1ldGFEYXRhPjwvTWV0YURhdGE+</body>
    </notification>
    <notification id="6b313534-bd23-12c2-44a3-d21f348e8a27"
                  partition="6">
        <queuedDateTime>2018-02-05T12:05:06.658Z</queuedDateTime>
        <headers>
            <header name="X-Badge-ID" value="ABC"/>
            <header name="X-Notification-Type" value="DMS"/>
            <header name="Content-Type" value="application/xml"/>
            <header name="ConversationID"
                    value="00001101-0000-1000-8000-00805f9b34fb"/>
        </headers>
        <body>PE1ldGFEYXRhPjwvTWV0YURhdGE+</body>
    </notification>
</notifications
```

6. Mailbox/Device ID Replacement

This service supersedes the existing CSP mailbox/Device ID service for CDS. It does not however remove the existing service which will be retained for messages currently being sent there.

The reasons for introducing a new service are:

- Support messaging with headers.
- Support concurrent consumers.
- Works in conjunction with pushed based mechanisms
- Align with RESTful API standards.

7. Partitions

Please be advised that not all CSP will initially provide support for partitioning. Please contact the individual CSP to enquire about their support for this option.

A polling consumer will need to retrieve a batch of notifications and then acknowledge those notifications before querying for another batch. If they query for a new batch before acknowledging one or more messages provided in the previous batch then those messages will be included in the new batch as well.

If the software house wishes to use multiple consumers they can do so by having the consumers coordinate which partitions of a topic they will pull from.

Each TOPIC will have 12 partitions (12 because it divides by 2, 3, 4 and 6). When a notification is persisted by the CSP it will assign the message to a partition.

A software product can then have their consumers coordinate which partitions each will consume from and as long as they are consuming from distinct partitions they will not need to coordinate their requests and acknowledgements.

All partitions must have a consumer to avoid messages being missed.

8. Description of Notification APIs

The following is brief overview of the APIs available from each CSP for the notification service. A more deep-dive look at each API can be found later in this document.

Configure Topic Consumer

This API is for configuring a TOPIC for push notifications. It allows a software product to configure the endpoint where the CSP can POST notifications to for a given topic. This should be a HTTPS endpoint. The value the CSP will provide in the HTTP Authorization header of the request can also be provided.

Get Topic Consumer

Returns the details of the currently configured endpoint used by the CSP for the given topic. The CSP will be sending HTTP POST requests, containing a single notification at a time, to this endpoint.

Send Heartbeat Notification

Allows someone to send a test notification that will be persisted to a given topic. This can be used to check the connectivity between the CSP and the software product. This heartbeat notification takes a specific format and is not intend as a mocking service for specific types of notifications.

Get Notification Batch

Queries for a batch of messages associated with a given topic.

Acknowledge Notification Batch

Notifies the CSP that the list of notification IDs have been successfully consumed from a given topic.

9. Get Notification Batch

Description

Queries for a batch of messages associated with a given topic. Only un-acknowledged notifications will be returned.

You should acknowledge each batch of messages before querying for a new batch. If messages are not acknowledged between two batches then messages in the first batch are likely to appear again in the second batch.

The TOPIC will need to be setup for your office by the CSP. If you submit declarations from multiple offices you will need at least one TOPIC for each such office.

You can only query for a batch of notifications if there is no topic consumer configured.

Endpoint

HTTP Method

GET

Request URL suffix

/notifications/{topic}

Path Variables

Name	Value / Example	Description
topic required	CDSTOPIC	CSP assigned name for the topic of message to read from.

Request Parameters

Name	Value / Example	Description
max optional	20	Allows you to limit the number of messages to return in a single request. Min 1, Max 100. Default 100.
partitionFrom optional	1	The first partition (inclusive to read from). Must be between 1 and 12.
partitionTo optional	12	The last partition (inclusive to read from). Must be between 1 and 12.
partitions optional	1,3,5	A comma separated list of partitions to pull from.
X-Badge-ID optional	ABC	Allows you to request messages for a specified X-Badge-ID. (Used where partitioning not implemented)

Request Headers

Name	Value / Example	Description
Authorization required	Basic am9lYmxvZ2dzOnBhc3N3b3Jk	CSP specific authentication information.
Accept required	application/vnd.csp.1.0+xml application/vnd.csp.1.0+json	Specifies the version of the API that you want to call. versioning.

User-Agent required	Vendor=Acme Software Vendor plc, Application=Your Application, Version=1.2.3, Badge=ABC, ClientID= A. B. Clearance Agents	See the User Agent section of the API Reference Guide

Partitioning

Partitioning is a means of supporting multiple consumers. If no partitioning settings are given a consumer will receive messages from all partitions. In the case of multiple consumers each consumer should be configured to pull from a distinct set of partitions.

Each queue is split into 12 such partitions. Notifications will be balanced across each partition by the CSP.

No guarantee of distinct order is given across partitions. CSPs should pull message from a single partition in sequence but this does not guarantee that this will correspond with the sequence from source.

You can select the partitions for each consumer in the request parameters that consumer calls.

You should provide either both partitionFrom and partitionTo or just partitions. Providing all 3 will result in an error.

Success Response when Messages are Available

Http status

