

SPEC-PO-RS

Last changed:	20.11.2025 10:53:23
Version:	3.0.0-rc.6
Creator:	VDV ETS

Table of Contents

1	Introduction	7
2	Product Owner	7
2.1	Product Owner System	10
2.2	Product Owner Back-Office Main Module	10
2.3	Product Owner Back-Office Ordered Action Management Module	10
2.4	Product Owner Back-Office Static Entitlement Module	11
3	Notification Process Patterns	11
3.1	Supporting activities	14
3.1.1	Perform transaction to XY	14
3.1.2	Prepare XY parameters	17
3.2	Supporting classes	18
3.2.1	Action parameters	18
3.2.2	XY parameters	18
3.2.3	SignedXYAttestation	18
3.2.4	Notification parameters	18
3.2.5	XYNotification	19
3.2.6	ForwardedXYNotification	19
3.2.7	tNotifyXY	19
3.2.8	tNotifyXYAborted	19
3.2.9	notifyXY	19
3.2.10	notifyXYResponse	19
3.2.11	notifyXYException	19
3.2.12	forwardXYNotification	19
3.2.13	forwardXYNotificationResponse	19
3.2.14	forwardXYNotificationException	19
3.3	Perform entitlement XY and notify	19
3.3.1	Perform entitlement XY and notify	20
3.3.2	T-Module::Perform entitlement XY and notify	21
3.3.3	Notify entitlement XY	22
3.3.4	Notify entitlement XY aborted	23
3.3.5	Notify entitlement XY aborted based on attestation	23
3.3.6	Notify XY (entitlement owned)	24
3.3.7	Notify XY (entitlement non-owned)	25
3.4	Perform application XY and notify	25
3.4.1	Perform application XY and notify	26
3.4.2	T-Module::Perform application XY and notify	26
3.4.3	Notify application XY	28
3.4.4	Notify application XY aborted	28

3.4.5	Notify XY (application owned)	29
3.4.6	Notify XY (application non-owned)	30
3.5	Handle entitlement XY notification from operational perspective	30
3.5.1	Handle entitlement XY notification from operational perspective	31
3.5.2	Role-BO-Module::Handle entitlement XY notification from operational perspective	31
3.5.3	Check and process entitlement XY notification from operational perspective	32
3.5.4	Role-BO-Module::Handle entitlement XY aborted notification from operational perspective	33
3.6	Handle application XY notification from operational perspective	34
3.6.1	Handle application XY notification from operational perspective	35
3.6.2	Role-BO-Module::Handle application XY notification from operational perspective	35
3.6.3	Check and process application XY notification from operational perspective	36
3.6.4	Role-BO-Module::Handle application XY aborted notification from operation perspective	38
3.7	Handle entitlement XY notification from product perspective	38
3.7.1	Handle entitlement XY notification from product perspective	39
3.7.2	PO-BO-Module::Handle entitlement XY notification from product perspective	39
3.7.3	Check and process entitlement XY notification from product perspective	41
3.7.4	Conditionally forward notification to pCCP	41
3.8	Handle entitlement XY notification from contractual perspective	42
3.8.1	Handle entitlement XY notification from contractual perspective	43
3.8.2	pCCP-BO-Module::Handle entitlement XY notification from contractual perspective	43
3.8.3	Check and process entitlement XY notification from contractual perspective	44
3.8.4	pCCP-BO-Module::Perform specific contractual tasks on entitlement XY notification	45
3.9	Handle application XY notification from contractual perspective	46
3.9.1	Handle application XY notification from contractual perspective	47
3.9.2	pCCP-BO-Module::Handle application XY notification from contractual perspective	47
3.9.3	Check and process application XY notification from contractual perspective	48
3.9.4	pCCP-BO-Module::Perform specific contractual tasks on application XY notification	49
4	Verifying Lists Updated via Increments	50
4.1	Checksum calculation for hotlist and action list verification	52
4.2	Example calculation for an action list inventory	52
4.3	Example calculation for an application hotlist inventory	53
4.4	Example calculation for an entitlement hotlist inventory	54
5	Basic Bundle Back-Office System	55
5.1	Overview	55
5.2	Use Cases	56
5.2.1	Handle discarded messages information	56
5.2.2	Set service as available for a participant	58
5.2.3	Set service as unavailable for a participant	60

5.2.4	Retrieve CV certificate over signing key	62
5.2.5	Retrieve and distribute the CA certificate repository	64
5.2.6	Retrieve and distribute the CV certificate revocation list	66
5.2.7	Notify events	68
5.2.8	Handle events notification	70
5.2.9	Demand application hotlisting	72
5.2.10	Determine user medium owner	74
5.2.11	Demand entitlement hotlisting	76
5.2.12	Demand SAM hotlisting	78
5.2.13	Determine SAM owner	80
5.2.14	Revoke application hotlisting demand	82
5.2.15	Revoke entitlement hotlisting demand	84
5.2.16	Retrieve entitlement hotlist	86
5.2.17	Optional: Retrieve entitlement hotlist with product information	88
5.2.18	Optional: Retrieve incremental entitlement hotlist	90
5.2.19	Optional: Verify entitlement hotlist updated via increments	92
5.2.20	Update organisation hotlist inventory	94
5.2.21	Update SAM hotlist inventory	96
5.2.22	Retrieve and distribute organisation list	98
6	Basic Bundle PO-System	100
6.1	Overview	100
6.2	Use Cases	100
6.2.1	Handle entitlement unblocked notification from product perspective	100
6.2.2	Handle entitlement blocked notification from product perspective	103
6.2.3	Update hotlist inventory from product perspective	105
6.2.4	Get unclaimed list information	107
6.2.5	Get product acceptance configuration list	109
6.2.6	Add product acceptance entry to hotlist configuration	111
6.2.7	Remove product acceptance entry from hotlist configuration	113
6.2.8	Remove product acceptance from participants	115
6.2.9	Monitor SAMs from product perspective	117
6.2.10	Analyse entitlement history from product perspective	119
6.2.11	Resolve notifications with timeout warnings	121
6.2.12	Check entitlement notifications against issuance notification from product perspective	123
7	Electronic Ticket Bundle PO-System	125
7.1	Overview	125
7.2	Use Cases	125
7.2.1	Handle entitlement issued notification from product perspective	125
7.2.2	Handle entitlement terminated notification from product perspective	128
7.2.3	Handle entitlement inspected notification from product perspective	130
8	Account-Based Payment Bundle PO-System	132

8.1	Overview	132
8.2	Use Cases	132
8.2.1	Handle entitlement issued notification from product perspective	132
8.2.2	Handle entitlement terminated notification from product perspective	135
8.2.3	Handle entitlement inspected notification from product perspective	137
9	Stored-Value Payment Bundle PO-System	139
9.1	Overview	139
9.2	Use Cases	139
9.2.1	Handle entitlement issued notification from product perspective	139
9.2.2	Handle entitlement terminated notification from product perspective	142
9.2.3	Handle entitlement inspected notification from product perspective	144
9.2.4	Handle stored-value payment method recharged notification from product perspective	146
9.2.5	Handle stored-value payment method reimbursed notification from product perspective	148
10	Sale Electronic Ticket via Account-Based Payment Bundle PO-System	150
10.1	Overview	150
10.2	Use Cases	150
10.2.1	Handle account-based payment method debited notification from product perspective	150
10.2.2	Handle account-based payment method credited notification from product perspective	152
11	Sale Electronic Ticket via Stored-Value Payment Bundle PO-System	154
11.1	Overview	154
11.2	Use Cases	154
11.2.1	Handle stored-value payment method debited notification from product perspective	154
11.2.2	Handle stored-value payment method credited notification from product perspective	156
12	IN-OUT Bundle PO-System	158
12.1	Overview	158
12.2	Use Cases	158
12.2.1	Handle check-in notification from product perspective	158
12.2.2	Handle check-out notification from product perspective	160
12.2.3	Handle account-based payment method debited notification from product perspective	162
12.2.4	Handle stored-value payment method debited notification from product perspective	164
12.2.5	Handle user tariff parameters changed notification from product perspective	166
12.2.6	Demand account-based payment method charging	168
13	Ordered Action Management Bundle PO-System	170
13.1	Overview	170
13.2	Use Cases	170

13.2.1	Handle entitlement issuance order	170
13.2.2	Handle entitlement unblocking order	173
13.2.3	Handle entitlement termination order	175
13.2.4	Handle entitlement blocking order	177
13.2.5	Handle order group	179
13.2.6	Handle order cancellation	181
13.2.7	Handle ordered entitlement issued notification from product perspective	183
13.2.8	Handle ordered entitlement unblocked notification from product perspective	185
13.2.9	Handle ordered entitlement terminated notification from product perspective	187
13.2.10	Handle ordered entitlement blocked notification from product perspective	189
13.2.11	Handle retrieval request for action list	191
13.2.12	Handle retrieval request for incremental action list	193
13.2.13	Handle verification request for action list updated via increments	195
13.2.14	Generate current action list	197
13.2.15	Put action list retrieval configuration	198
13.2.16	Analyse order history from product perspective	200
13.2.17	Check for order obsolescence	202
14	Static Entitlements Bundle PO-System	204
14.1	Overview	204
14.2	Use Cases	204
14.2.1	Handle static entitlement issued notification from product perspective	204
14.2.2	Handle static entitlement inspected notification from product perspective	207
14.2.3	Handle static entitlement terminated notification from product perspective	209
14.2.4	Check static entitlement notifications against issuance notification from product perspective	211
14.2.5	Check static entitlement notifications for plausibility	213
15	Miscellaneous Bundle PO-System	214
15.1	Overview	214
15.2	Use Cases	214
15.2.1	Determine valid entitlements for given app instance ID	214
15.2.2	Handle entitlement validated notification from product perspective	216

1 Introduction

From the EFM perspective, the product owner system receives messages from the various participant systems regarding interactions with different travel authorisations and payment methods. It performs comprehensive monitoring and checks the plausibility of incoming messages.

In the CICO area, the system evaluates incoming CICO messages with regard to fares and sends corresponding preliminary calculations to the responsible CCP system.

This reference specification contains all functionality bundles and use cases for a product owner back-office system.

Depending on the deployment variant, different functionality bundles can be combined.

However, the functionality of all participating systems assigned to the product owner must be mapped and the processing of all messages must be guaranteed.

2 Product Owner

This chapter contains all components and interfaces concerning the [Product Owner](#).

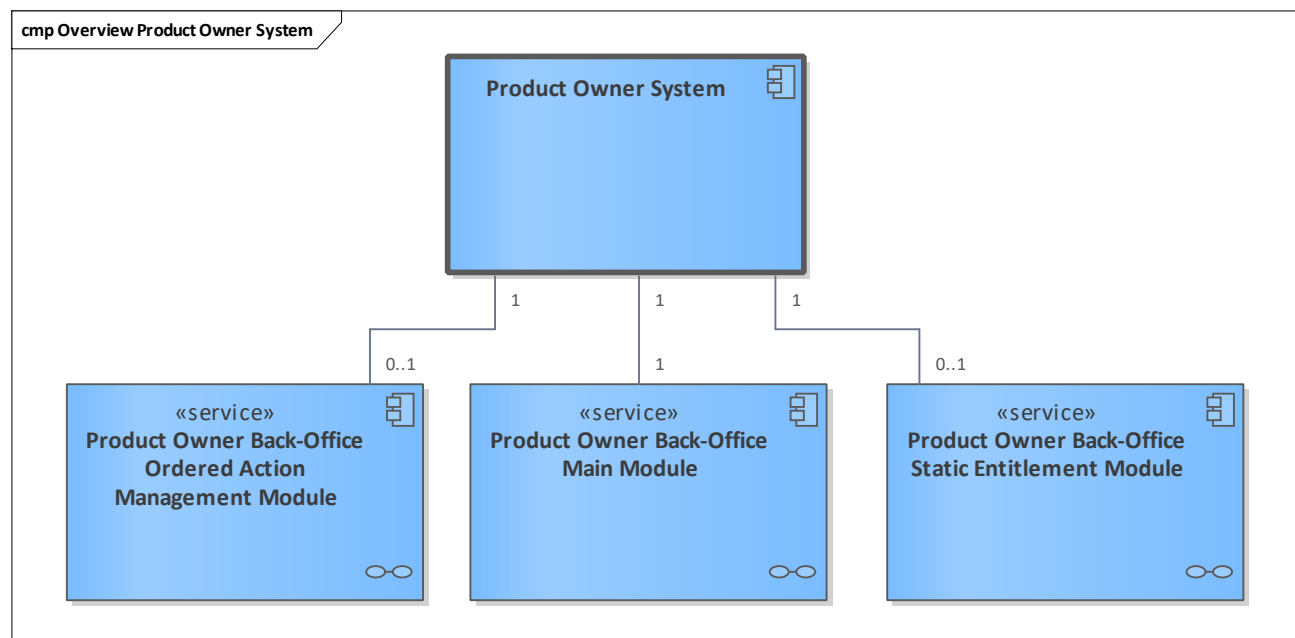


Figure 1: Overview Product Owner System

Shows the composition of a [Product Owner System](#).

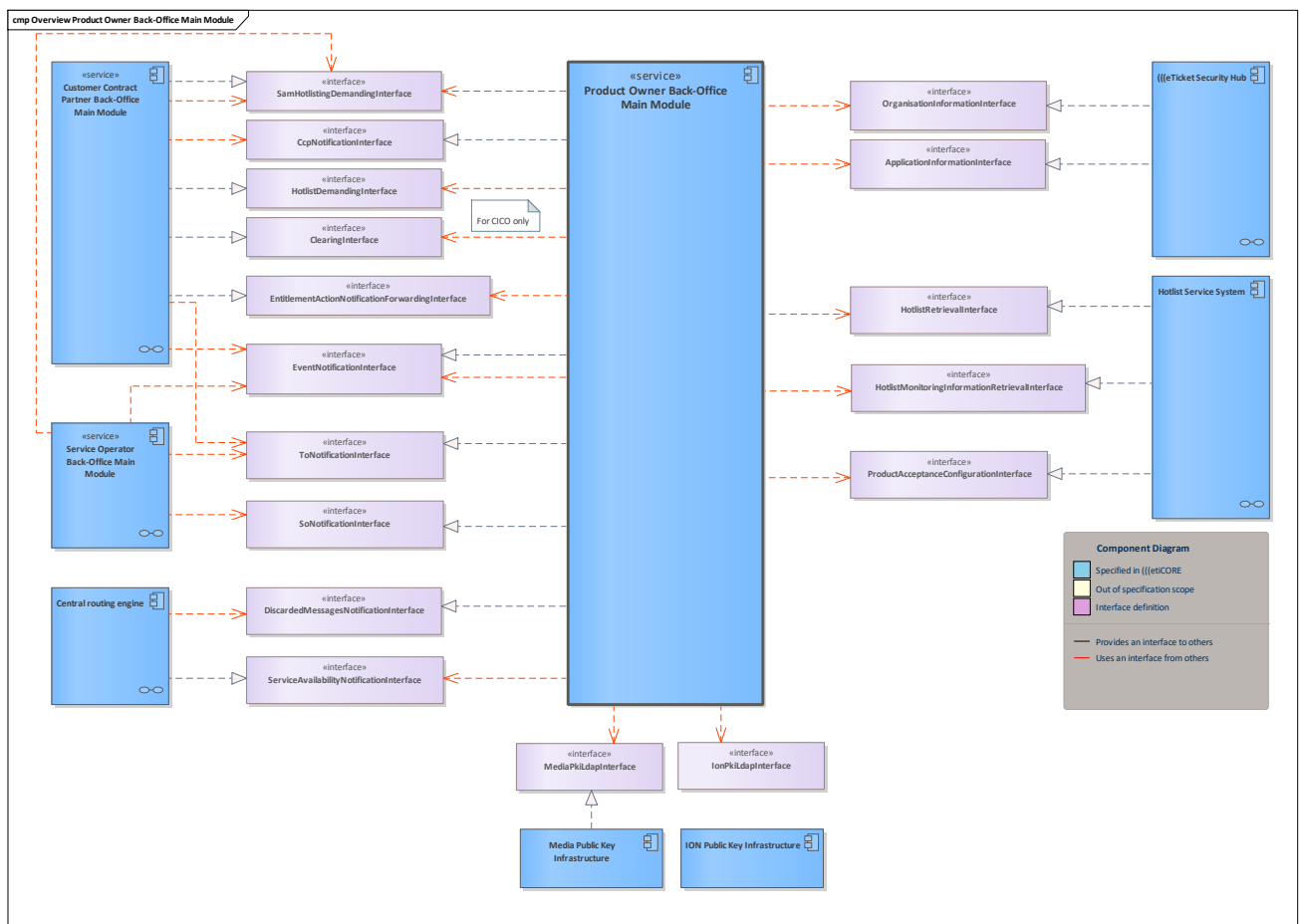


Figure 2: Overview Product Owner Back-Office Main Module

Shows the interaction of a [Product Owner Back-Office Main Module](#) via interfaces with other components.

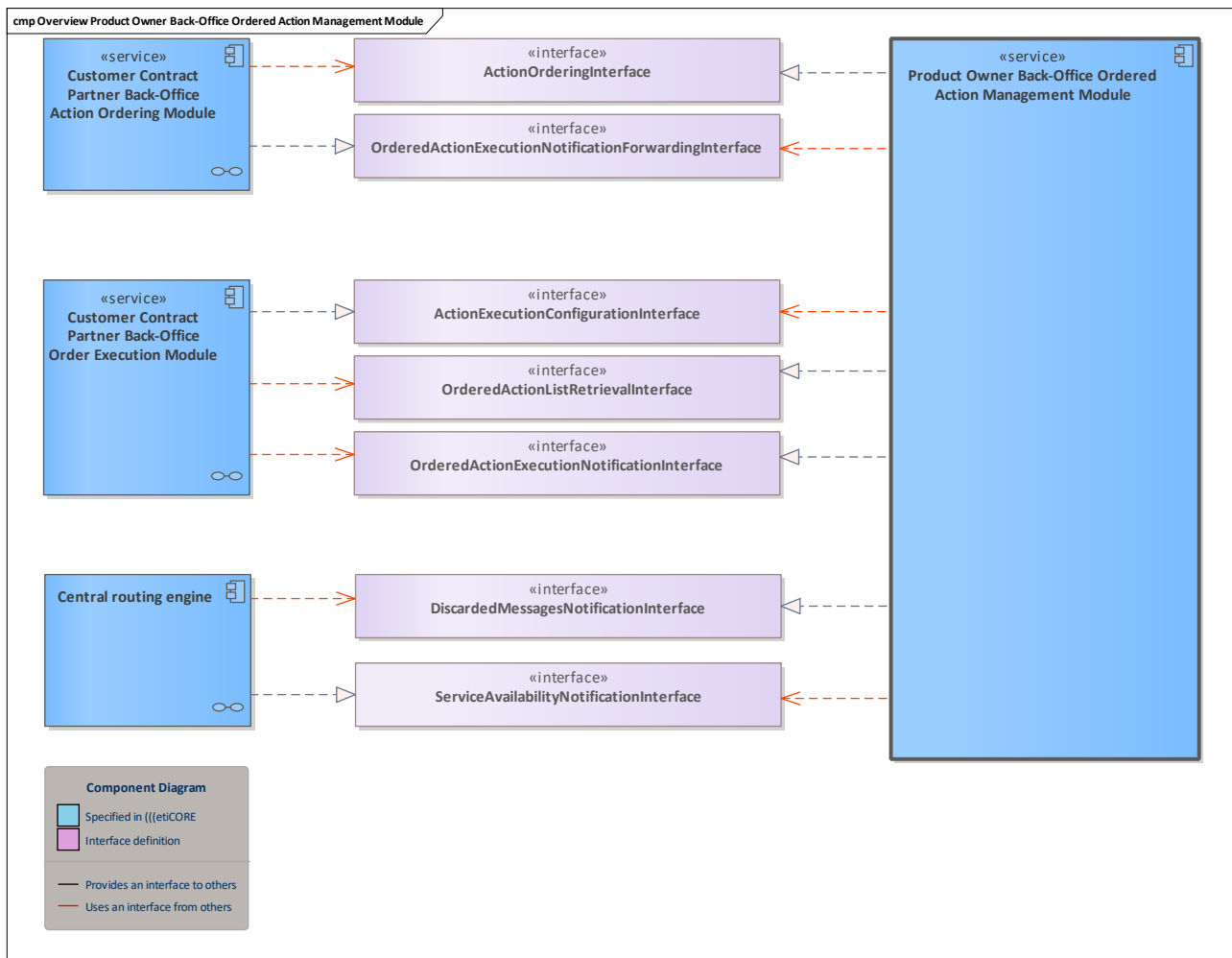


Figure 3: Overview Product Owner Back-Office Ordered Action Management Module
Shows the interaction of a [Product Owner Back-Office Ordered Action Management Module](#) via interfaces with other components.

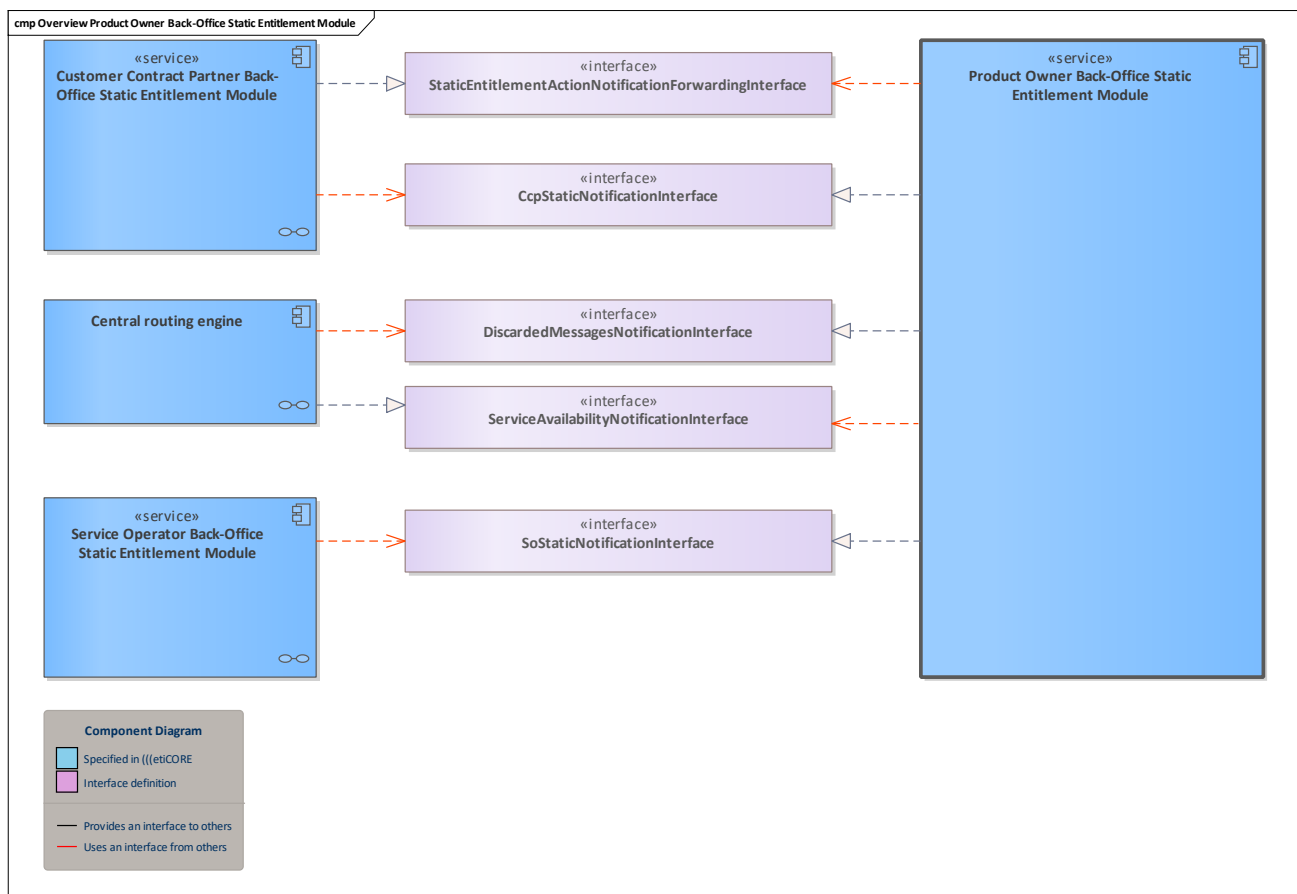


Figure 4: Overview Product Owner Back-Office Static Entitlement Module

Shows the interaction of a [Product Owner Back-Office Static Entitlement Module](#) via interfaces with other components.

2.1 Product Owner System

Component for a product owner system. Depending on the implemented functionality bundles, the product owner system consists of

- [Product Owner Back-Office Main Module](#) (always)
- [Product Owner Back-Office Ordered Action Management Module](#) (if the PO supports action management for actions to be executed on user media later on terminals via action lists)
- [Product Owner Back-Office Static Entitlement Module](#) (if the PO also works with static entitlements)

2.2 Product Owner Back-Office Main Module

Component which implements the basic functionality for a [Product Owner](#) and [Product Owner System](#).

2.3 Product Owner Back-Office Ordered Action Management Module

System (component) which implements the necessary functionality for the [Product Owner](#) and [Product Owner System](#) that supports action management for actions on user media to be executed on terminals via action lists.

2.4 Product Owner Back-Office Static Entitlement Module

System (component) which implements the necessary functionality for the [Product Owner](#) and [Product Owner System](#) that works with static entitlements.

3 Notification Process Patterns

This chapter deals with the patterns which are widely used in the model, especially for notification processes.

If you understand these patterns, you will understand the structure of most of the use cases.

The different actions that can be executed by the user medium fall into different categories with respect to who is authorised to execute them. The important distinction here is whether the executing organisation is also the owner of the target object (UM application for actions targeting the application, entitlement for actions targeting the entitlement).

For some actions, the executing organisation is never the object owner (i.e. for all actions only relevant to Service Operators, which are never owners of UMs or entitlements). Others may only be performed by the owner, i.e. unblocking (outside of ordered action execution). The rest may be performed by organisations that may or may not be the object owner, e.g. debiting a payment method or blocking entitlement or application.

The actions (and their notifications) fall into four different categories :

- Entitlement owned
- Entitlement non-owned
- Application owned
- Application non-owned

Depending on the category the action falls into, the handling of their notifications with respect to potential forwarding to the owner and with respect to when the contractual processing happens may be different.

Templates for the execution of actions and the handling of their notifications for the categories entitlement owned and non-owned, as well as application owned and non-owned are shown.

Within these categories, the notification handling is very similar.

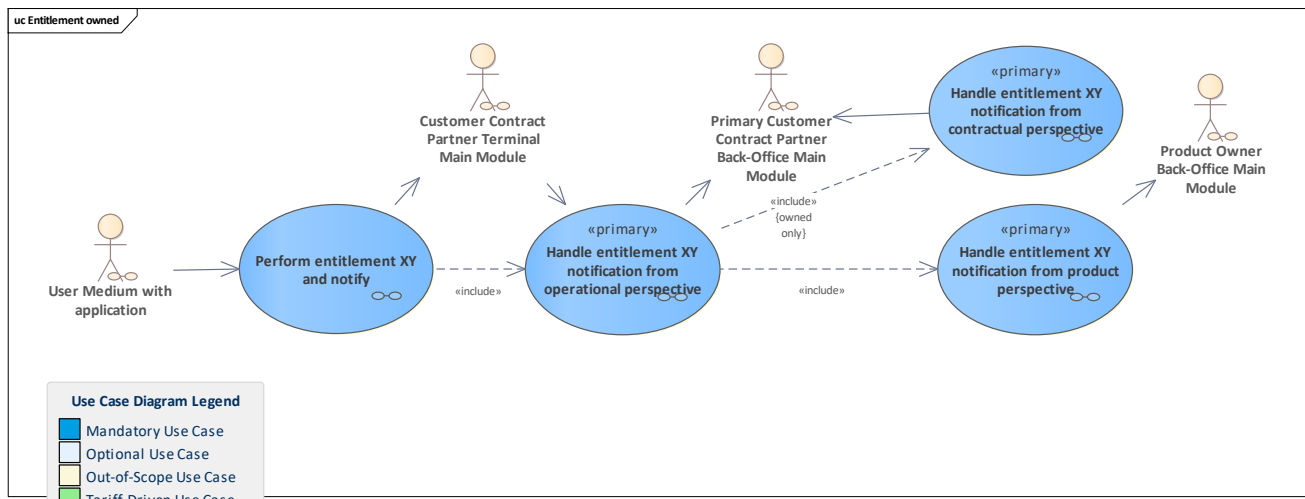


Figure 5: Entitlement owned

An entitlement-specific notification can only be created by the owner of the entitlement. The PO is informed about the action by the party performing the action, which always is the owner in this scenario.

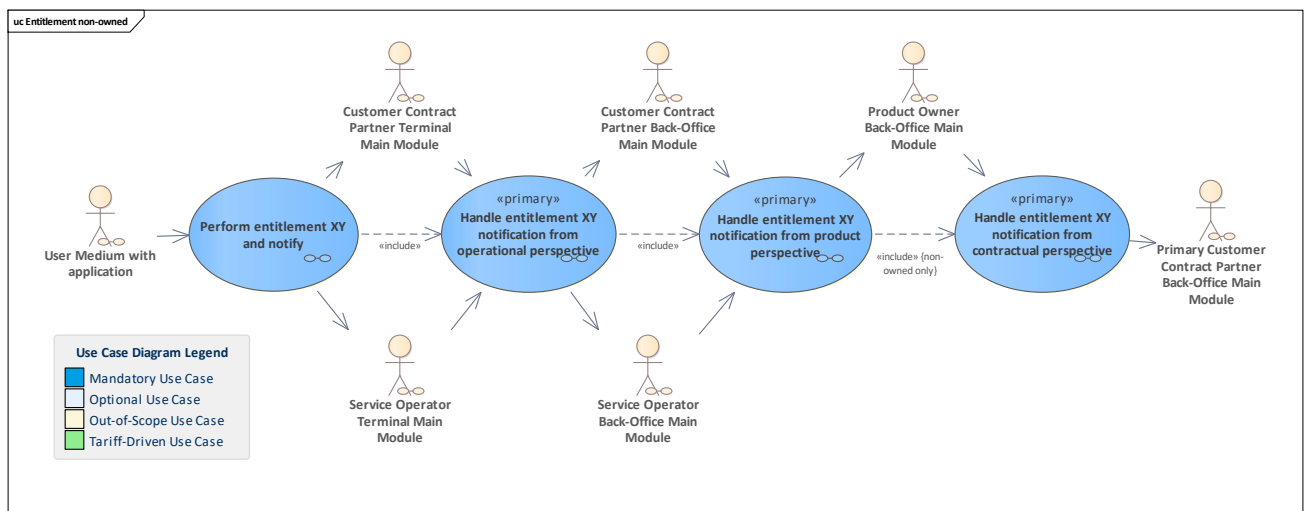


Figure 6: Entitlement non-owned

An entitlement-specific notification can only be created by parties other than the owner of the entitlement. The PO is informed about the action by the party performing the action. The owner is informed about the action by the PO.

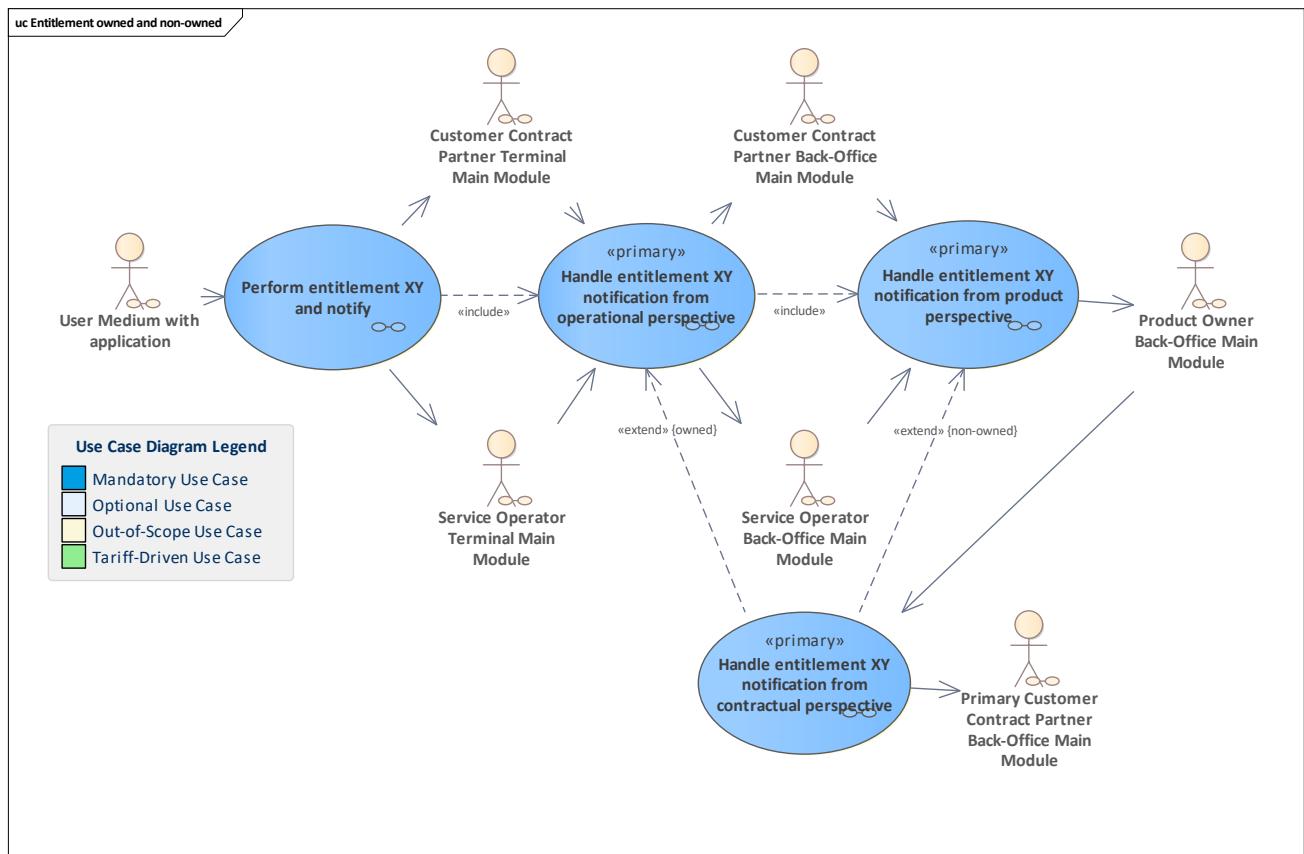


Figure 7: Entitlement owned and non-owned

An entitlement-specific notification can be created by the owner of the entitlement and other parties.

The PO is informed about the action by the party performing the action.

The owner is informed about the action by the PO if he did not perform the action himself.

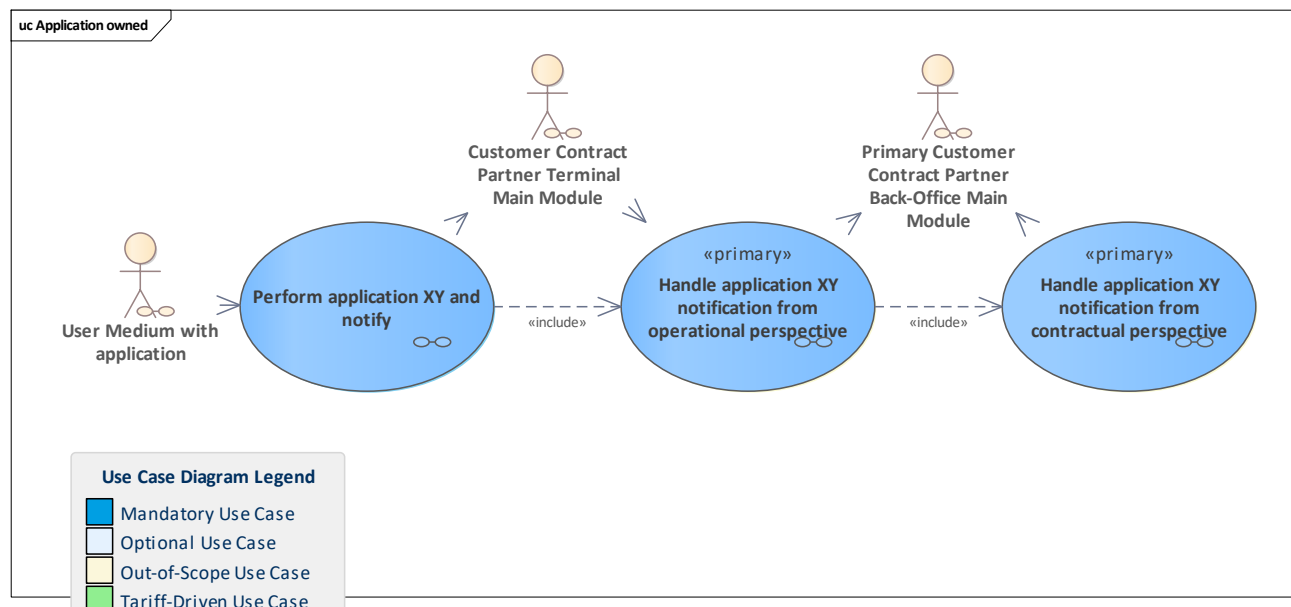


Figure 8: Application owned

An application-specific notification can only be created by the owner of the application.

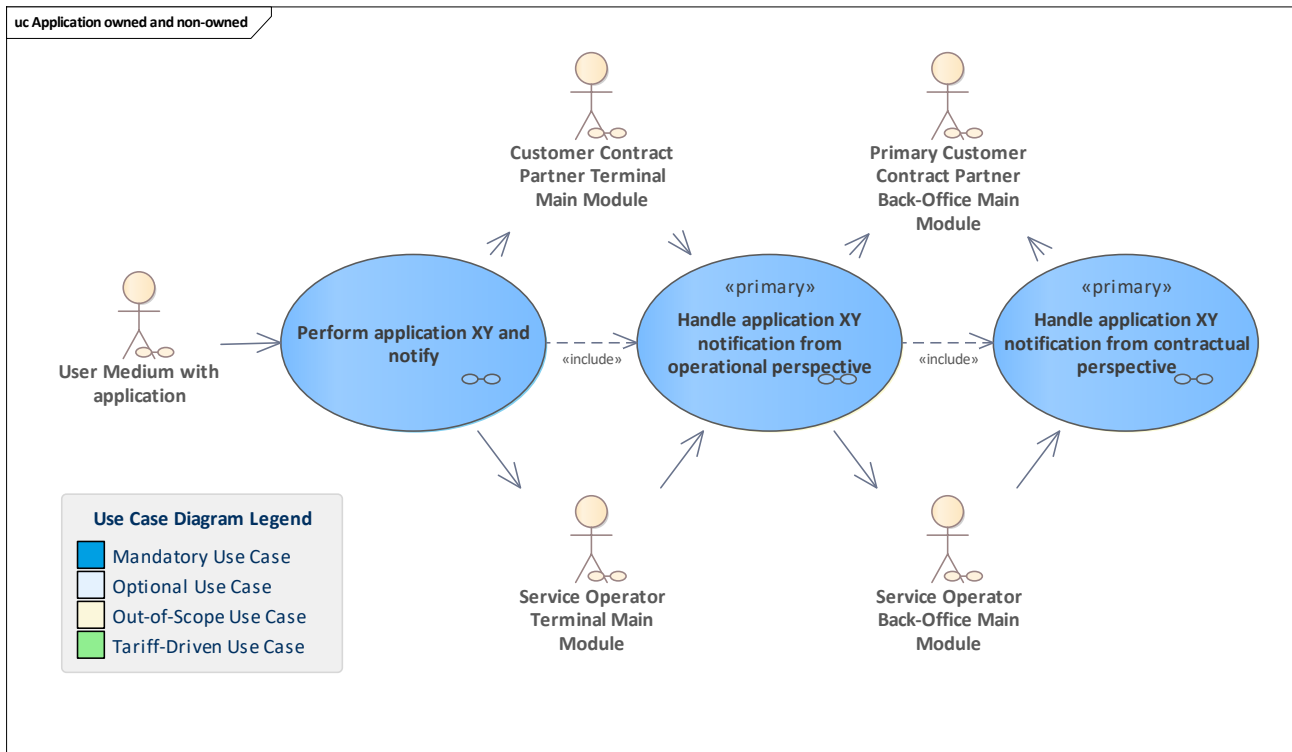


Figure 9: Application owned and non-owned

An application-specific notification can be created by the owner of the application and other parties.

The owner is informed about the action by the party performing the action if he did not perform the action himself.

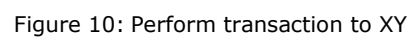
3.1 Supporting activities

This chapter contains activities that are used in more than one use case.

3.1.1 Perform transaction to XY

Shows the actual transaction including XY. May start by ensuring, that a session is established. This is left out if there is a business requirement to securely retrieve the entitlement in question before performing the operation since that already had to be done in a session. The next step is to prepare the XY parameters for the action to be performed. Finally, it contains a transaction following the schema described in [Transaction](#).

In some special cases, a transaction may involve more than one action execution. In that case, there will be a variation of this diagram type leaving out the commit transaction step at the end (and the corresponding timeout error situation). This variation is then used in conjunction with a classic version of this diagram type within a single [Role-T-Module::Perform entitlement XY and notify](#) diagram.



3.1.1.2 SAM::Authorise XY

Call the SAM operation AUTHORISE_ENTITLEMENT/AUTHORISE_ACTION using the prepared parameters.

Since the SAM performs a roll-back in case of an error, no special error handling is shown here. If the SAM configuration does not allow the transaction to be completed, the error scenario "Action aborted" should be followed. This is not shown here, since the terminal should be able to anticipate this situation, e.g. by retrieving the SAM product issuance rights before trying to AUTHORISE_ENTITLEMENT to make sure the SAM is configured to issue the product in question.

3.1.1.3 SAM::Validate XY and authorise commit

Call the SAM operation VALIDATE_ACTION using the secured attestation prepared by the UM. The only relevant error code in this situation is the error code *SM data objects incorrect*. All other error codes should either not happen at all or be easy to correct by the terminal. If that is not true for any reason, proceed as if the action had to be aborted.

If the connection to the SAM or UM is lost during the execution of this command, the transaction is also aborted.

3.1.1.4 UM::Commit transaction

Call the UM operation COMMIT_TRANSACTION using the secured attestation prepared by the UM.

The only relevant error code in this situation is the error code *SM data objects incorrect*. All other error codes should either not happen at all or be easy to correct by the terminal. If that is not true for any reason, proceed as if the action had to be aborted.

If the connection to the UM is lost during the execution of this command such that it cannot be ruled out that the command successfully ran on the UM side (and only the response was not transmitted), the *commit timeout* error scenario has to be followed.

3.1.1.5 UM::Execute XY

Call the UM operation ISSUE_ENTITLEMENT / EXECUTE_ACTION using the Command APDU readily prepared by the SAM.

In case of an error, the terminal cannot correct the message to eliminate the problem since it is, and has to be, secured (encrypted and MAC-signed) using the session keys only SAM and UM know. Thus, the operation has to be re-authorised requiring the use of further SAM counters, effectively aborting the transaction since it may - on a business level - also affect other actions within the same transaction or may already have (in case of the error code *SM data objects incorrect*) reset the session on the UM side. For the sake of simplicity, this is not distinguished here and the whole transaction is aborted.

If the connection to the SAM or UM is lost during the execution of this command, the transaction is also aborted.

3.1.1.6 Update shadow copies of SAM counters

Update the shadow copies of the SAM counters.

For AUTHORISE_ACTION this means incrementing the SAM action counter by one.

For AUTHORISE_ENTITLEMENT this means incrementing the SAM entitlement issuance counter and the product issuance counter corresponding to the issued product by one.

3.1.2 Prepare XY parameters

Composes the parameters of the UM operation (ISSUE_ENTITLEMENT/EXECUTE_ACTION) that are set by the terminal for action parameters. These parameters are then given to the authorising operation of the SAM (AUTHORISE_ENTITLEMENT/AUTHORISE_ACTION) and further adjusted by the SAM yielding a fully prepared Command APDU to be used for the UM.