```
200 (OK)
```

Response body

The response body will contain a list of notifications. Each notification consists of a body and a list of headers. The body will be a base 64 encode binary data. The format of the body is dependent on the format received from the source system and does not reflect the requested content-type which only dictates the wrapping payload. e.g. DMS provides XML messages so the body will be a base64 encoded XML message even if the notifications pull has a Content-Type of application/vnd.csp.1.0+json.

For push notifications the schema will be as per the pull notification. It will contain only one notification and will be in XML format.

Schema:



pull-notifications-response.xsd

Sample XML API response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<notifications topic="CDSTOPIC" count="1">
    <notification id="5231b533-ba17-4787-98a3-f2df37de2aD">
        <queuedDateTime>2018-02-05T12:05:06.658Z</queuedDateTime>
        <headers>
            <header name="X-Badge-ID" value="ABC"/>
            <header name="X-CSP-ID" value="ABC-1234567890123"/>
            <header name="X-Notification-Type" value="API"/>
            <header name="Content-Type" value="application/xml"/>
            <header name="ConversationID"
                    value="00001101-0000-1000-8000-00805f9b34fb"/>
        </headers>
        <body>PE1ldGFEYXRhPjwvTWV0YURhdGE+</body>
    </notification>
</notifications
```

Sample CDS generated DMS XML response (ConverstionID included)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<notifications topic="CDSTOPIC" count="1">
    <notification id="5231b533-ba17-4787-98a3-f2df37de2aD">
        <queuedDateTime>2018-02-05T12:05:06.658Z</queuedDateTime>
        <headers>
            <header name="X-Badge-ID" value="ABC"/>
            <header name="X-Notification-Type" value="DMS"/>
            <header name="Content-Type" value="application/xml"/>
            <header name="ConversationID"
                    value="00001101-0000-1000-8000-00805f9b34fb"/>
        </headers>
        <body>PE1ldGFEYXRhPjwvTWV0YURhdGE+</body>
    </notification>
</notifications
```

Sample CSP generated DMS Inventory pre check failure XML response (X-CSP-ID included)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<notifications topic="CDSTOPIC" count="1">
    <notification id="5231b533-ba17-4787-98a3-f2df37de2aD">
        <queuedDateTime>2018-02-05T12:05:06.658Z</queuedDateTime>
        <headers>
            <header name="X-Badge-ID" value="ABC"/>
            <header name="X-Notification-Type" value="DMS"/>
            <header name="Content-Type" value="application/xml"/>
            <header name="X-CSP-ID" value="ABC-1234567890123"/>
        </headers>
        <body>PE1ldGFEYXRhPjwvTWV0YURhdGE+</body>
    </notification>
</notifications
```

Sample JSON API response


```
{
    “topic”: “CDSTOPIC”,
    “count”: 1,
    “notifications”: [
        {
            “id”: “5231b533-ba17-4787-98a3-f2df37de2ad7”,
            “partition”: 1,
            “queuedDateTime”: “2018-02-05T12:05:06.658”,
            “headers”: [
                {
                    “name”: “X-Badge-ID”,
                    “value”: “ABC”
                },
                {
                    “name”: ”X-CSP-ID”:,
                    “value”: ”ABC-1234567890123”
                },
                {
                    “name”: ”X-Notification-Type”:,
                    “value”: ”API”
                },
                {
                    “name”: “Content-Type”,
                    “value”: “application/xml”
                },
                {
                    “name”: “ConversationID”,
                    “value”: “00001101-0000-1000-8000-00805f9b34fb”
                }
            ],
            “body”: “PE1ldGFEYXRhPjwvTWV0YURhdGE+”
       }
    ]
}
```

Notification Headers

Name	Value/Example	Description
ConversationID	00001101-0000-1000-8000-00805f9b34fb	CDS generated conversation Id, not returned for (Inventory pre-check failures)
X-Badge-ID	ABC	Associated X-Badge-Identifier
X-CSP-ID*	ABC-1234567890123	**Only returned for MDTP (API) Notifications or CSP generated DMS Notifications (Inventory pre-check failures)**
X-Notification-Type	API	API = MDTP Notifications DMS = DMS Notifications

*X-CSP-ID, only applicable to MDTP notifications **or DMS notifications where an inventory pre-check has failed prior to submitting to CDS**, it is not stored therefore, not returned with any subsequent CDS generated DMS notifications.

Success Response when No Notifications are Available

Http Status

```
204 (No Content)
```

Response Body

There will be no response body returned in this instance.

Error scenarios

The specific error responses have a consistently formed XML body, including:

Name	Type	Description
Code	string	A machine-readable error code. This is unique for each error scenario.
Message	string	A human-readable explanation for the error.

There might be other error-specific information (such as a list of fields that are in error).

Here is an example error response in XML format

```
<errorResponse>
```

```
  <code>NOT_AUTHORIZED</code>
  <message>You do not have permissions to access this
topic</message>
</errorResponse>
```

Status	Code	Scenario
403 (forbidden)	NOT_AUTHORIZED	You do not authorization to access the requested TOPIC
403 (forbidden)	INVALID_BADGE_ID	The provided X-Badge-ID is not associated with the requested TOPIC
400 (Bad Request)	PARTITION_PARAM_MISS_MATCH	An invalid combination of partition parameters was specified.
404 (Not Found)	TOPIC_NOT_FOUND	No topic was found matching the given name.
423 (Locked)	LOCKED_PUSH_MESSAGING_ACTIVE	The topic is locked because push messaging is currently active.
500 (Internal Server Error)	INTERNAL_SERVER_ERROR	The CSP encountered an unexpected error while processing the request. Please contact the relevant CSP helpdesk.

Other error scenarios may occur. For more details on common error conditions that may occur please see the API reference guide.

Implementation Restrictions

Implementers are requested to limit polling frequency to no more than once every 30 seconds when no notifications are available (i.e. after receives a 204 response). Users wishing more immediate response should consider implementing push notifications.

You may query for a new batch immediate after receiving a 200 OK response from an acknowledgement request.

10. Acknowledge a batch of notifications

For acknowledging that a collection of notifications from a given topic have been successfully received by the recipient.

The notification IDs to acknowledge can be extracted from the response to a get batch request.

Any messages not acknowledged from a batch will still be available to be consumed in future batch request.

Software products should ensure that a notification has been persisted within their systems before the notification is acknowledged. The CSP is not required to support retaining previously acknowledged notifications and could choose to delete them at any point after receiving a successful acknowledgement.

This endpoint supports wrapping the batch of notifications as either an XML or JSON payload. The wrapping type is not dependent on the content type of the individual notifications. Notifications included in a single batch do not need to have the same content type. Each notifications own body is base64 encoded and will need to be decoded before parsing. Notifications themselves are not restricted to XML or JSON and can include other types such as a plain text report. Please check for the Content-Type header on a per notification bases to determine the content type of the notification. For CDS all currently known notification types that can be delivered to a topic are all XML based. You can request from your CSP to route notifications with different content types to different topics to avoid having a mix of notification types appearing in the same batch.

You only need to use this acknowledgement request in conjunction with the Get Notification Batch request. If push based notifications are configured for a topic then requests will be auto-acknowledged when the CSP receives a 2XX response.