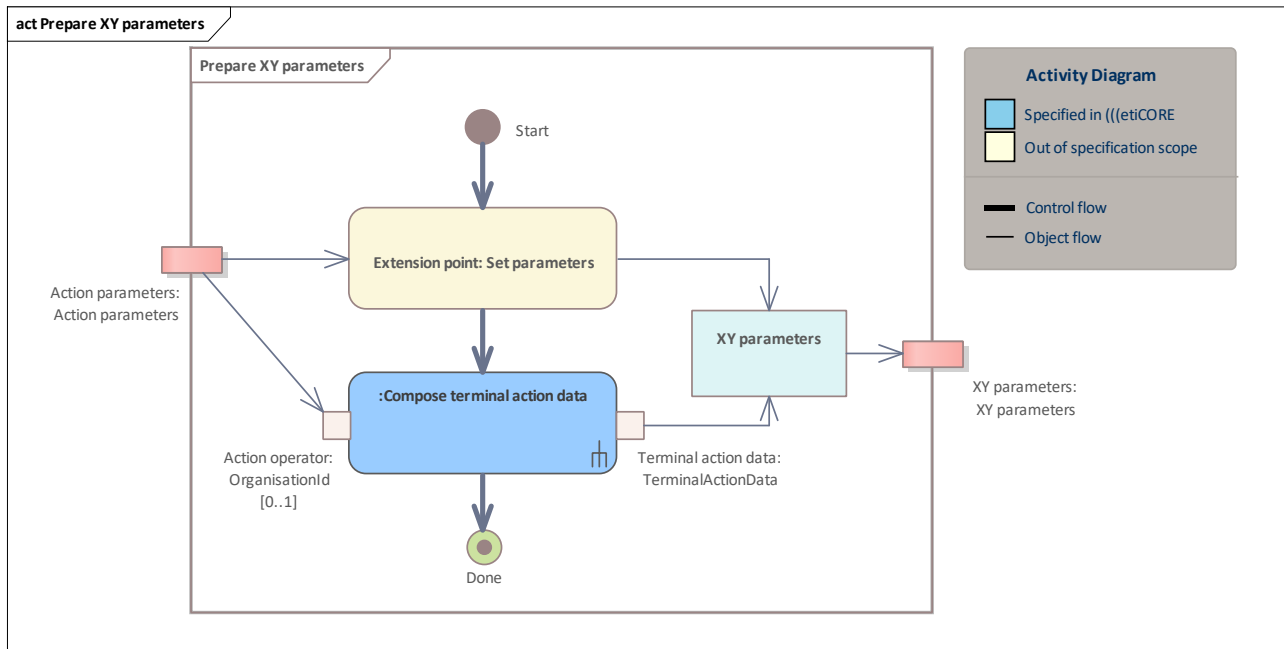


Figure 11: Prepare XY parameters

3.1.2.1 Extension point: Set parameters

This extension point is a placeholder for the use case-specific preparatory steps to create an action parameters.

For entitlement actions, the product ID given via the parameters is always part of the action parameters.

For application actions, a pseudo product ID is constructed using the UM Owner ID as the PO Org ID.

3.2 Supporting classes

Gives an overview of the supporting classes used for the pattern diagrams.

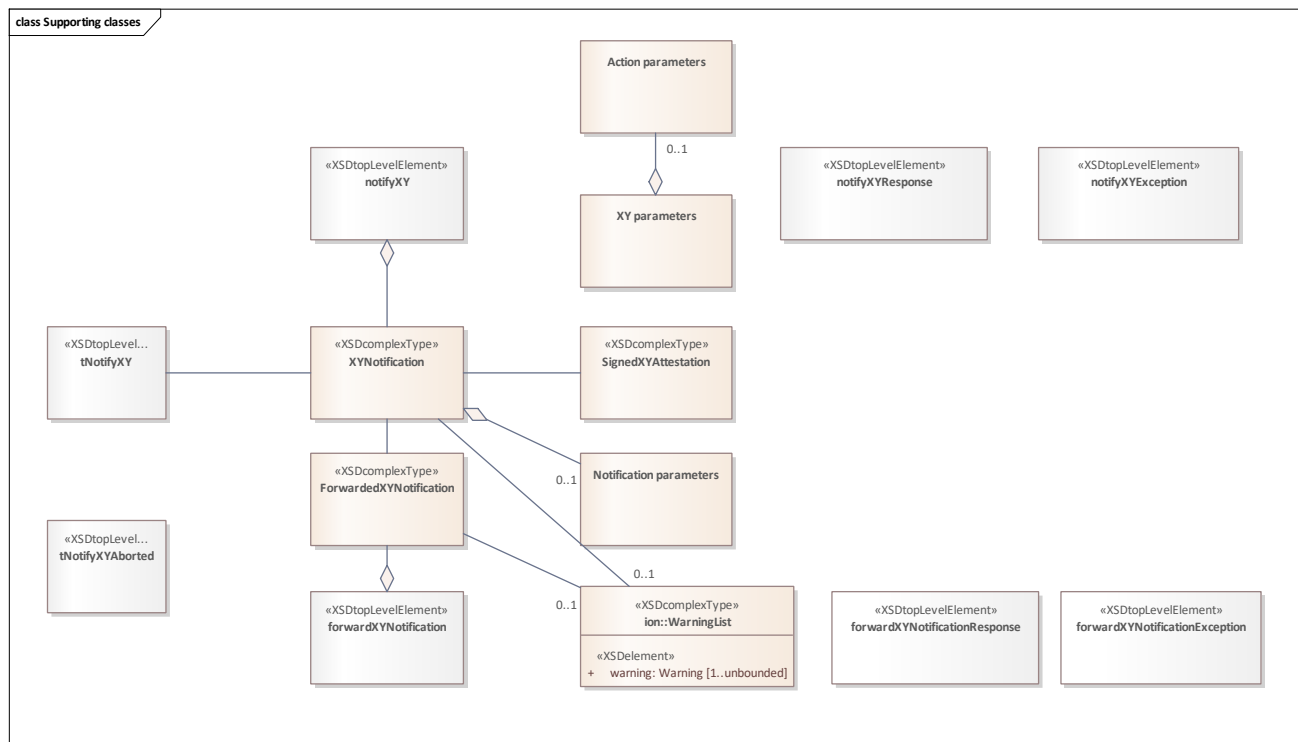


Figure 12: Supporting classes

3.2.1 Action parameters

Only applicable to entitlement actions.

Always contains the entry ID and the product ID.

May contain additional, use case-specific parameters, e.g. action tariff parameters.

3.2.2 XY parameters

Parameters for the action used to call the SAM operation authorising the action on the UM side.

3.2.3 SignedXYAttestation

An attestation of successful action execution signed by the UM involved.

Also identifies the UM so that back-office systems can retrieve the corresponding certificate and verify the signature.

3.2.4 Notification parameters

Additional parameters to insert into the notification bearing the signed attestation.



3.2.5 XYNotification

Every XY notification consists of the following three building blocks:

- an XY attestation signed by the UM
- for some use cases: additional information (see [Notification parameters](#))
- optional: a warning list

3.2.6 ForwardedXYNotification

A forwarded XY notification is a wrapper around the XY notification that may carry additional events in a separate event list.

3.2.7 tNotifyXY

Element used to transmit an XY notification from the terminal to its back-office system.

3.2.8 tNotifyXYAborted

Element used to transmit the information about an aborted XY action from the terminal to its back-office system.

3.2.9 notifyXY

Element used to inform the product owner about an entitlement action execution and to inform the owner of an application about an application action execution.

3.2.10 notifyXYResponse

Element used in response to [notifyXY](#) for normal process termination.

3.2.11 notifyXYException

Element used in reply to [notifyXY](#) for abnormal process termination.

3.2.12 forwardXYNotification

Element used to inform the entitlement owner about an entitlement action execution by a third party.

3.2.13 forwardXYNotificationResponse

Element used in response to [forwardXYNotification](#) for normal process termination.

3.2.14 forwardXYNotificationException

Element used in reply to [forwardXYNotification](#) for abnormal process termination.

3.3 Perform entitlement XY and notify

See [Perform entitlement XY and notify](#)

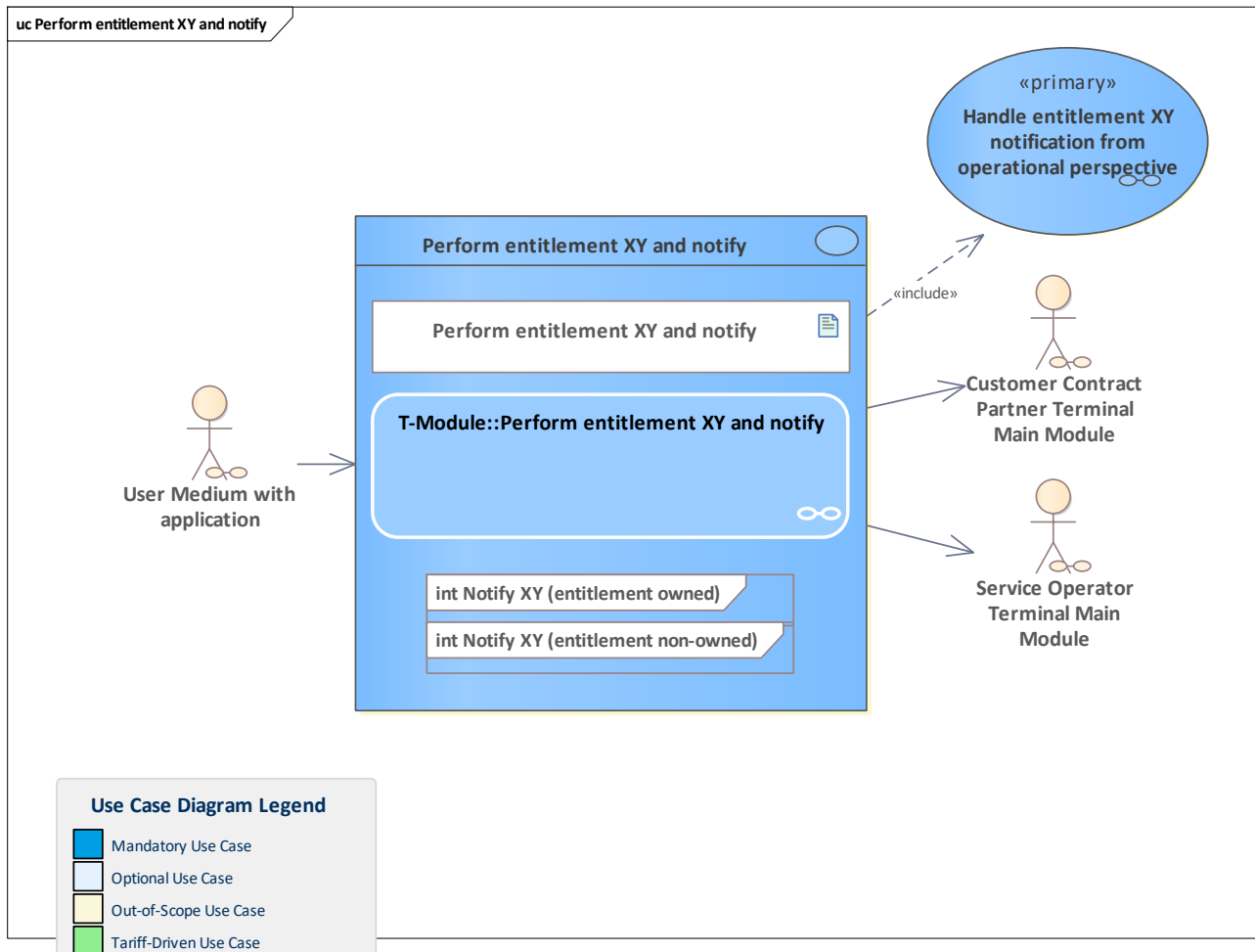


Figure 13: Perform entitlement XY and notify

3.3.1 Perform entitlement XY and notify

The user medium-based entitlement action XY is performed by the terminal in conjunction with a SAM and the back-office system corresponding to the terminal is notified about the action execution.

Combines the transaction including the action XY and the notification processes triggered by this. Shows the interaction between the UM transaction and the notification processes. The timeout warning is generated in the corresponding diagram if needed.

The business preconditions to perform the transaction (e.g. determining the actual need to block an entitlement or making sure that a stored-value payment method is sufficiently charged) are already satisfied before the process shown in this diagram begins.

Note:

Executing orders in the context of ordered action management involves the same UM operations as outside of that scope, but leads to different notification processes (partly belonging to different notification categories). In this case, only one diagram "Perform transaction to XY" may be needed, which is used in two diagrams "T-XYZ::Perform XY and notify" combining it with different notification activities (i.e. the ordered and the regular variant).

3.3.2 T-Module::Perform entitlement XY and notify

See [Perform entitlement XY and notify](#)

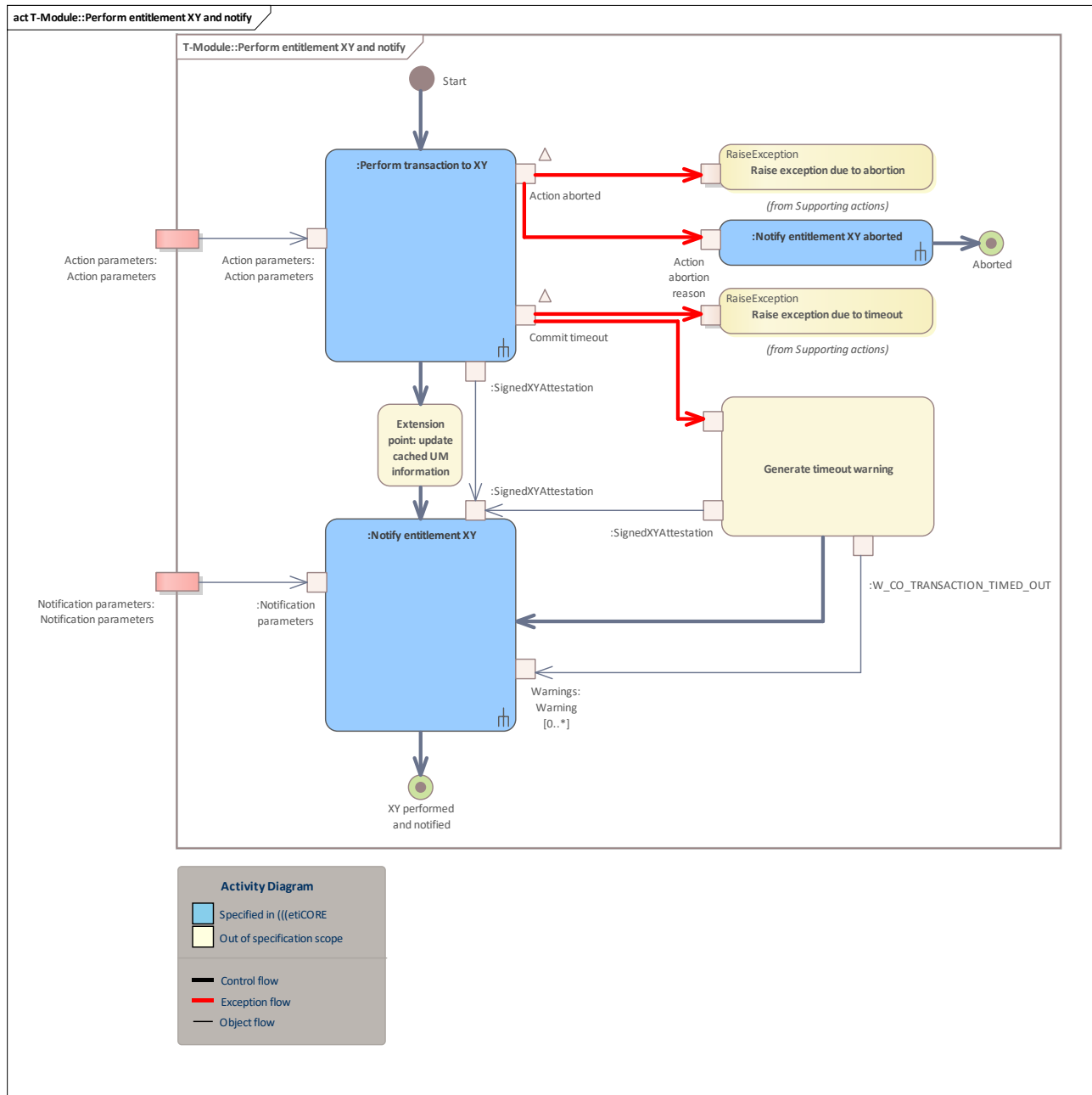


Figure 14: T-Module::Perform entitlement XY and notify

3.3.2.1

See [Perform transaction to XY](#)

3.3.2.2

See [Notify entitlement XY](#)

3.3.2.3

See [Notify entitlement XY aborted](#)

3.3.2.4 Extension point: update cached UM information

Update the cached information about the UM as appropriate for the executed action, e.g. entitlement action counter, application status, etc.

3.3.2.5 Generate timeout warning

Generate a timeout warning to be placed into the notification sent about the executed action.

3.3.3 Notify entitlement XY

The terminal notifies its back-office system about a successful action execution.

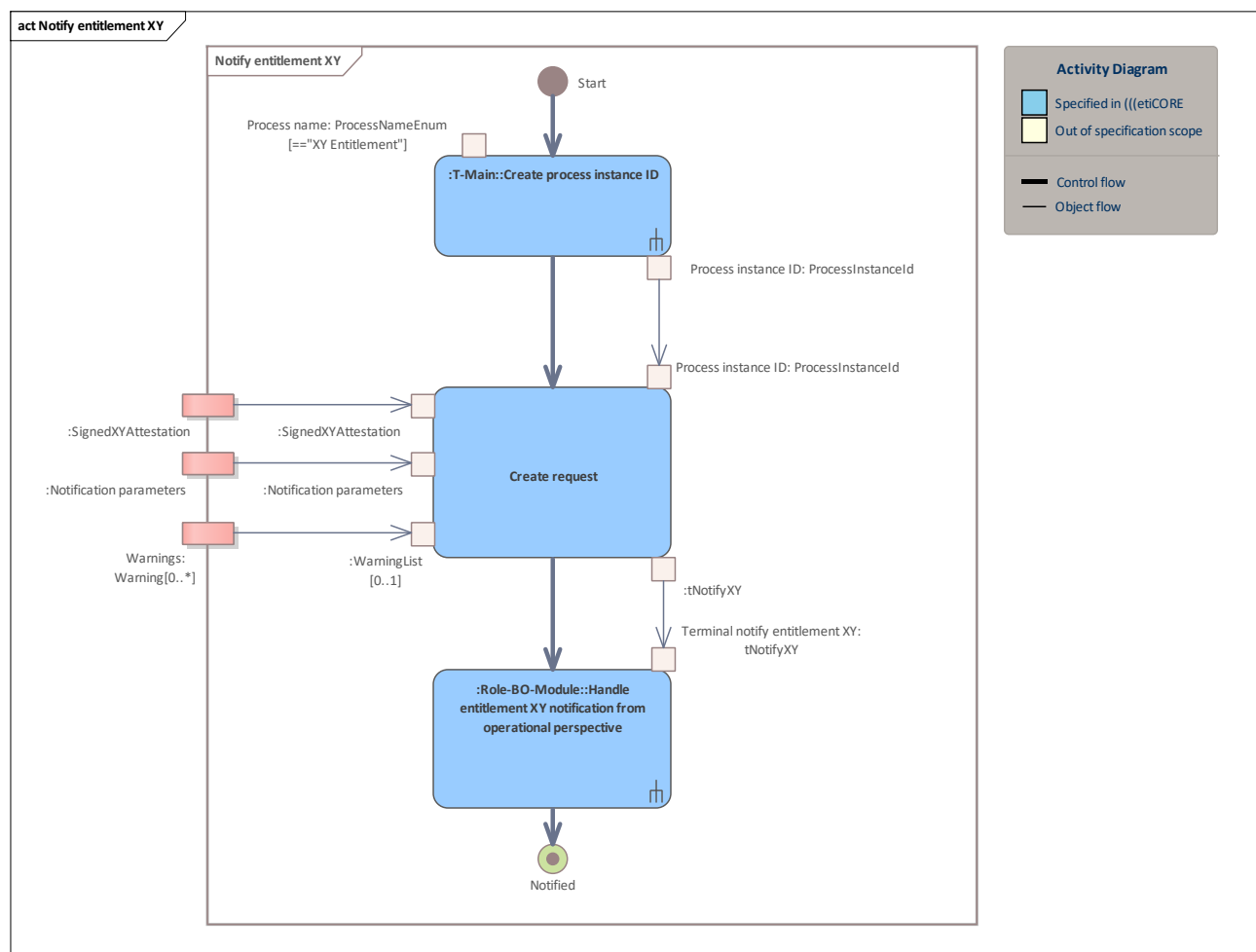


Figure 15: Notify entitlement XY

3.3.4 Notify entitlement XY aborted

The terminal notifies its back-office system about an aborted action.

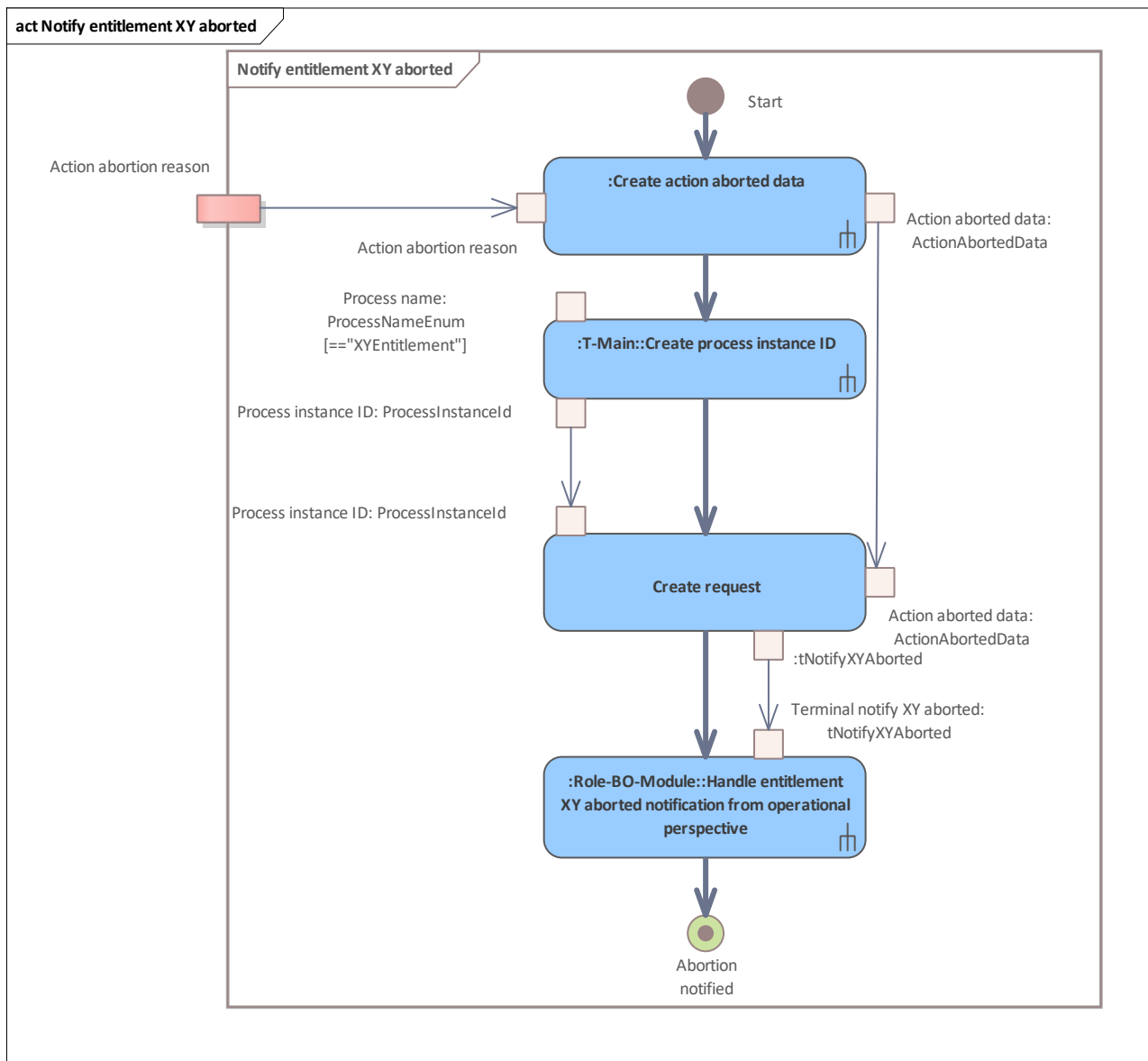


Figure 16: Notify entitlement XY aborted

3.3.5 Notify entitlement XY aborted based on attestation

If there are several actions within a transaction, this variant is to be used for the actions already completed (but not committed).

Since the attestation is already available, we can extract the required data from it instead of reproducing it.

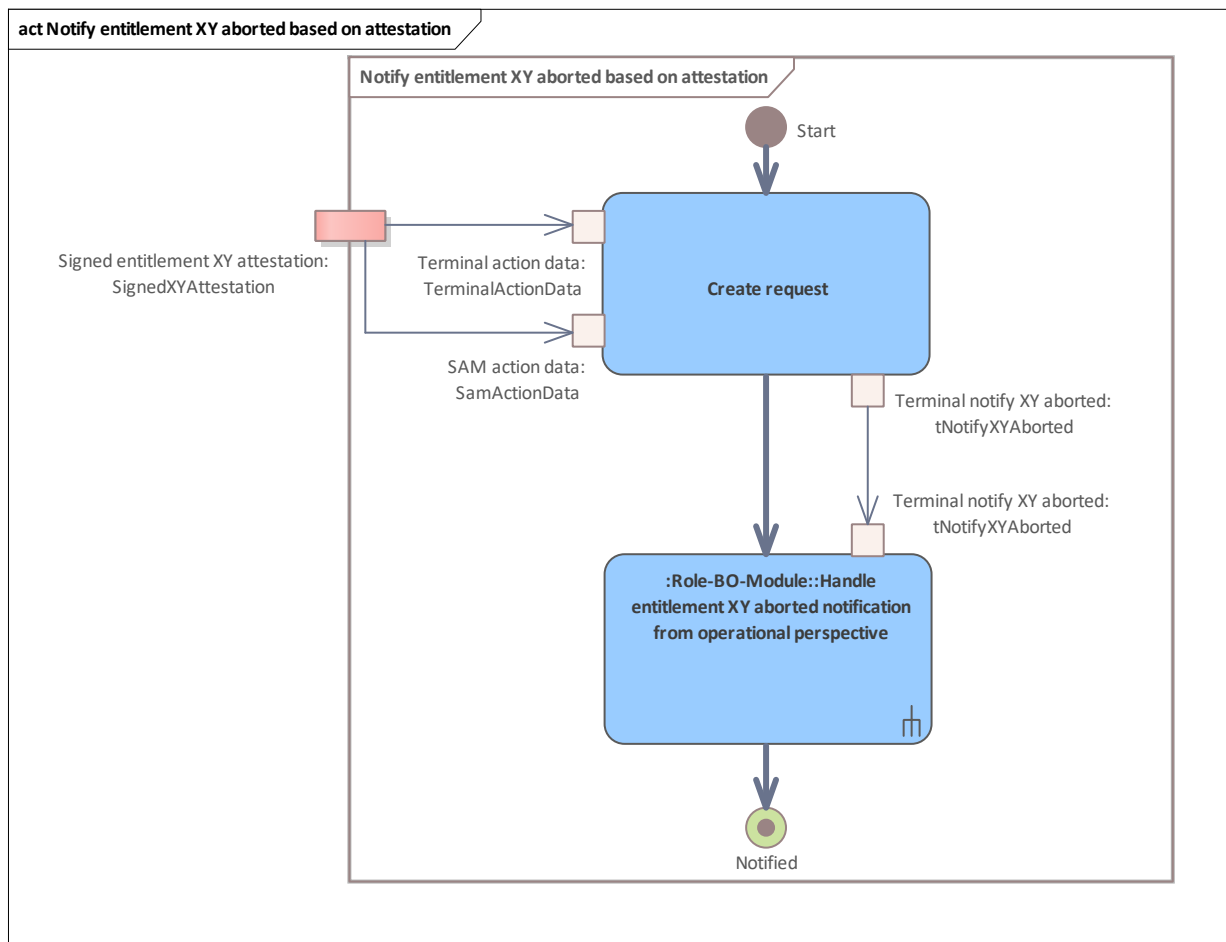


Figure 17: Notify entitlement XY aborted based on attestation

3.3.6 Notify XY (entitlement owned)

Shows the message chain starting in a terminal (normally customer contract partner terminal), sent to a terminal operator back-office system (primary customer contract partner system) and finally sent to the product owner system.

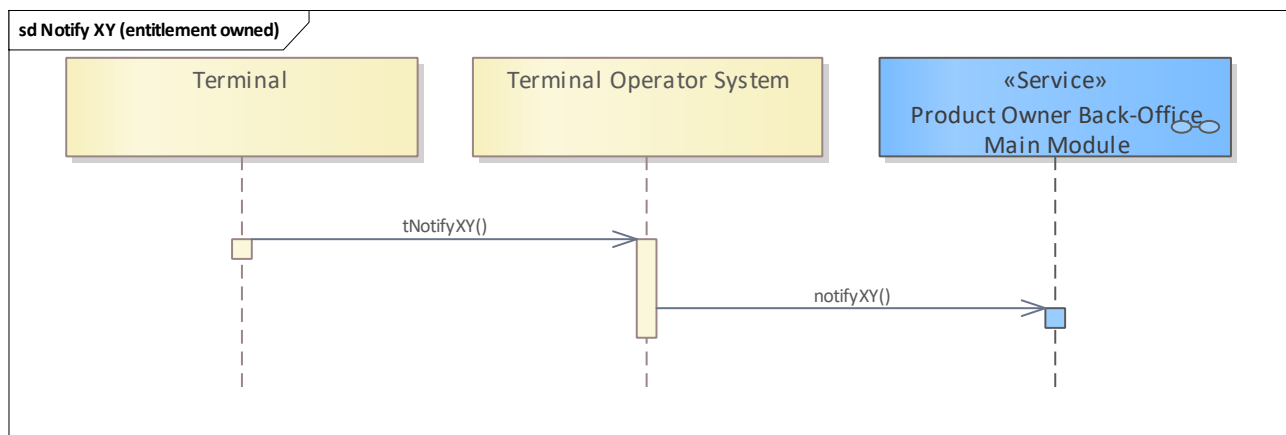


Figure 18: Notify XY (entitlement owned)

See [Notify XY \(entitlement owned\)](#)

3.3.7 Notify XY (entitlement non-owned)

Shows the message chain starting in a terminal (customer contract partner terminal or service operator terminal), sent to a terminal operator back-office system (customer contract partner system or service operator system) then sent to the product owner system and finally to the owning primary customer contract partner system.

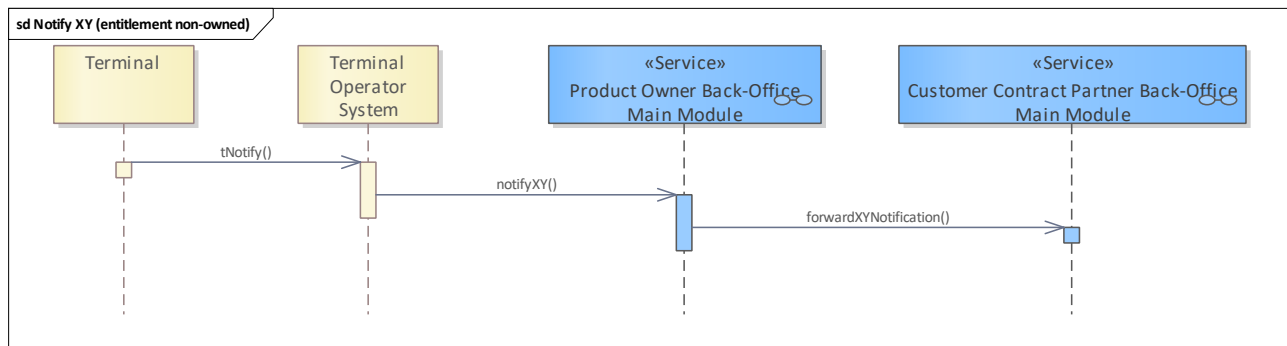


Figure 19: Notify XY (entitlement non-owned)

See [Notify XY \(entitlement non-owned\)](#).

3.4 Perform application XY and notify

See [Perform application XY and notify](#)

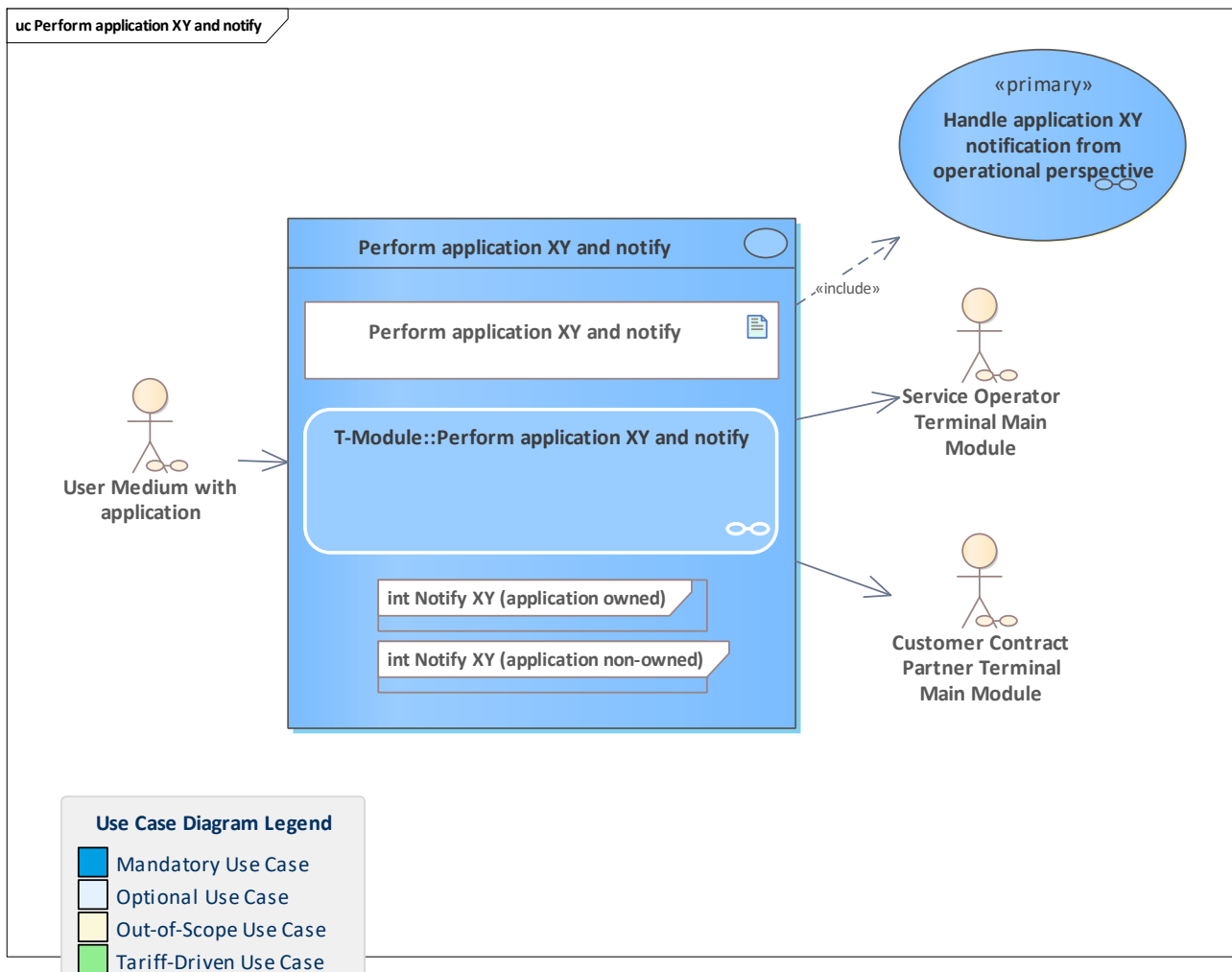


Figure 20: Perform application XY and notify

3.4.1 Perform application XY and notify

The UM application action XY is performed by the terminal in conjunction with a SAM and the back-office system corresponding to the terminal is notified about the action execution.

Combines the transaction including the action XY and the notification processes triggered by this. Shows the interaction between the UM transaction and the notification processes. The timeout warning is generated in this diagram if needed.

The business preconditions to perform the transaction (e.g. determining the actual need to block an application) are already satisfied before the process shown in this diagram begins.

3.4.2 T-Module::Perform application XY and notify

See [Perform application XY and notify](#)

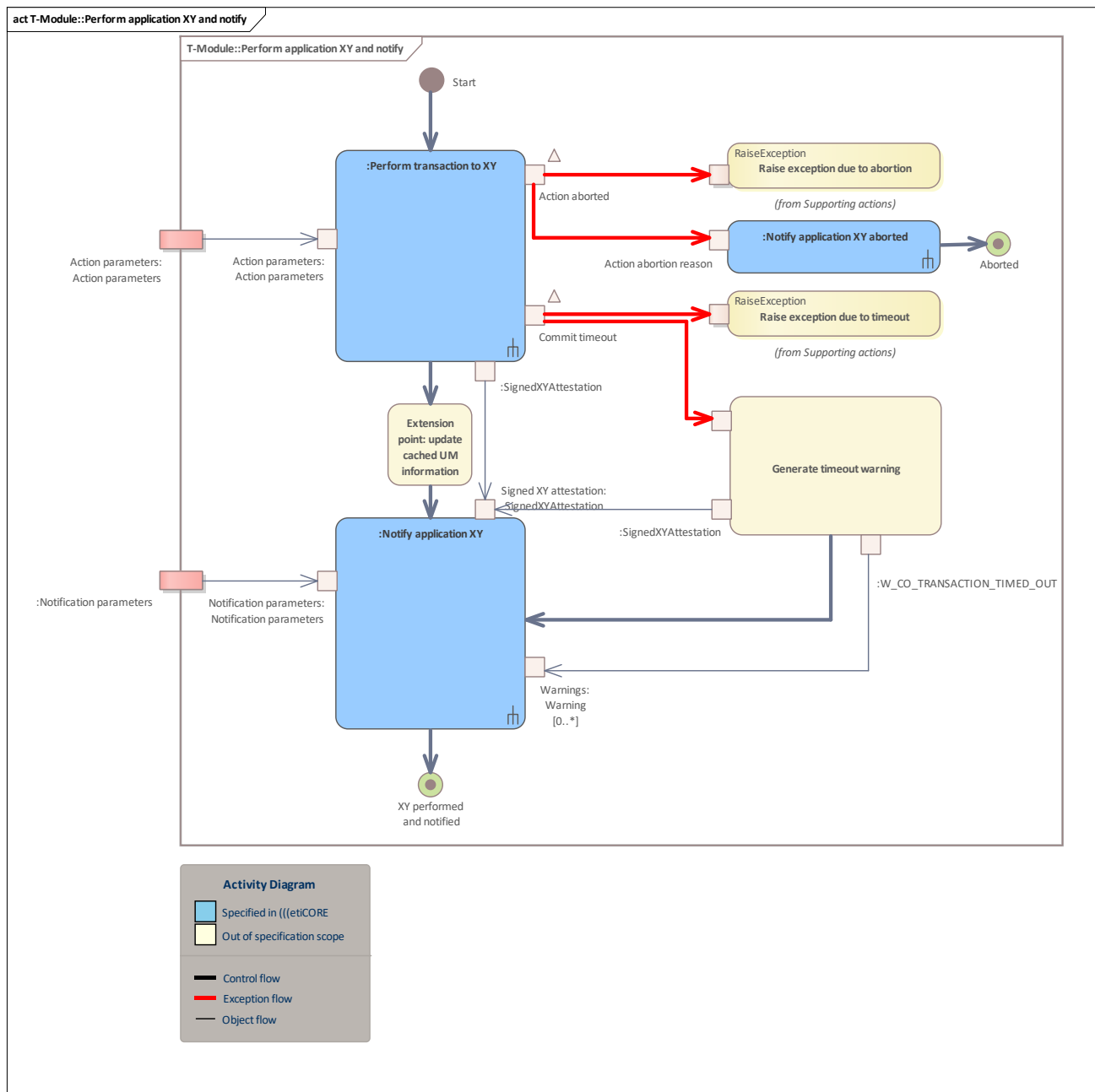


Figure 21: T-Module::Perform application XY and notify

3.4.2.1

See [Perform transaction to XY](#)

3.4.2.2

See [Notify entitlement XY aborted](#)

3.4.2.3

See [Notify entitlement XY](#)

3.4.2.4 Extension point: update cached UM information

Update the cached information about the UM as appropriate for the executed action, e.g. entitlement action counter, application status, etc.

3.4.2.5 Generate timeout warning

Generate a timeout warning to be placed into the notification sent about the executed action.

3.4.3 Notify application XY

The terminal notifies its back-office system about a successful action execution.

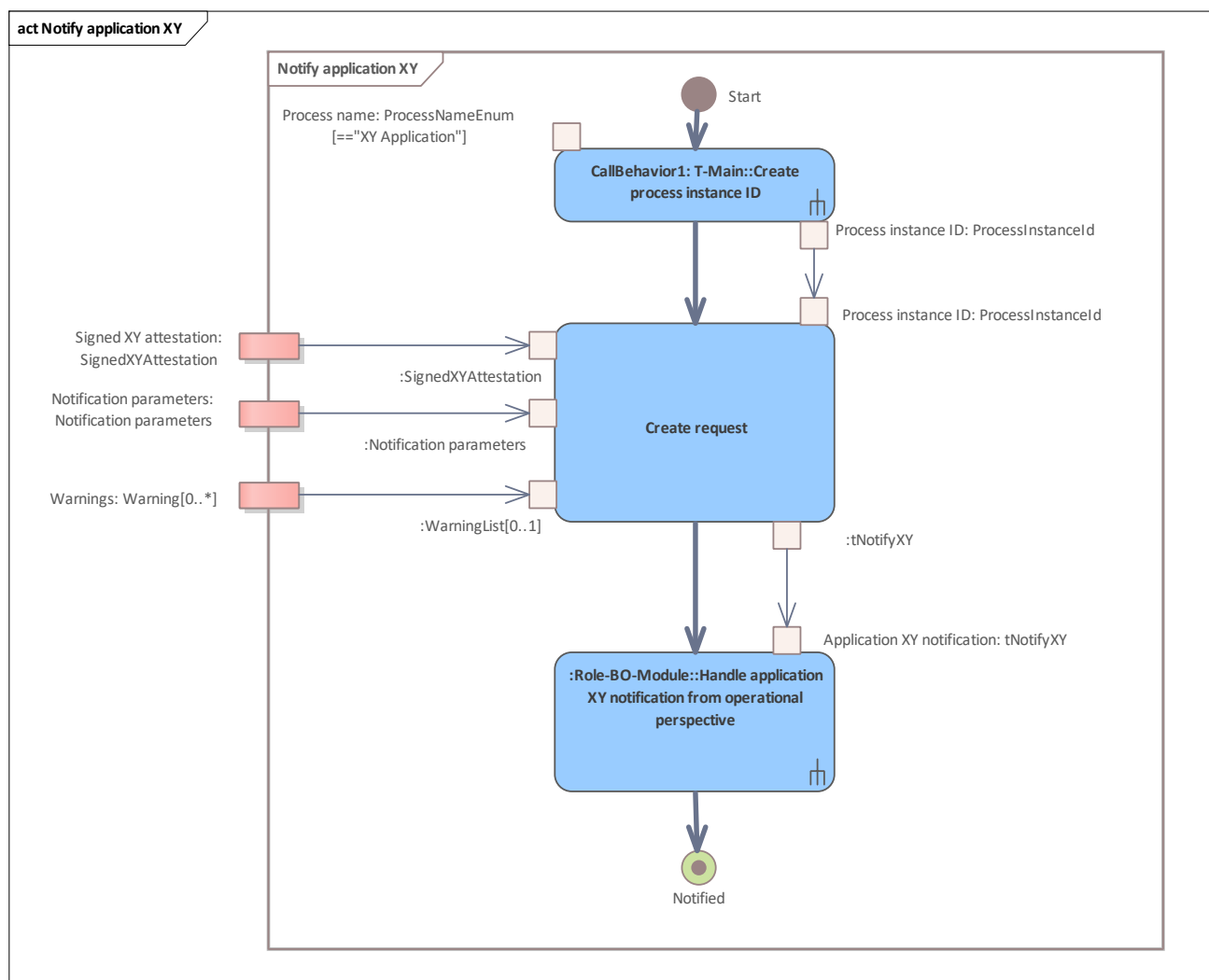


Figure 22: Notify application XY

3.4.4 Notify application XY aborted

The terminal notifies its back-office system about an aborted action.