Endpoint

HTTP Method

DELETE

Request URL suffix

/notifications/{topic}

Request Body

Request can be submitted as XML or JSON
XML Schema



pull-notifications-acknowledgement.xsd

Sample XML Body

```
<notifications>
  <id>5231b533-ba17-4787-98a3-f2df37de2ad7</id>
  <id>1231d433-fc27-4787-98a3-f2da38a82aa7</id>
</notifications>
```

Sample JSON Body

```
[
  “5231b533-ba17-4787-98a3-f2df37de2ad7”,
  “1231d433-fc27-4787-98a3-f2da38a82aa7”
]
```

Success Response

Indicates that the CSP has processed the acknowledgement request and that you can query for a new batch when ready.

HTTP status

```
200 (OK)
```

Error scenarios

The specific error responses have a consistently formed XML body, including:

Name	Type	Description
Code	string	A machine-readable error code. This is unique for each error scenario.
Message	string	A human-readable explanation for the error.

There might be other error-specific information (such as a list of fields that are in error).

Here is an example error response in XML format

```
<errorResponse>
  <code>NOT_AUTHORIZED</code>
  <message>You do not have permissions to access this
topic</message>
</errorResponse>
```

Status	Code	Scenario
403 (forbidden)	NOT_AUTHORIZED	You do not have authorization to access the requested TOPIC
404 (Not Found)	TOPIC_NOT_FOUND	No topic was found matching the given name.
500 (Internal Server Error)	INTERNAL_SERVER_ERROR	The CSP encountered an unexpected error while processing the request. Please

		contact the relevant CSP helpdesk.
423 (Locked)	LOCKED_PUSH_MESSAGING_ACTIVE	The topic is locked because push messaging is currently active.

Other error scenarios may occur. For more details on common error conditions that may occur please see the API reference guide.

11. Configure Topic Consumer

Allows a declaration to set a consumer for the given topic. Messages sent to the TOPIC will be pushed to the given consumer using a HTTPS POST.

Only one consumer can be configured per TOPIC. You can use the same consumer against multiple topics but these will all need to be configured separately.

Endpoint

Http Method

PUT

Request URL suffix

/notifications/{topic}/consumer

Request body schema

consumer.xsd

Request Body Sample

```
<consumer
endpointUrl="https://api.abcenterprise.co.uk/cds/notifications"
authorization="Basic ABC">
</consumer>
```

Path Variables

Name	Value / Example	Description
topic required	CDSTOPIC	CSP assigned name for the topic to assign the consumer to.

Request Headers

Name	Value / Example	Description

Authorization required	Basic am9lYmxvZ2dzOnBhc3N3b3Jk	CSP specific authentication information.
Content-Type required	application/xml; application/json	The format of the message payload. Either XML or JSON.
Accept required	application/vnd.csp.1.0+xml; application/vnd.csp.1.0+json	Specifies the version of the API that you want to call. versioning.
User-Agent required	Vendor=Acme Software Vendor plc, Application=Your Application, Version=1.2.3, Badge=ABC, ClientID= A. B. Clearance Agents	See the User Agent section of the API Reference Guide

Body Elements

Name	Value / Example	Description
consumer@endpointUrl required	https://api.abcenterprise.co.uk/cds /notifications	Provides the https url endpoint to which notifications should be posted. Can be blank to disable the consumer.
consumer@authorization required	Basic ABC	Specifies the authorization header to set in any POST requests.

It is recommended that you swap the authorization header on a semi-regular bases – at least once a year. You can do so by using this configure topic consumer API. If your endpoint does not support rolling updates you should remember to disable the consumer during this change.

Success Response Messages Available

Http status

```
200 (OK)
```

Error scenarios

The specific error responses have a consistently formed XML body, including:

Name	Type	Description
error/code	string	A machine-readable error code. This is unique for each error scenario.
error/message	string	A human-readable explanation for the error.

There might be other error-specific information (such as a list of fields that are in error).

Here is an example error response in XML format

```
<errorResponse>
  <code>NOT_AUTHORIZED</code>
  <message>You do not have permissions to access this
topic</message>
</errorResponse>
```

Status	Code	Scenario
403 (forbidden)	NOT_AUTHORIZED	You do not authorization to assign a consumer to this TOPIC
404 (Not Found)	TOPIC_NOT_FOUND	No topic was found matching the given name.
422 (Unprocessable Entity)	HTTPS_NOT_SPECIFIED	The endpoint given was not a HTTPS endpoint.
500 (Internal Server Error)	INTERNAL_SERVER_ERROR	The CSP encountered an unexpected error while processing the request. Please contact the relevant CSP helpdesk.

Other error scenarios may occur. For more details on common error conditions that may occur please see the API reference guide.

12. Get Topic Consumer

Retrieves details of the currently configured TOPIC consumer.

Endpoint

Http Method

GET

Request URL suffix

/notifications/{topic}/consumer

Path Variables

Name	Value / Example	Description
topic required	CDSTOPIC	CSP assigned name for the topic to assign the consumer to.

Request Headers

Name	Value / Example	Description
Authorization required	Basic am9lYmxvZ2dzOnBhc3N3b3Jk	CSP specific authentication information.
Accept required	application/vnd.csp.1.0+xml; application/vnd.csp.1.0+json	Specifies the version of the API that you want to call. versioning.
User-Agent required	Vendor=Acme Software Vendor plc, Application=Your Application, Version=1.2.3, Badge=ABC, ClientID= A. B. Clearance Agents	See the User Agent section of the API Reference Guide

Success Response

HTTP Status

```
200 (OK)
```

Response Body Schema

XML Schema:



consumer.xsd

Response Body Sample

```
<consumer
endpointUrl="https://api.abcenterprise.co.uk/cds/notifications"
authorization="Basic ABC">
</consumer>
```

13. Send Heartbeat Notification

Sends a heartbeat notification to this topic. This can help test connectivity to the topic by sending a test message to the topic that will be sent to the consumer.

Endpoint

Http Method

POST

Request URL Suffix

/notifications/{topic}/heartbeat

Request Headers

Name	Value / Example	Description
Authorization required	Basic am9lYmxvZ2dzOnBhc3N3b3Jk	CSP specific authentication information.
Accept required	application/vnd.csp.1.0+xml; application/vnd.csp.1.0+json	Specifies the version of the API that you want to call. versioning. This value controls the format of the heartbeat notification.
User-Agent required	Vendor=Acme Software Vendor plc, Application=Your Application, Version=1.2.3, Badge=ABC, ClientID= A. B. Clearance Agents	See the User Agent section of the API Reference Guide

Request Body

Request body not required and any content put here will be ignored.

Success Response

HTTP status

```
200 (OK)
```

Asynchronous Notification

On successful submission a notification will be raised against the chosen topic.

The notification will contain the following headers:

Name	Value / Example	Description

Content-Type	application/xml; application/json	Based on the requested content in the Accept header of the original POST
Test	Test	Indicates that this is a test message.
From	Testuser	The submitter of the original heartbeat request.

The body of the notification will conform to the following XML schema:



heartbeat.xsd

Sample XML Heartbeat Notification

```
<heartbeat requestDateTime="2018-02-06T09:04:00.000Z">
</heartbeat>
```

Sample JSON Heartbeat Notification

```
{
  "type": "heartbeat",
  "requestDateTime": "2018-02-06T09:04:00.000Z"
}
```

Error scenarios

The specific error responses have a consistently formed XML body, including:

Name	Type	Description
error/code	string	A machine-readable error code. This is unique for each error scenario.
error/message	string	A human-readable explanation for the error.

There might be other error-specific information (such as a list of fields that are in error).

Here is an example error response in XML format

```
<errorResponse>
  <code>NOT_AUTHORIZED</code>
```

```
  <message>You do not have permissions to access this
topic</message>
</errorResponse>
```

Status	Code	Scenario
403 (forbidden)	NOT_AUTHORIZED	You do not authorization to access the requested TOPIC
404 (Not Found)	TOPIC_NOT_FOUND	No topic was found matching the given name.
500 (Internal Server Error)	INTERNAL_SERVER_ERROR	The CSP encountered an unexpected error while processing the request. Please contact the relevant CSP helpdesk.