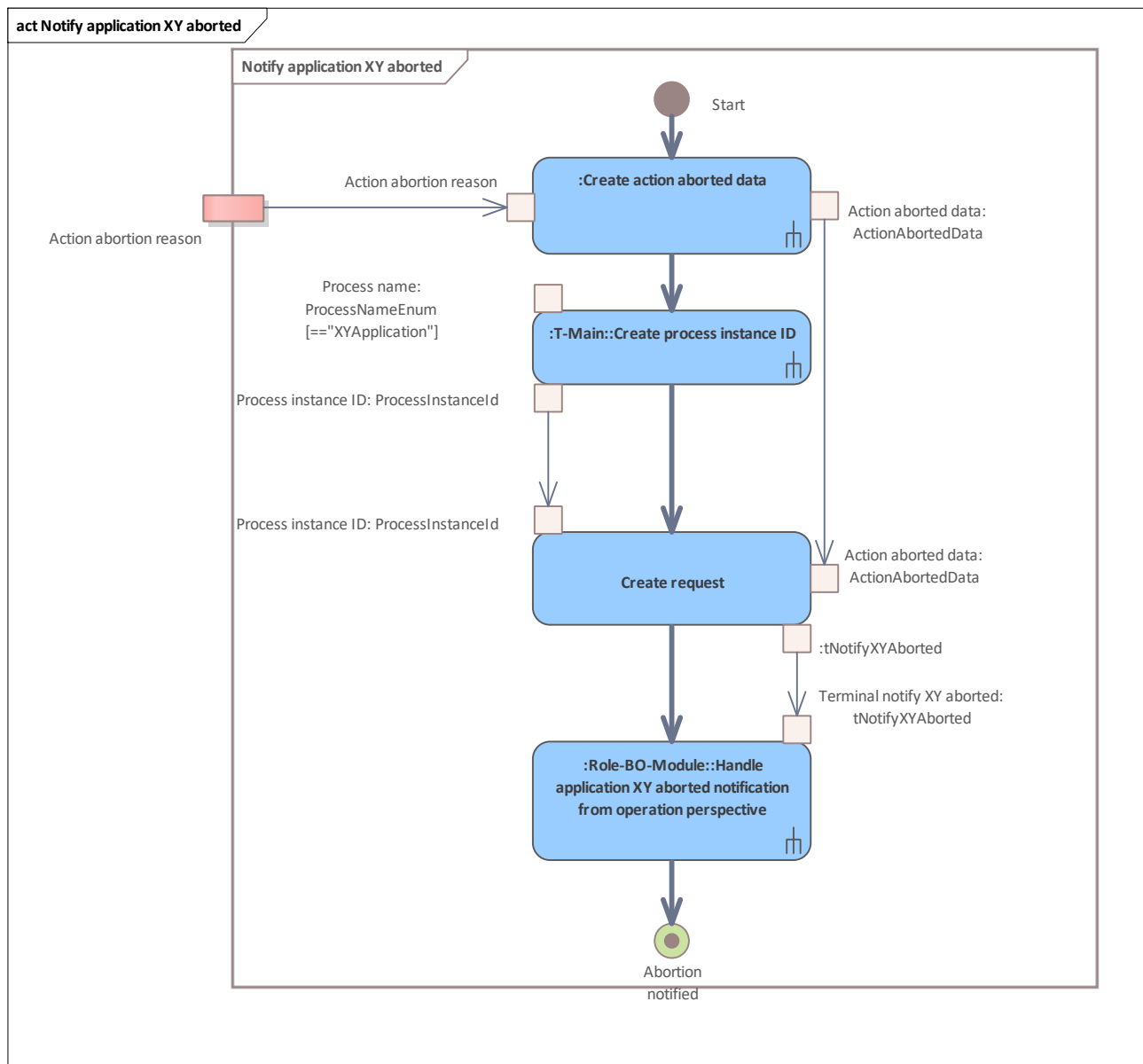


Figure 23: Notify application XY aborted

3.4.5 Notify XY (application owned)

Shows the message chain starting in a terminal (normally customer contract partner terminal), sent to a terminal operator back-office system (primary customer contract partner system).

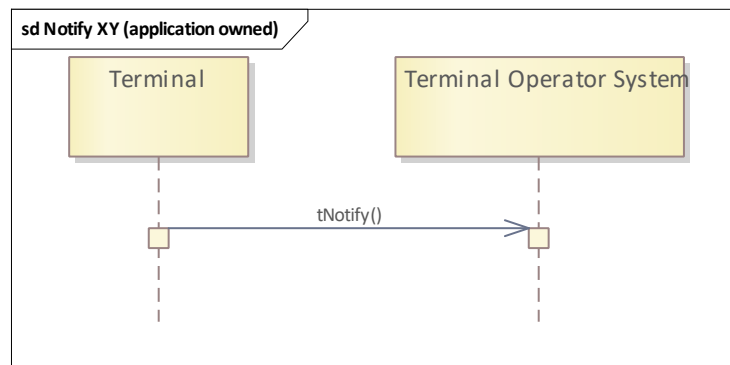


Figure 24: Notify XY (application owned)

See [Notify XY \(application owned\)](#).

3.4.6 Notify XY (application non-owned)

Shows the message chain starting in a terminal (customer contract partner terminal or service operator terminal), sent to a terminal operator back-office system (customer contract partner system or service operator system) and finally to the owning primary customer contract partner system.

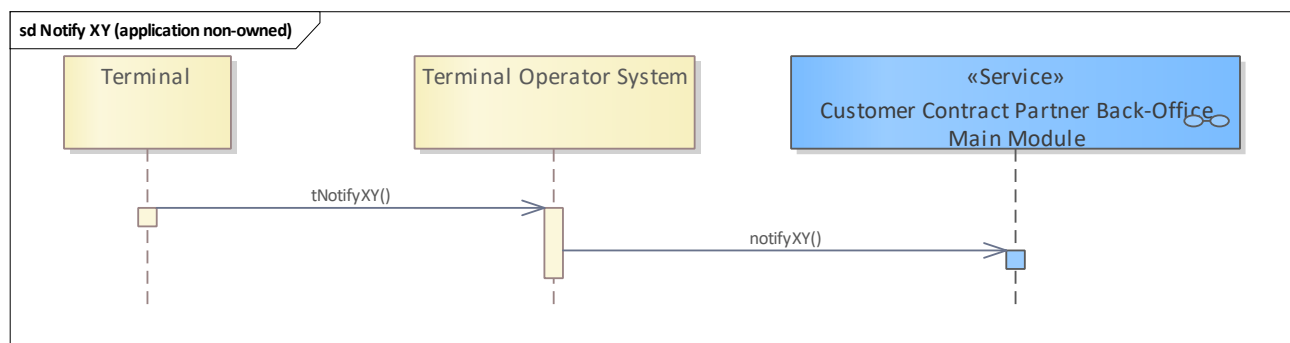


Figure 25: Notify XY (application non-owned)

See [Notify XY \(application non-owned\)](#).

3.5 Handle entitlement XY notification from operational perspective

See [Handle entitlement XY notification from operational perspective](#)

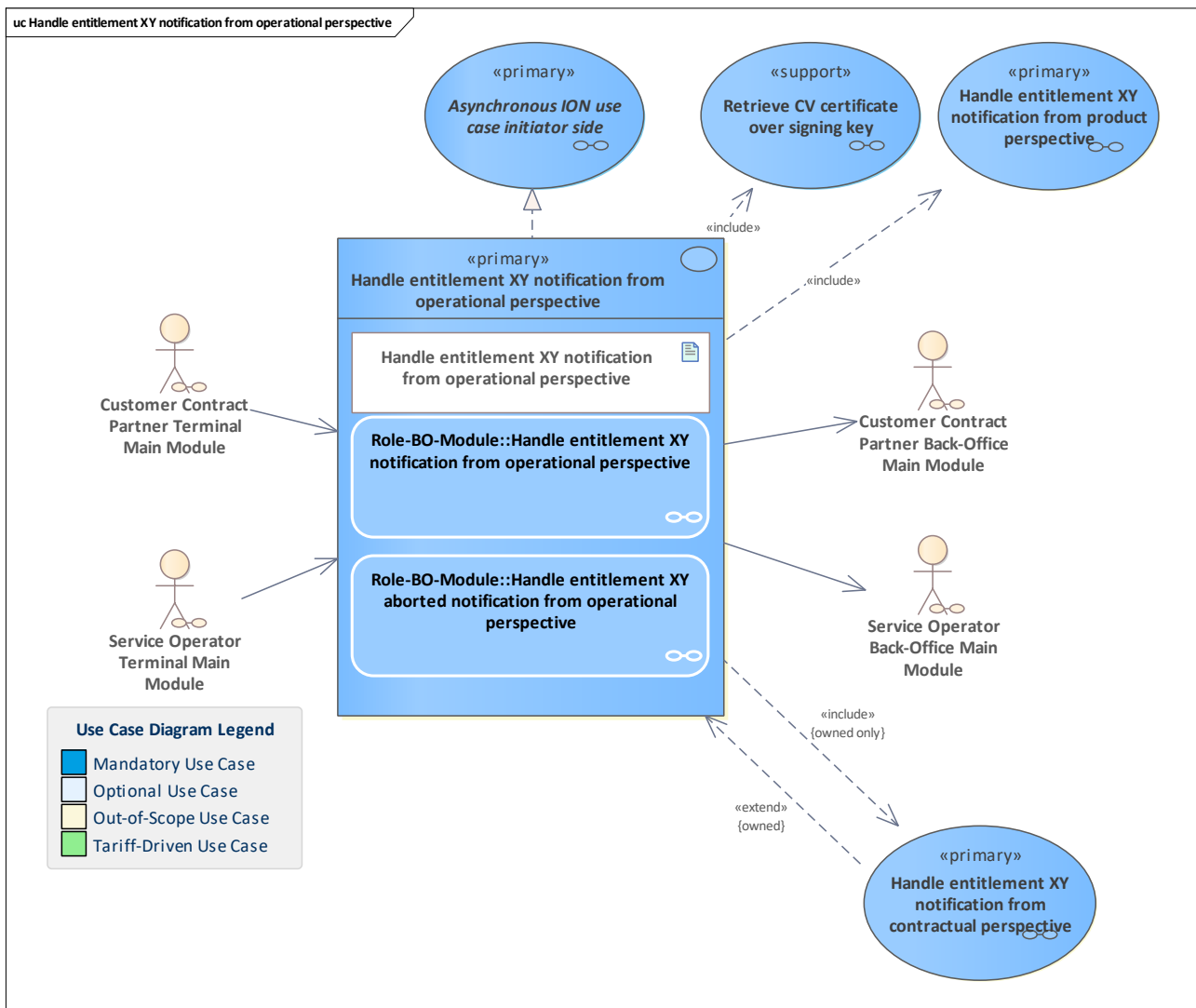


Figure 26: Handle entitlement XY notification from operational perspective

3.5.1 Handle entitlement XY notification from operational perspective

A back-office system belonging to a terminal that executed a UM action with an entitlement processes the notification about that action execution.

The processing is done from the operational perspective, focusing on the terminal-side of the action execution, i.e. logging the used SAM counter values.

Depending on the use case and the ownership of the entitlement the action was executed on, other back-office systems are informed about the action execution.

3.5.2 Role-BO-Module::Handle entitlement XY notification from operational perspective

See [Handle entitlement XY notification from operational perspective](#)

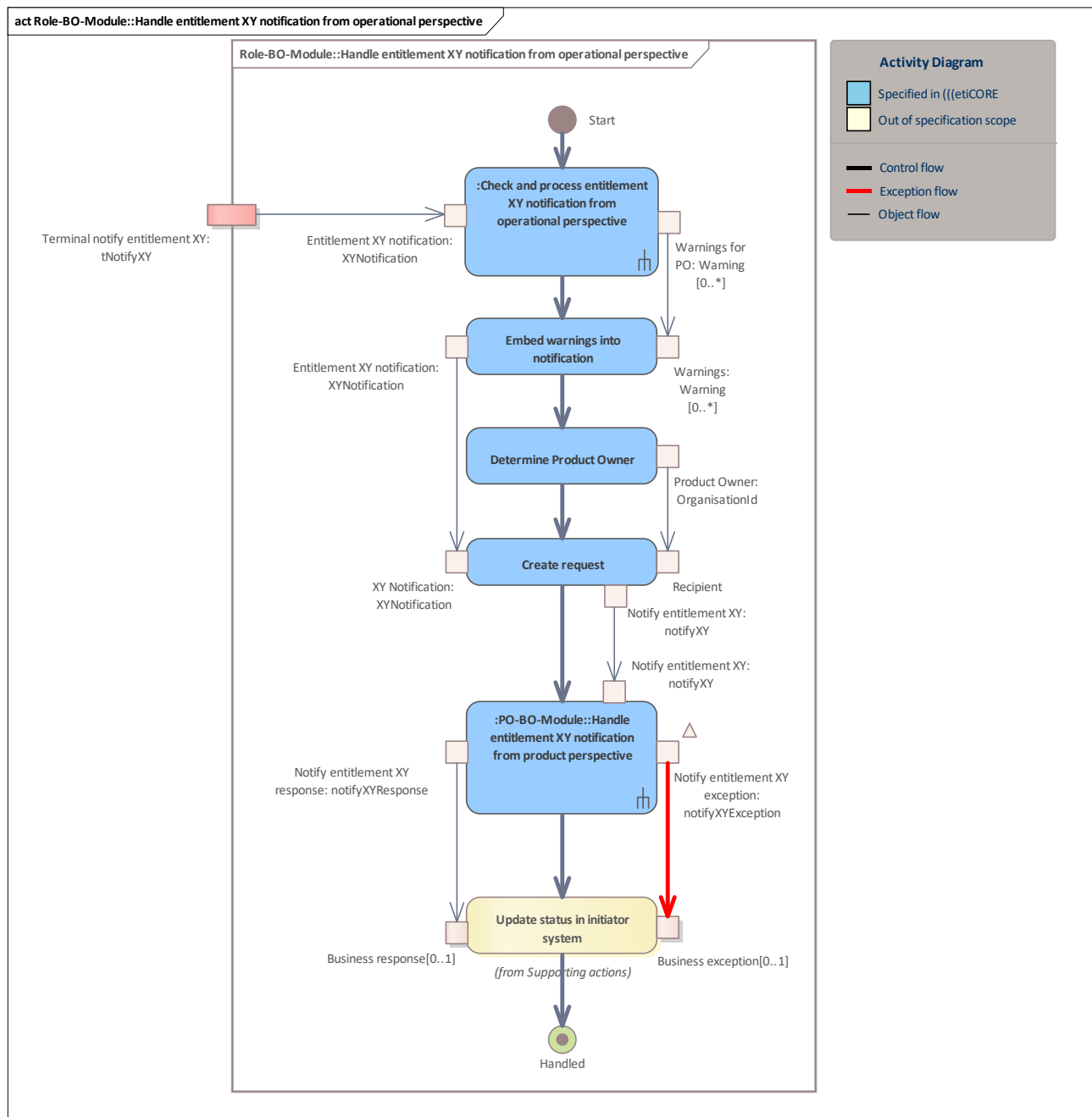


Figure 27: Role-BO-Module::Handle entitlement XY notification from operational perspective

3.5.3 Check and process entitlement XY notification from operational perspective

This activity performs the system internal processing of an entitlement-based notification coming from a terminal. All necessary monitoring checks are performed, divided into general checks and notification-specific checks.

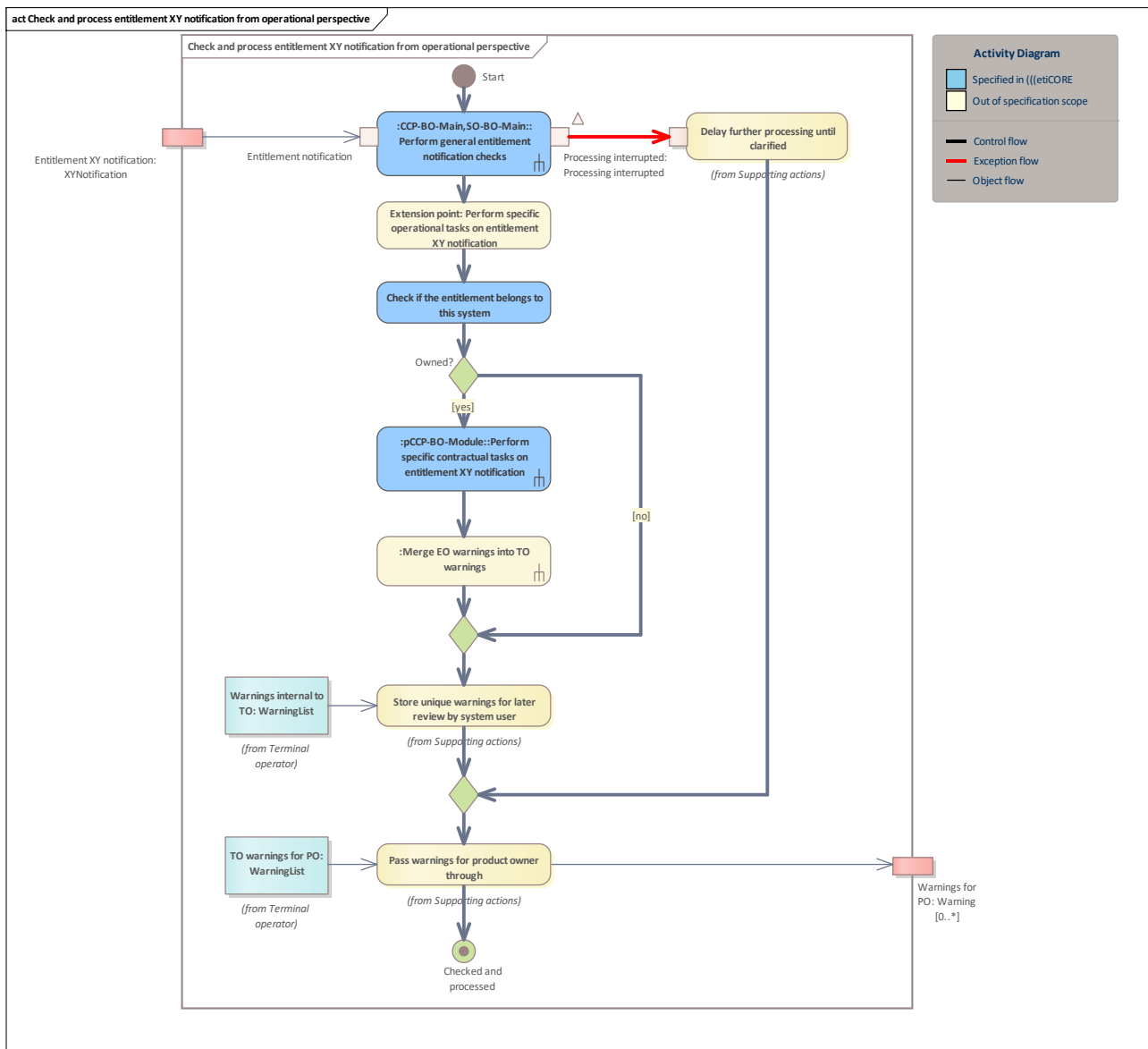


Figure 28: Check and process entitlement XY notification from operational perspective

3.5.3.1 Extension point: Perform specific operational tasks on entitlement XY notification

Log SAM action counter value / SAM entitlement issuance counter value / product issuance counter value usage, etc.

3.5.4 Role-BO-Module::Handle entitlement XY aborted notification from operational perspective

The back-office system belonging to the terminal handles a notification about an abortion during entitlement XY from an operational perspective.

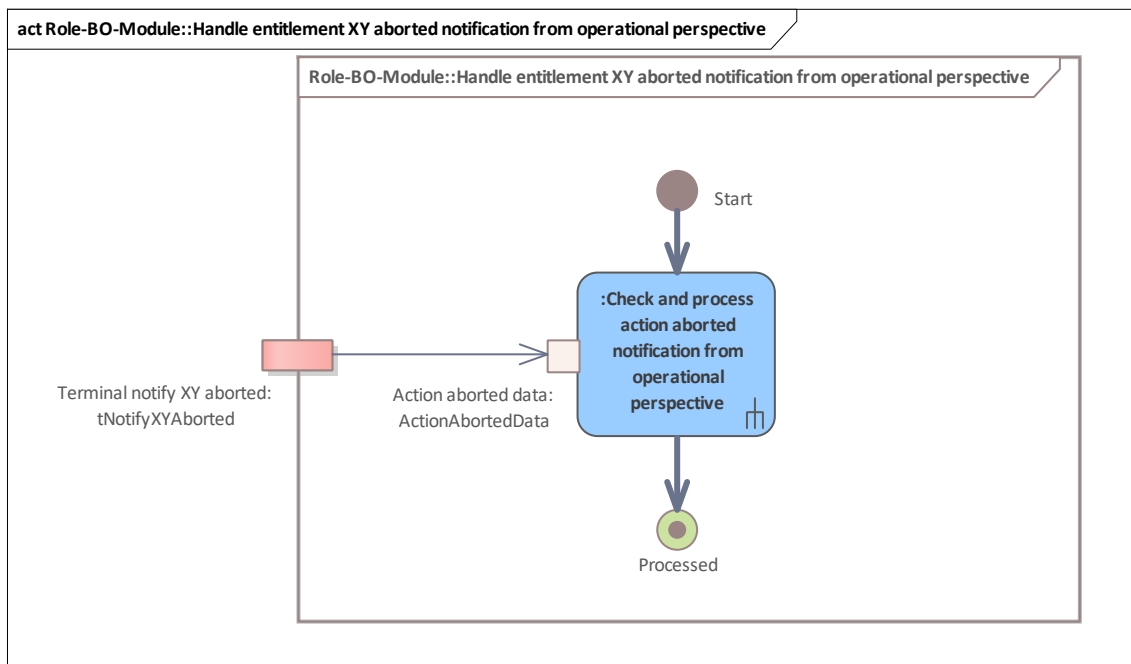


Figure 29: Role-BO-Module::Handle entitlement XY aborted notification from operational perspective

3.6 Handle application XY notification from operational perspective

See [Handle application XY notification from operational perspective](#)

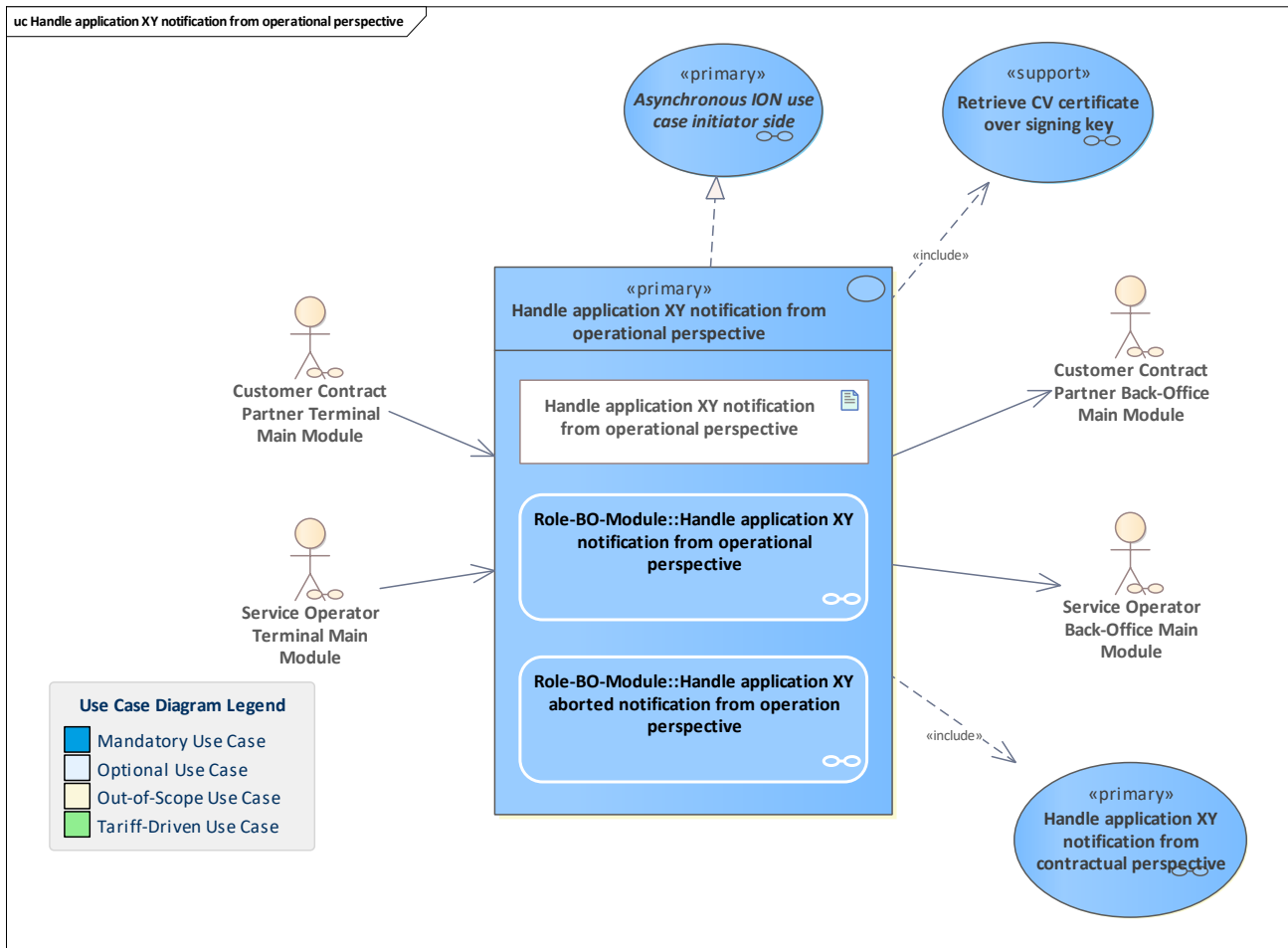


Figure 30: Handle application XY notification from operational perspective

3.6.1 Handle application XY notification from operational perspective

A back-office system belonging to a terminal that executed a UM action with an application processes the notification about that action execution.

The processing is done from the operational perspective, focusing on the terminal-side of the action execution, i.e. logging the used SAM counter values.

Depending on the use case and the ownership of the application the action was executed on, other back-office systems are informed about the action execution.

3.6.2 Role-BO-Module::Handle application XY notification from operational perspective

See [Handle application XY notification from operational perspective](#)

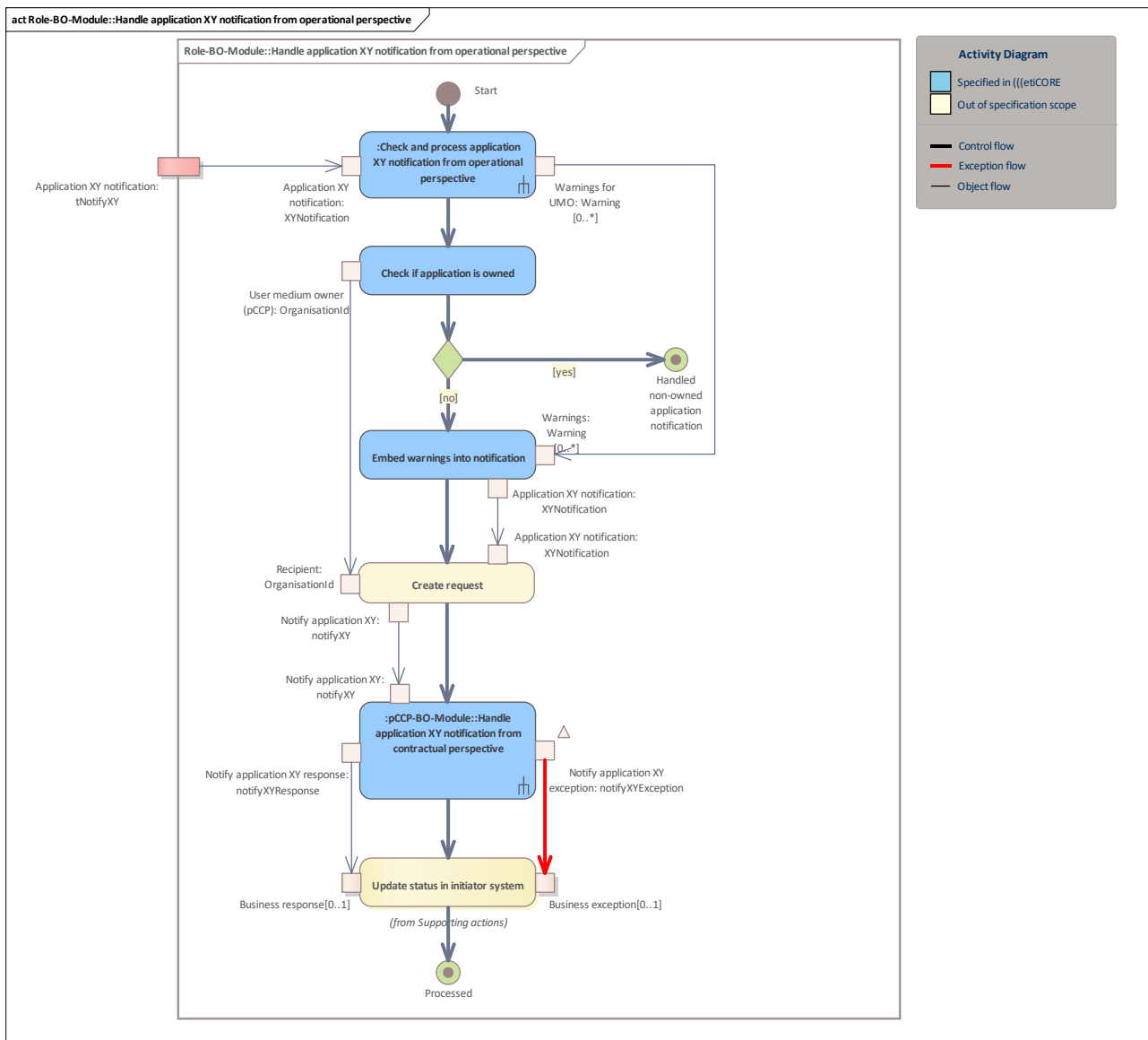


Figure 31: Role-BO-Module::Handle application XY notification from operational perspective

3.6.3 Check and process application XY notification from operational perspective

This activity performs the system internal processing of an application-based notification coming from a terminal. All necessary monitoring checks are performed, divided into general checks and notification specific checks.

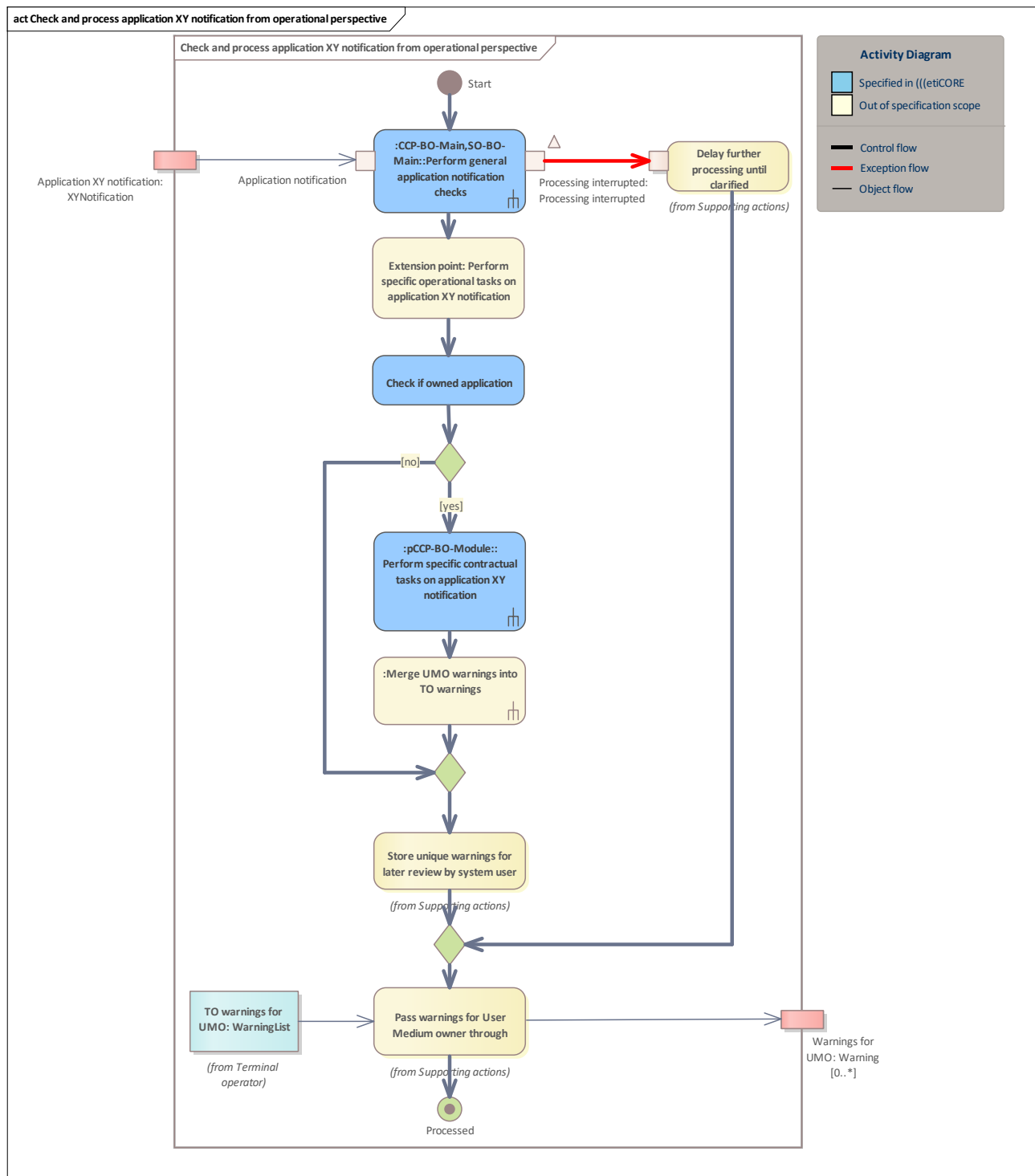


Figure 32: Check and process application XY notification from operational perspective

3.6.3.1 Extension point: Perform specific operational tasks on application XY notification

Log SAM action counter value, etc.

3.6.4 Role-BO-Module::Handle application XY aborted notification from operation perspective

The back-office system belonging to the terminal handles a notification about an abortion during application XY from an operational perspective.

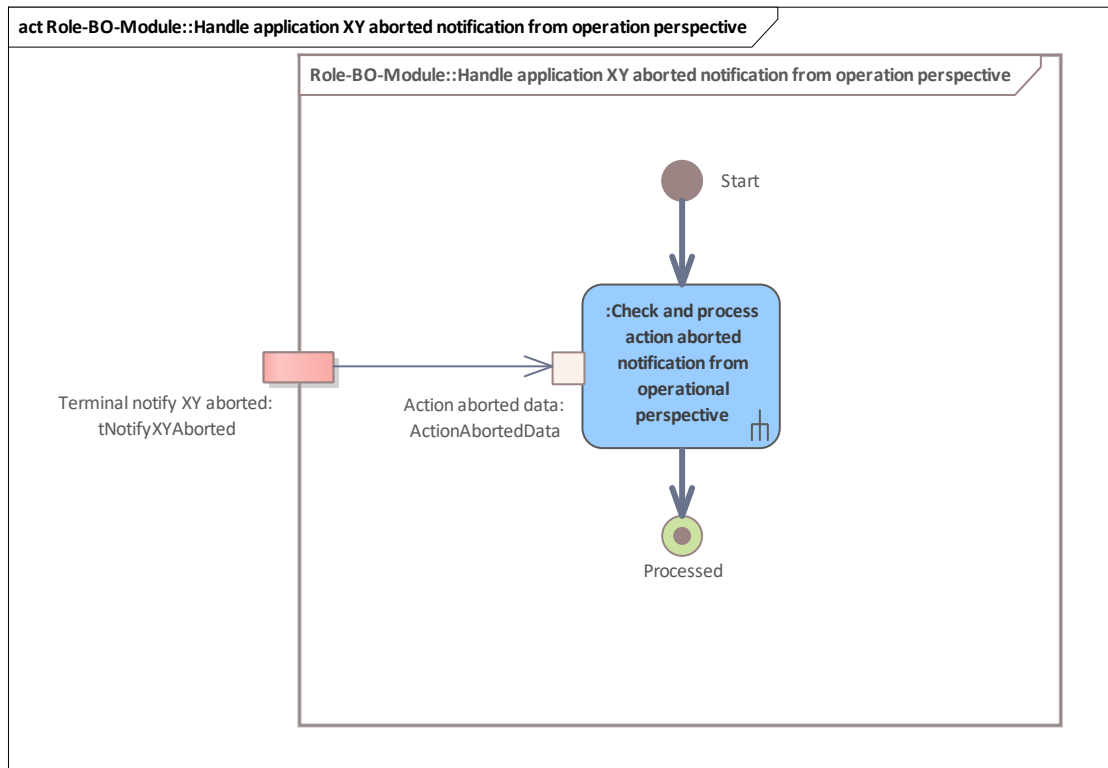


Figure 33: Role-BO-Module::Handle application XY aborted notification from operation perspective

3.7 Handle entitlement XY notification from product perspective

See [Handle entitlement XY notification from product perspective](#)

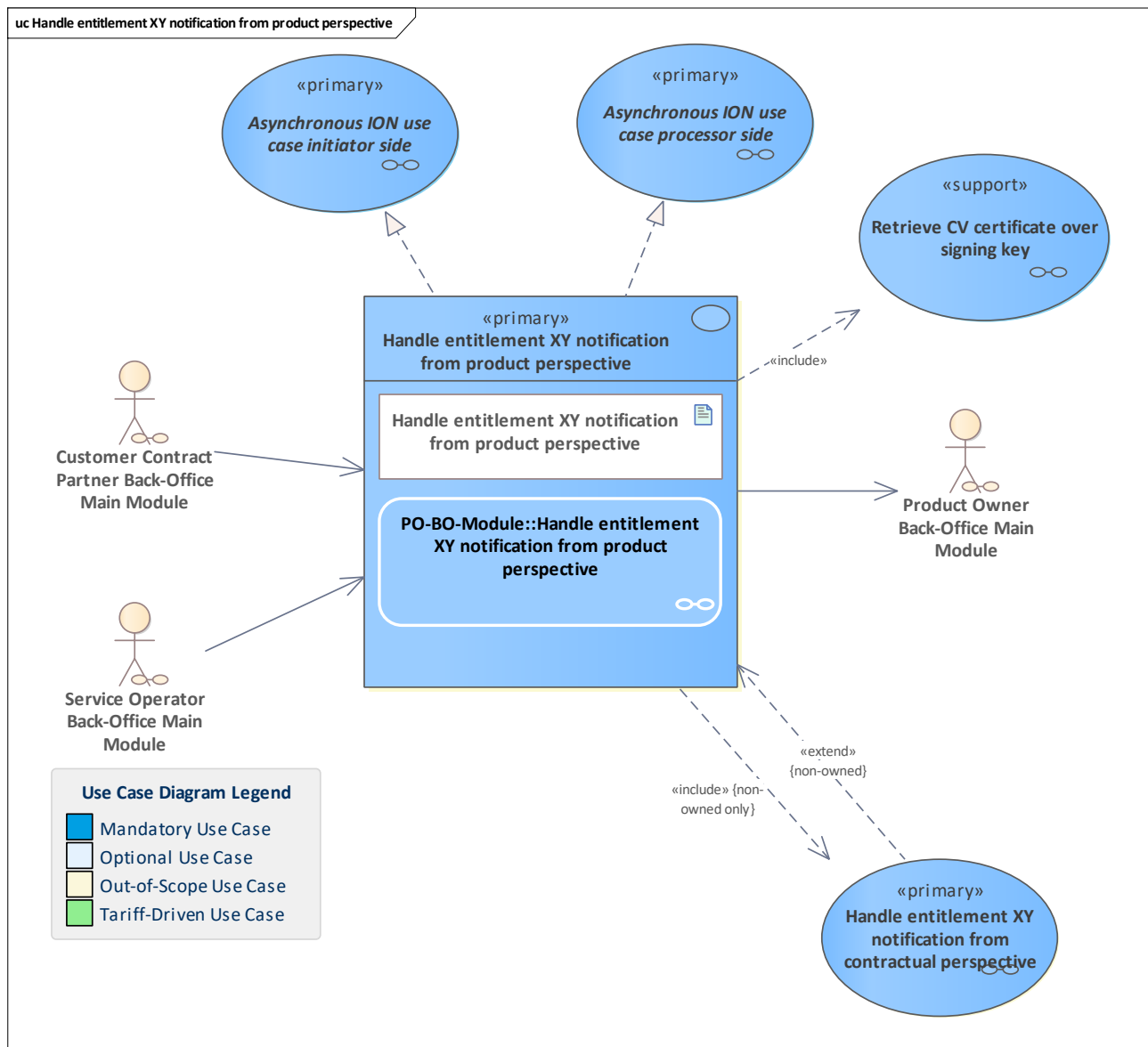


Figure 34: Handle entitlement XY notification from product perspective

3.7.1 Handle entitlement XY notification from product perspective

A PO system processes a notification about an entitlement action executed on an entitlement that is an instance of an owned product. The processing is done from the product perspective, focusing on the entitlement lifecycle and tariff aspects.

3.7.2 PO-BO-Module::Handle entitlement XY notification from product perspective

See [Handle entitlement XY notification from product perspective](#)

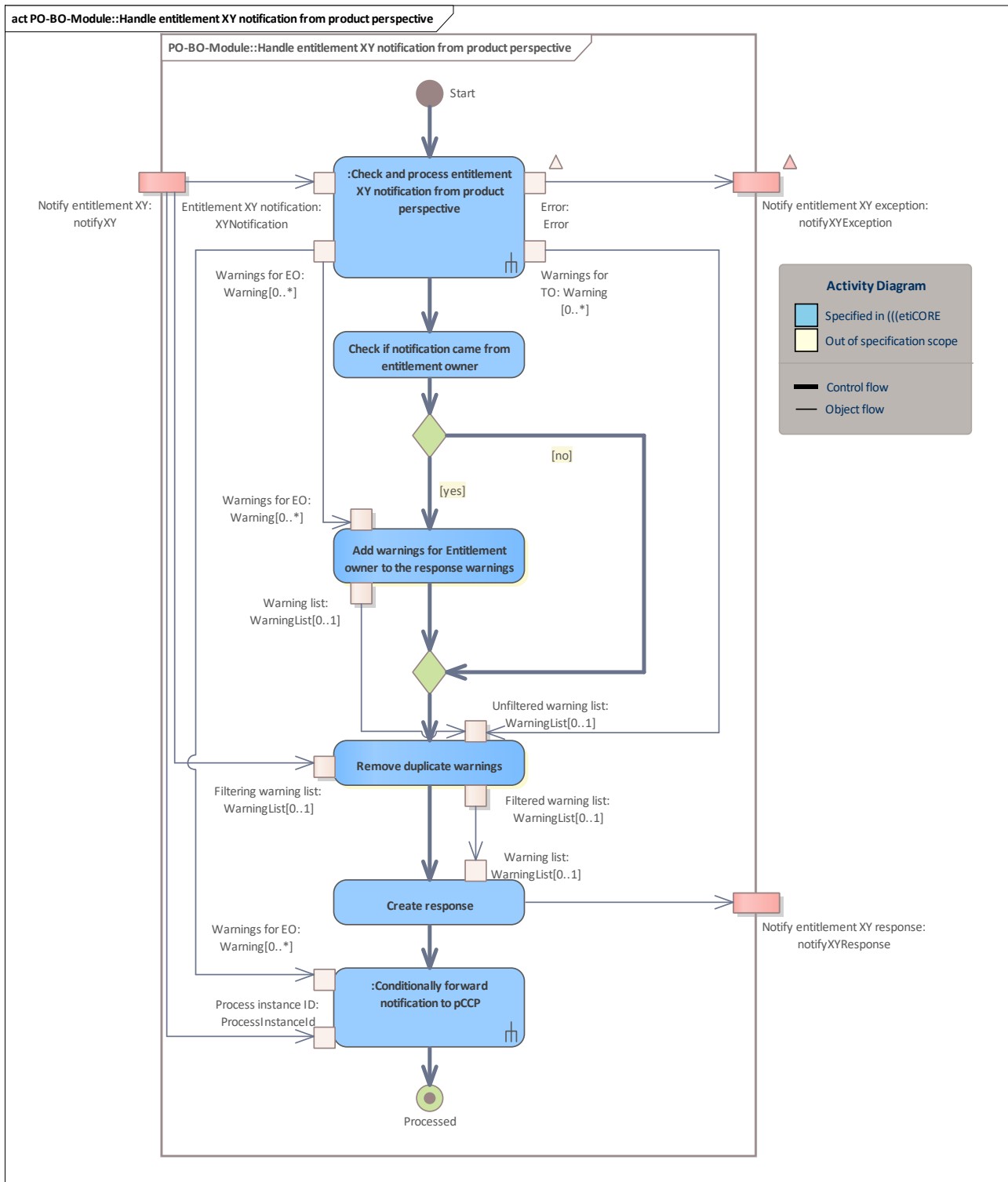


Figure 35: PO-BO-Module::Handle entitlement XY notification from product perspective

3.7.3 Check and process entitlement XY notification from product perspective

This activity performs the system internal processing of an entitlement-based notification coming from the terminal operator system (SO or CCP). All necessary monitoring checks are performed, divided into general checks and notification specific checks.

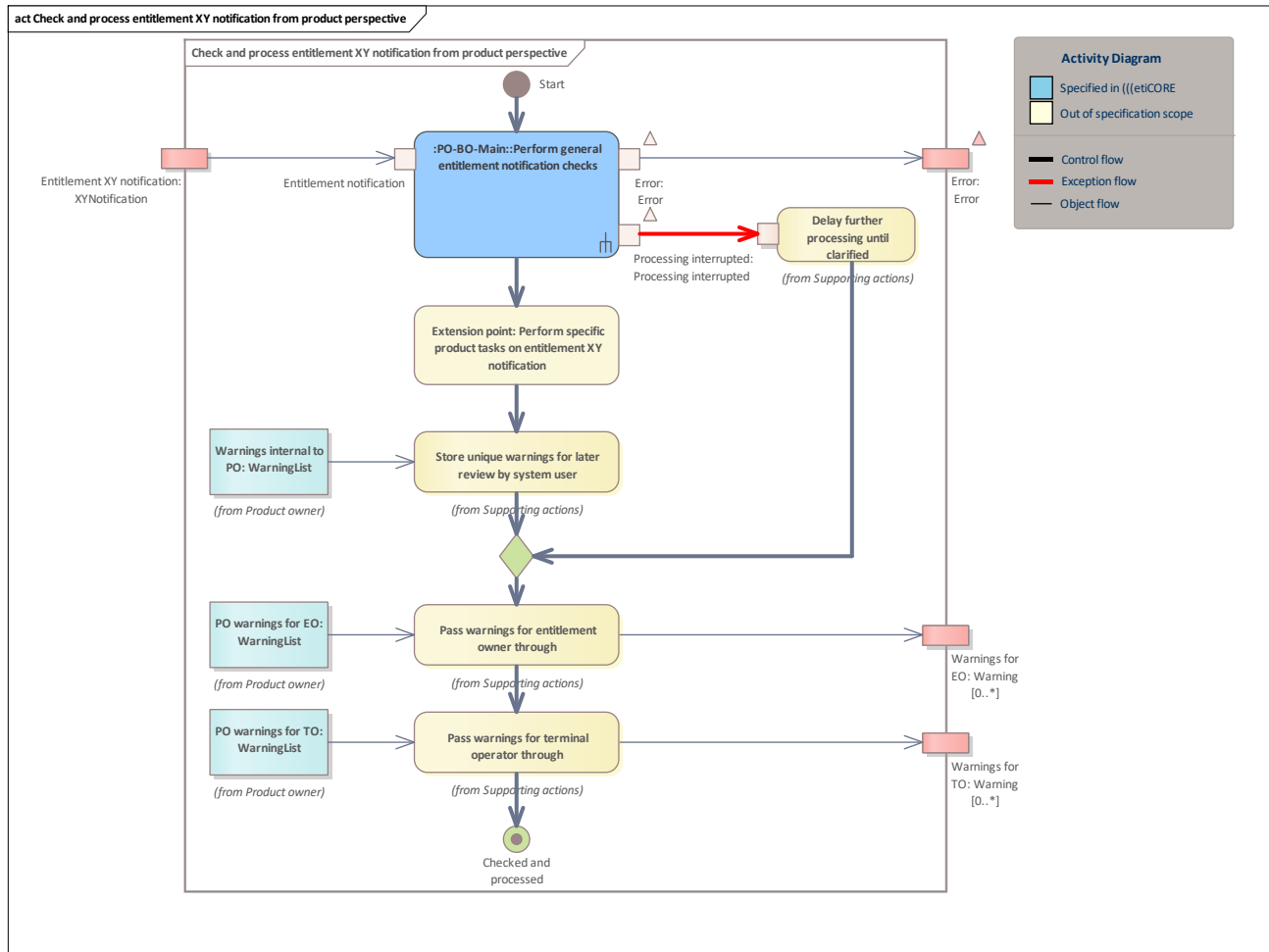


Figure 36: Check and process entitlement XY notification from product perspective

3.7.4 Conditionally forward notification to pCCP

Activity that is used if the sender of the notification was not owner of the entitlement. In this case, the notification is forwarded to the primary customer contract partner.

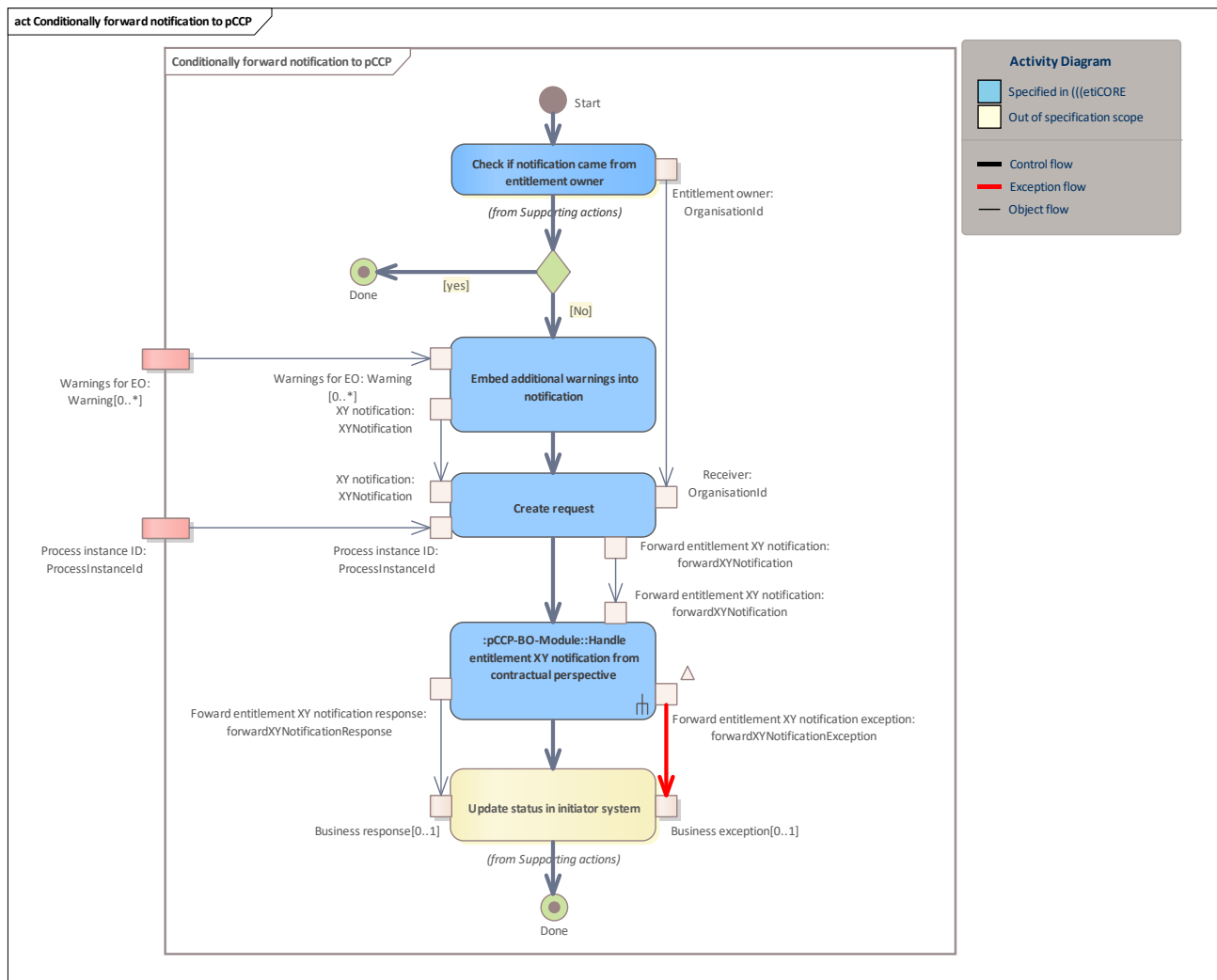


Figure 37: Conditionally forward notification to pCCP

3.7.4.1 Embed additional warnings into notification

Embed the given warnings into the notification currently being handled.

3.8 Handle entitlement XY notification from contractual perspective

See [Handle entitlement XY notification from contractual perspective](#)

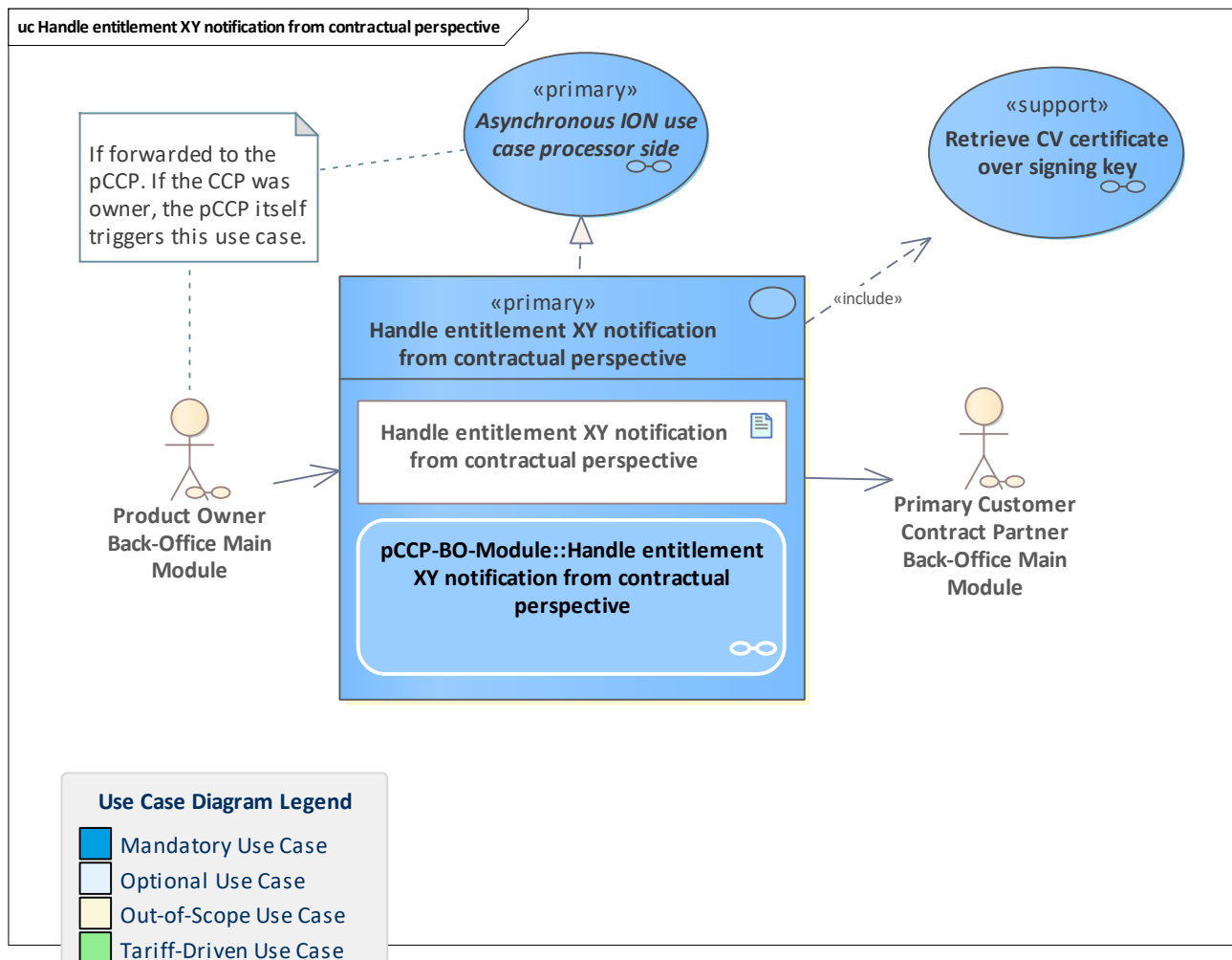


Figure 38: Handle entitlement XY notification from contractual perspective

3.8.1 Handle entitlement XY notification from contractual perspective

A CCP system processes a notification about an action executed on an owned entitlement. The processing is done from the contractual perspective focusing on the entitlement lifecycle and payment aspects.

This use case has two entry points:

- [pCCP-BO-Module::Handle entitlement XY notification from contractual perspective](#)
This entry point is used in the non-owned scenarios
- [pCCP-BO-Module::Perform specific contractual tasks on entitlement XY notification](#)
This entry point is used in the owned scenarios

The first includes the latter, in which most of the processing is done.

3.8.2 pCCP-BO-Module::Handle entitlement XY notification from contractual perspective

See [Handle entitlement XY notification from contractual perspective](#)

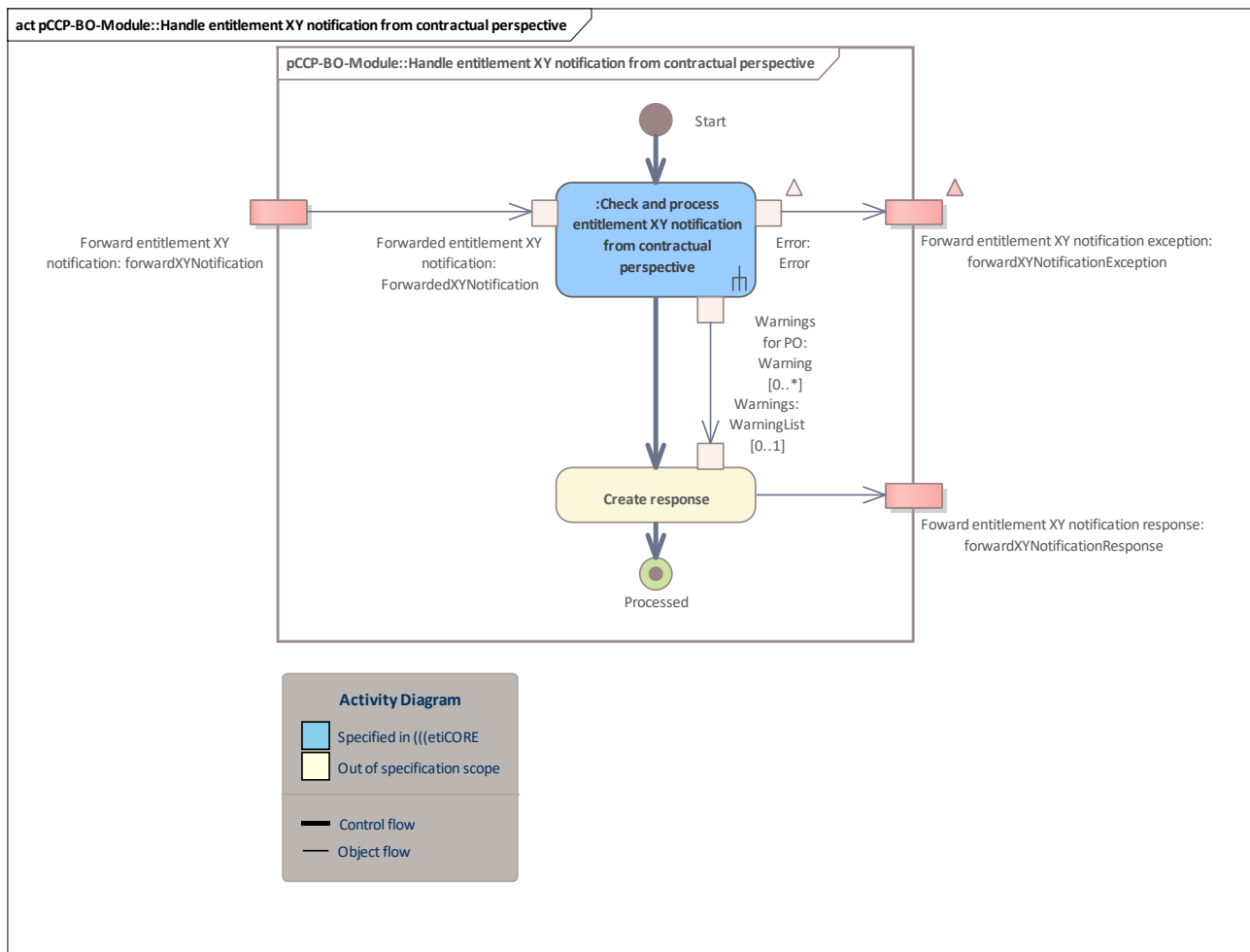


Figure 39: pCCP-BO-Module::Handle entitlement XY notification from contractual perspective

3.8.3 Check and process entitlement XY notification from contractual perspective

This activity performs the system internal processing of an entitlement-based notification either forwarded by the PO system or triggered directly after [Handle entitlement XY notification from operational perspective](#), if the current actor is the owner of the entitlement . All necessary monitoring checks are performed, divided into general checks and notification specific checks.

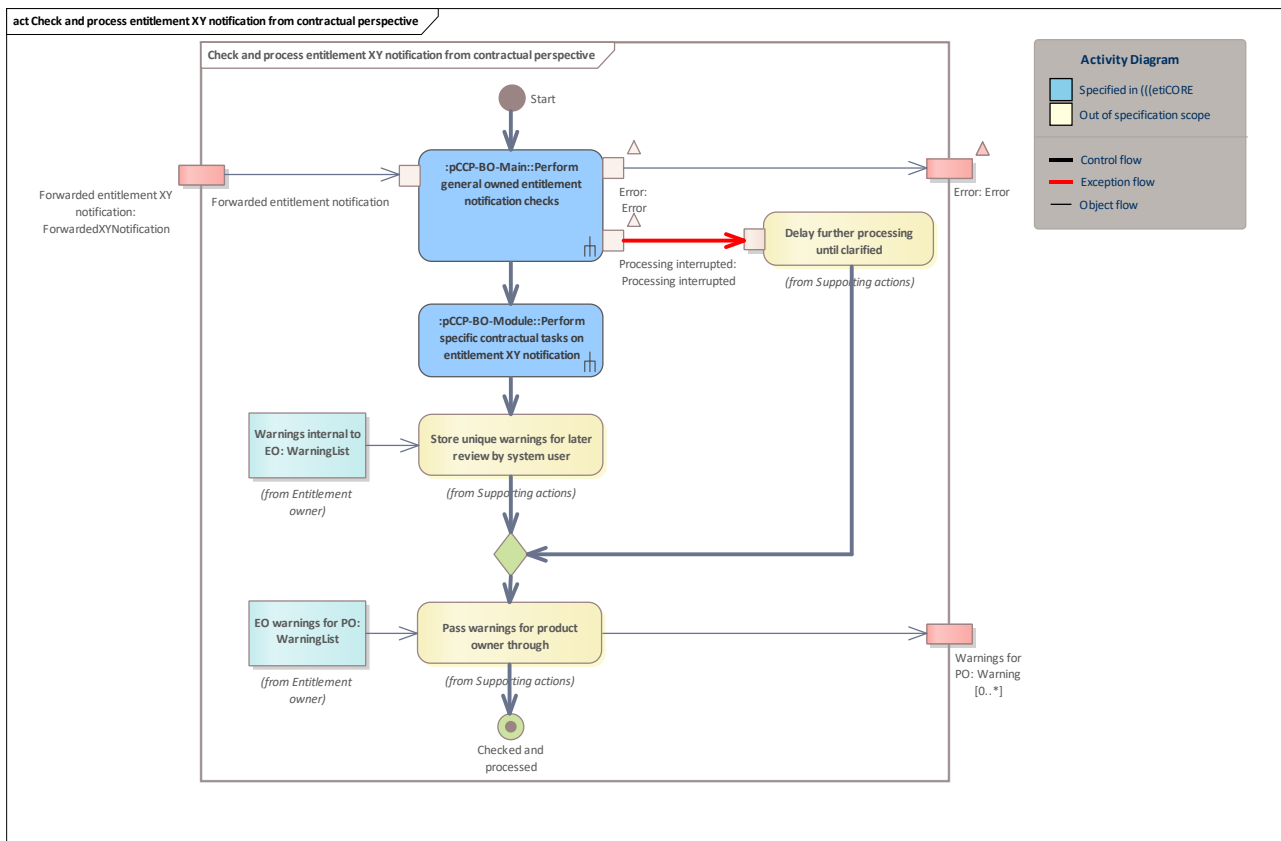


Figure 40: Check and process entitlement XY notification from contractual perspective

3.8.4 pCCP-BO-Module::Perform specific contractual tasks on entitlement XY notification

See [Handle entitlement XY notification from contractual perspective](#)

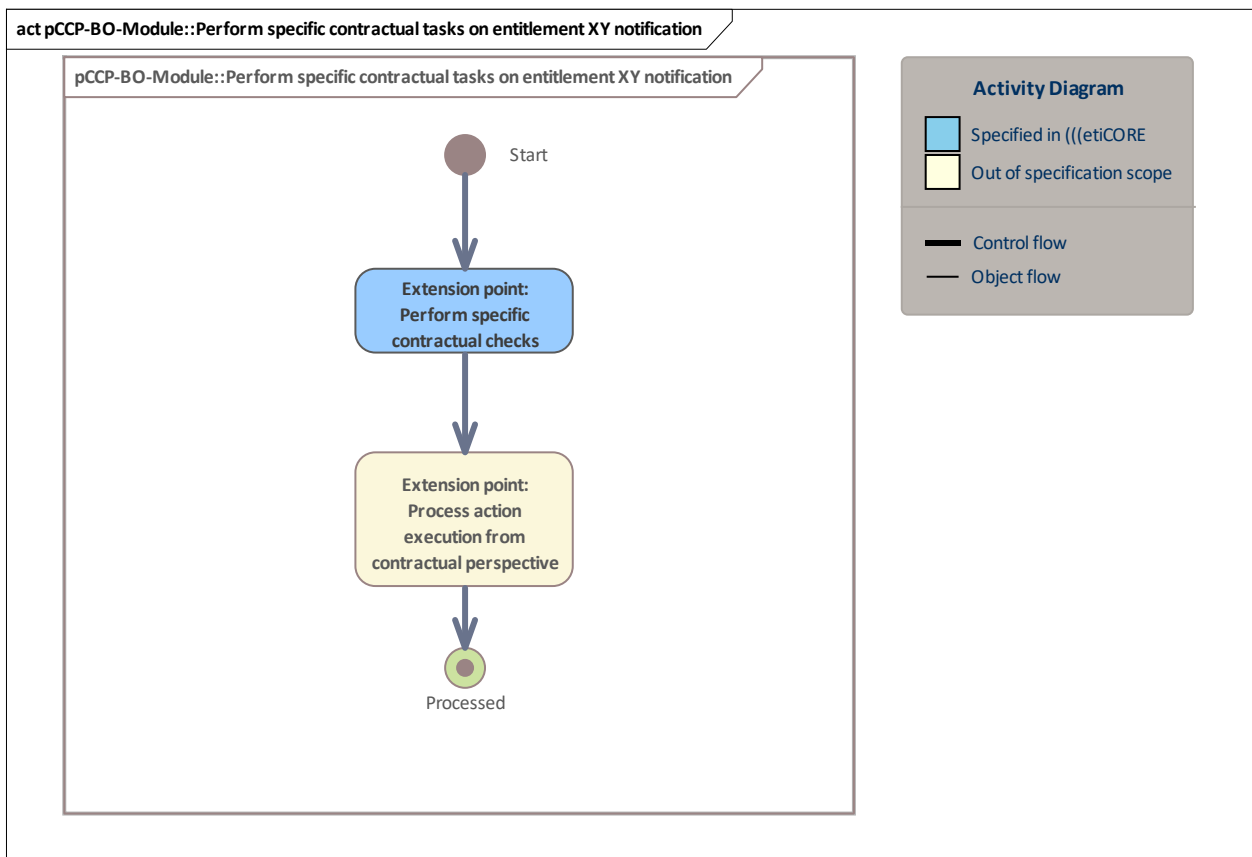


Figure 41: pCCP-BO-Module::Perform specific contractual tasks on entitlement XY notification

3.8.4.1 Extension point: Process action execution from contractual perspective

Update sales register, update entitlement state, etc.

3.9 Handle application XY notification from contractual perspective

See [Handle application XY notification from contractual perspective](#)

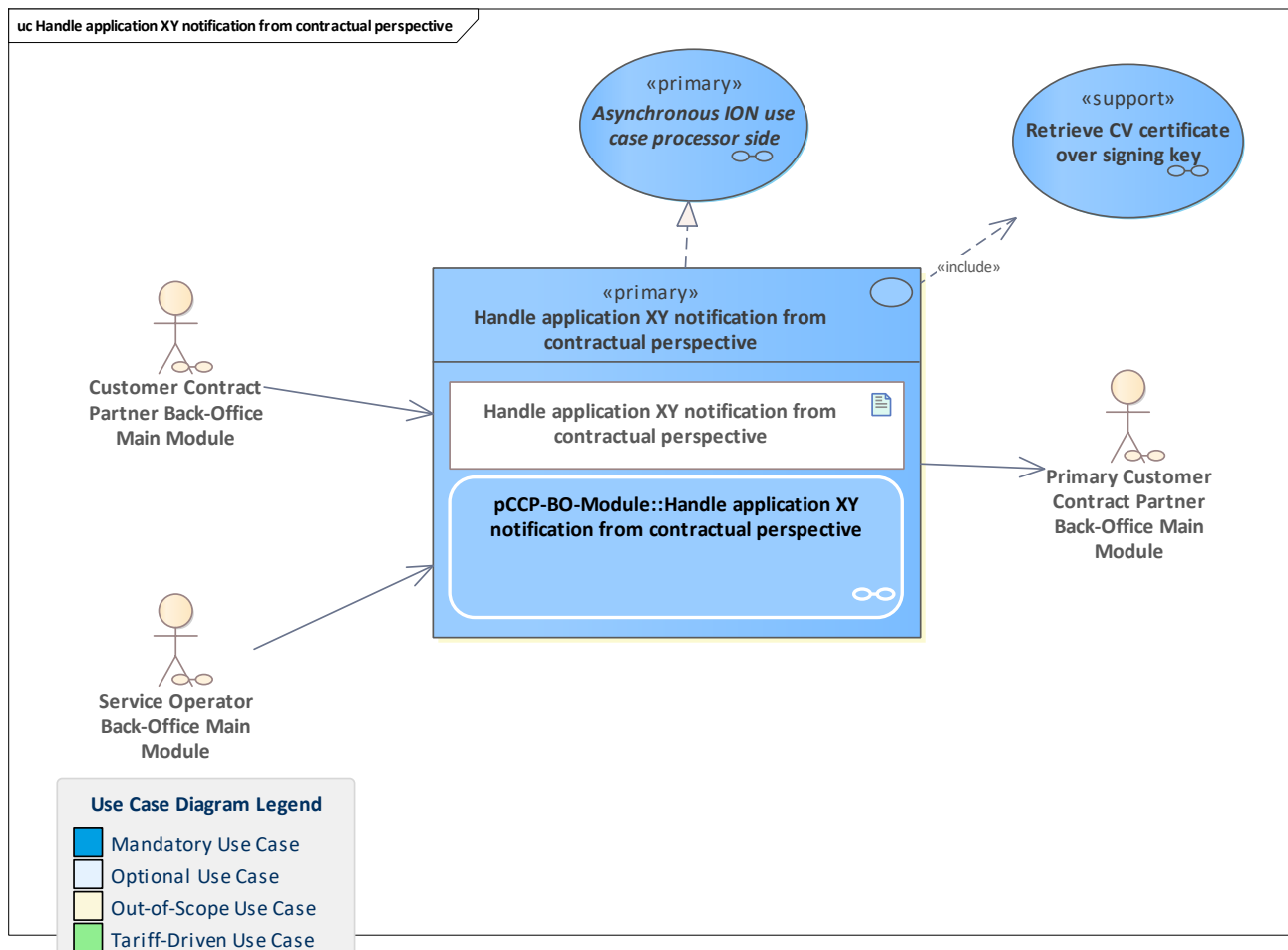


Figure 42: Handle application XY notification from contractual perspective

3.9.1 Handle application XY notification from contractual perspective

A CCP system processes a notification about an action executed on an owned application. The processing is done from the contractual perspective focusing on the application lifecycle aspects.

This use case has two entry points:

- [pCCP-BO-Module::Handle application XY notification from contractual perspective](#)
This entry point is used in the non-owned scenarios
- [pCCP-BO-Module::Perform specific contractual tasks on application XY notification](#)
This entry point is used in the owned scenarios

The first includes the latter, in which most of the processing is done.

3.9.2 pCCP-BO-Module::Handle application XY notification from contractual perspective

See [Handle application XY notification from contractual perspective](#)

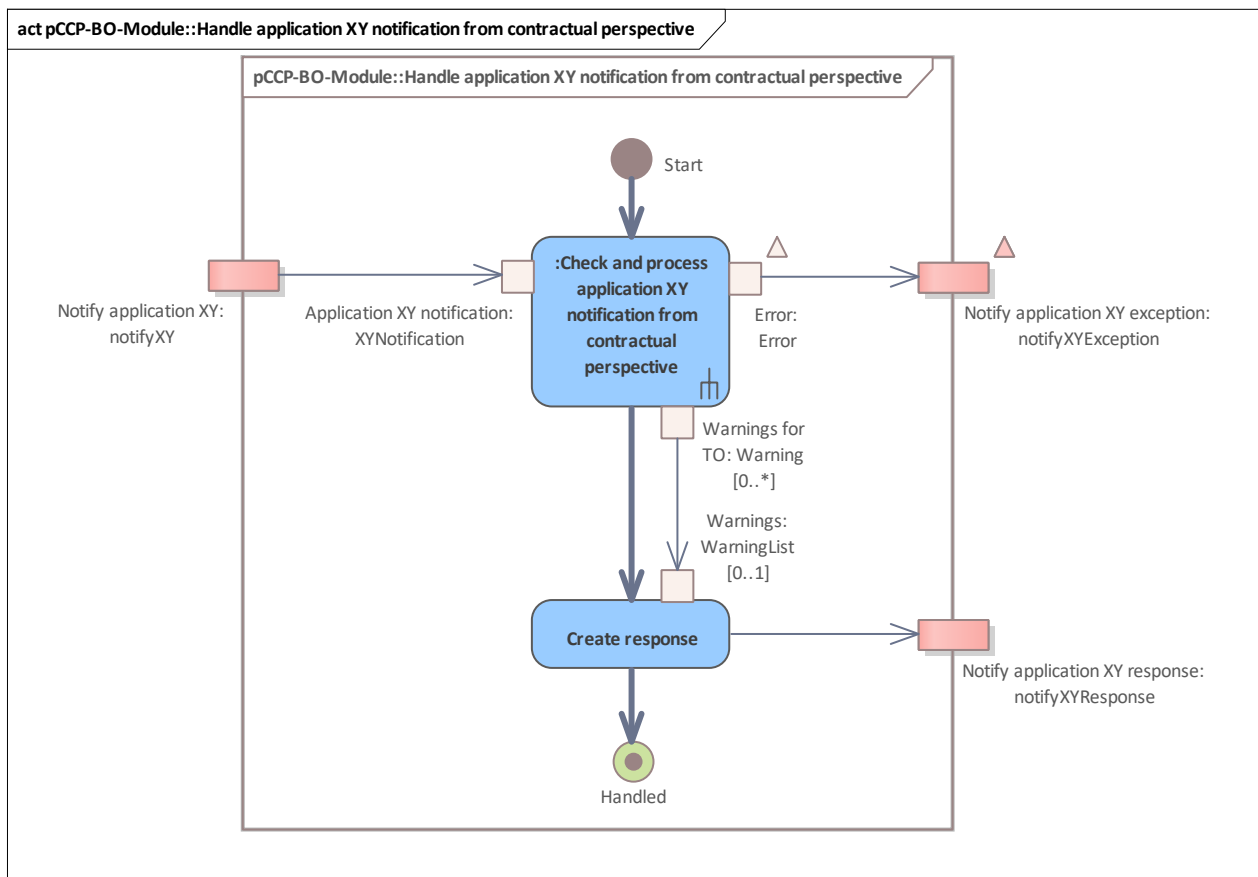


Figure 43: pCCP-BO-Module::Handle application XY notification from contractual perspective

3.9.3 Check and process application XY notification from contractual perspective

This activity performs the system internal processing of an application based notification either sent from another terminal operator system (SO or sCCP) or triggered directly after the executed use case [Handle application XY notification from operational perspective](#), if the current actor is the owner of the application instance. All necessary monitoring checks are performed, divided into general checks and notification specific checks.

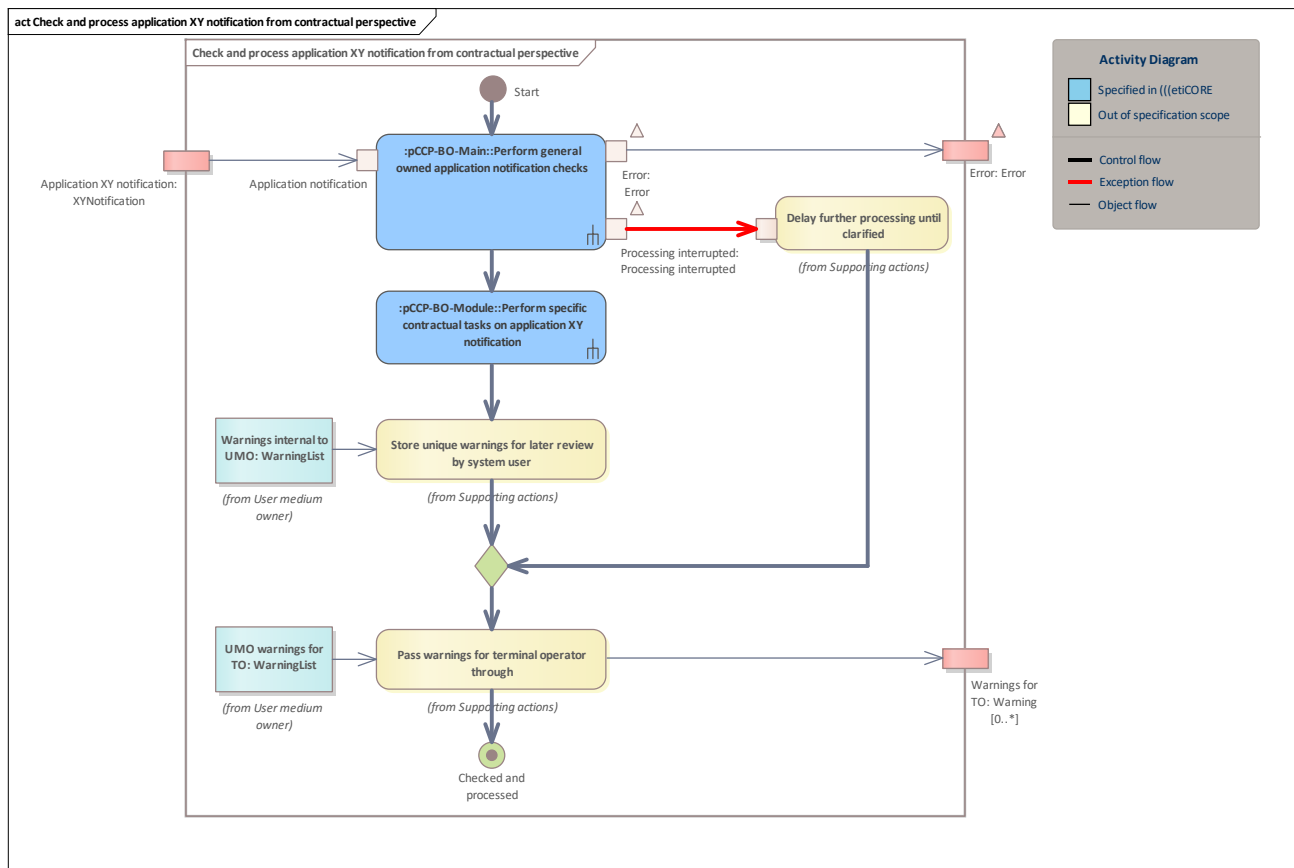


Figure 44: Check and process application XY notification from contractual perspective

3.9.4 pCCP-BO-Module::Perform specific contractual tasks on application XY notification

See [Handle application XY notification from contractual perspective](#)

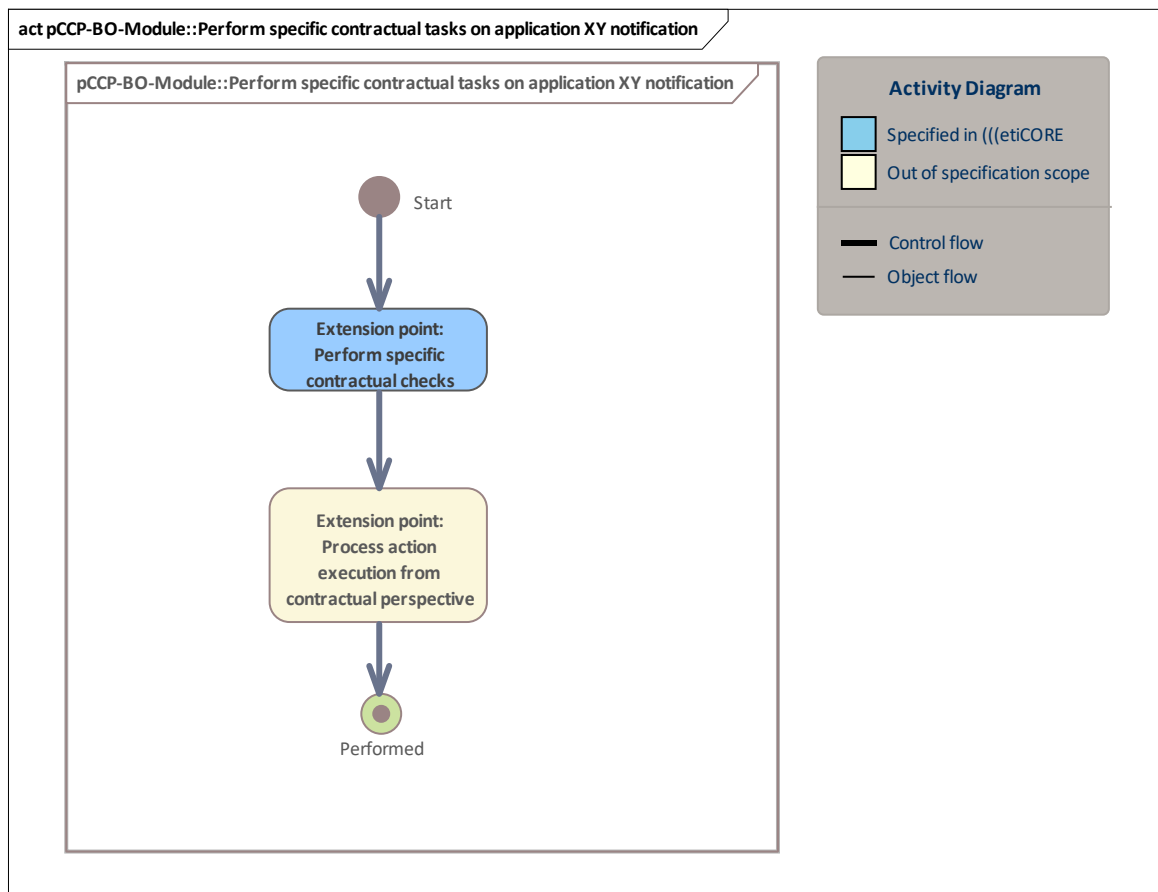


Figure 45: pCCP-BO-Module::Perform specific contractual tasks on application XY notification

4 Verifying Lists Updated via Increments

Incremental lists are used for large lists containing thousands of entries. These lists occur in the hotlist service for lists of hotlisted application instances and entitlements, as well as in the action list service when fetching the newest action list.

Instead of fetching a full list, an incremental list can be used. This list only contains the differences between a specified last list and the current one.

These differences have to be merged into the current inventory. To verify that the updated inventory is equal to the inventory of a full list, a certain checksum algorithm is used.

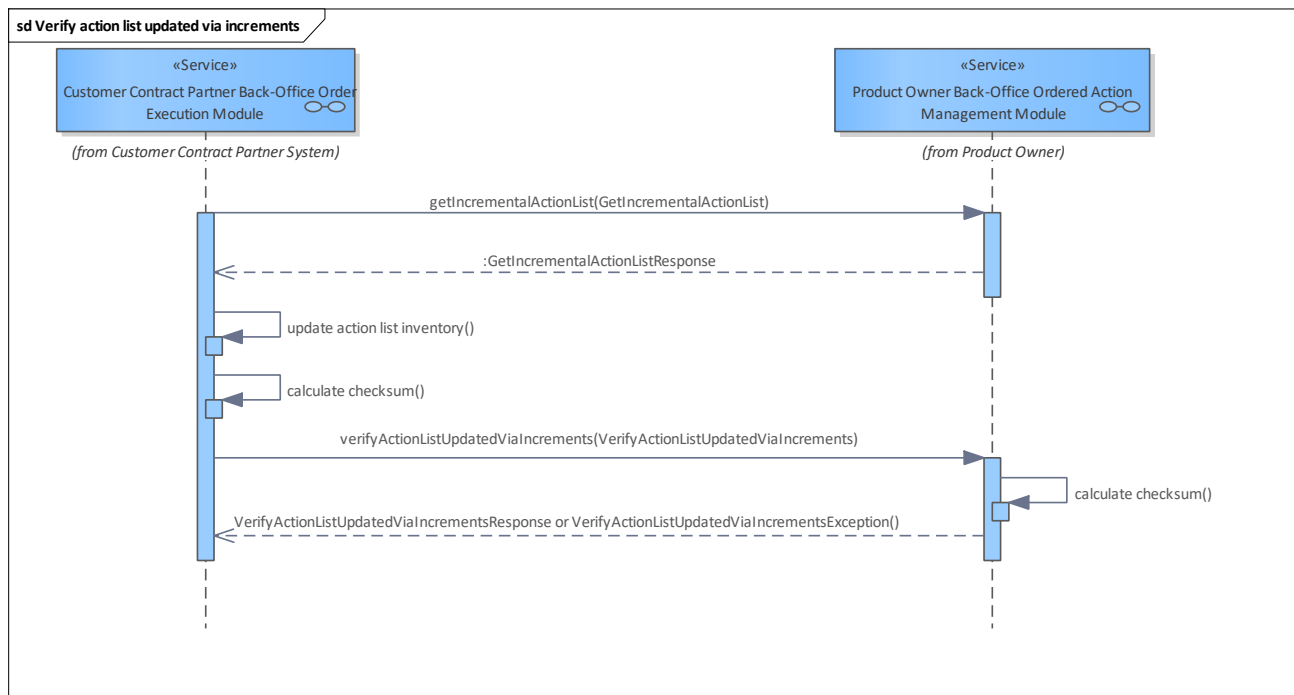


Figure 46: Verify action list updated via increments

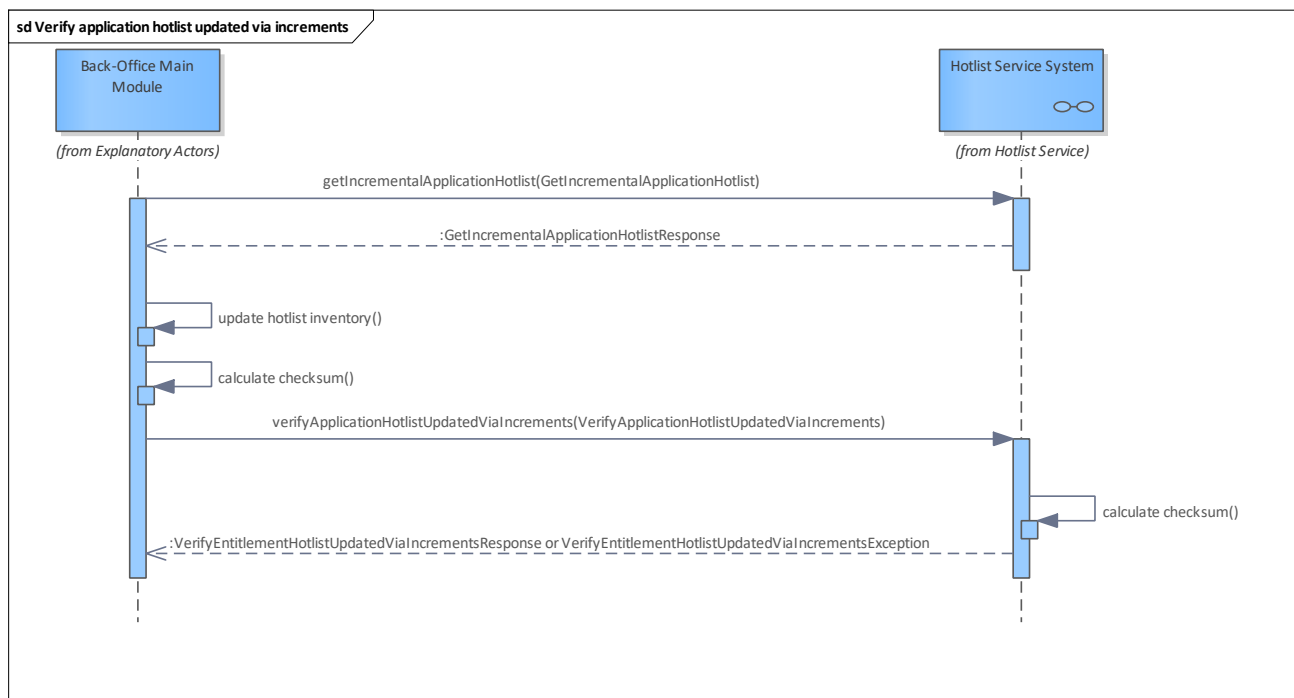


Figure 47: Verify application hotlist updated via increments

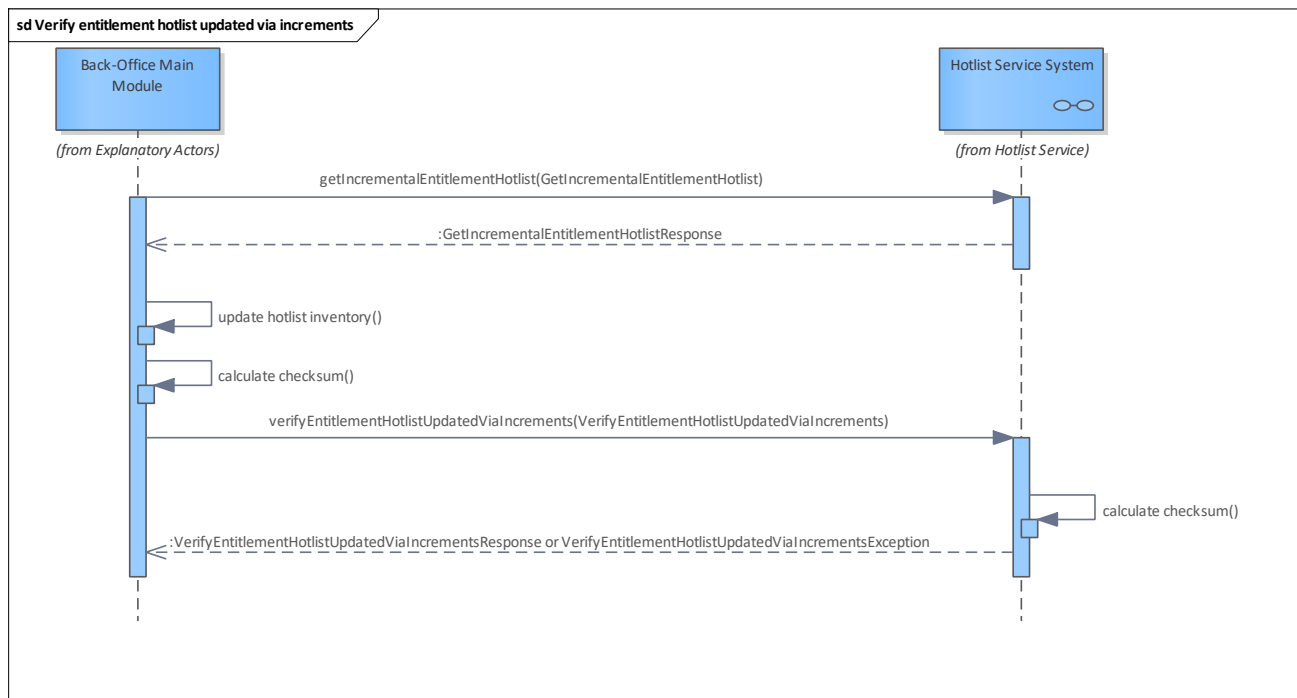


Figure 48: Verify entitlement hotlist updated via increments

4.1 Checksum calculation for hotlist and action list verification

The actual calculation of the checksum differs per list and is given in the corresponding use cases. Here, we describe some aspects that apply to all of them.