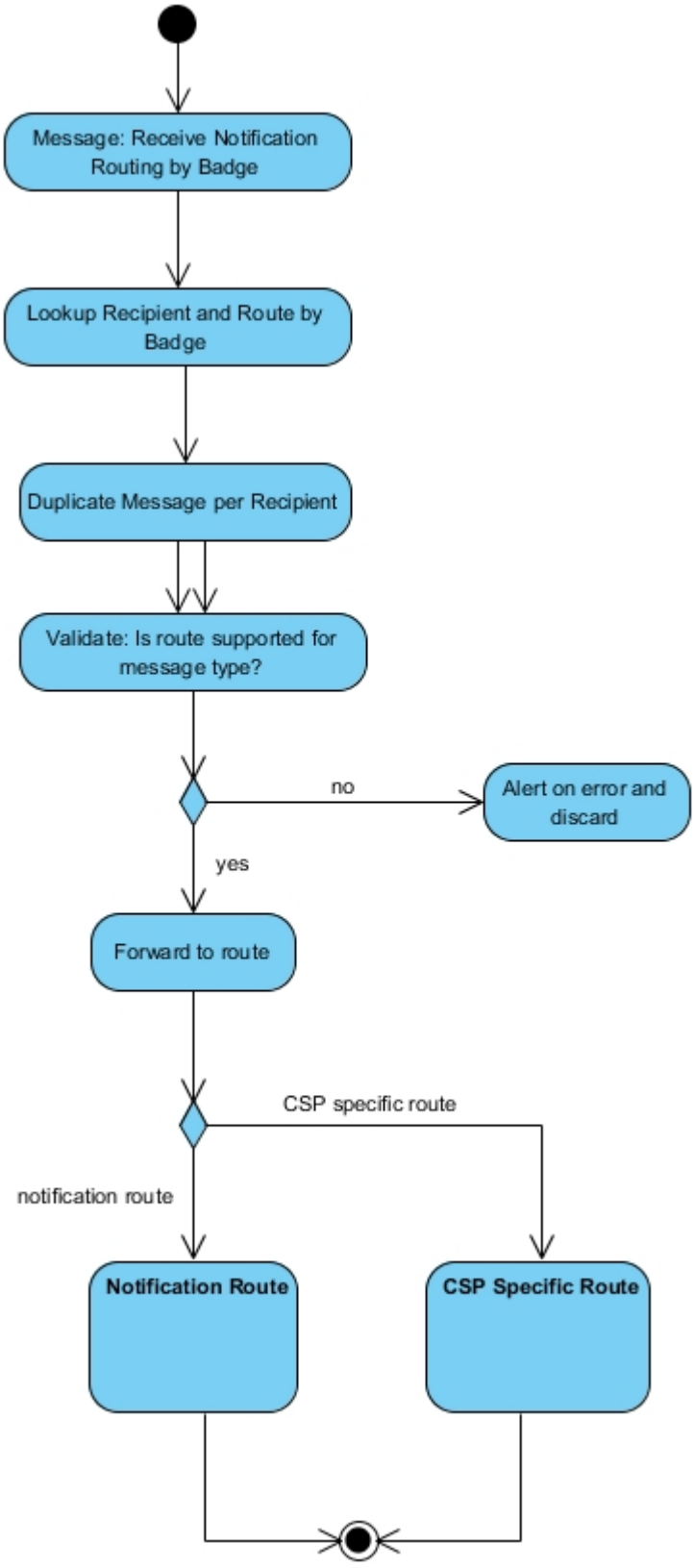
Other error scenarios may occur. For more details on common error conditions that may occur please see the API reference guide.

14. Diagrams

This section provides a range of diagrams that provide a general guidance on how the notification service will be implemented within a CSPs system.

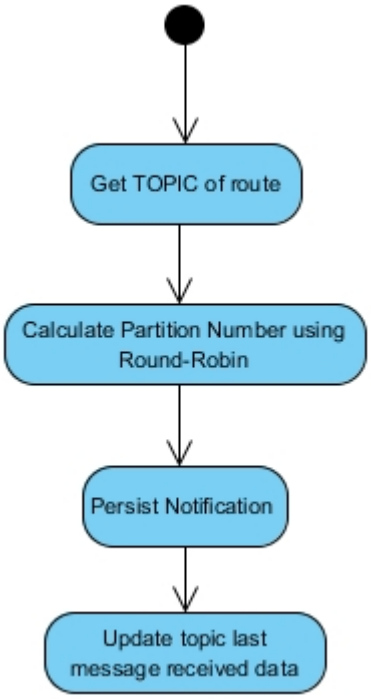
Notification Routing

This is an activity diagram that shows how a CSP might handle routing a notification that has been raised in its system against a specific badge.