The data objects are encoded according to the distinguished encoding rules (DER) per the corresponding ASN.1 schemata including tag and length.

The hash algorithm to use is SHA-256.

We always sort lexicographically in ascending order.

Pseudo code to calculate the hash value:

```
hash( concatenate( sort( [ serialise(entry) for entry in entries ] ) ) )
```

4.2 Example calculation for an action list inventory

Action list inventory (reduced to the parts relevant to hashing) to verify:

```
[
  {
    "entitlementUnblockingActionListEntry": {
      "actionListEntryBaseInformation": {
        "orderId": {
          "orderNumber": 108,
          "orderingCcp": 123
        }
      }
    }
  },
  {
    "entitlementUnblockingActionListEntry": {
      "actionListEntryBaseInformation": {
        "orderId": {
          "orderNumber": 109,
          "orderingCcp": 124
        }
      }
    }
  }
]
```

```
{
  "entitlementIssuanceActionListEntry": {
    "actionListEntryBaseInformation": {
      "orderId": {
        "orderNumber": 109,
        "orderingCcp": 123
      }
    }
  },
  "entitlementTerminationActionListEntry": {
    "actionListEntryBaseInformation": {
      "orderId": {
        "orderNumber": 111,
        "orderingCcp": 123
      }
    }
  },
  "entitlementBlockingActionListEntry": {
    "actionListEntryBaseInformation": {
      "orderId": {
        "orderNumber": 110,
        "orderingCcp": 123
      }
    }
  }
}
```

Intermediate result after serialisation:

```
[0x4D017B5A016C,
0x4D017B5A016D,
0x4D017B5A016F,
0x4D017B5A016E]
```

Intermediate result after sorting:

```
[0x4D017B5A016C,
0x4D017B5A016D,
0x4D017B5A016E,
0x4D017B5A016F]
```

Final result after applying SHA-256:

```
0x1376FF1481CB9B73DDADDC5F4588478A506EE4129AA345B2D408E37DF1FB51E8
```

4.3 Example calculation for an application hotlist inventory

Application hotlist inventory to verify:

```
[
  {
    "applicationInstanceId": {
      "registrationAuthorityId": 10004,
```

```
        "subjectRole": "0x1F",
        "subjectNumber": "0x00000004"
    },
    "applicationTransitionCounter": 0,
    "applicationBlockingMode": 10
},
{
    "applicationInstanceId": {
        "registrationAuthorityId": 10004,
        "subjectRole": "0x1B",
        "subjectNumber": "0x00000002"
    },
    "applicationTransitionCounter": 6,
    "applicationBlockingMode": 0
},
{
    "applicationInstanceId": {
        "registrationAuthorityId": 10004,
        "subjectRole": "0x1B",
        "subjectNumber": "0x00000001"
    },
    "applicationTransitionCounter": 0,
    "applicationBlockingMode": 0
}
]
```

Intermediate result after serialisation:

```
[0x640D4D02271480011F82040000000458010002010A,
0x640D4D02271480011B820400000002580106020100,
0x640C4D02271480011B8203000001580100020100]
```

Intermediate result after sorting:

```
[0x640C4D02271480011B8203000001580100020100,
0x640D4D02271480011B820400000002580106020100,
0x640D4D02271480011F82040000000458010002010A]
```

Final result after applying SHA-256:

```
0x98152207B15DE600F0A98974CA510DDB6A3034E661BFE4287BF808128C61C6E2
```

4.4 Example calculation for an entitlement hotlist inventory

Entitlement hotlist inventory to verify:

```
[
    {
        "entitlementBlockingMode": 0,
        "entitlementId": {
            "ccpOrgId": 348,
            "entitlementIssuanceCounter": 1005,
            "samId": {
                "registrationAuthorityId": 2713,
                "subjectNumber": "0x00000002",
                "subjectRole": "0x13"
            }
        }
    },
    {
        "entitlementBlockingMode": 0,
        "entitlementId": {
            "ccpOrgId": 348,
            "entitlementIssuanceCounter": 1005,
            "samId": {
                "registrationAuthorityId": 2713,
                "subjectNumber": "0x00000001",
                "subjectRole": "0x13"
            }
        }
    }
]
```

```
"entitlementTransitionCounter": 0
},
{
  "entitlementBlockingMode": 20,
  "entitlementId": {
    "ccpOrgId": 5730,
    "entitlementIssuanceCounter": 120,
    "samId": {
      "registrationAuthorityId": 2713,
      "subjectNumber": "0x00000002",
      "subjectRole": "0x13"
    }
  }
},
"entitlementTransitionCounter": 0
},
{
  "entitlementBlockingMode": 0,
  "entitlementId": {
    "ccpOrgId": 36,
    "entitlementIssuanceCounter": 48905,
    "samId": {
      "registrationAuthorityId": 2713,
      "subjectNumber": "0x00000002",
      "subjectRole": "0x13"
    }
  }
},
"entitlementTransitionCounter": 6
}
]
```

Intermediate result after serialisation:

```
[0x6E174D02015C640D4D020A998001138204000000024B0203ED580100020100,
0x6E164D021662640D4D020A998001138204000000024B0178580100020114,
0x6E174D0124640D4D020A998001138204000000024B0300BF09580106020100]
```

Intermediate result after sorting:

```
[0x6E164D021662640D4D020A998001138204000000024B0178580100020114,
0x6E174D0124640D4D020A998001138204000000024B0300BF09580106020100,
0x6E174D02015C640D4D020A998001138204000000024B0203ED580100020100]
```

Final result after applying SHA-256:

```
0xC58276BE0B189DEAB6350B85588F80EF016153D01002B14F192B85B1FAA38124
```

5 Basic Bundle Back-Office System

This functionality bundle contains use cases that all back-office systems of the CCP, SO and PO must implement.

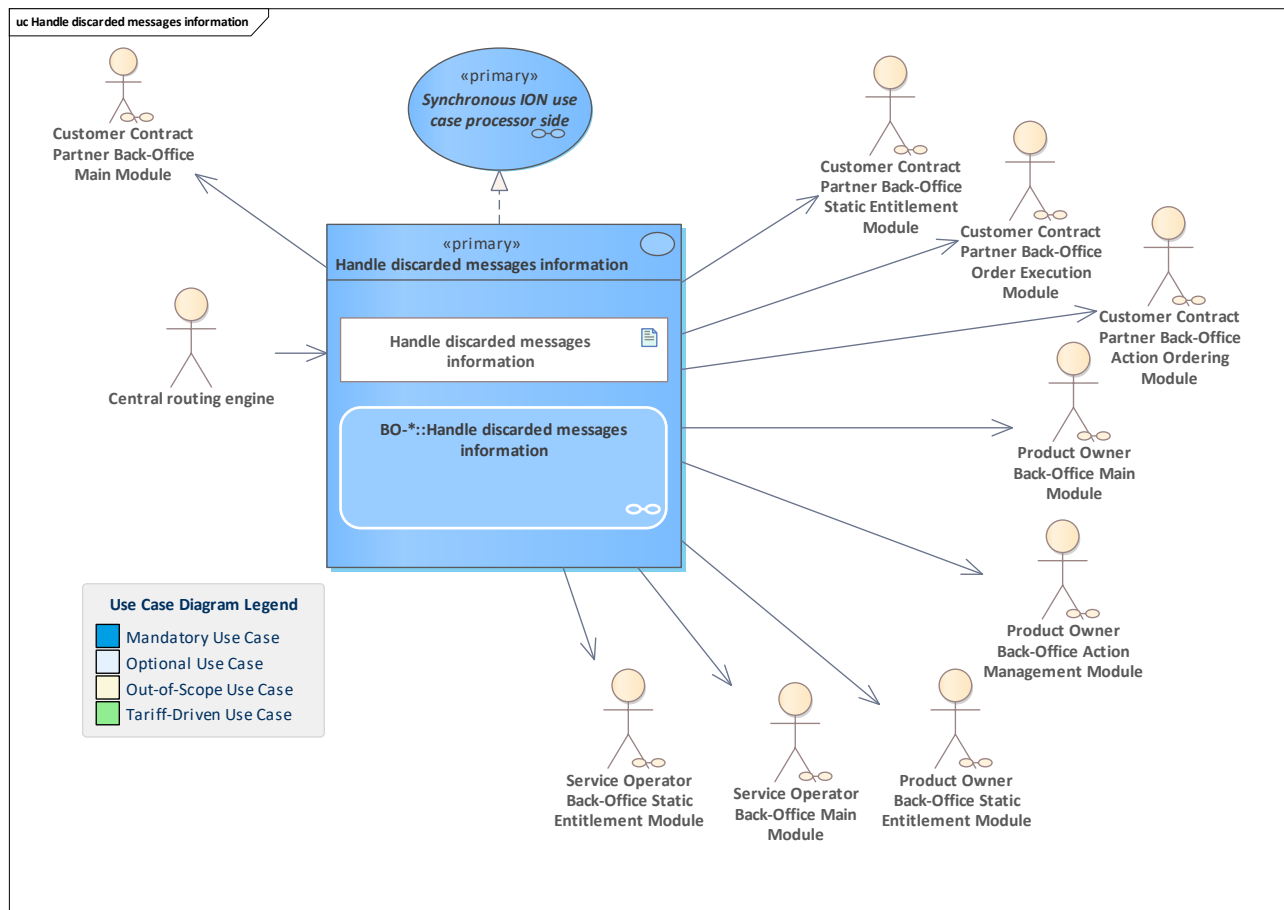
5.1 Overview

[Handle discarded messages information](#)
[Set service as available for a participant](#)
[Set service as unavailable for a participant](#)
[Retrieve CV certificate over signing key](#)

[Retrieve and distribute the CA certificate repository](#)
[Retrieve and distribute the CV certificate revocation list](#)
[Notify events](#)
[Handle events notification](#)
[Demand application hotlisting](#)
[Determine user medium owner](#)
[Demand entitlement hotlisting](#)
[Demand SAM hotlisting](#)
[Determine SAM owner](#)
[Revoke application hotlisting demand](#)
[Revoke entitlement hotlisting demand](#)
[Retrieve entitlement hotlist](#)
[Optional: Retrieve entitlement hotlist with product information](#)
[Optional: Retrieve incremental entitlement hotlist](#)
[Optional: Verify entitlement hotlist updated via increments](#)
[Update organisation hotlist inventory](#)
[Update SAM hotlist inventory](#)
[Retrieve and distribute organisation list](#)

5.2 Use Cases

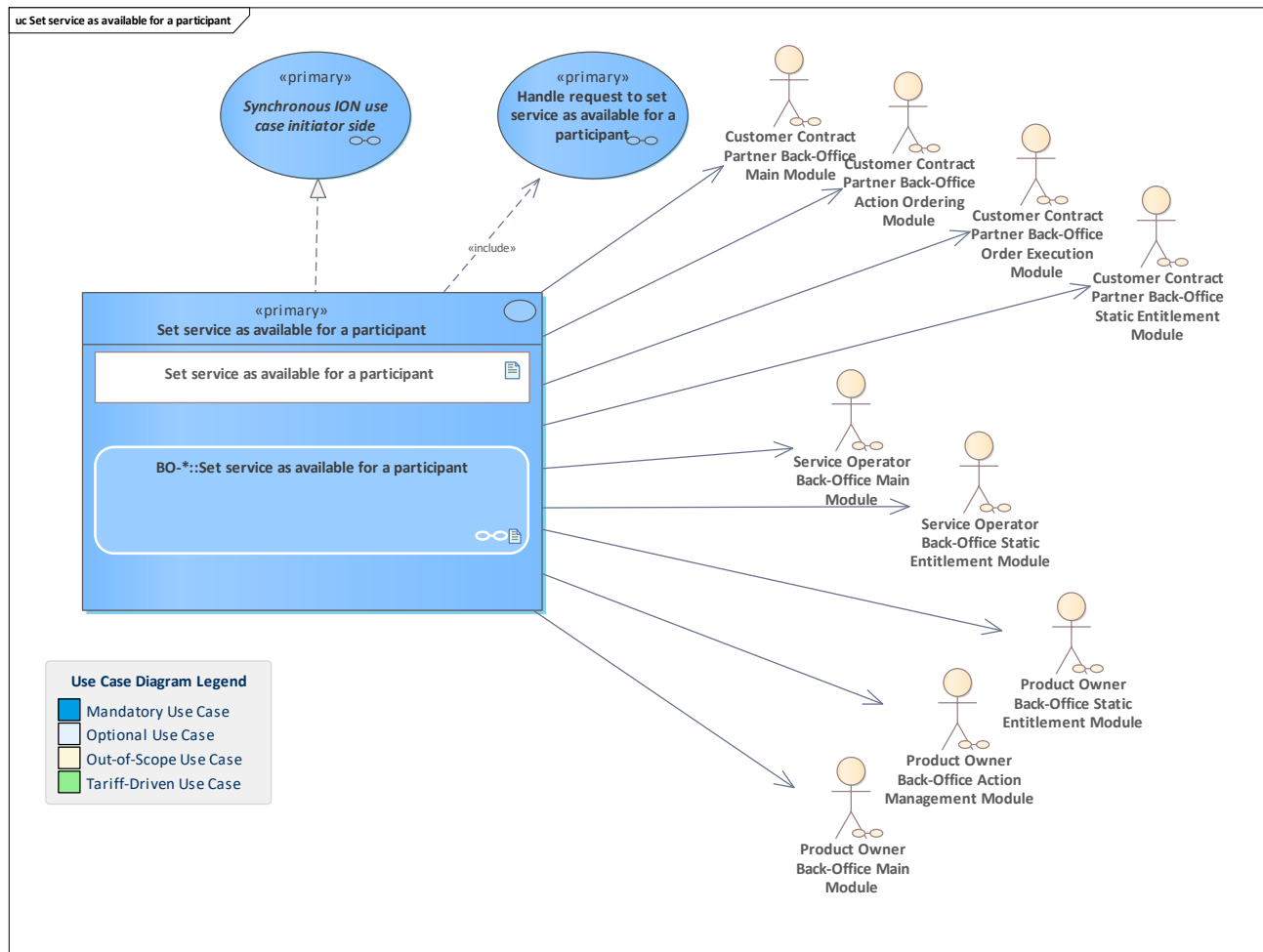
5.2.1 Handle discarded messages information





Description	<p>Process information about discarded asynchronous messages which were provided for store & forward but could not be delivered to the specified recipient.</p> <p>The CRE will send message information to a</p> <ul style="list-style-type: none">• Customer Contract Partner System• Ordering Customer Contract Partner System• Executing Customer Contract Partner System• Customer Contract Partner STE System• Service Operator System• Service Operator STE System• Product Owner System• Product Owner Action Management System• Product Owner STE System <p>These systems and modules work with asynchronous messages which can possibly be discarded.</p>
Initiating Actor	Central routing engine
Reacting Actor	Initiator Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module Service Operator Back-Office Main Module Product Owner Back-Office Action Management Module Customer Contract Partner Back-Office Static Entitlement Module Customer Contract Partner Back-Office Order Execution Module Product Owner Back-Office Static Entitlement Module Service Operator Back-Office Static Entitlement Module Customer Contract Partner Back-Office Action Ordering Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	discarded messages information : notifyDiscardedMessages
Outputs	response : notifyDiscardedMessagesResponse
Error Cases	business exception : notifyDiscardedMessagesException
Activity Diagram	BO-*::Handle discarded messages information

5.2.2 Set service as available for a participant

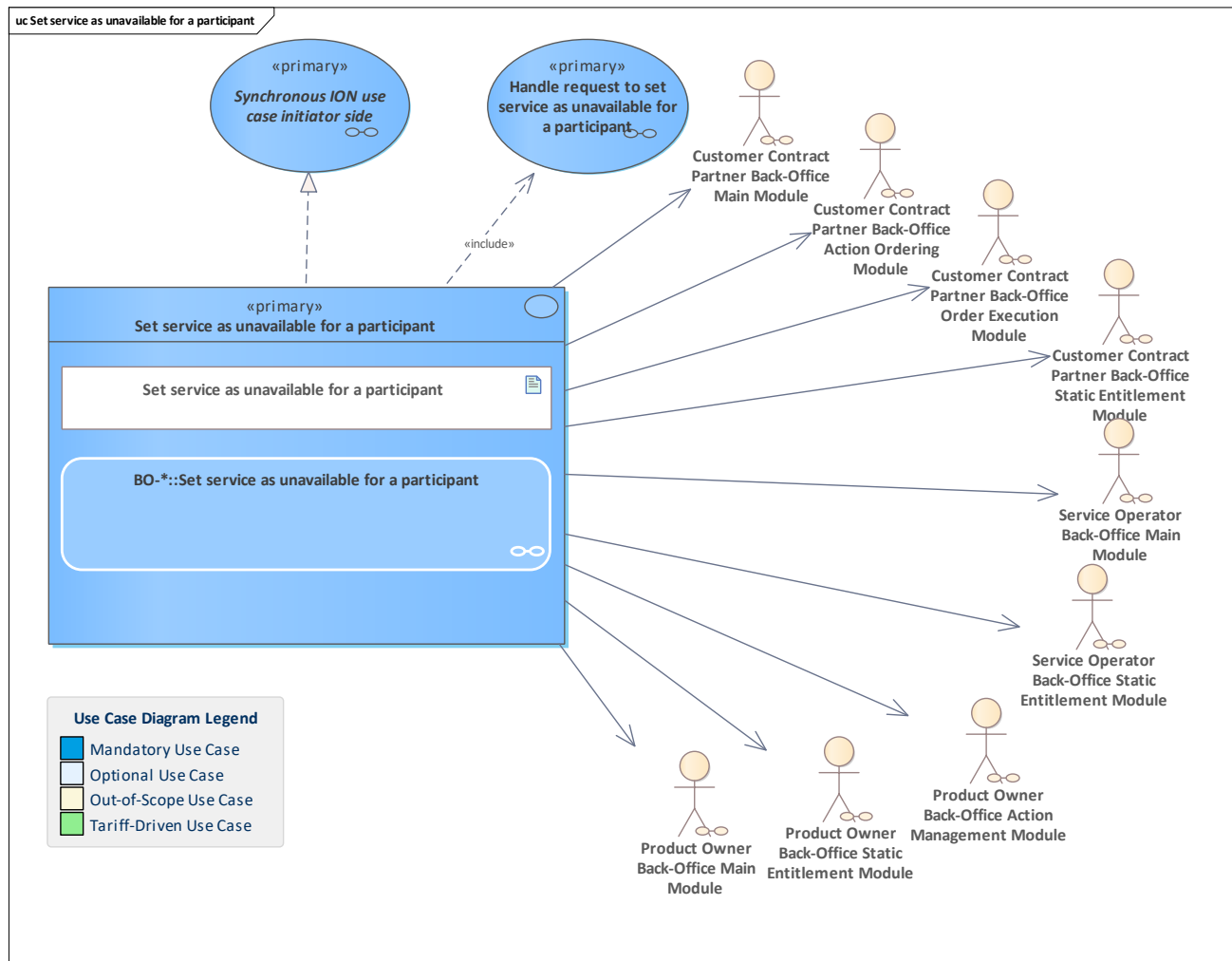


Use Case	Set service as available for a participant
Description	In this use case, a participant sets a certain service as being available in the central routing engine (CRE). The purpose of this use case is to enable receiving messages for use cases in an asynchronous context. Note: if a service is announced to be available (again) the CRE will send stored messages to this service if any stored messages are still in the CRE's message queue.
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Customer Contract Partner Back-Office Action Ordering Module Customer Contract Partner Back-Office Order Execution Module Customer Contract Partner Back-Office Static Entitlement Module Service Operator Back-Office Main Module Service Operator Back-Office Static Entitlement Module Product Owner Back-Office Action Management Module Product Owner Back-Office Main Module Product Owner Back-Office Static Entitlement Module
Preconditions	
Postconditions	
Linked Use Cases	



(Extended By)	
Linked Use Cases (Includes)	Handle request to set service as available for a participant
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	BO-*::Set service as available for a participant

5.2.3 Set service as unavailable for a participant

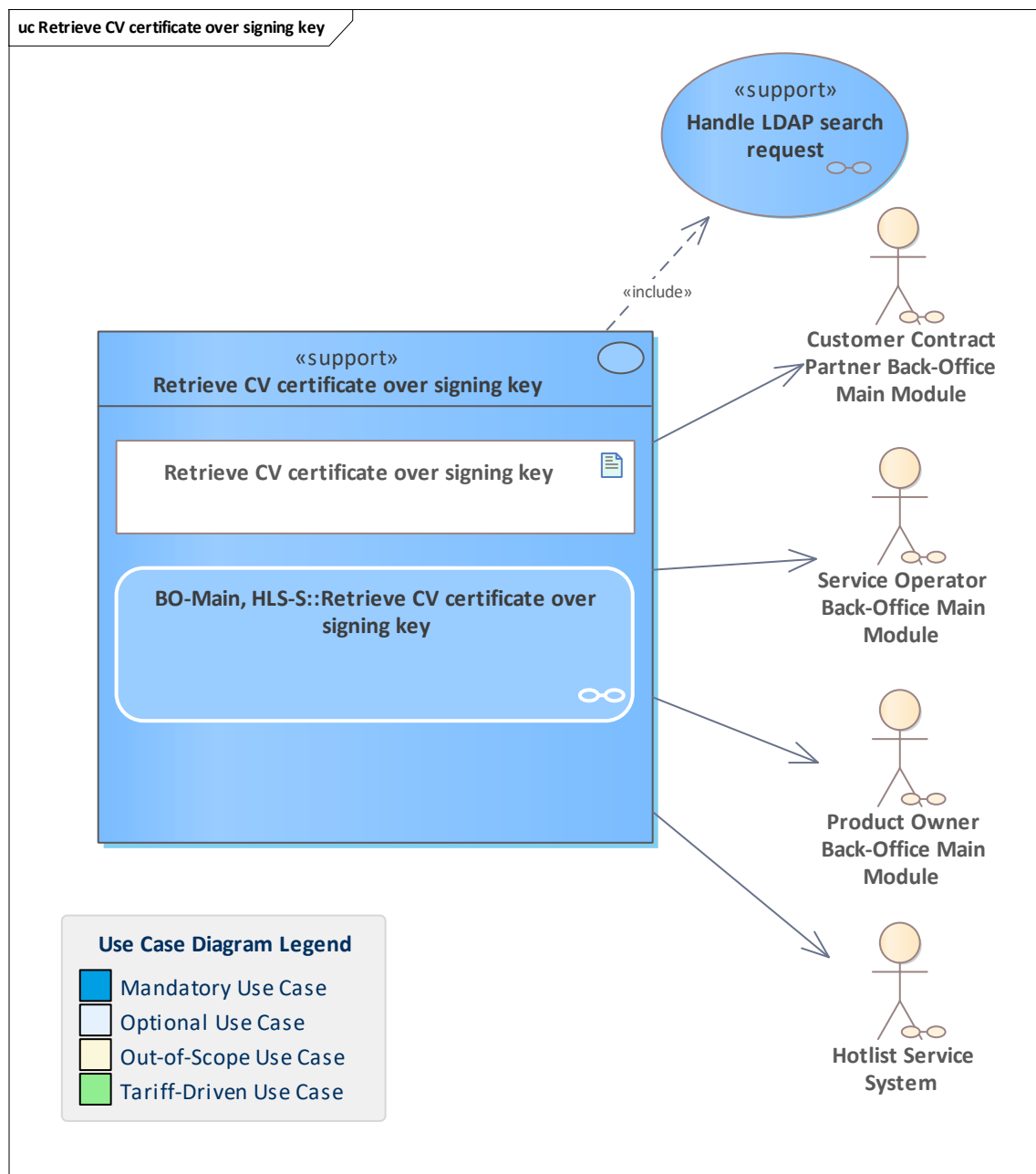


Use Case	<u>Set service as unavailable for a participant</u>
Description	<p>In this use case, a participant sets a certain service as being unavailable in the central routing engine (CRE).</p> <p>The purpose of this use case is to disable receiving messages for use cases in an asynchronous context. This means, that from now on, the CRE will store messages for a later delivery instead of routing them directly to the system that implements the service.</p> <p>Note: participants must configure their services via the ESH in a preparatory step.</p>
Initiating Actor	
Reacting Actor	<u>Customer Contract Partner Back-Office Main Module</u> <u>Customer Contract Partner Back-Office Action Ordering Module</u> <u>Customer Contract Partner Back-Office Order Execution Module</u> <u>Customer Contract Partner Back-Office Static Entitlement Module</u> <u>Service Operator Back-Office Main Module</u> <u>Service Operator Back-Office Static Entitlement Module</u> <u>Product Owner Back-Office Static Entitlement Module</u> <u>Product Owner Back-Office Main Module</u> <u>Product Owner Back-Office Action Management Module</u>
Preconditions	



Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle request to set service as unavailable for a participant
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	BO-*::Set service as unavailable for a participant

5.2.4 Retrieve CV certificate over signing key

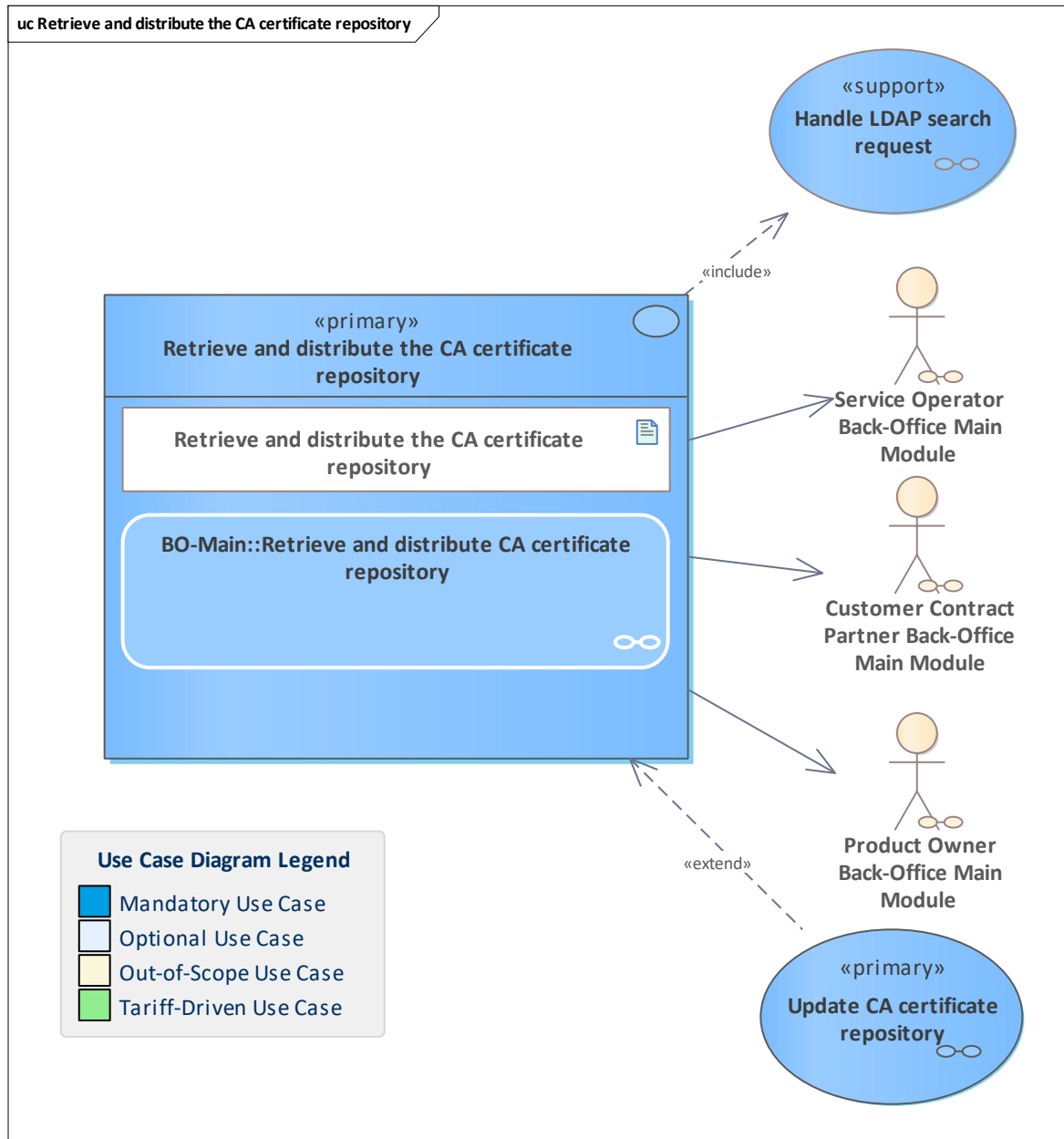


Use Case	Retrieve CV certificate over signing key
Description	Retrieve the latest certificate over the signing key of an end entity. Note that there might be several certificates for this end entity that are not relevant here: superseded certificates and certificates over keys not used for signature purposes.
Initiating Actor	
Reacting Actor	Service Operator Back-Office Main Module Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module Hotlist Service System
Preconditions	



Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle LDAP search request
Linked Use Cases (Realises)	
Base Activity	
Inputs	App instance ID : AppInstanceId
Outputs	CV certificate over signing key : CvCertificate
Error Cases	M-PKI not reachable Unknown app instance ID
Activity Diagram	BO-Main, HLS-S::Retrieve CV certificate over signing key

5.2.5 Retrieve and distribute the CA certificate repository

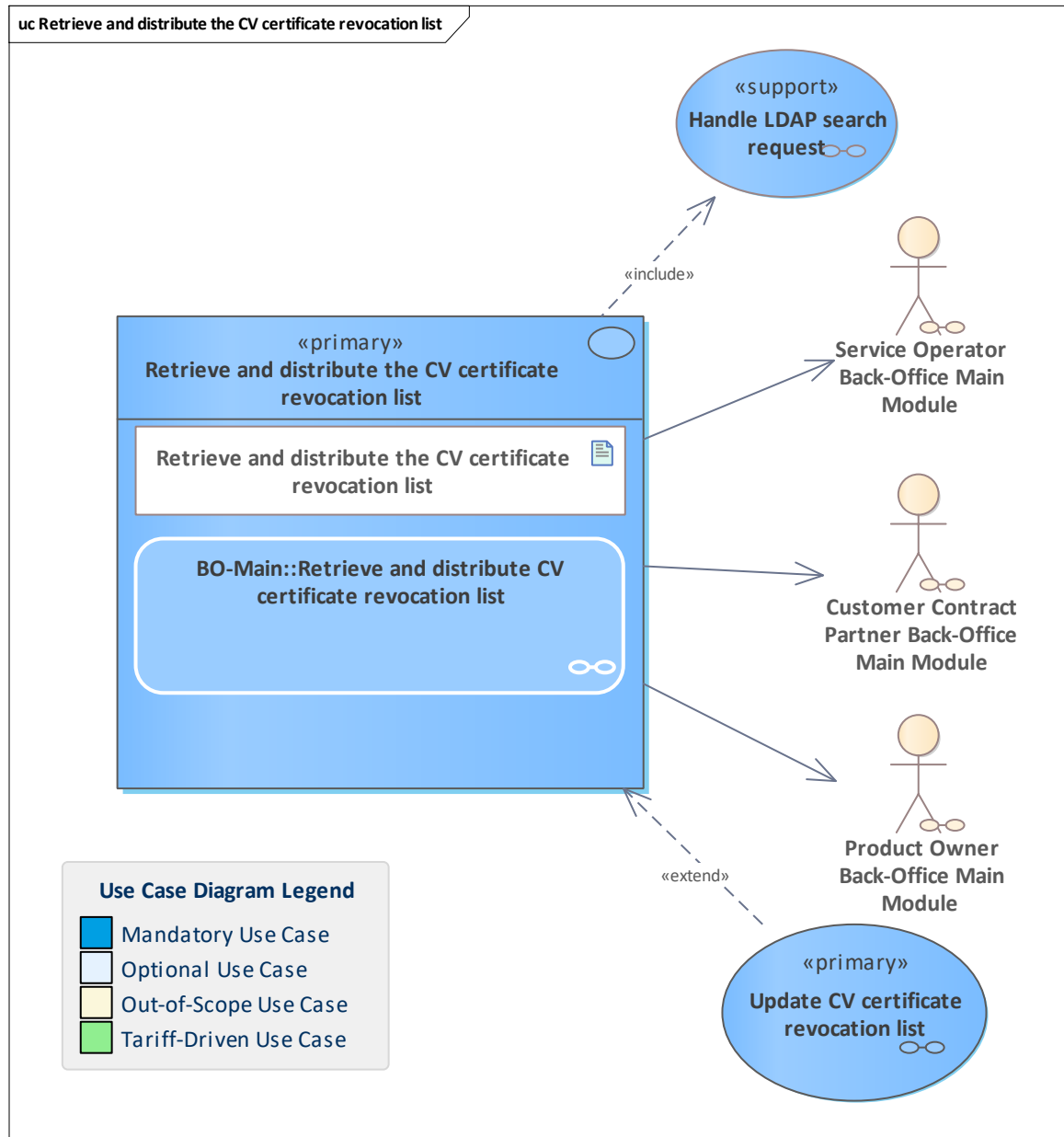


Use Case	Retrieve and distribute the CA certificate repository
Description	The back-office system retrieves the CA certificate repository from the media-PKI (M-PKI). If the requestor operates terminals that belong to back-office this system, the CA certificate repository is also updated in all of them. This process needs to run periodically to keep the CA certificate repository up to date.
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module



	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Update CA certificate repository
Linked Use Cases (Includes)	Handle LDAP search request
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	BO-Main::Retrieve and distribute CA certificate repository

5.2.6 Retrieve and distribute the CV certificate revocation list

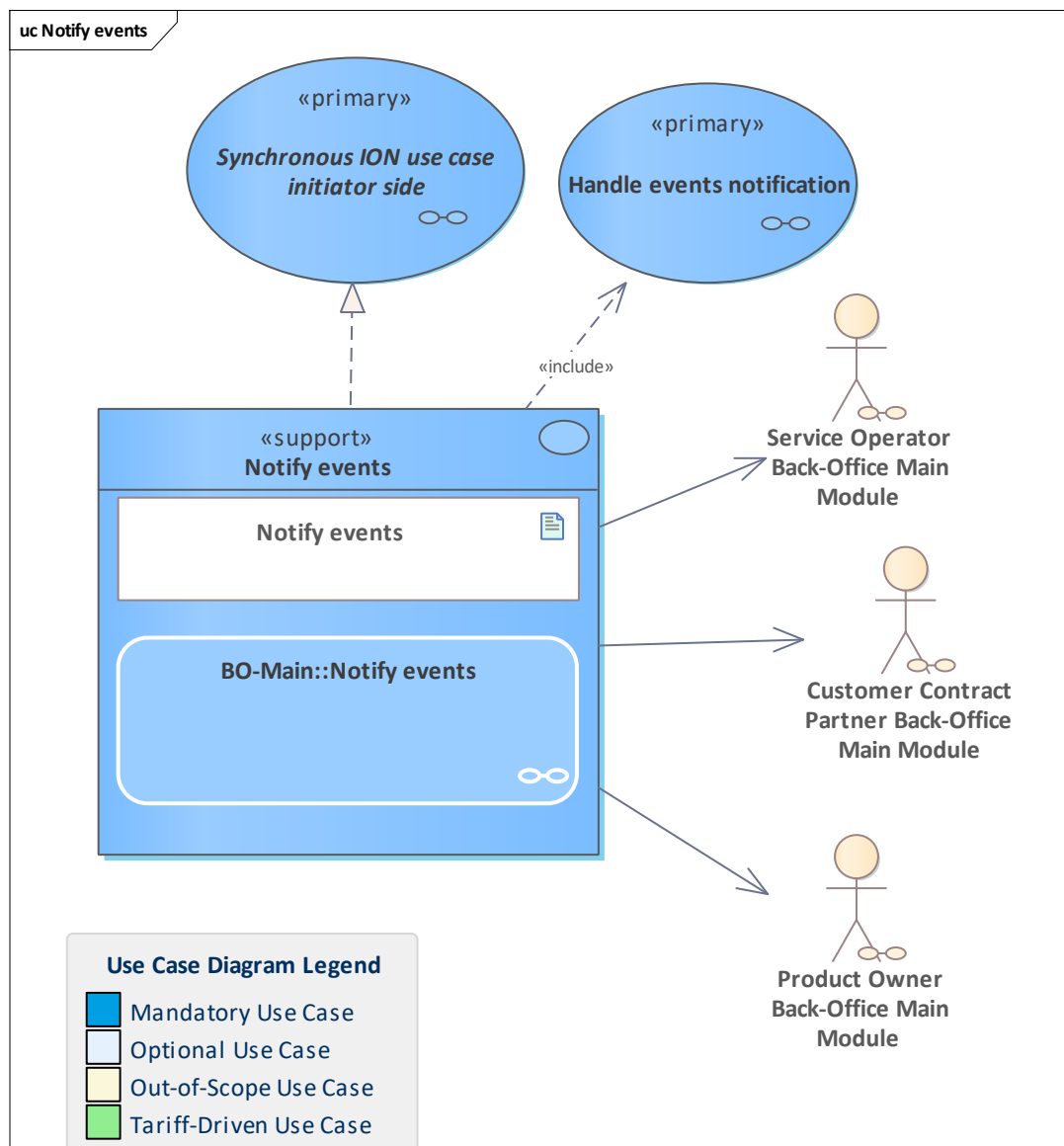


Use Case	Retrieve and distribute the CV certificate revocation list
Description	The back-office system retrieves the CV certificate revocation list from the media-PKI (M-PKI). If the requestor operates terminals that belong to back-office this system, the CV certificate revocation list is also updated in all of them. This process needs to run periodically to keep the CV certificate revocation list up to date.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module Service Operator Back-Office Main Module Customer Contract Partner Back-Office Main Module



Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Update CV certificate revocation list
Linked Use Cases (Includes)	Handle LDAP search request
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	BO-Main::Retrieve and distribute CV certificate revocation list

5.2.7 Notify events

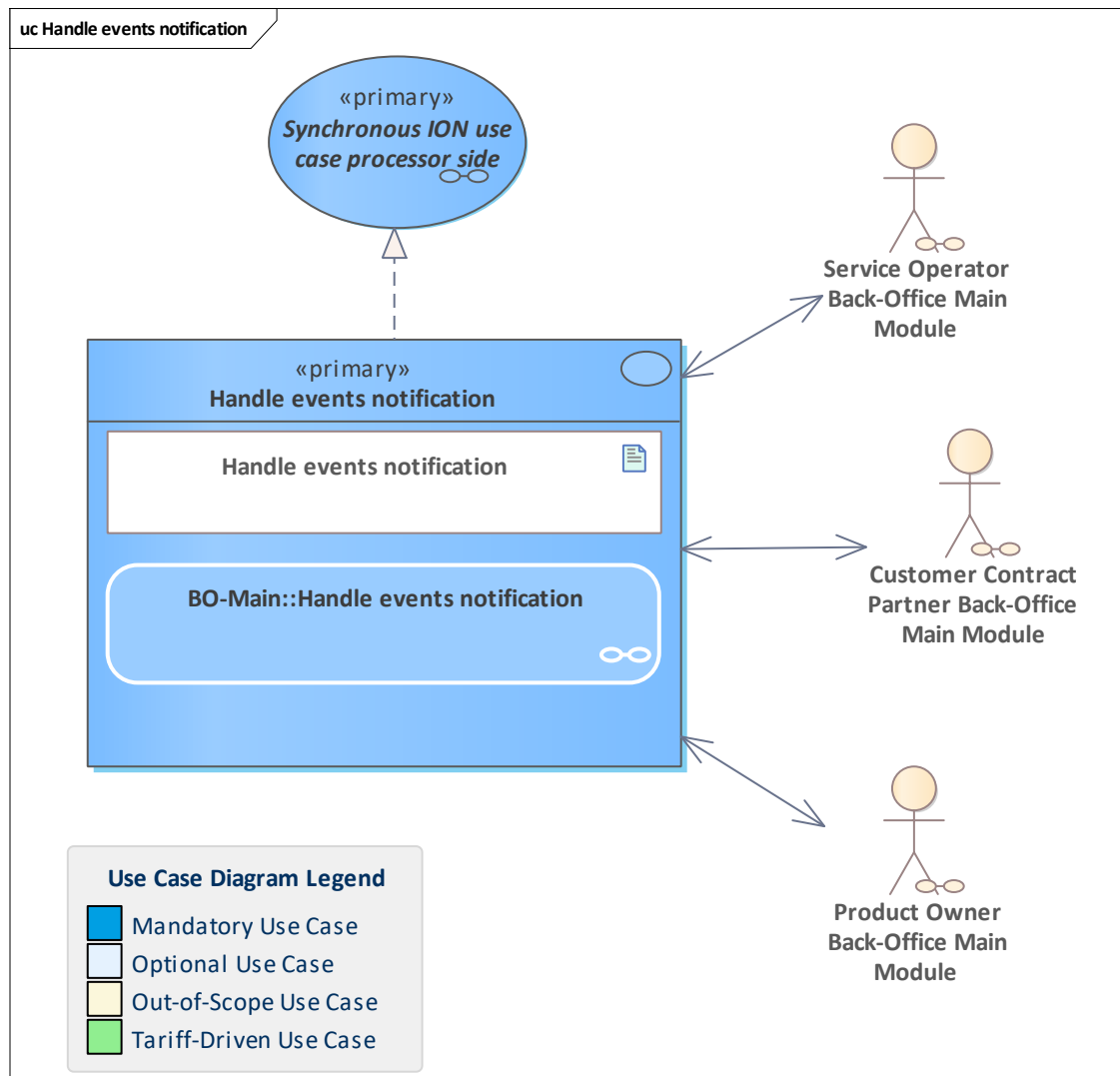


Use Case	Notify events
Description	Notify a participant about warnings which occurred during the downstream monitoring or similar.
Initiating Actor	
Reacting Actor	Service Operator Back-Office Main Module Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle events notification
Linked Use Cases (Realises)	Synchronous ION use case initiator side



Base Activity	
Inputs	Role : PartnerRoleCode Warnings : Warning Receiver : OrganisationId
Outputs	
Error Cases	
Activity Diagram	BO-Main::Notify events

5.2.8 Handle events notification

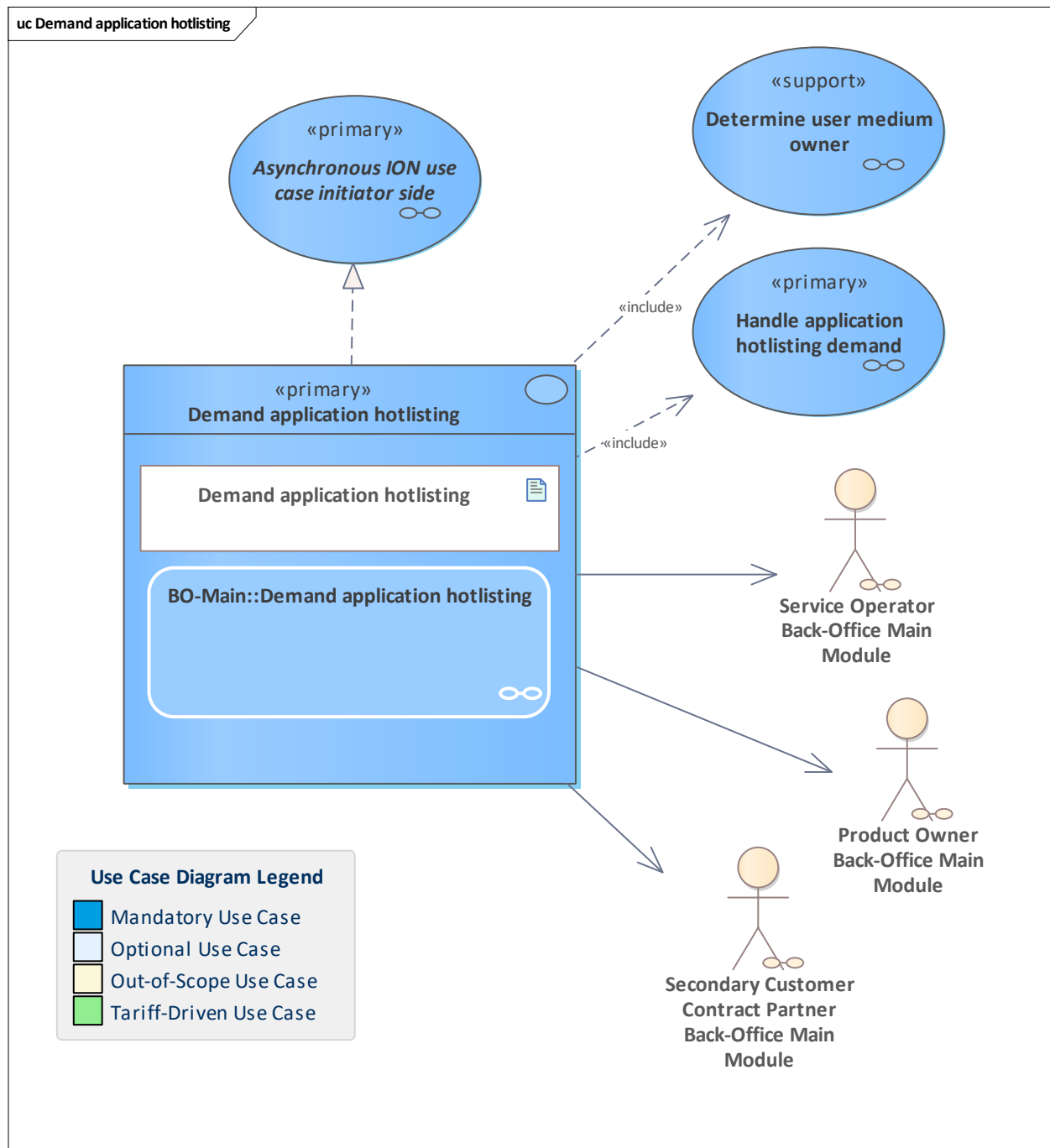


Use Case	Handle events notification
Description	A participant is informed about warnings.
Initiating Actor	Service Operator Back-Office Main Module Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module
Reacting Actor	Service Operator Back-Office Main Module Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	



Inputs	Notify events : notifyEvents
Outputs	Notify events response : notifyEventsResponse
Error Cases	Notify events exception : notifyEventsException
Activity Diagram	BO-Main::Handle events notification

5.2.9 Demand application hotlisting

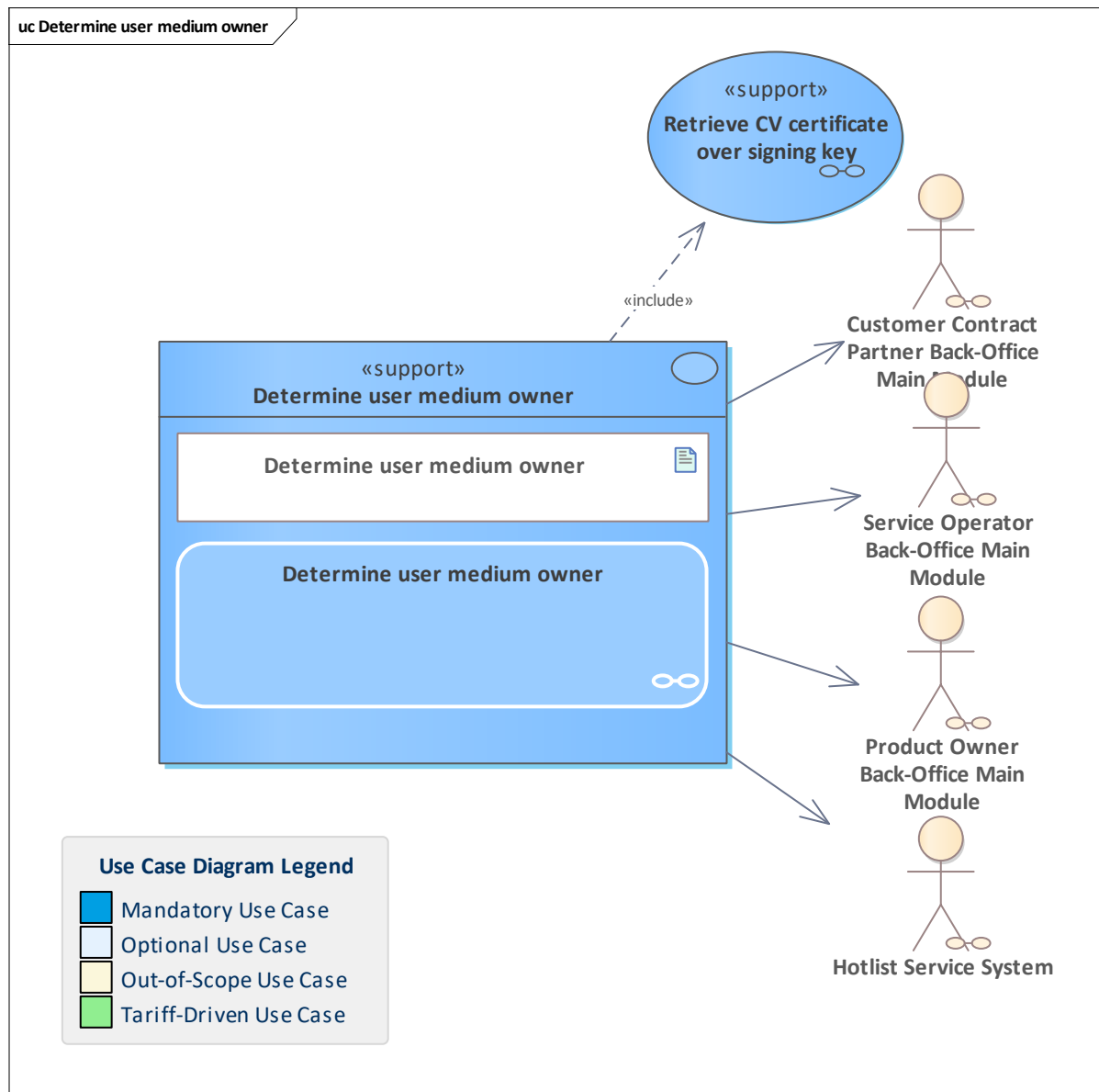


Use Case	Demand application hotlisting
Description	<p>The PO, sCCP or SO as sender demands the pCCP that issued the application to the customer to hotlist the application in question. The sender adds a reason (SO e.g. in case of a defective user medium, sCCP e.g. in case of a lost user medium) and further hotlisting parameters.</p> <p>This application can be either a user medium application with an application instance ID or a MOTICS app with an SCE ID.</p>



	In most cases, the hotlist demand from a third party will be caused by monitoring.
Initiating Actor	
Reacting Actor	Service Operator Back-Office Main Module Secondary Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle application hotlisting demand / Determine user medium owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side
Base Activity	
Inputs	Application transition counter : TransitionCounter Blocking reason : BlockingReason Hotlist entry expiration time : ZonedDateTime Application blocking mode : ApplicationBlockingModeCode App instance ID : AppInstanceId
Outputs	
Error Cases	
Activity Diagram	BO-Main::Demand application hotlisting

5.2.10 Determine user medium owner

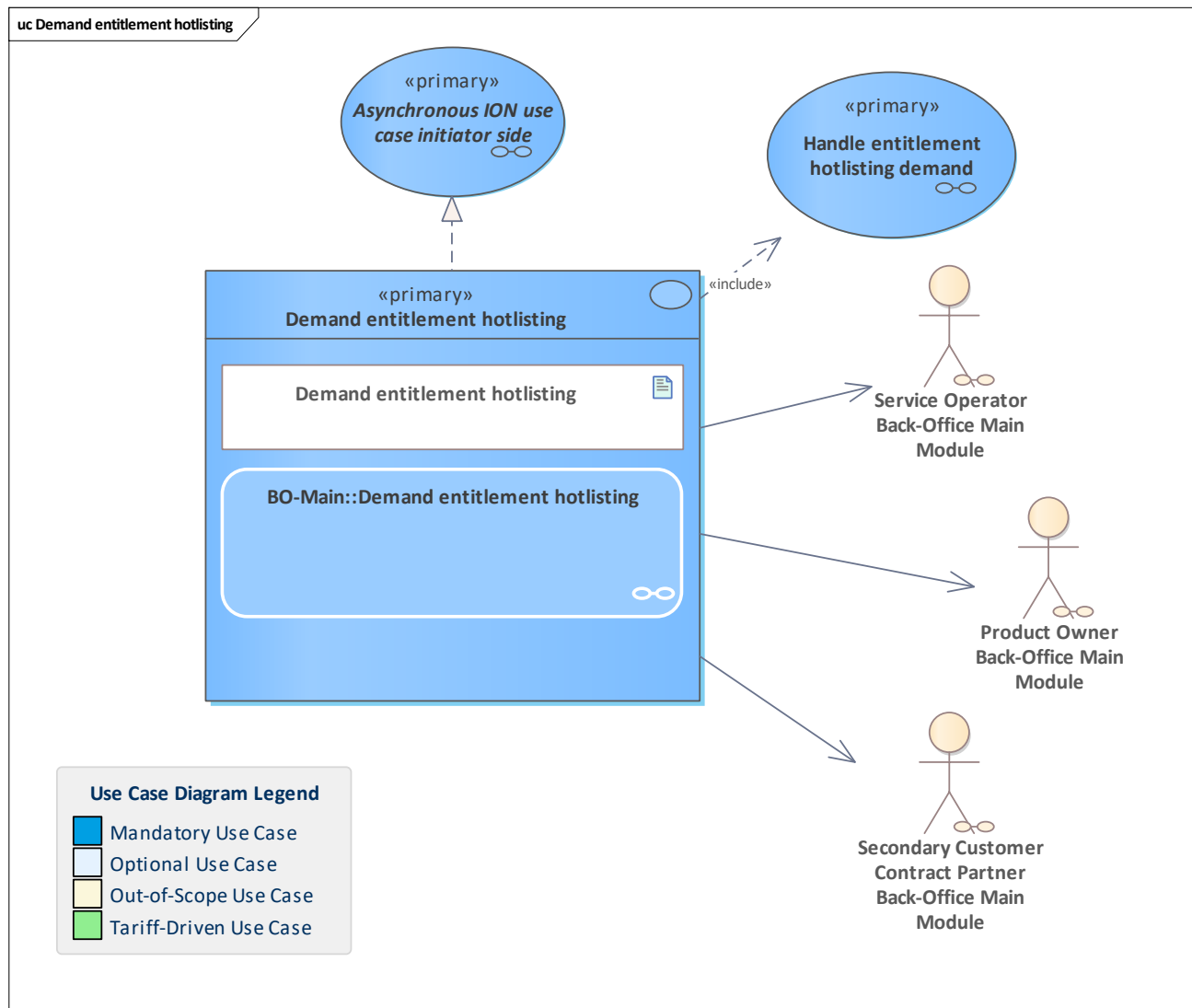


Use Case	<u>Determine user medium owner</u>
Description	<p>Determine the owner of a user medium with an application or SCE (in case of MOTICS with static entitlement) in a back-office system using the ownership information contained in the corresponding certificate.</p> <p>To determine the user medium owner, the matching CV certificate is to be fetched, so the caller retrieves the latest certificate over the signing key of an end entity.</p> <p>Note that there might be several certificates for this end entity, that are not relevant here: superseded certificates and certificates for keys not used for signature purposes. Thus, the right certificate that delivers the owner organisation ID has to be filtered.</p>
Initiating Actor	
Reacting Actor	<u>Customer Contract Partner Back-Office Main Module</u>



	Service Operator Back-Office Main Module Product Owner Back-Office Main Module Hotlist Service System
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Retrieve CV certificate over signing key
Linked Use Cases (Realises)	
Base Activity	
Inputs	App instance ID : AppInstanceId
Outputs	Org ID of user medium owner : OrganisationId
Error Cases	M-PKI not reachable Unknown app instance ID
Activity Diagram	Determine user medium owner

5.2.11 Demand entitlement hotlisting

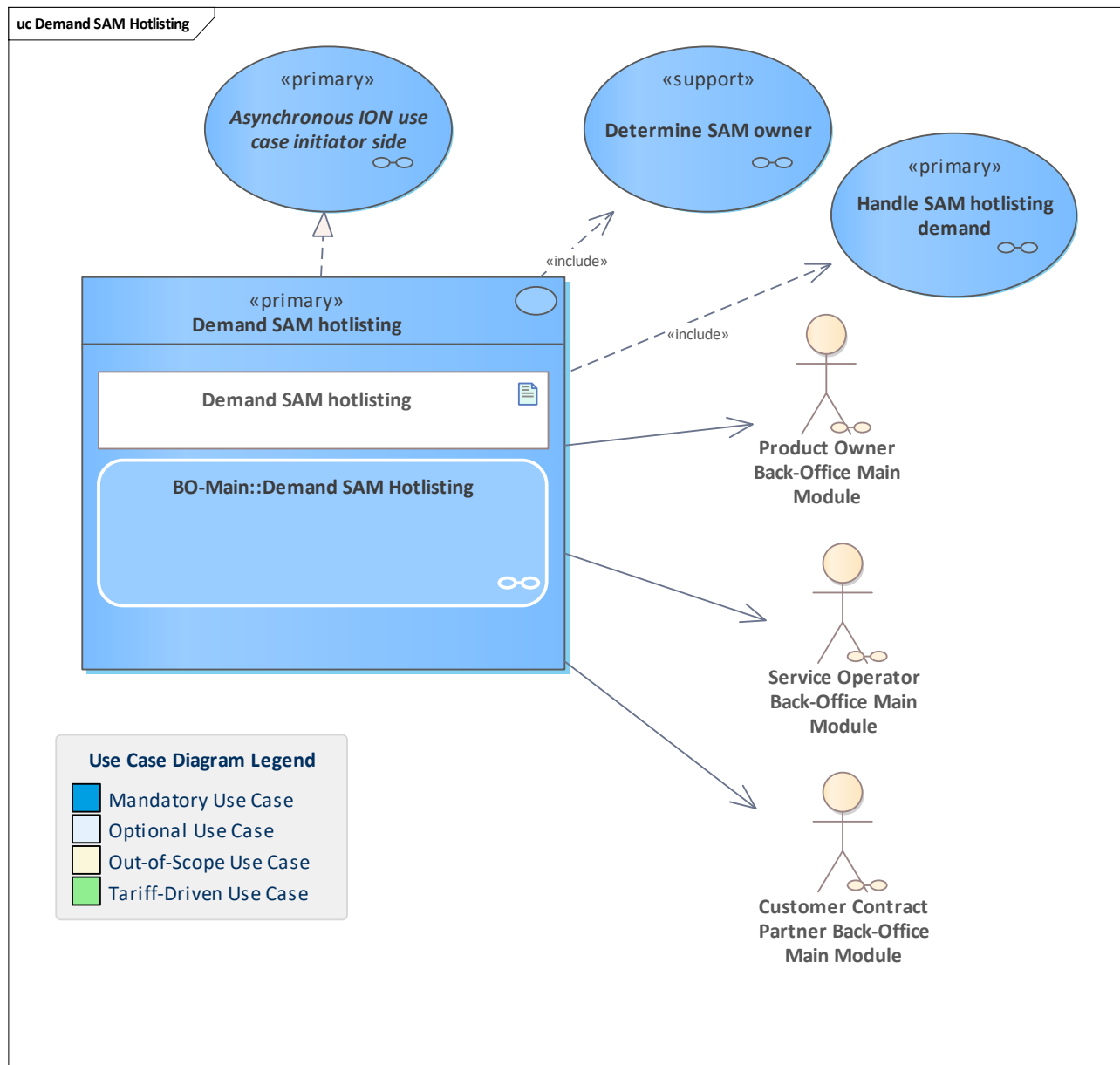


Use Case	Demand entitlement hotlisting
Description	<p>In this use case, the PO, sCCP or SO as sender demands the pCCP that issued the entitlement to the customer to hotlist the entitlement in question.</p> <p>The sender adds a reason and further hotlisting parameters. This entitlement can be either located on a user medium with an application or a static entitlement coming from a barcode or a MOTICS app.</p> <p>In most cases, the hotlisting demand from a third party will be caused by monitoring.</p> <p>Possible reasons are:</p> <ul style="list-style-type: none"> Inconsistencies have occurred during monitoring (the most likely reason) Offence against terms of carriage Payment delay Referenced product was deactivated



	<ul style="list-style-type: none">Entitlement with the same ID already existsUser medium/application which contains entitlements of a secondary customer contract partner (CCP) was replaced
Initiating Actor	
Reacting Actor	Service Operator Back-Office Main Module Secondary Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle entitlement hotlisting demand
Linked Use Cases (Realises)	Asynchronous ION use case initiator side
Base Activity	
Inputs	Blocking reason : BlockingReason Hotlist entry expiration time : ZonedDateTime Entitlement blocking mode : EntitlementBlockingModeCode Hotlist entry effective time : ZonedDateTime UM or SCE-ID : AppInstanceId Entitlement transition counter : TransitionCounter Product ID : ProductId Entitlement ID : EntitlementId
Outputs	
Error Cases	
Activity Diagram	BO-Main::Demand entitlement hotlisting

5.2.12 Demand SAM hotlisting

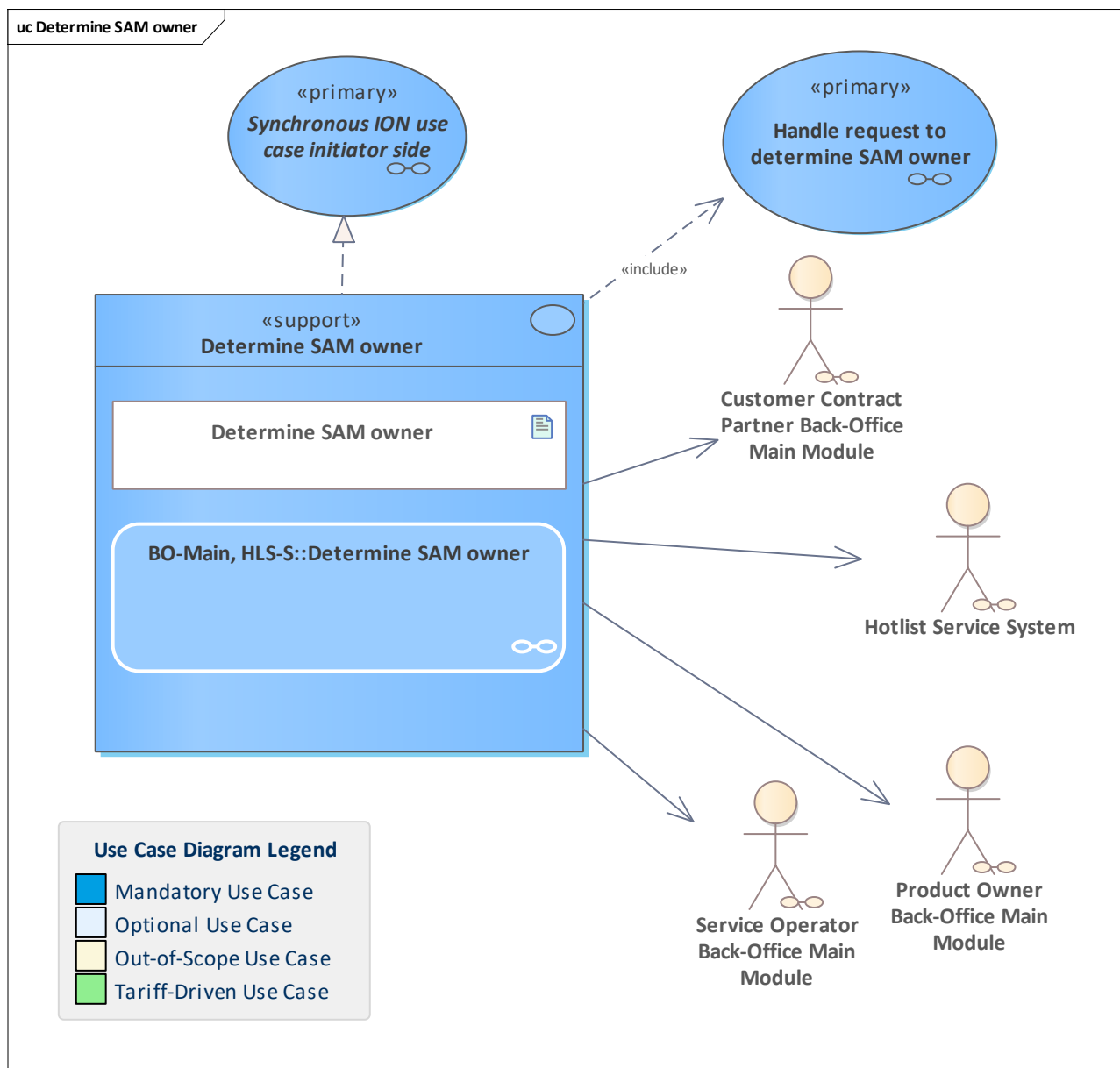


Use Case	Demand SAM hotlisting
Description	<p>A PO, SO or CCP sends a demand for hotlisting to the SAM Owner (SO or CCP). Reasons for this demand could either be loss or theft of a SAM that was used in one of the SO or CCP terminals. Another reason could be suspected fraud, which could be detected by monitoring.</p> <p>Before demanding the SAM hotlisting, the demander must find out the SAM owner to place the demand to the right receiver.</p>
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	



Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle SAM hotlisting demand / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side
Base Activity	
Inputs	Blocking reason : BlockingReason SAM action counter : ActionCounter SAM ID : AppInstanceId SAM entitlement issuance counter : EntitlementIssuanceCounter
Outputs	
Error Cases	
Activity Diagram	BO-Main::Demand SAM Hotlisting

5.2.13 Determine SAM owner



Use Case	Determine SAM owner
Description	Determine the SAM Owner (organisation ID and role) for a given SAM ID using the service provided by the ESH.
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module Hotlist Service System Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases	Handle request to determine SAM owner

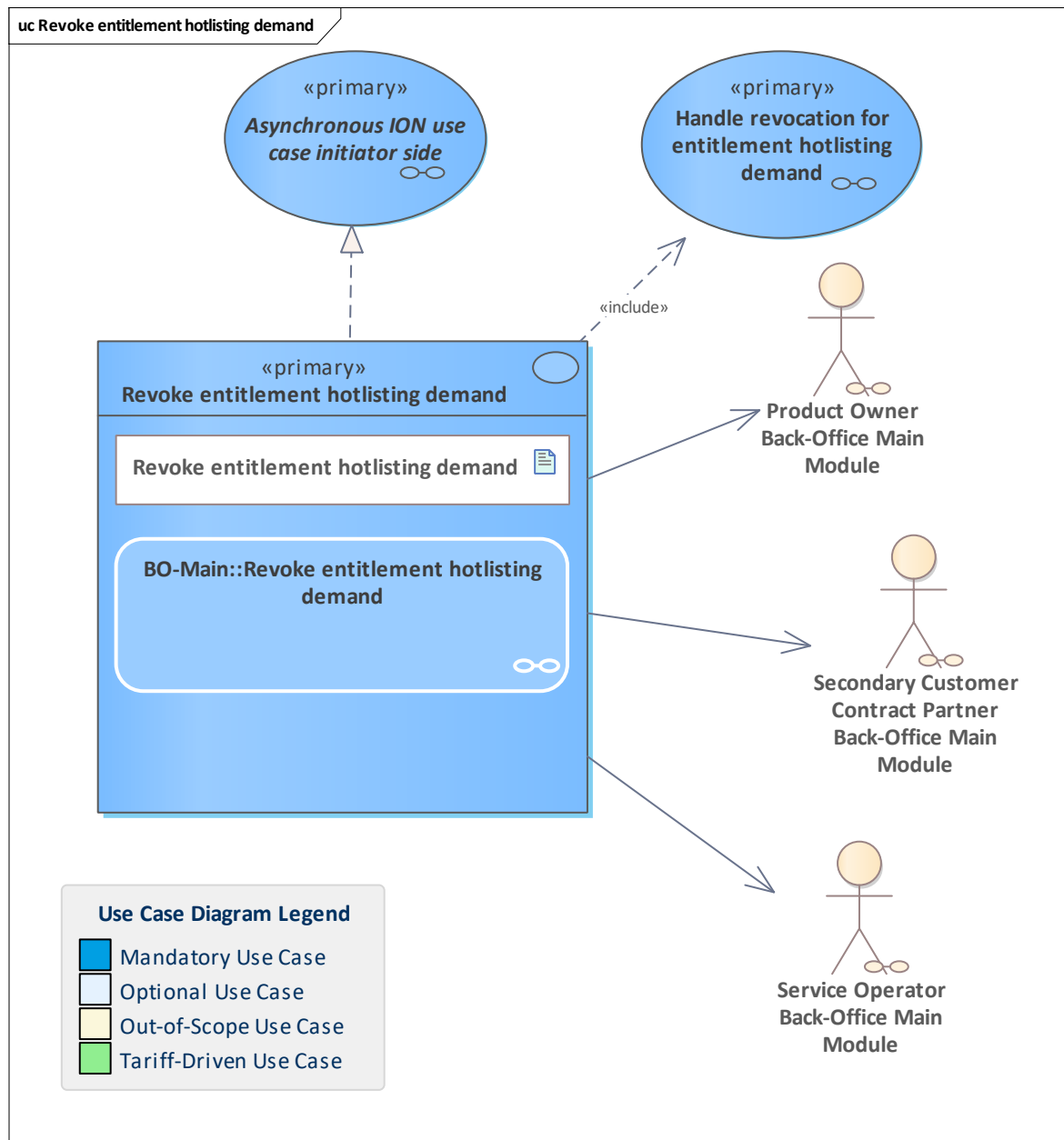


(Includes)	
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	SAM ID : AppInstanceId
Outputs	SAM owner : OrganisationalUnit
Error Cases	E_CO_APP_INSTANCE_ID_UNKNOWN Unknown SAM : E_CO_APP_INSTANCE_ID_UNKNOWN Timeout
Activity Diagram	BO-Main, HLS-S::Determine SAM owner



	The new application owner must have transferred the information about old hotlisting demands (if any), especially the ION message references.
Initiating Actor	
Reacting Actor	Secondary Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle revocation for application hotlisting demand / Determine user medium owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side
Base Activity	
Inputs	ION message id : IonMessageId App instance ID : AppInstanceId
Outputs	
Error Cases	
Activity Diagram	BO-Main::Revoke application hotlisting demand

5.2.15 Revoke entitlement hotlisting demand



Use Case	Revoke entitlement hotlisting demand
Description	<p>A PO, SO or sCCP sends a demand for hotlisting revocation to the entitlement owner (pCCP).</p> <p>The pCCP will check the revocation demand and, in case of a positive decision, have the entitlement removed from the hotlist. The revocation references the ION message of the previous hotlisting demand.</p> <p>This is a rarely used use case.</p>
Initiating Actor	
Reacting Actor	Service Operator Back-Office Main Module Secondary Customer Contract Partner Back-Office Main Module Product Owner Back-Office Main Module

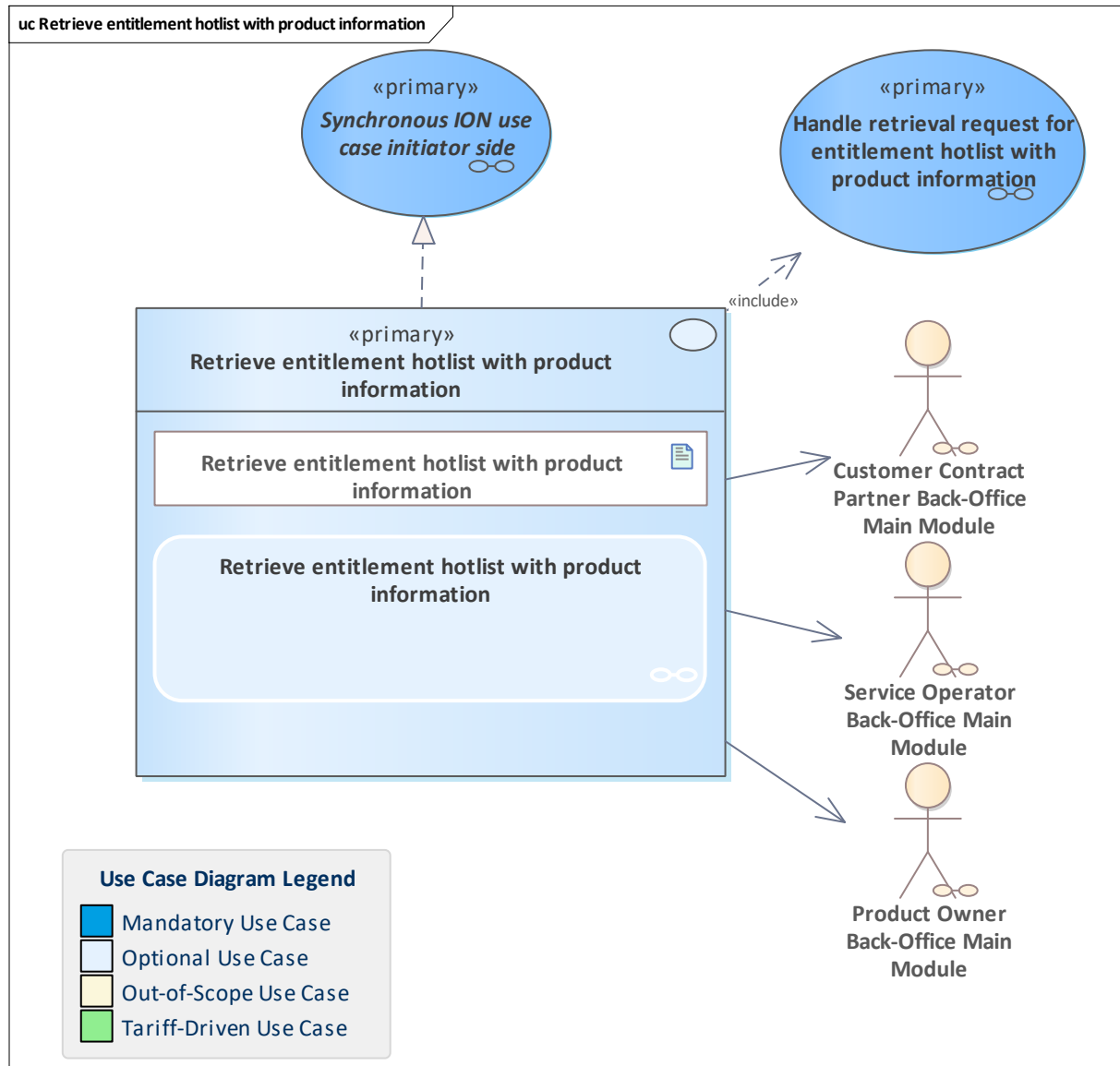


Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle revocation for entitlement hotlisting demand
Linked Use Cases (Realises)	Asynchronous ION use case initiator side
Base Activity	
Inputs	Entitlement Id : EntitlementId ION message ID : IonMessageId
Outputs	
Error Cases	
Activity Diagram	BO-Main::Revoke entitlement hotlisting demand



Linked Use Cases (Includes)	Handle retrieval request for entitlement hotlist
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Process instance ID : ProcessInstanceId
Outputs	Entitlement hotlist : EntitlementHotlist
Error Cases	
Activity Diagram	Retrieve entitlement hotlist

5.2.17 Optional: Retrieve entitlement hotlist with product information

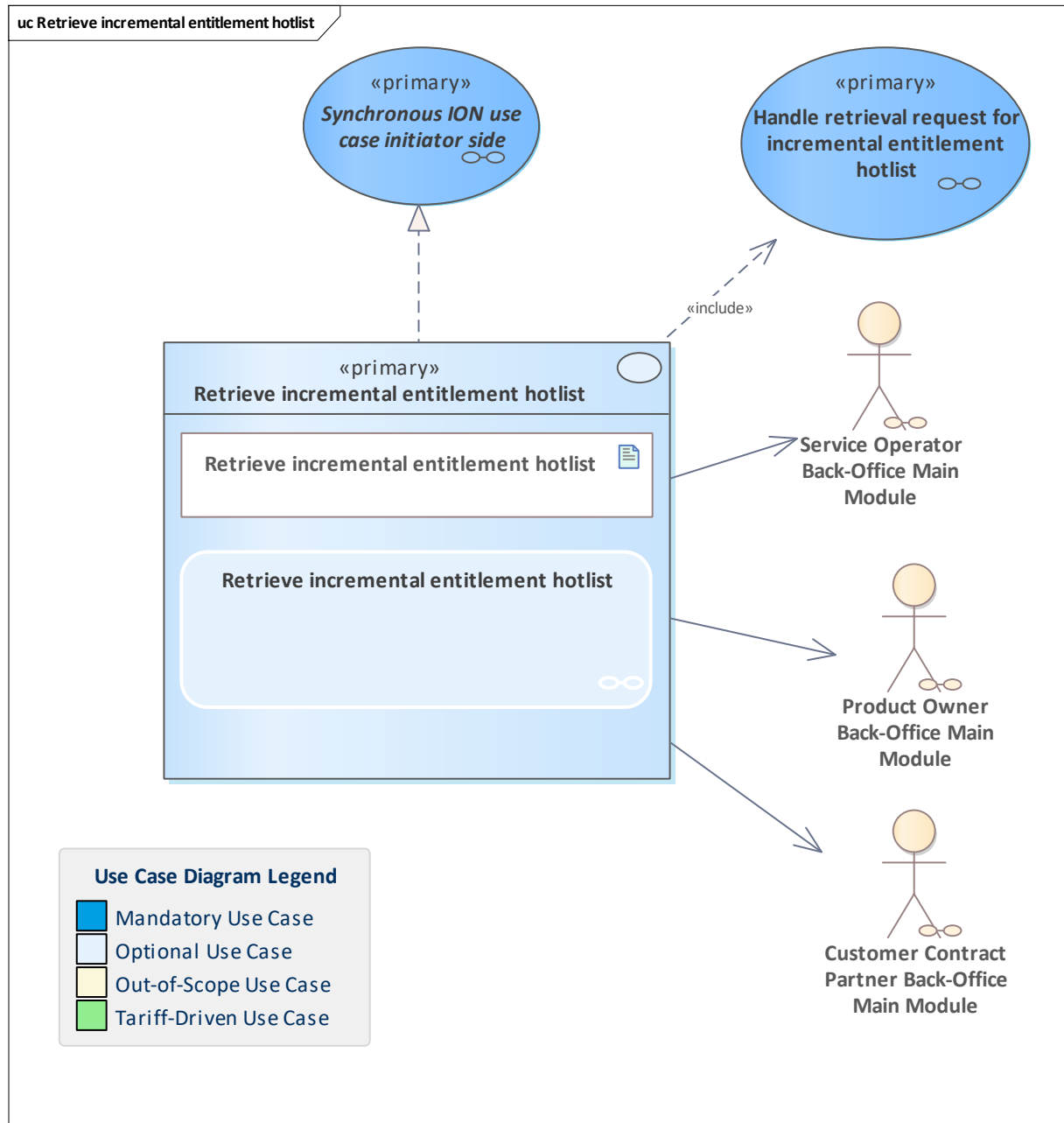


Use Case	Retrieve entitlement hotlist with product information
Description	The SO, PO and CCP want to update their entitlement hotlist inventory by retrieving the entitlement hotlist with additionally contained product information from the hotlist service system. Please note that the entitlement hotlist can be retrieved either as an incremental or a total hotlist, as well as a hotlist with product information (this use case).
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	



Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle retrieval request for entitlement hotlist with product information
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Process instance ID : ProcessInstanceId
Outputs	Entitlement hotlist : EntitlementHotlist
Error Cases	
Activity Diagram	Retrieve entitlement hotlist with product information

5.2.18 Optional: Retrieve incremental entitlement hotlist

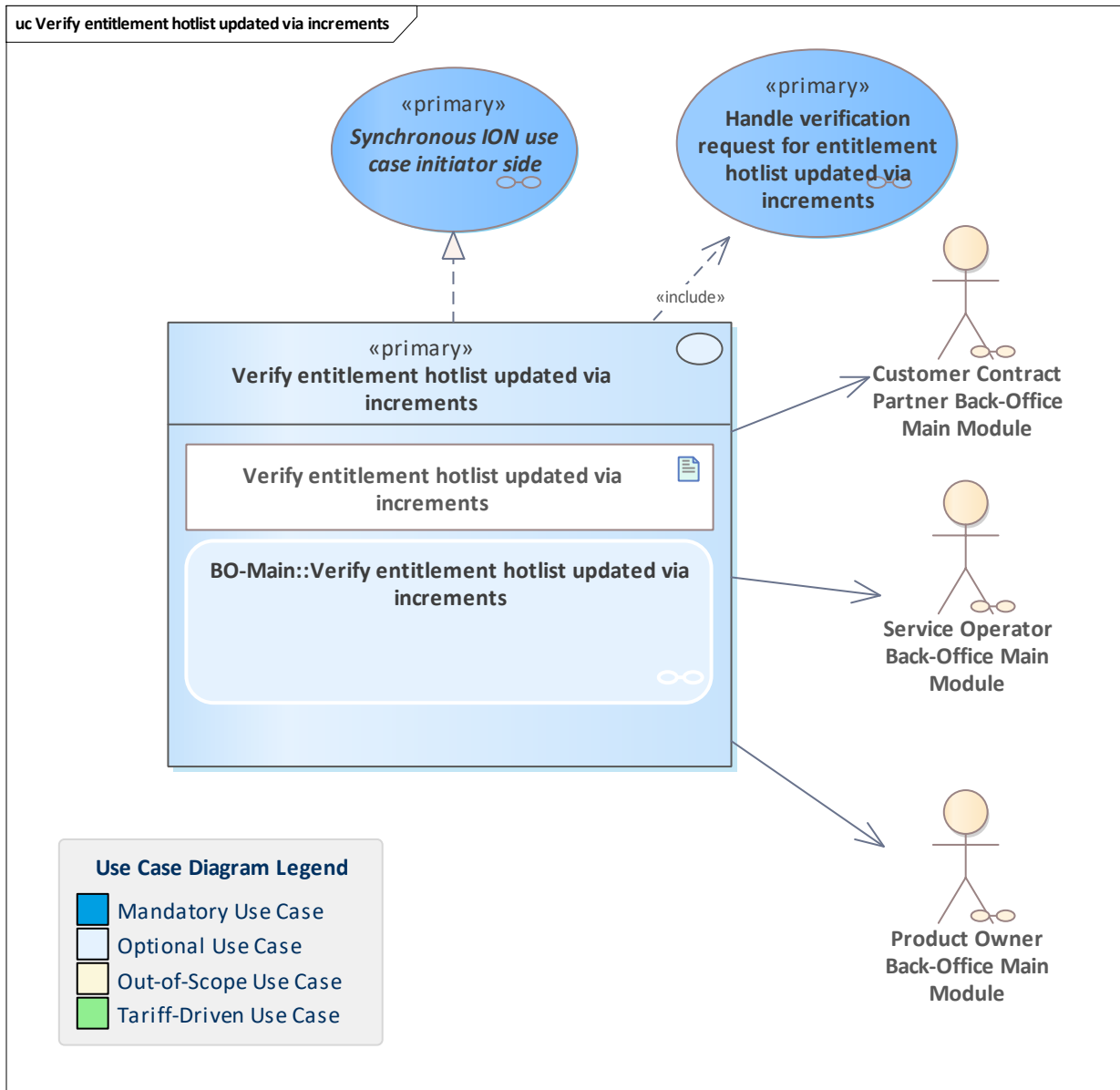


Use Case	Retrieve incremental entitlement hotlist
Description	The SO, PO and CCP want to update their entitlement hotlist inventory by retrieving the current incremental entitlement hotlist from the hotlist service system. Please note that the entitlement hotlist can be retrieved either as an incremental (this use case) or a total hotlist, as well as a hotlist with product information.
Initiating Actor	
Reacting Actor	Service Operator Back-Office Main Module



	Product Owner Back-Office Main Module Customer Contract Partner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle retrieval request for incremental entitlement hotlist
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Process instance ID : ProcessInstanceId
Outputs	Updated entitlement hotlist : EntitlementHotlist
Error Cases	
Activity Diagram	Retrieve incremental entitlement hotlist

5.2.19 Optional: Verify entitlement hotlist updated via increments

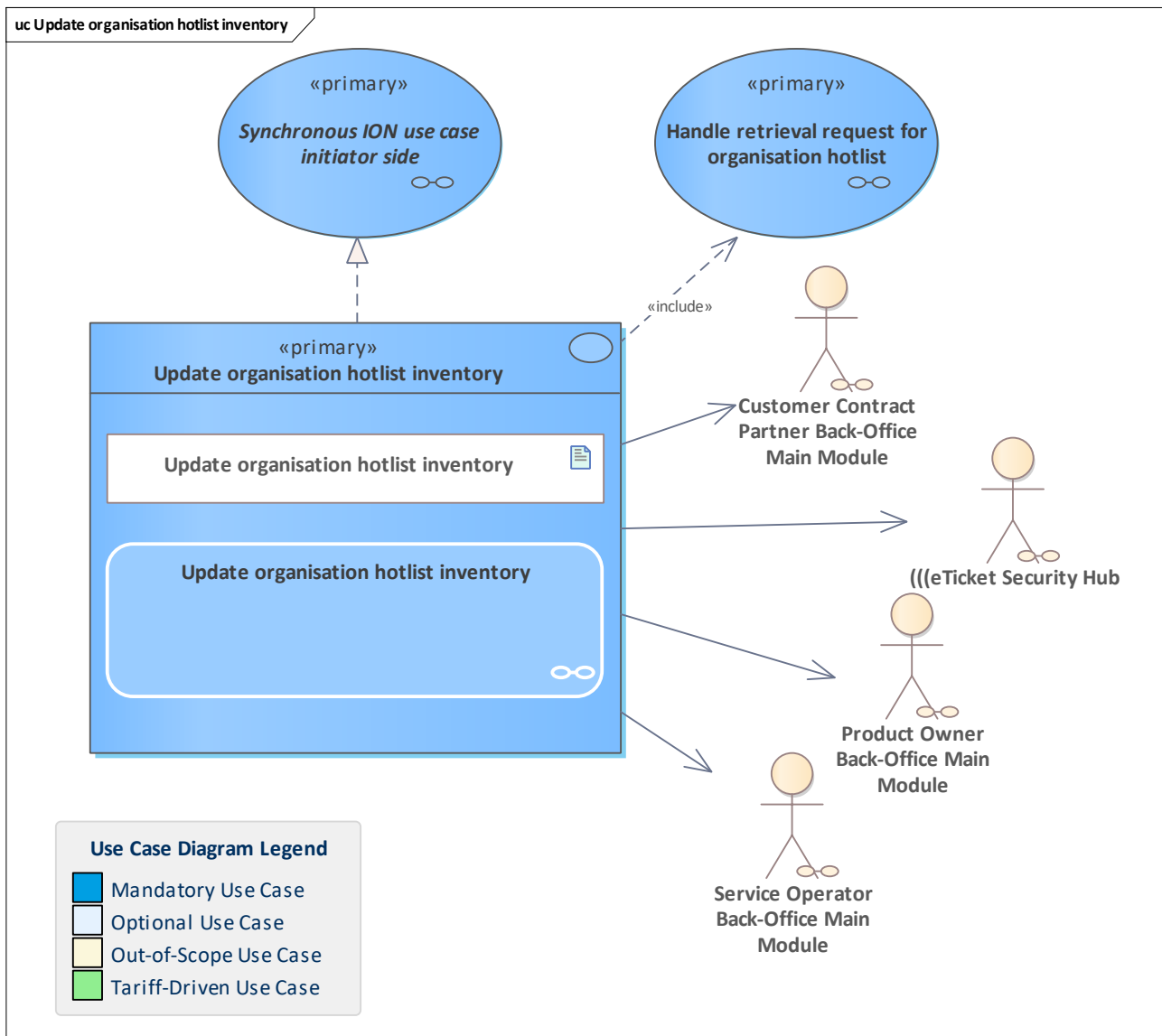


Use Case	Verify entitlement hotlist updated via increments
Description	After updating the entitlement hotlist inventory via the last incremental entitlement hotlist, the participant must ensure that the updated hotlist inventory is the same as the hotlist inventory of the hotlist service system (filtered to the participant's products). For that reason, participants compute the checksum of all the entitlement IDs in their inventory and send it to the hotlist service system to verify the value of the checksum. See also Checksum calculation for hotlist and action list verification and Example calculation for an entitlement hotlist inventory .
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module