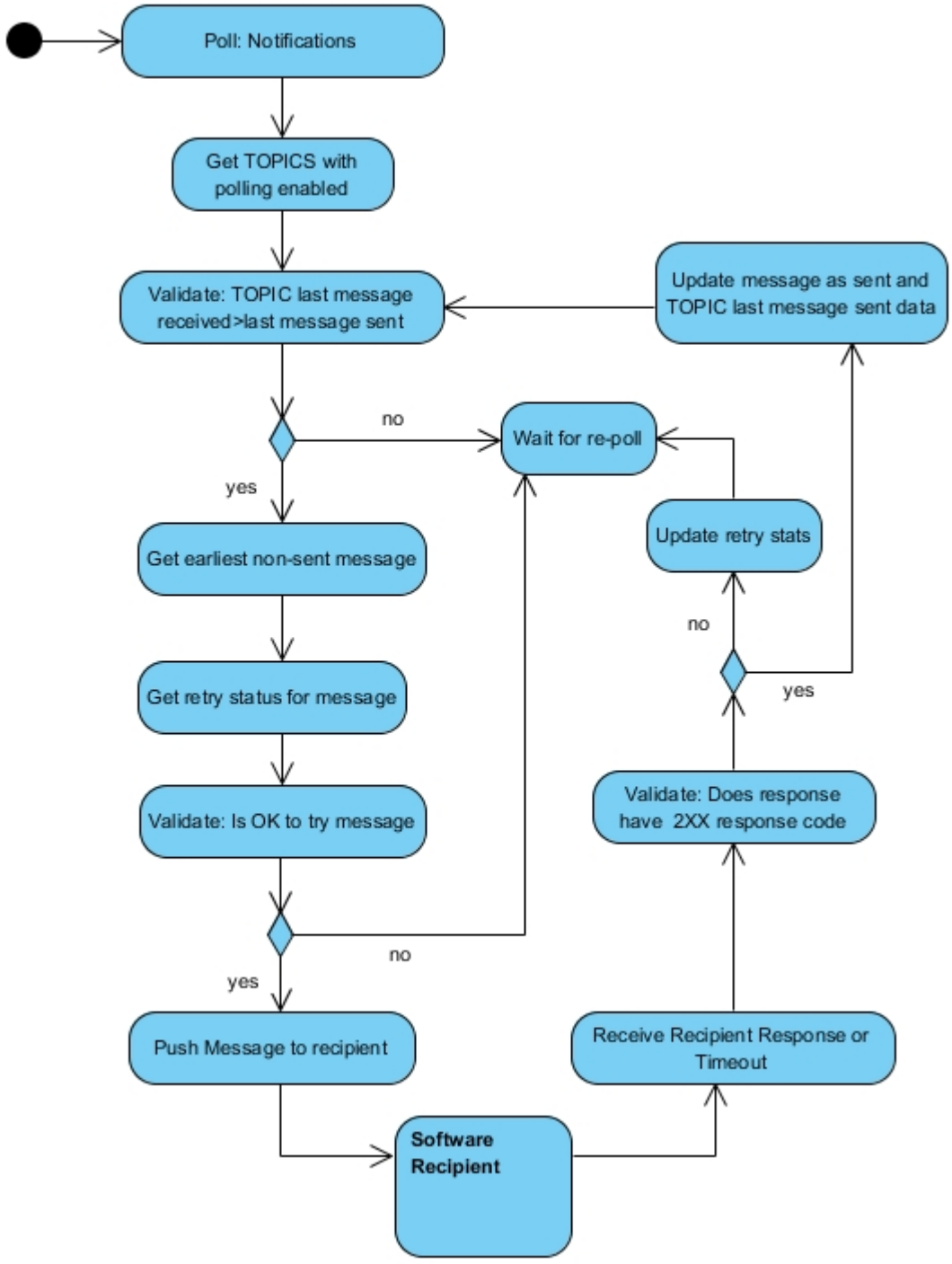
Notification Service Receive Notification

Activity diagram showing what a CSP might do when its notification service receives a message for a specific TOPIC.



Notification Push Mechanism

Activity diagram showing how a CSP might handle pushing notifications to a registered endpoint consumer.



Notification Pull Mechanism

An activity diagram showing how a CSP might handle the batch and acknowledgement process.