	Service Operator Back-Office Main Module Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle verification request for entitlement hotlist updated via increments
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Process instance ID : ProcessInstanceId Updated entitlement hotlist : EntitlementHotlist
Outputs	
Error Cases	
Activity Diagram	BO-Main::Verify entitlement hotlist updated via increments

5.2.20 Update organisation hotlist inventory

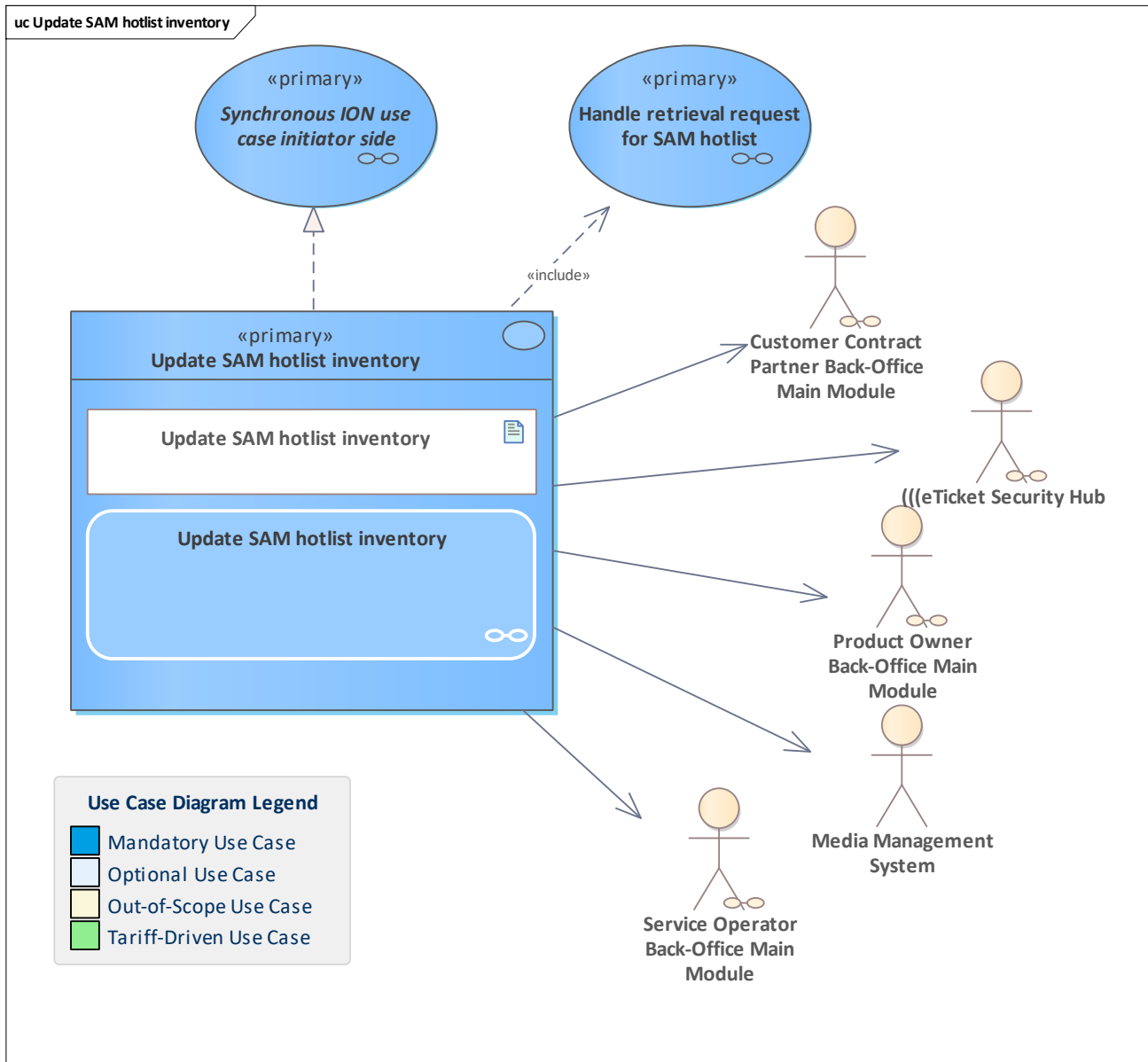


Use Case	Update organisation hotlist inventory
Description	The SO, CCP, PO and the scheme manager's ESH want to update their organisation hotlist inventory by retrieving the current organisation hotlist from the hotlist service system and processing it into their organisation hotlist inventory.
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module Product Owner Back-Office Main Module ((eTicket Security Hub
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases	Handle retrieval request for organisation hotlist



(Includes)	
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	
Outputs	Updated organisation hotlist inventory
Error Cases	
Activity Diagram	Update organisation hotlist inventory

5.2.21 Update SAM hotlist inventory

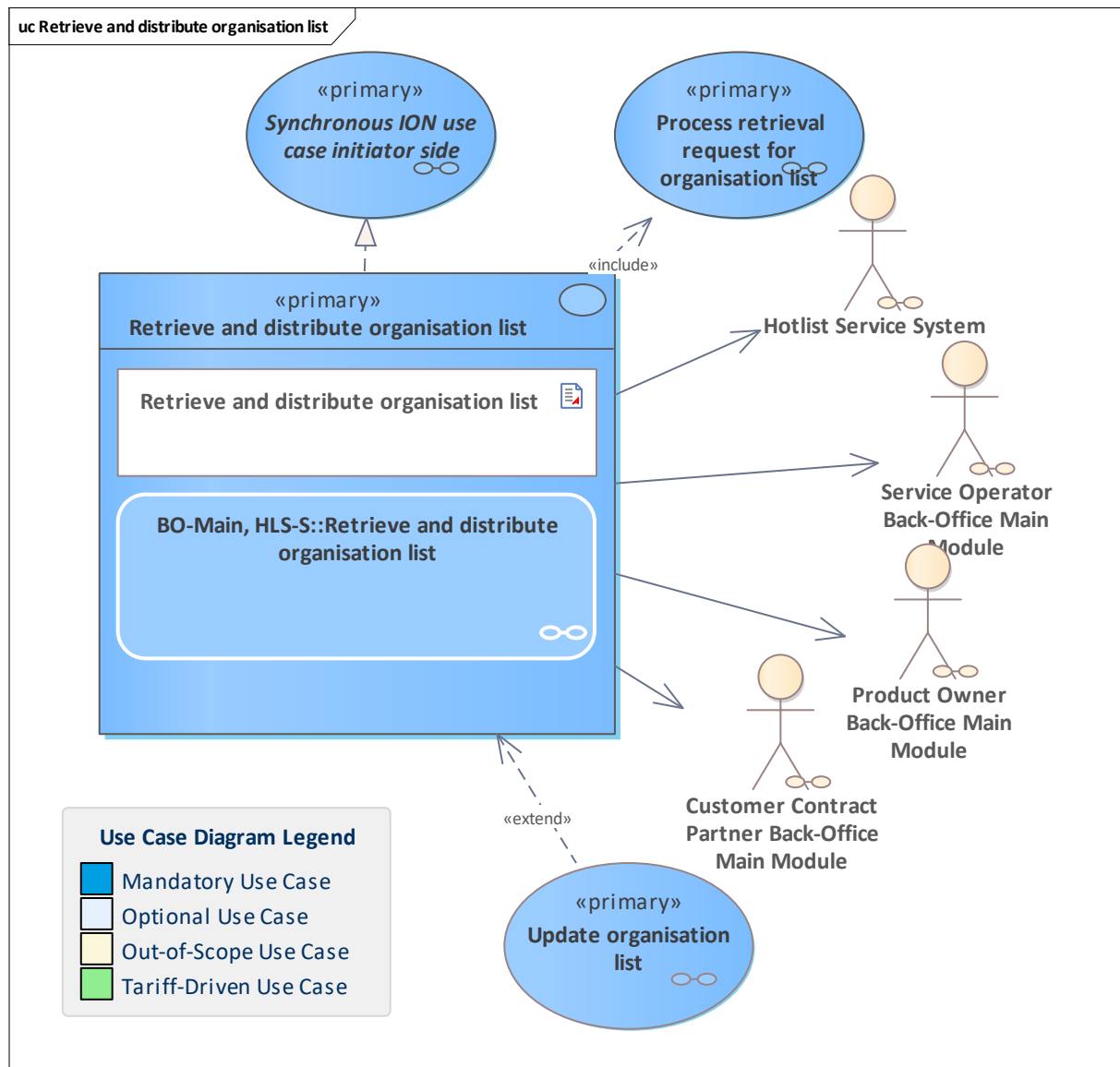


Use Case	<u>Update SAM hotlist inventory</u>
Description	The SO, CCP, PO and the scheme manager's ESH and MMS want to update their SAM hotlist inventory by retrieving the current SAM hotlist from the hotlist service system and processing it into the SAM hotlist inventory.
Initiating Actor	
Reacting Actor	<u>Customer Contract Partner Back-Office Main Module</u> <u>Service Operator Back-Office Main Module</u> <u>Product Owner Back-Office Main Module</u> <u>Media Management System</u> <u>(((eTicket Security Hub</u>
Preconditions	
Postconditions	
Linked Use Cases	



(Extended By)	
Linked Use Cases (Includes)	Handle retrieval request for SAM hotlist
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	
Outputs	Updated SAM hotlist inventory
Error Cases	
Activity Diagram	Update SAM hotlist inventory

5.2.22 Retrieve and distribute organisation list



Use Case	Retrieve and distribute organisation list
Description	The registrar, as part of the scheme manager, provides a list of organisations. This list can be retrieved by all participants daily. An organisation list is distributed to the terminals by CCP and SO. Please note that the list does not involve any IP addresses of the organisations due to data protection.
Initiating Actor	
Reacting Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module Product Owner Back-Office Main Module Hotlist Service System
Preconditions	
Postconditions	
Linked Use Cases	Update organisation list



(Extended By)	
Linked Use Cases (Includes)	Process retrieval request for organisation list
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	BO-Main, HLS-S::Retrieve and distribute organisation list



6 Basic Bundle PO-System

Functionality bundle that covers all the basic use cases required for the PO back-office system.

6.1 Overview

[Handle entitlement unblocked notification from product perspective](#)

[Handle entitlement blocked notification from product perspective](#)

[Update hotlist inventory from product perspective](#)

[Get unclaimed list information](#)

[Get product acceptance configuration list](#)

[Add product acceptance entry to hotlist configuration](#)

[Remove product acceptance entry from hotlist configuration](#)

[Remove product acceptance from participants](#)

[Monitor SAMs from product perspective](#)

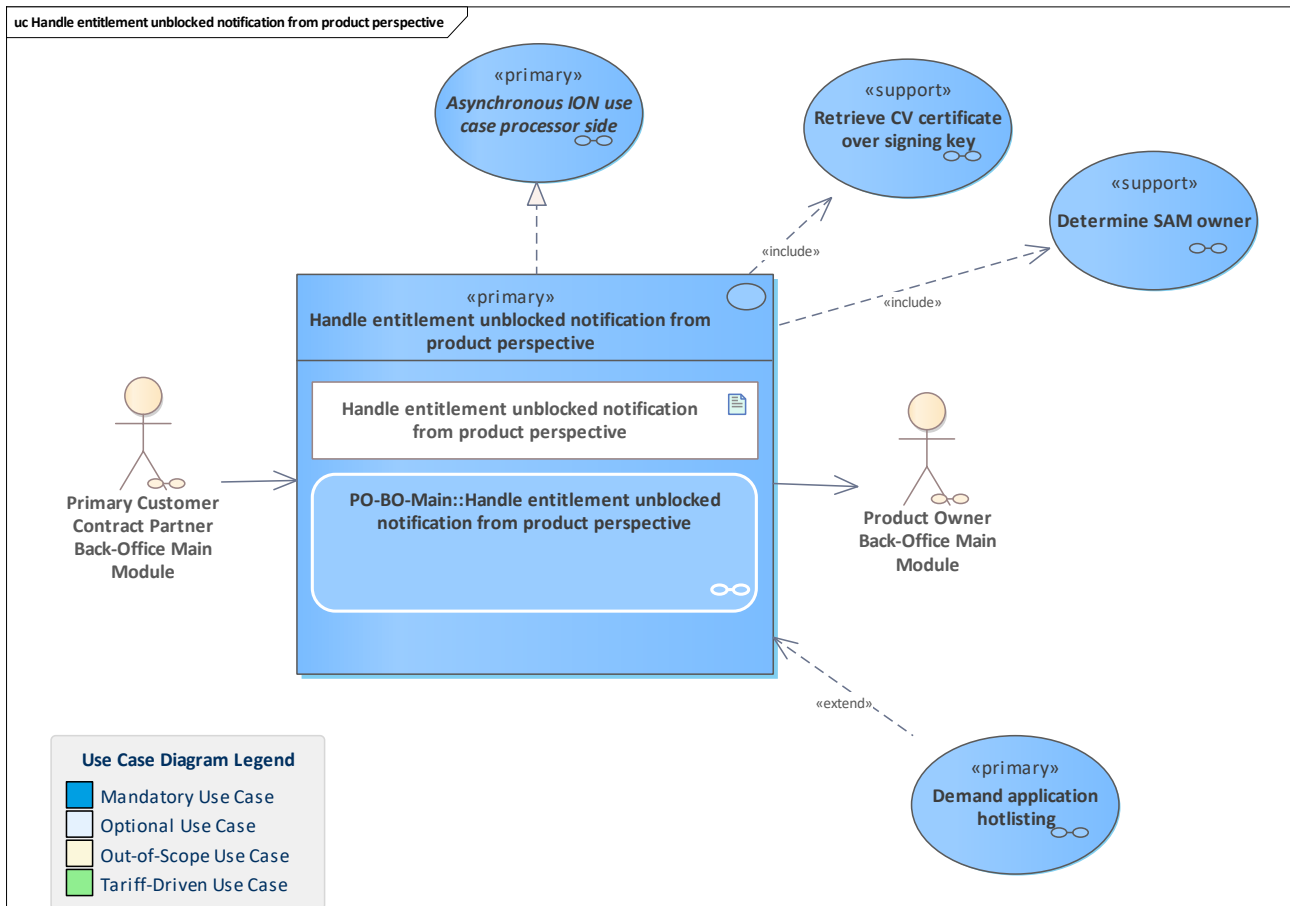
[Analyse entitlement history from product perspective](#)

[Resolve notifications with timeout warnings](#)

[Check entitlement notifications against issuance notification from product perspective](#)

6.2 Use Cases

6.2.1 Handle entitlement unblocked notification from product perspective

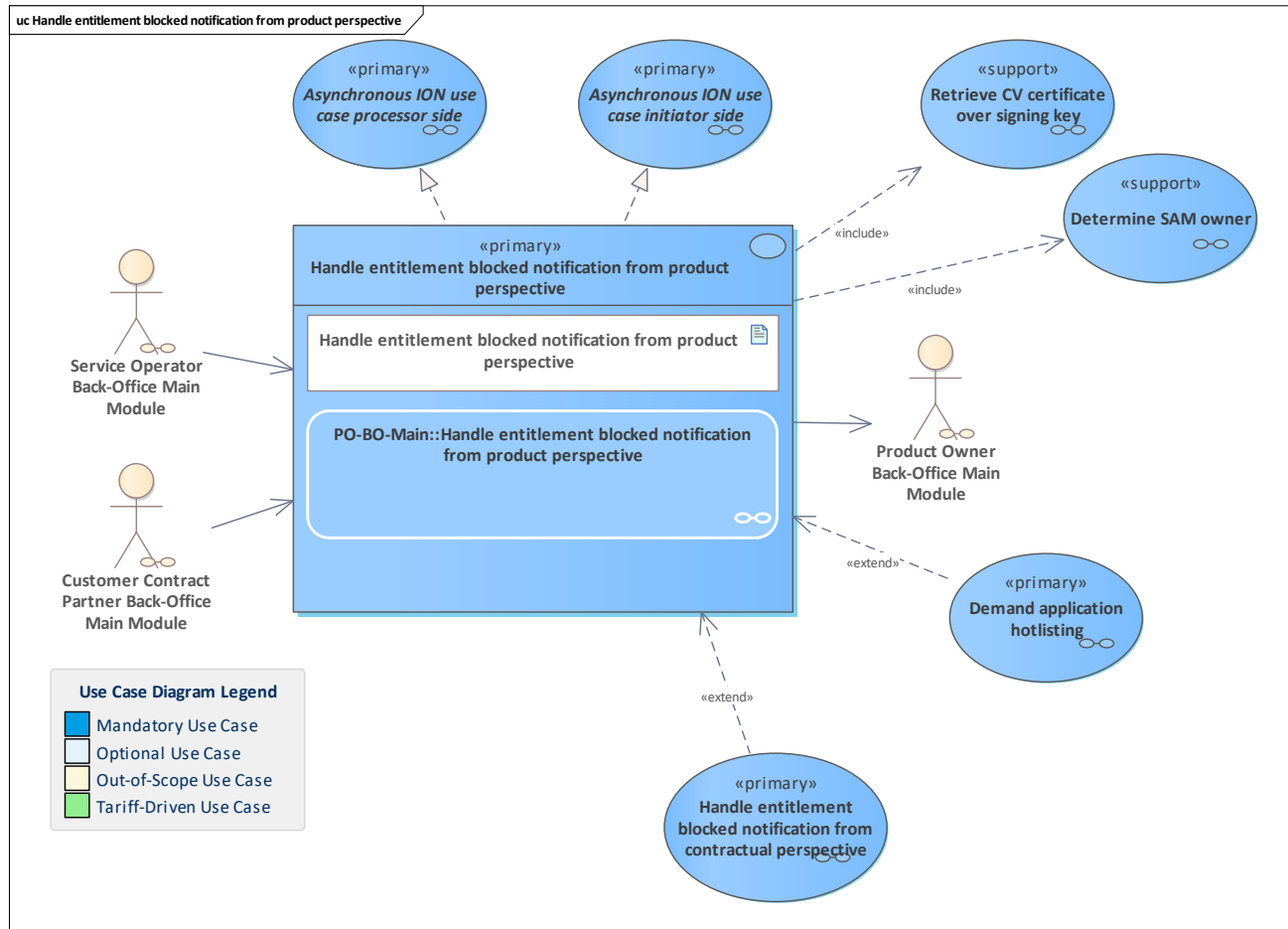


Use Case	Handle entitlement unblocked notification from product perspective
Description	<p>Handle an entitlement unblocked notification from the product perspective.</p> <p>The entitlement unblocked notification is received by the PO. The PO registers entitlement unblocked notification and does its checks and monitoring from the product perspective regarding the correct execution of unblocking. In this context, the signature of the embedded attestation is verified and the SAM owner of the SAM that performed the action is determined.</p> <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case as initiator due to the asynchronous call to the pCCP.</p>
Initiating Actor	Primary Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement unblocked : notifyEntitlementUnblocked



Outputs	<u>Notify entitlement unblocked response :</u> <u>notifyEntitlementUnblockedResponse</u>
Error Cases	<u>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</u> <u>E CO BINARY STRUCTURE CANNOT BE PROCESSED</u> <u>E CO WRONG ATTESTATION IN NOTIFICATION</u> <u>E CO WRONG SECURITY LEVEL</u> <u>E CO DUPLICATE UM ATTESTATION</u> <u>Notify entitlement unblocked exception :</u> <u>notifyEntitlementUnblockedException</u>
Activity Diagram	<u>PO-BO-Main::Handle entitlement unblocked notification from</u> <u>product perspective</u>

6.2.2 Handle entitlement blocked notification from product perspective

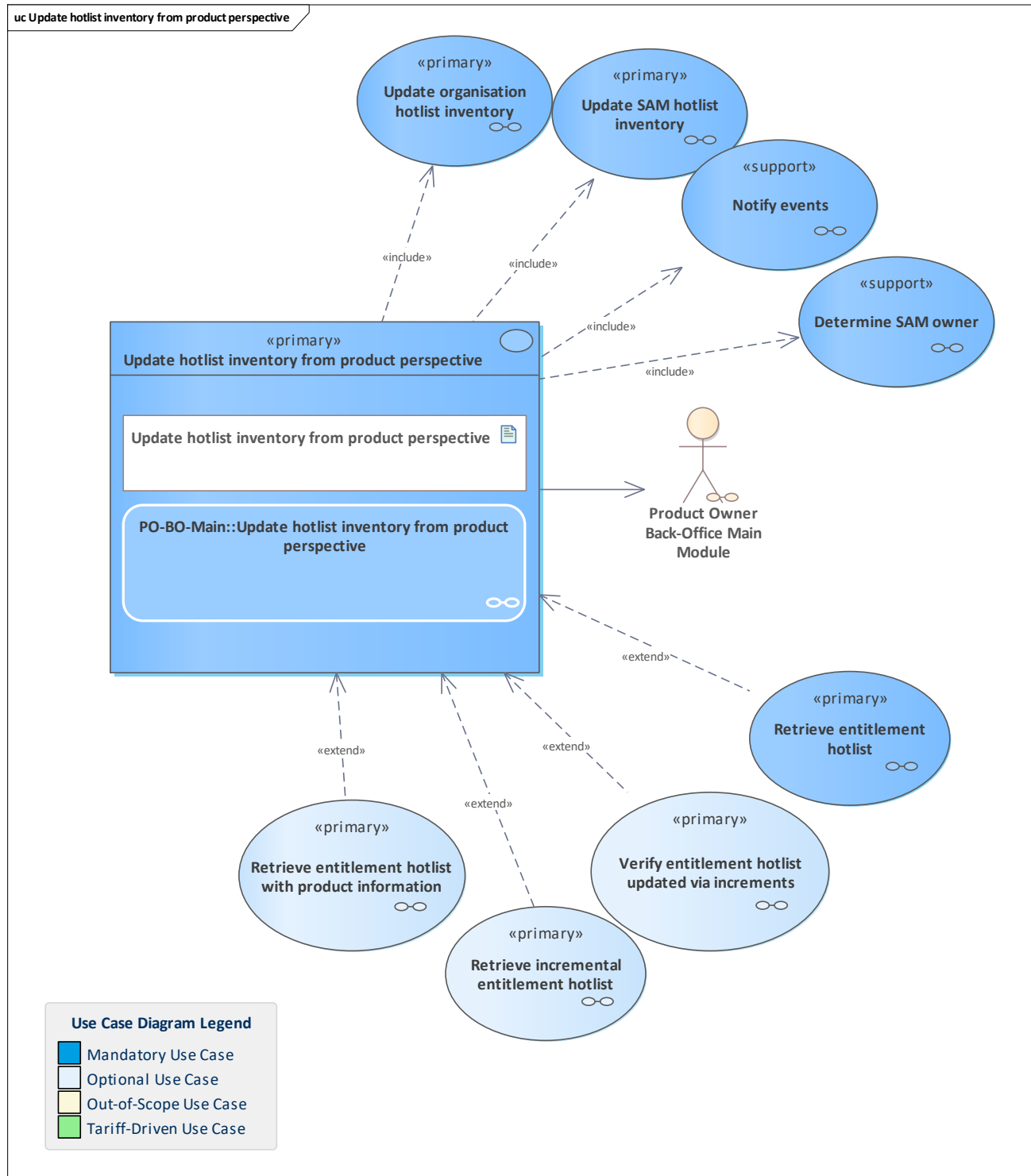


Use Case Description	Handle entitlement blocked notification from product perspective The entitlement blocked notification is received by the PO. The PO registers the entitlement blocked notification and performs its checks and monitoring from the product perspective regarding the correct execution of blocking. In this context, the signature of the blocking attestation is verified and the SAM owner of the SAM that performed the blocking is determined. <ul style="list-style-type: none"> if the sender is a sCCP or SO, forward the notification to pCCP if the sender is the pCCP, do not forward the notification. In this case, the current use case is not an Asynchronous ION use case initiator side.
Initiating Actor	Service Operator Back-Office Main Module Secondary Customer Contract Partner Back-Office Main Module Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle entitlement blocked notification from contractual perspective / Demand application hotlisting / Demand application hotlisting



Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side
Base Activity	
Inputs	Notify entitlement blocked : notifyEntitlementBlocked
Outputs	Notify entitlement blocked response : notifyEntitlementBlockedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notfiy entitlement blocked exception : notifyEntitlementBlockedException
Activity Diagram	PO-BO-Main::Handle entitlement blocked notification from product perspective

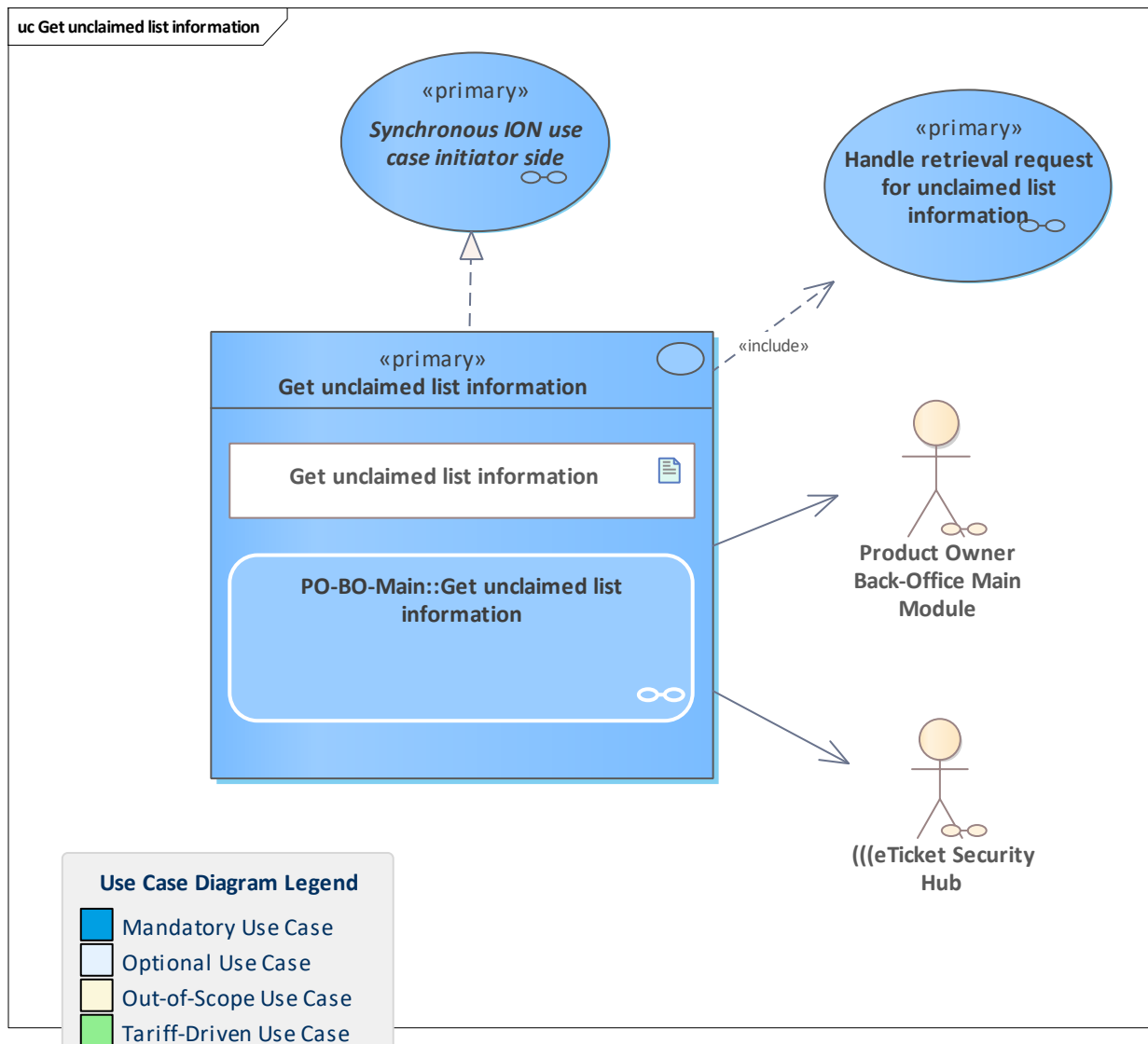
6.2.3 Update hotlist inventory from product perspective



Use Case	<u>Update hotlist inventory from product perspective</u>
Description	A product owner wants to retrieve the currently available hotlists and analyse them by checking if hotlisted elements are already

	<p>blocked in its inventory. The available hotlists are:</p> <ul style="list-style-type: none"> Entitlement hotlist: <ul style="list-style-type: none"> either total entitlement hotlist or incremental entitlement hotlist or total entitlement hotlist with product information SAM hotlist Organisation hotlist
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Retrieve entitlement hotlist / Retrieve entitlement hotlist with product information / Verify entitlement hotlist updated via increments / Retrieve incremental entitlement hotlist
Linked Use Cases (Includes)	Update organisation hotlist inventory / Update SAM hotlist inventory / Notify events / Determine SAM owner
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Update hotlist inventory from product perspective

6.2.4 Get unclaimed list information

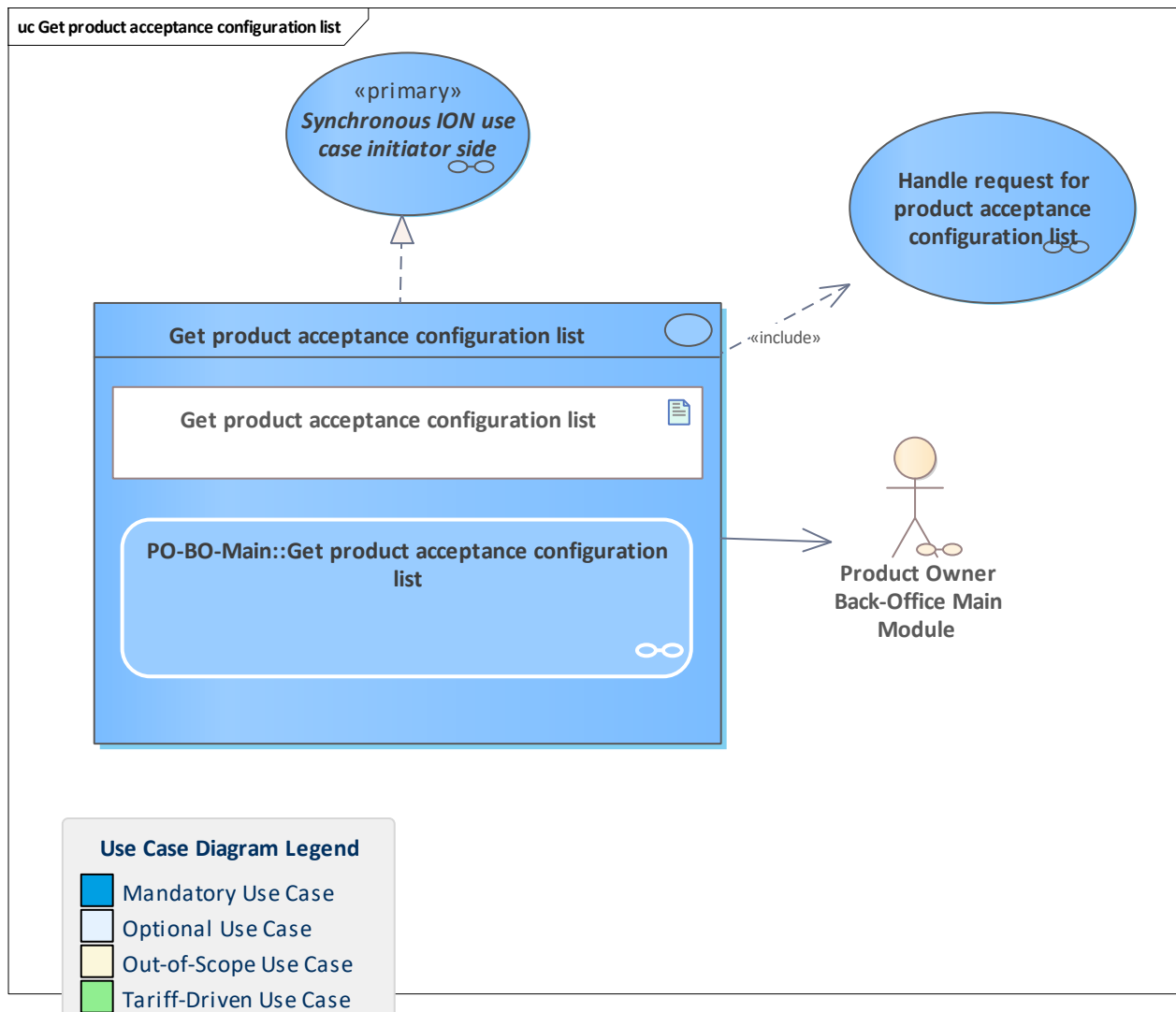


Use Case	<u>Get unclaimed list information</u>
Description	<p>Use case that allows the PO to monitor the collection behaviour of its partner companies with the role SO and CCP. The scheme manager monitors all SO, CCP and PO organisations. The requestor gets the unclaimed list information from the hotlist service system for a period between the passed list cycle and the current list cycle. This information is registered and can be gathered for a regular report.</p> <p>The following list types are expected for the organisation's roles:</p> <ul style="list-style-type: none"> CCP and SO: application-, entitlement-, SAM-, organisation- and authentication-key-hotlist PO (included if the actor is the scheme manager): entitlement-, SAM- and organisation-hotlist
Initiating Actor	
Reacting Actor	<u>Product Owner Back-Office Main Module</u> <u>(((eTicket Security Hub</u>



Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle retrieval request for unclaimed list information
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	List cycle number : ListCycleNumber
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Get unclaimed list information

6.2.5 Get product acceptance configuration list

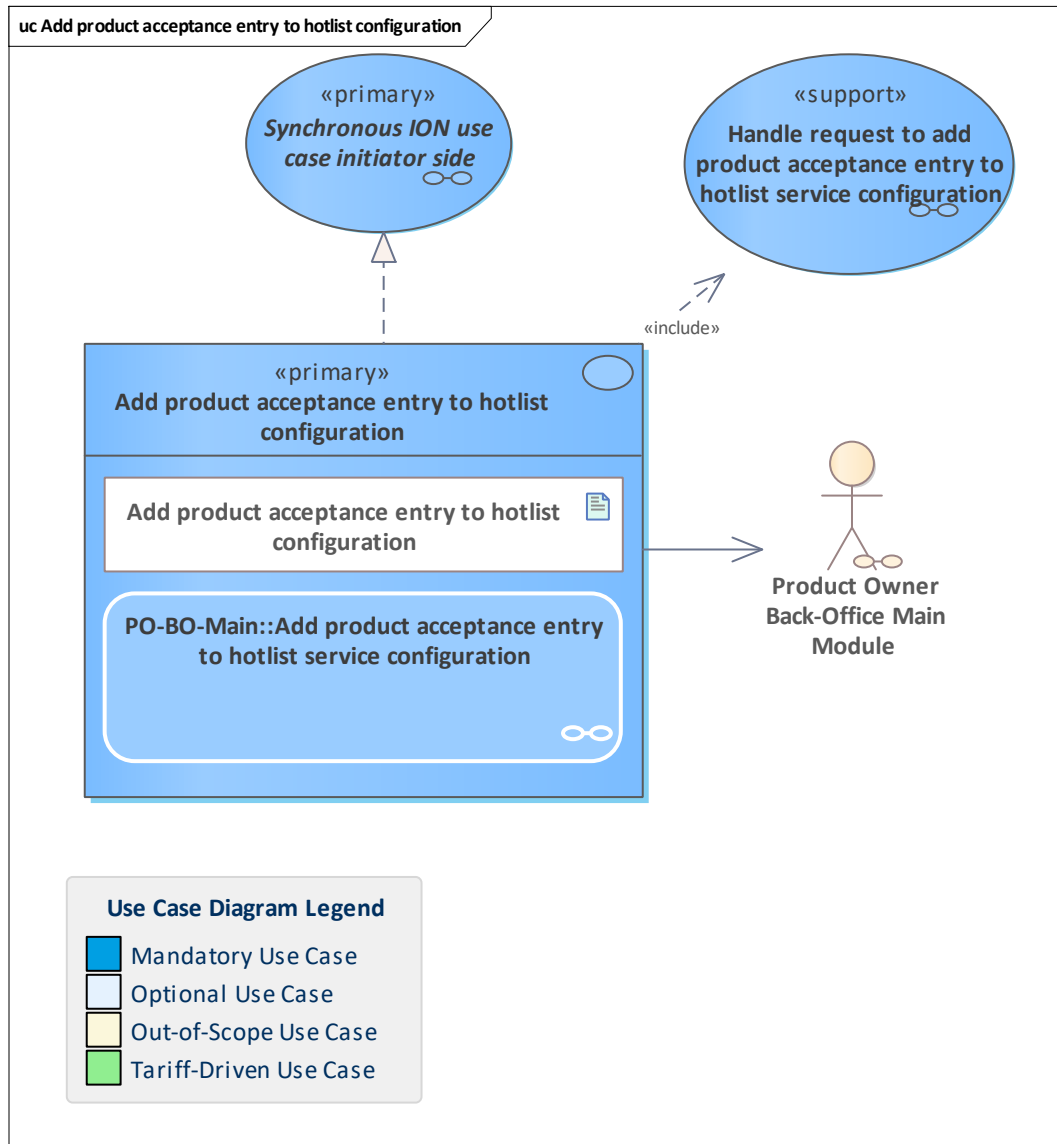


Use Case	Get product acceptance configuration list
Description	The PO retrieves its product acceptance configuration from the hotlist service system as a compressed list of product acceptance entries, where each entry contains the ID of the accepting organisation (SO or CCP), the product owner ID and, optionally, the product number. If the product number is omitted, the organisation accepts all products of the PO.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle request for product acceptance configuration list
Linked Use Cases (Realises)	Synchronous ION use case / Synchronous ION use case initiator side



Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Get product acceptance configuration list

6.2.6 Add product acceptance entry to hotlist configuration

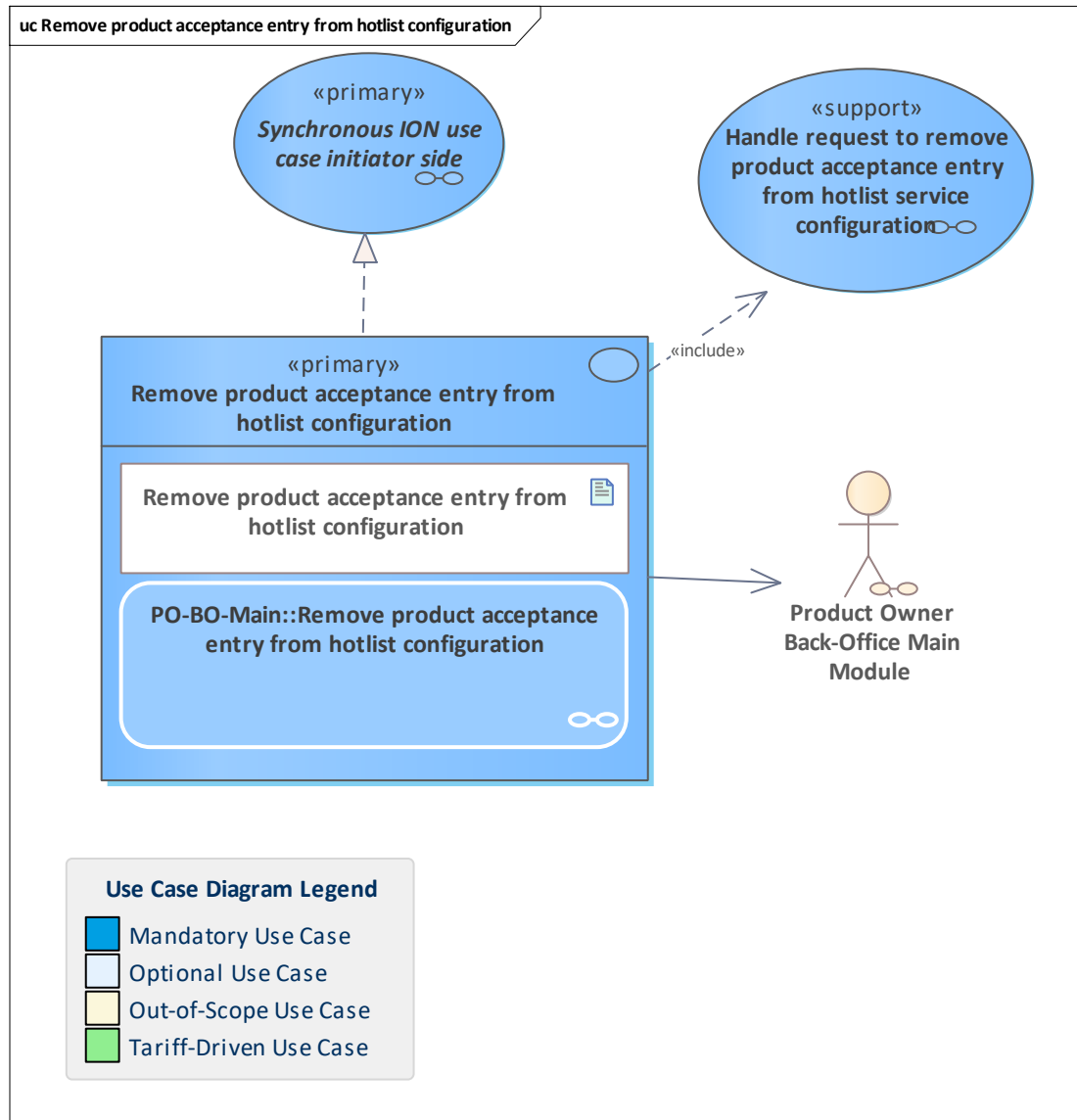


Use Case	<u>Add product acceptance entry to hotlist configuration</u>
Description	<p>To facilitate specific generation of hotlists, the hotlist service system uses information provided by the PO about which organisation accepts which products.</p> <p>For each of the products accepted by a certain organisation, the PO sends an acceptance request containing the accepting organisation ID, the PO's organisation ID and a product number. If the organisation accepts all products of a PO, the product number is omitted in the request.</p> <p>Please note that interoperable products are not allowed to be added to the configuration list.</p>
Initiating Actor	
Reacting Actor	<u>Product Owner Back-Office Main Module</u>



Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle request to add product acceptance entry to hotlist service configuration
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Product owner ID : OrganisationId Product number : ProductNumber Accepting Organisation ID : OrganisationId
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Add product acceptance entry to hotlist service configuration

6.2.7 Remove product acceptance entry from hotlist configuration

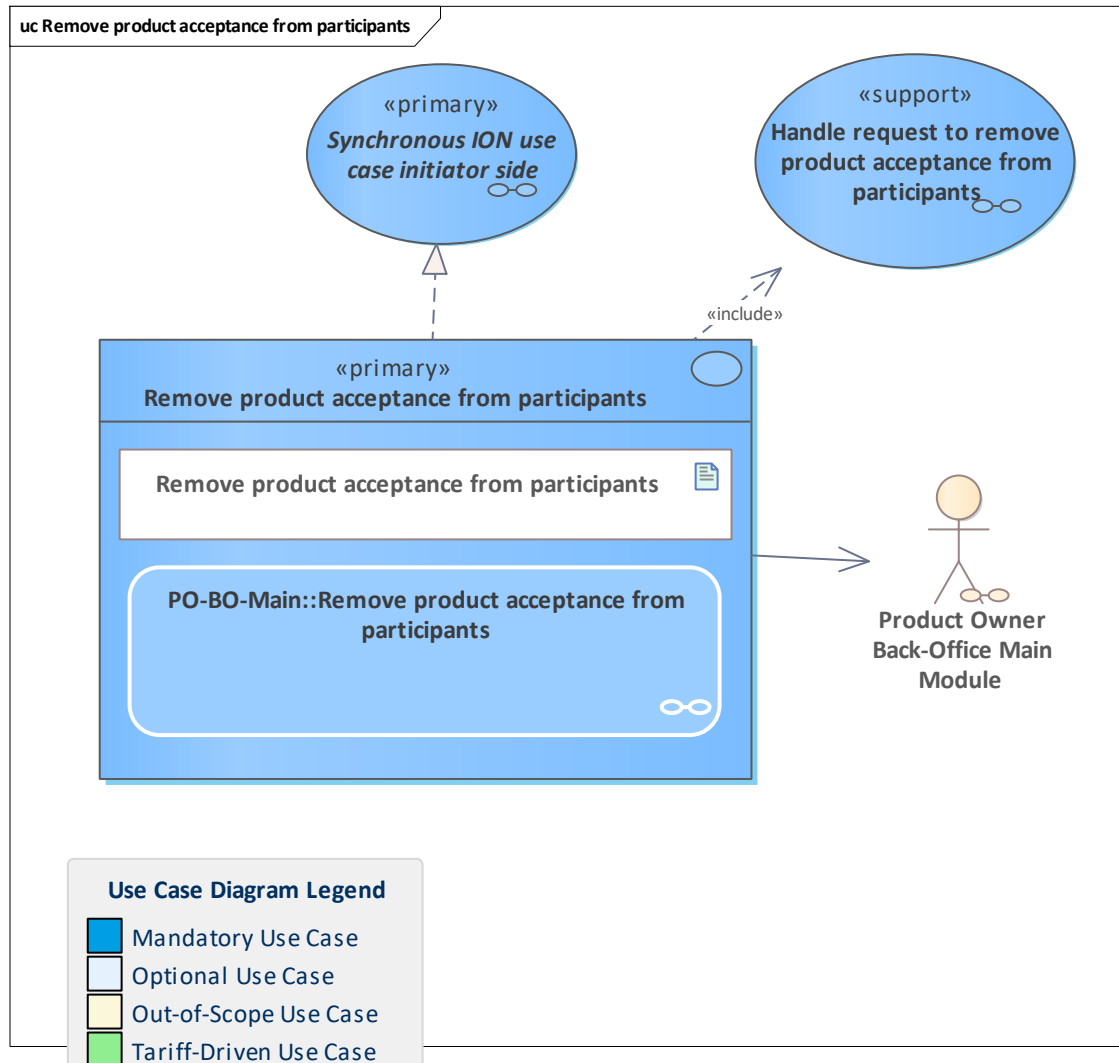


Use Case	Remove product acceptance entry from hotlist configuration
Description	<p>To facilitate specific generation of hotlists, the hotlist service system uses information provided by the PO about which organisation accepts which products.</p> <p>If a product is no longer accepted by an organisation, the PO sends an acceptance removal request containing the accepting organisation ID and a product number. If all products of the PO should be removed from the hotlist configuration for a certain organisation, the product number is omitted in the request.</p> <p>To remove a product from all accepting organisations, for example, if the product no longer exists, there is a specific use case to Remove product acceptance from participants.</p> <p>Please note that, as interoperable products are not a part of the</p>



	configuration list, a removal request for an interoperable product is not allowed.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle request to remove product acceptance entry from hotlist service configuration
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Product number : ProductNumber Accepting Organisation ID : OrganisationId
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Remove product acceptance entry from hotlist configuration

6.2.8 Remove product acceptance from participants

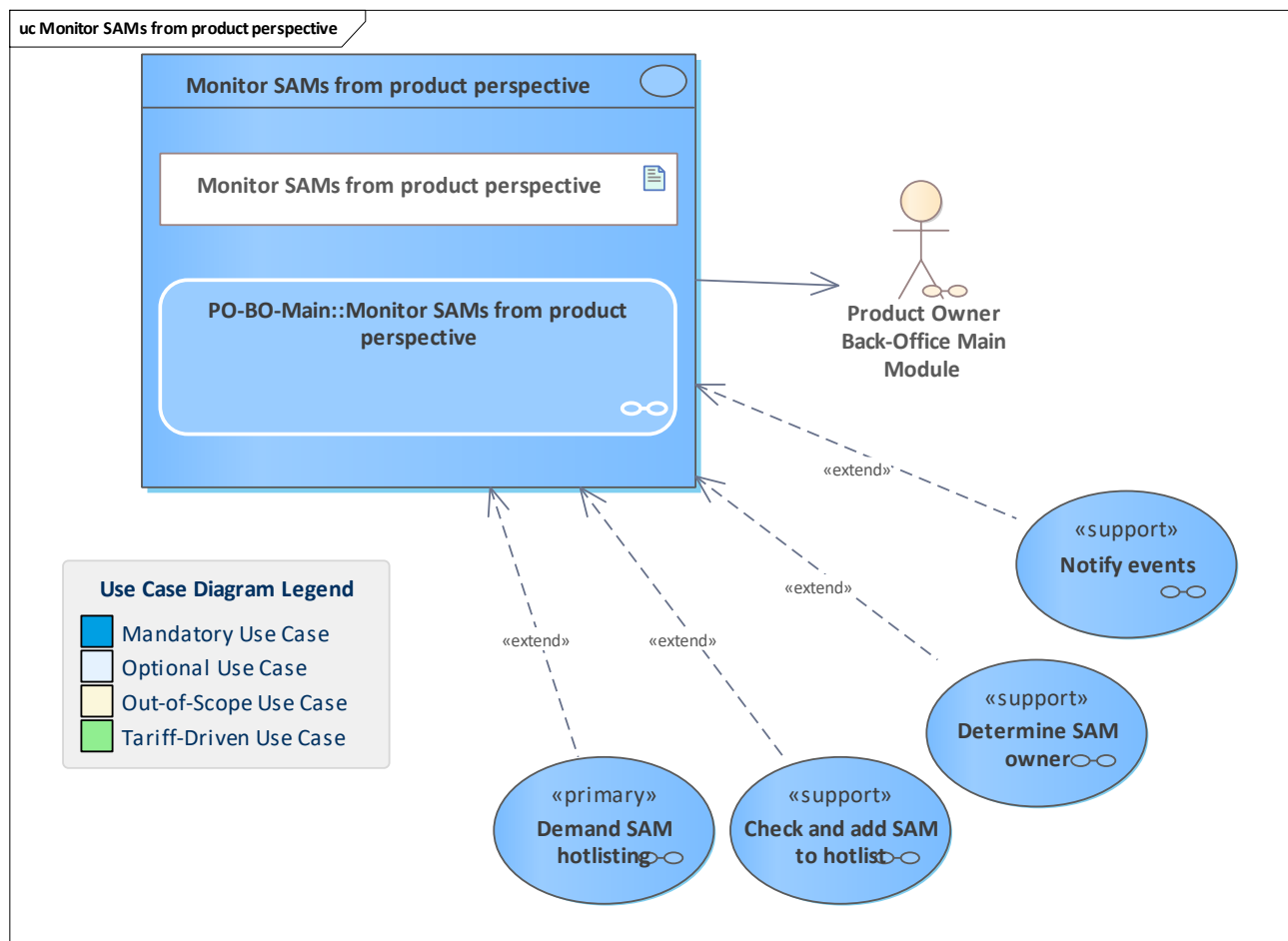


Use Case	Remove product acceptance from participants
Description	A product owner sends a request to remove product acceptance for one of its products from all organisations. The product number is mandatory in this request.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle request to remove product acceptance from participants
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	
Inputs	Product expiry date : ZonedDate Product number : ProductNumber



Outputs	
Error Cases	
Activity Diagram	<u>PO-BO-Main::Remove product acceptance from participants</u>

6.2.9 Monitor SAMs from product perspective

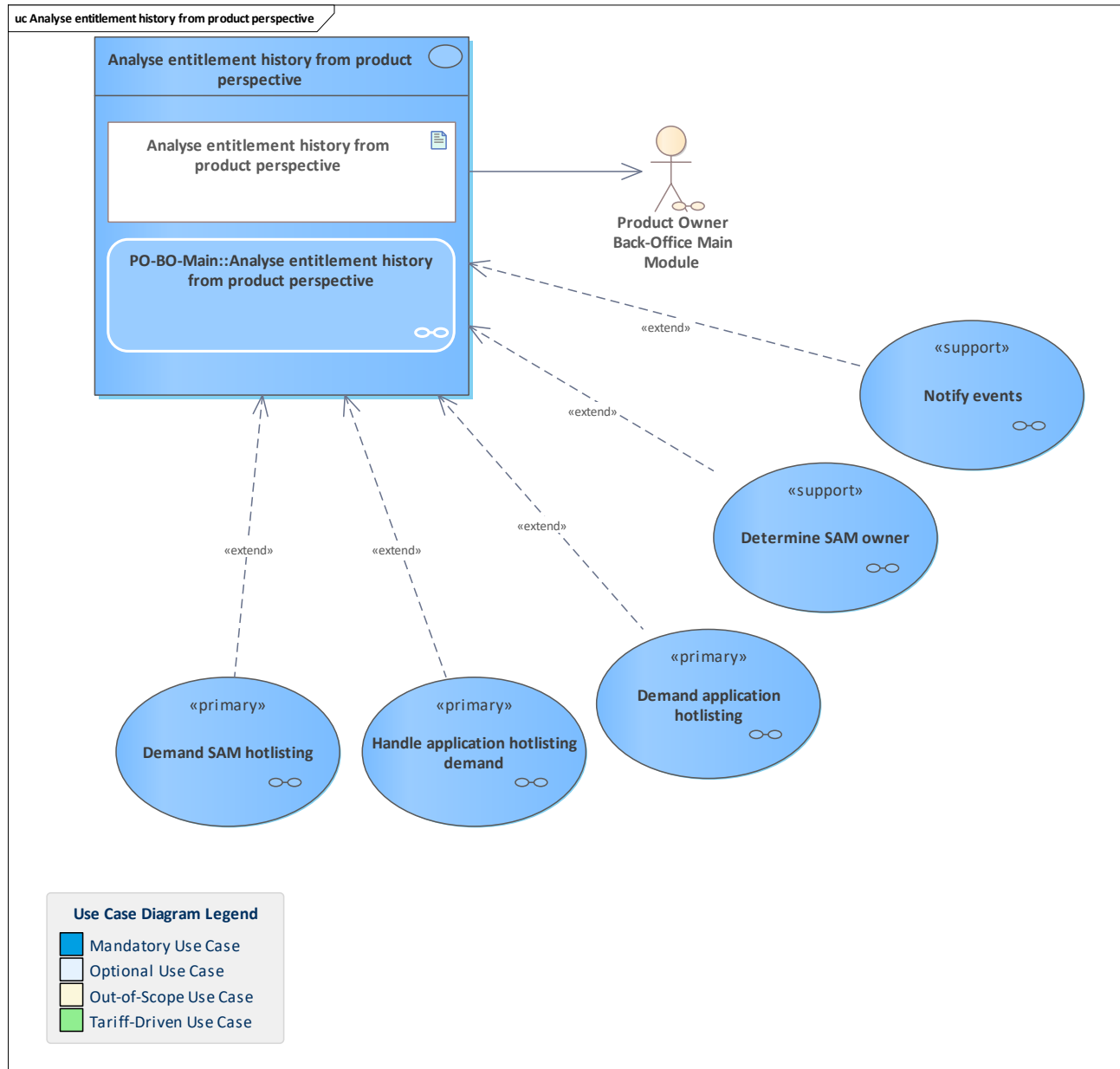


Use Case	Monitor SAMs from product perspective
Description	The product owner has to monitor the usage of the product owner tokens configured into SAMs. Every issue found shall only be reported once.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Notify events / Determine SAM owner / Check and add SAM to hotlist / Demand SAM hotlisting
Linked Use Cases (Includes)	Determine SAM owner
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	



Activity Diagram	PO-BO-Main::Monitor SAMs from product perspective
-------------------------	---

6.2.10 Analyse entitlement history from product perspective

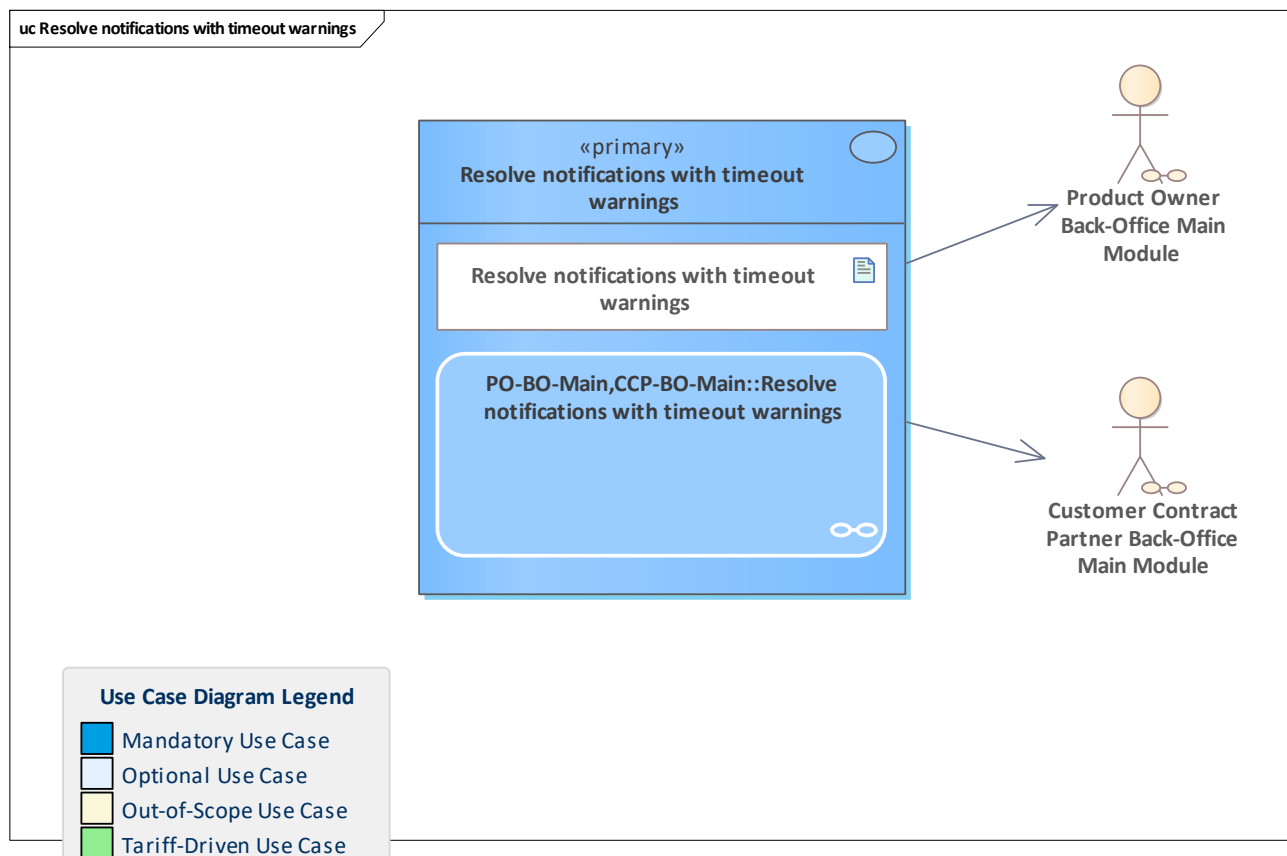


Use Case	Analyse entitlement history from product perspective
Description	Analyse the entitlement history for inconsistencies or gaps in the usage history. PO and pCCP both need to have all information about their entitlements, which means they need to have all notifications regarding them. Every issue found shall only be reported once.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases	Notify events / Determine SAM owner / Demand application



(Extended By)	hotlisting / Handle application hotlisting demand / Demand SAM hotlisting
Linked Use Cases (Includes)	Handle application hotlisting demand / Determine SAM owner
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Analyse entitlement history from product perspective

6.2.11 Resolve notifications with timeout warnings

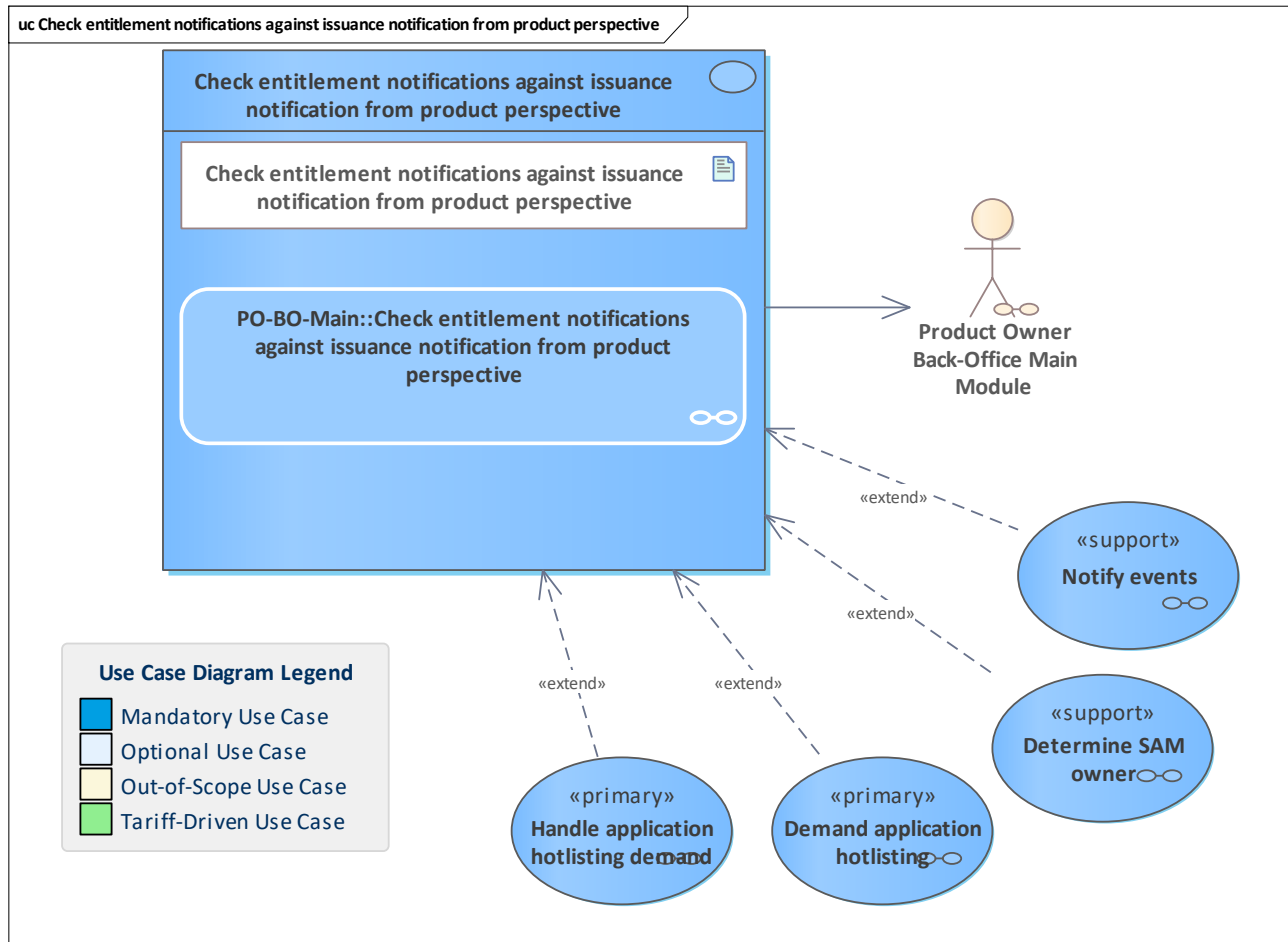


Use Case	Resolve notifications with timeout warnings
Description	<p>The system periodically tries to resolve notifications with timeout warnings. A timeout warning can be resolved positively (determining that the action permanently changed the user medium) or negatively (determining the user medium performed a roll-back and was unchanged).</p> <p>When a notification has its timeout warning positively resolved, it can enter regular system processing.</p>
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module Customer Contract Partner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	



Error Cases	
Activity Diagram	PO-BO-Main,CCP-BO-Main::Resolve notifications with timeout warnings

6.2.12 Check entitlement notifications against issuance notification from product perspective



Use Case	Check entitlement notifications against issuance notification from product perspective
Description	Non-issuing entitlement notifications need to be checked against details that are only contained in the issuance notification, e.g., the entitlement validity period. This also makes sure that every entitlement seen live is known to the system / has been reported as issued.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Notify events / Handle application hotlisting demand / Determine SAM owner
Linked Use Cases (Includes)	Handle application hotlisting demand
Linked Use Cases (Realises)	
Base Activity	
Inputs	



Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Check entitlement notifications against issuance notification from product perspective

7 Electronic Ticket Bundle PO-System

Functionality bundle that covers the use cases for a PO back-office system with electronic tickets placed on a user medium with application (e.g. chip card).

7.1 Overview

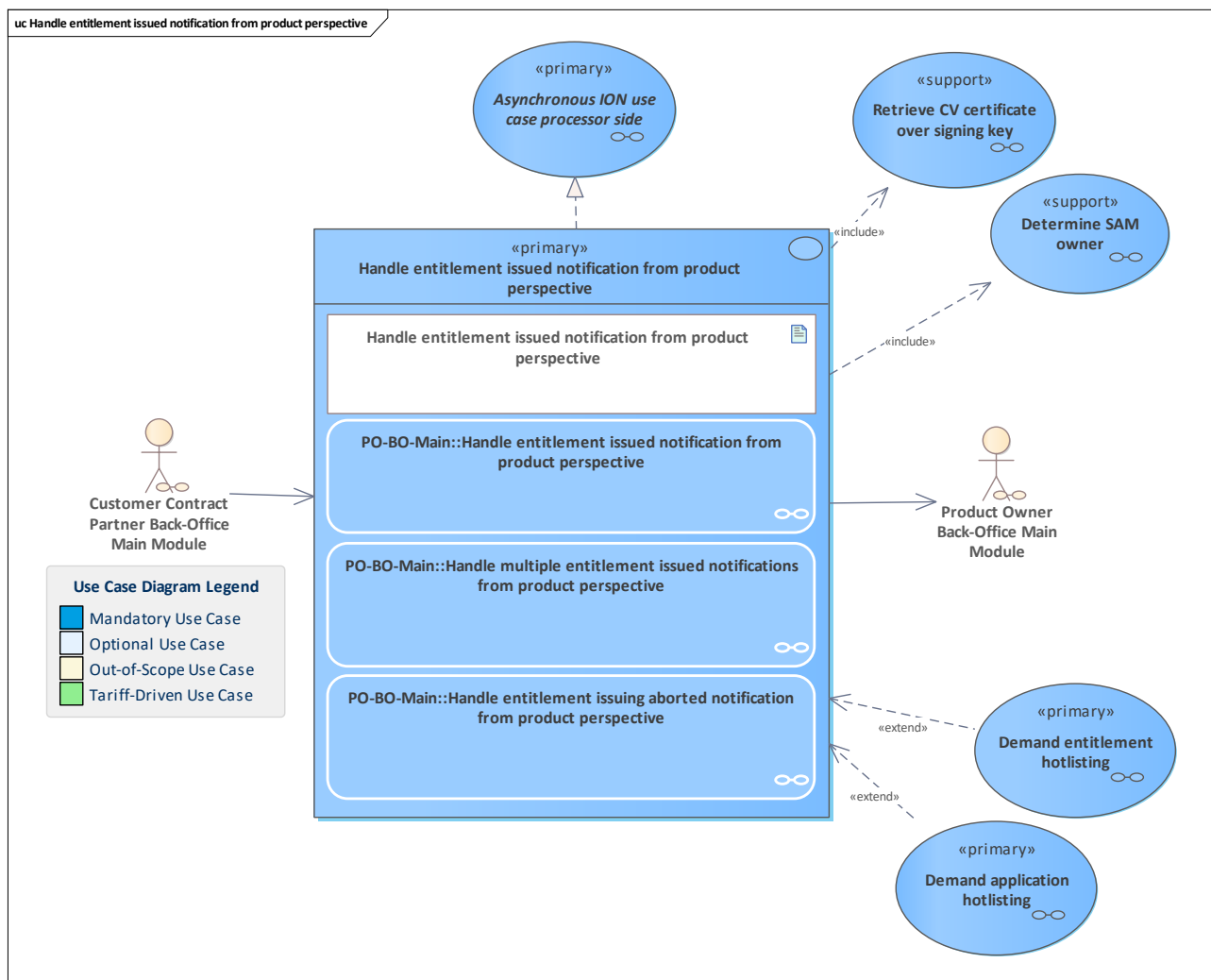
Handle entitlement issued notification from product perspective

Handle entitlement terminated notification from product perspective

Handle entitlement inspected notification from product perspective

7.2 Use Cases

7.2.1 Handle entitlement issued notification from product perspective

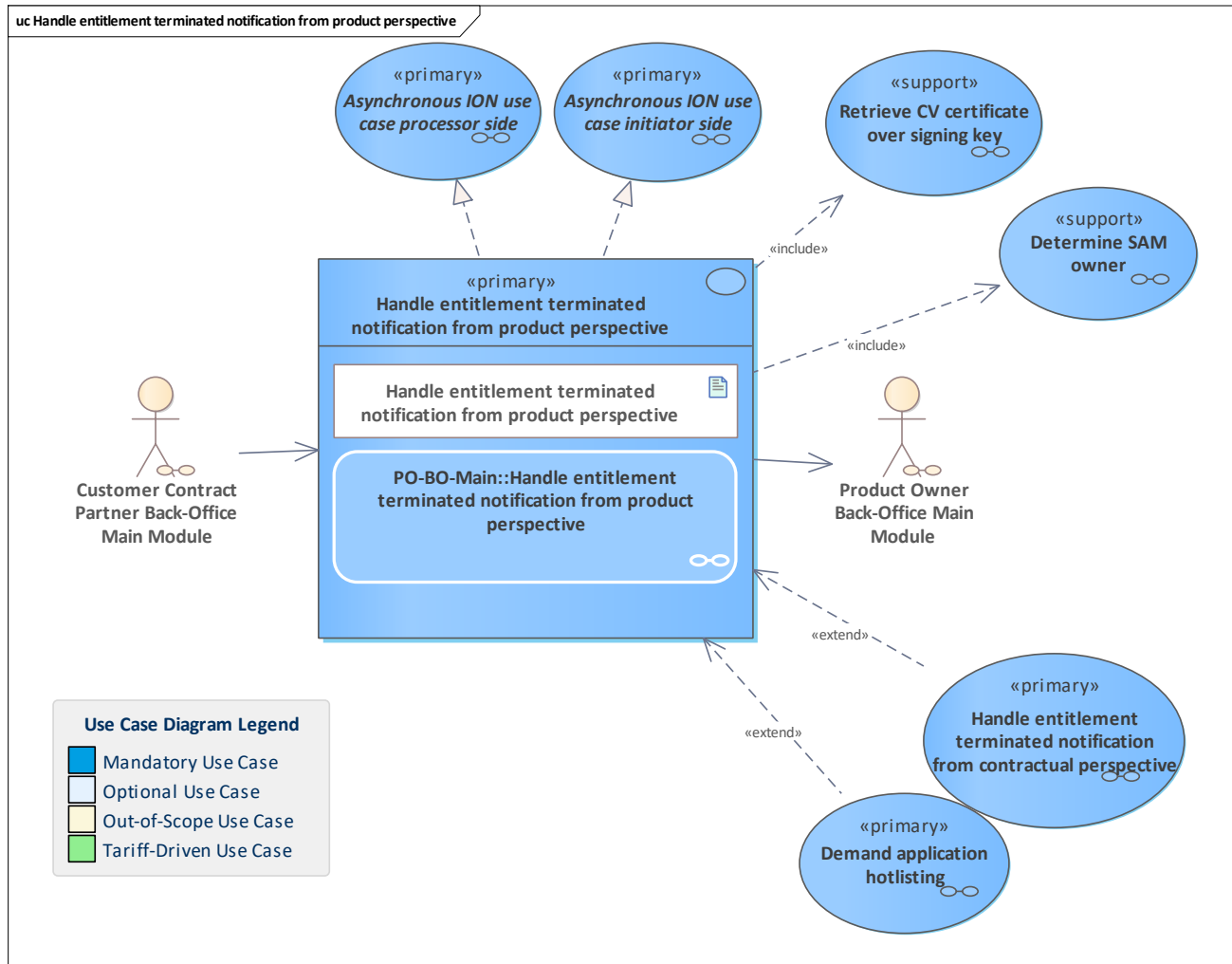


Description	<p>The PO back-office system receives the notification about an entitlement issuance and handles it from the product perspective. It registers the notification and does checks and monitoring. In the case of a list of notifications, this list is processed instead of a single notification.</p> <p>In the case of an abortion notification, the PO system has to register the SAM and product issuance counter for consistent monitoring.</p> <p>Note: the application instance ID of the user medium the entitlement was issued to can be uniquely identified via the request field <i>umAppInstanceId</i>, which is part of the SignedEntitlementIssuedAttestation that is contained in the EntitlementIssuedNotification.</p> <p>Note: for this use case, the issuance was not triggered by an action order.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand entitlement hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement issued : notifyEntitlementIssued
Outputs	Notify entitlement issued response : notifyEntitlementIssuedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement issued exception : notifyEntitlementIssuedException
Activity Diagram	PO-BO-Main::Handle entitlement issued notification from product perspective
Alternative 1	
Inputs	Process entitlement issued notification list : processEntitlementIssuedNotificationList
Outputs	Process entitlement issued notification list response : processEntitlementIssuedNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process entitlement issued notification list exception : processEntitlementIssuedNotificationListException
Activity Diagram	PO-BO-Main::Handle multiple entitlement issued notifications from product perspective
Alternative 2	



Inputs	Notify entitlement issuing aborted : notifyEntitlementIssuingAborted
Outputs	Notify entitlement issuing aborted response : notifyEntitlementIssuingAbortedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO WRONG SECURITY LEVEL Notify entitlement issuing aborted exception : notifyEntitlementIssuingAbortedException
Activity Diagram	PO-BO-Main::Handle entitlement issuing aborted notification from product perspective

7.2.2 Handle entitlement terminated notification from product perspective

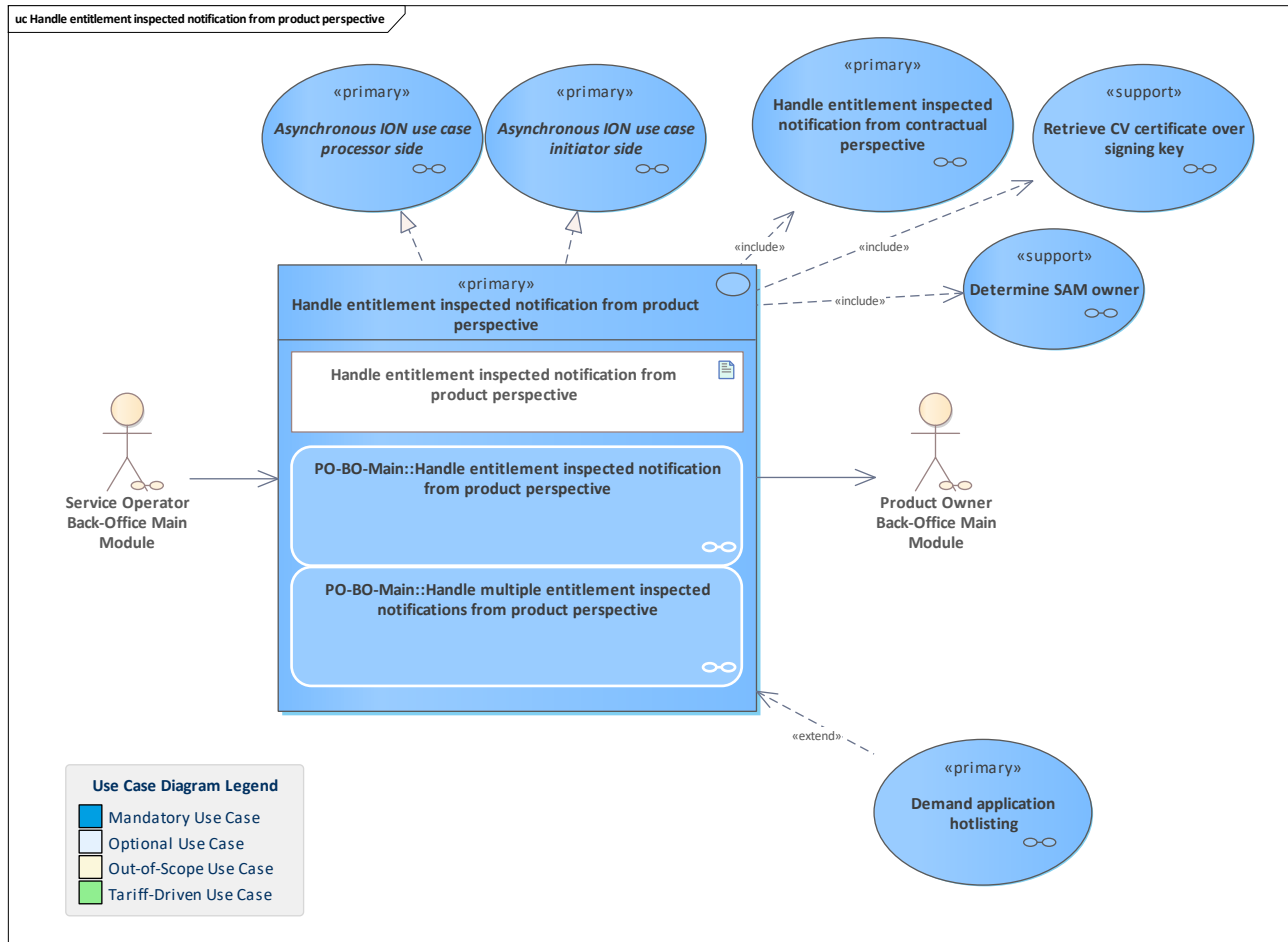


Use Case	Handle entitlement terminated notification from product perspective
Description	<p>Handle an entitlement terminated notification from the product perspective.</p> <p>The entitlement terminated notification is received by the PO. The PO registers the entitlement terminated notification and performs its checks and monitoring from the product perspective regarding the correct execution of the termination. In this context, the signature of the termination attestation is verified and the SAM owner of the SAM that performed the termination is determined.</p> <ul style="list-style-type: none"> if the sender is a sCCP: forward the notification to the pCCP if the sender is the pCCP, do not forward the notification. In this case, the current use case is not an asynchronous ION use case as initiator (Asynchronous ION use case initiator side). <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case</p>



	as initiator due to the asynchronous call to the pCCP.
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle entitlement terminated notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement terminated : notifyEntitlementTerminated
Outputs	Notify entitlement terminated response : notifyEntitlementTerminatedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement terminated exception : notifyEntitlementTerminatedException
Activity Diagram	PO-BO-Main::Handle entitlement terminated notification from product perspective

7.2.3 Handle entitlement inspected notification from product perspective



Use Case	Handle entitlement inspected notification from product perspective
Description	This use case describes the processing of the notification of an inspected entitlement in the PO back-office system. The SO sends the notification, the PO registers it and does its monitoring checks. After these checks, the notification is forwarded to the CCP back-office system. Furthermore, entitlement inspected notifications can be collected in the SO system and then sent as a list in a scheduled process to the PO.
Initiating Actor	Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle entitlement inspected notification from contractual perspective / Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side /



	<u>Asynchronous ION use case processor side</u>
Base Activity	
Inputs	<u>Notify entitlement inspected : notifyEntitlementInspected</u>
Outputs	<u>Notify entitlement inspected response : notifyEntitlementInspectedResponse</u>
Error Cases	<u>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</u> <u>E CO BINARY STRUCTURE CANNOT BE PROCESSED</u> <u>E CO WRONG ATTESTATION IN NOTIFICATION</u> <u>E CO WRONG SECURITY LEVEL</u> <u>E CO DUPLICATE UM ATTESTATION</u> <u>Notify entitlement inspected exception : notifyEntitlementInspectedException</u>
Activity Diagram	<u>PO-BO-Main::Handle entitlement inspected notification from product perspective</u>
Alternative 1	
Inputs	<u>Process entitlement inspected notification list : processEntitlementInspectedNotificationList</u>
Outputs	<u>Process entitlement inspected notification list response : processEntitlementInspectedNotificationListResponse</u>
Error Cases	<u>E CO CANNOT DECOMPRESS DATA</u> <u>E CO CANNOT DECOMPRESS DATA</u> <u>E CO WRONG ELEMENT IN COMPRESSED DATA</u> <u>Process entitlement inspected notification list exception : processEntitlementInspectedNotificationListException</u>
Activity Diagram	<u>PO-BO-Main::Handle multiple entitlement inspected notifications from product perspective</u>

8 Account-Based Payment Bundle PO-System

Functionality bundle that covers the use cases for a PO back-office system forming the basis for using the account-based payment method.

8.1 Overview

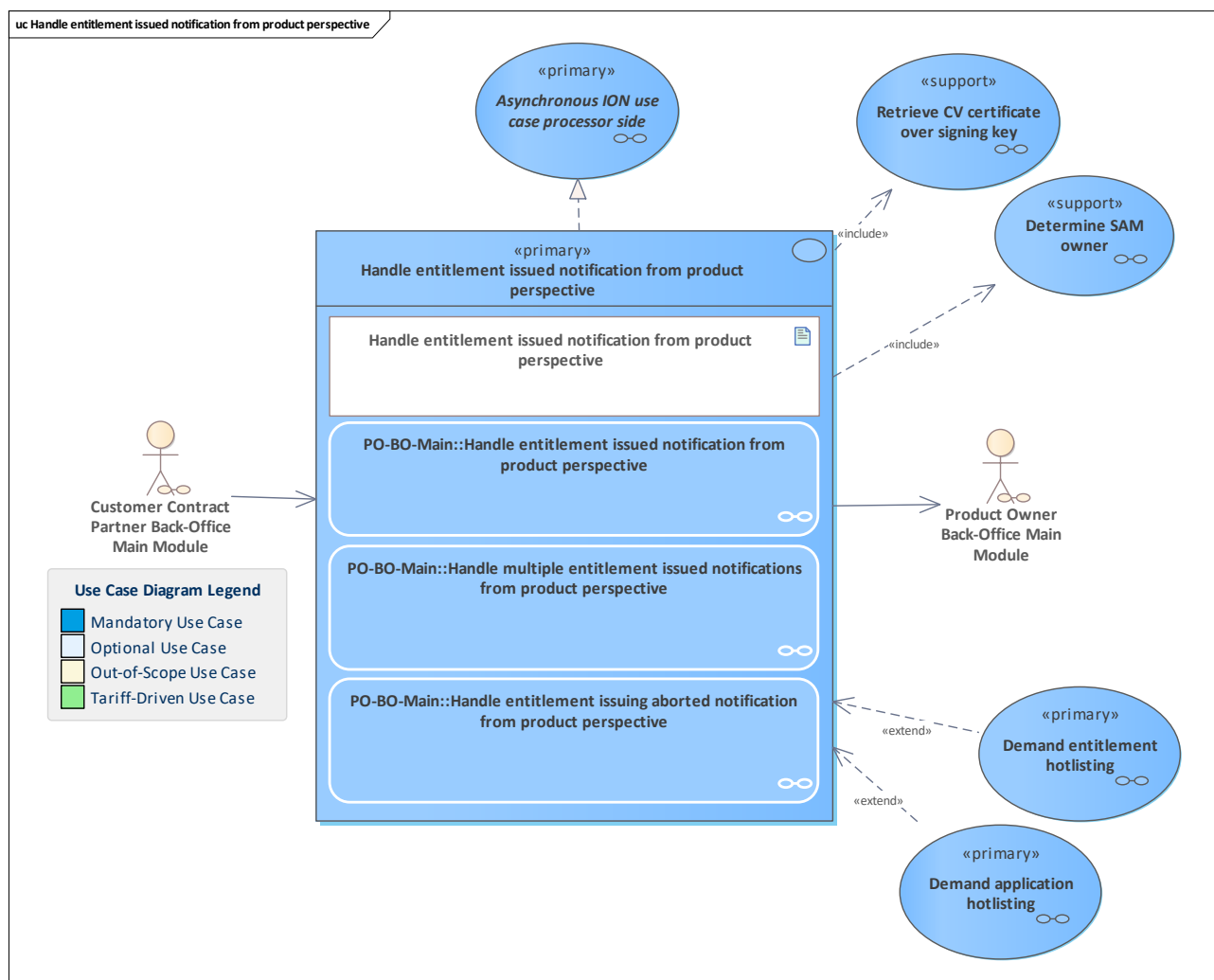
Handle entitlement issued notification from product perspective

Handle entitlement terminated notification from product perspective

Handle entitlement inspected notification from product perspective

8.2 Use Cases

8.2.1 Handle entitlement issued notification from product perspective

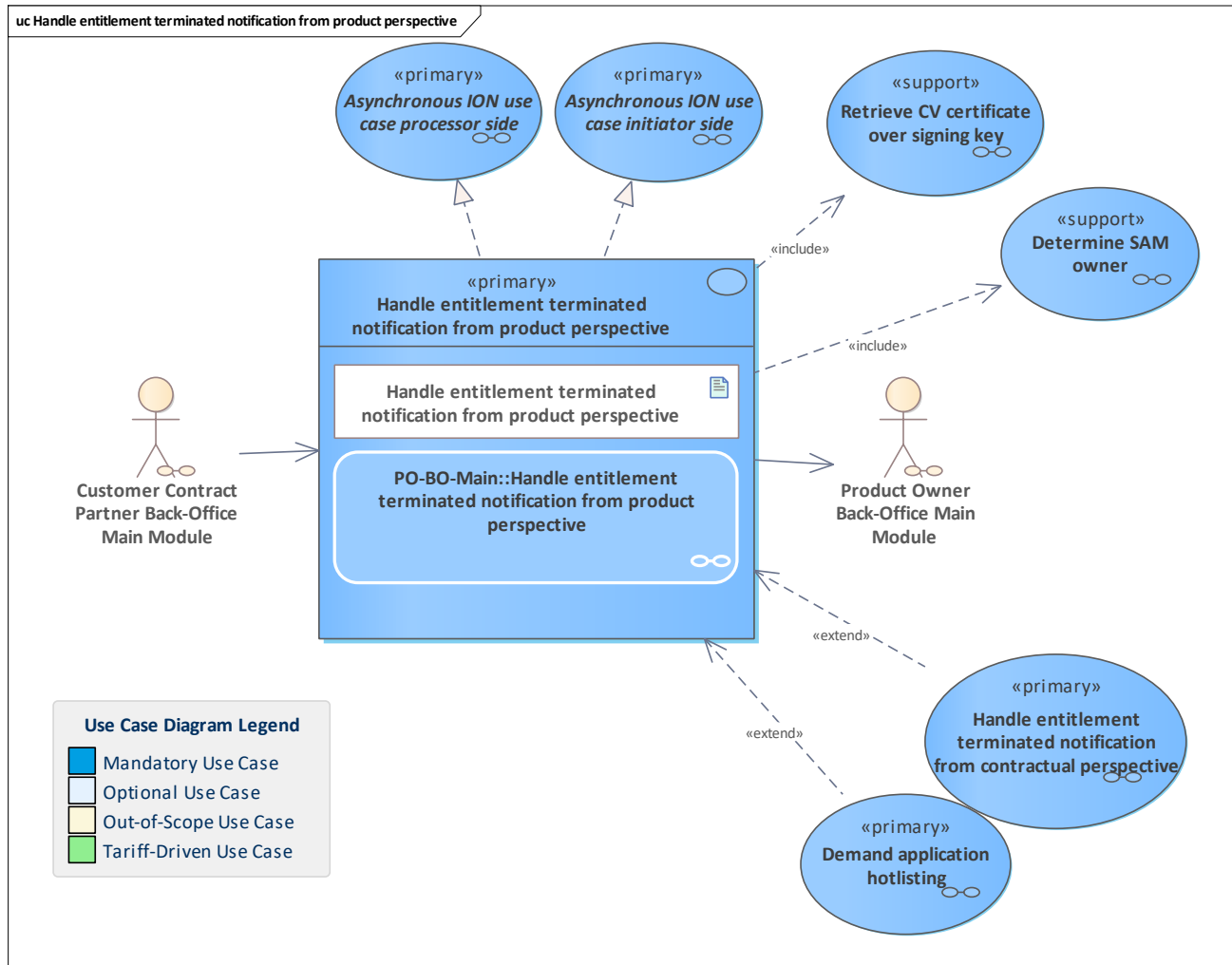


Use Case	Handle entitlement issued notification from product perspective
Description	<p>The PO back-office system receives the notification about an entitlement issuance and handles it from the product perspective. It registers the notification and does checks and monitoring. In the case of a list of notifications, this list is processed instead of a single notification.</p> <p>In the case of an abortion notification, the PO system has to register the SAM and product issuance counter for consistent monitoring.</p> <p>Note: the application instance ID of the user medium the entitlement was issued to can be uniquely identified via the request field <i>umAppInstanceId</i>, which is part of the SignedEntitlementIssuedAttestation that is contained in the EntitlementIssuedNotification.</p> <p>Note: for this use case, the issuance was not triggered by an action order.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand entitlement hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement issued : notifyEntitlementIssued
Outputs	Notify entitlement issued response : notifyEntitlementIssuedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement issued exception : notifyEntitlementIssuedException
Activity Diagram	PO-BO-Main::Handle entitlement issued notification from product perspective
Alternative 1	
Inputs	Process entitlement issued notification list : processEntitlementIssuedNotificationList
Outputs	Process entitlement issued notification list response : processEntitlementIssuedNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process entitlement issued notification list exception : processEntitlementIssuedNotificationListException
Activity Diagram	PO-BO-Main::Handle multiple entitlement issued notifications from product perspective



Alternative 2	
Inputs	<u>Notify entitlement issuing aborted :</u> <u>notifyEntitlementIssuingAborted</u>
Outputs	<u>Notify entitlement issuing aborted response :</u> <u>notifyEntitlementIssuingAbortedResponse</u>
Error Cases	<u>E_PO_RECEIVER_IS_NOT_PRODUCT_OWNER_OF_OBJECT</u> <u>E_CO_WRONG_SECURITY_LEVEL</u> <u>Notify entitlement issuing aborted exception :</u> <u>notifyEntitlementIssuingAbortedException</u>
Activity Diagram	<u>PO-BO-Main::Handle entitlement issuing aborted notification from</u> <u>product perspective</u>

8.2.2 Handle entitlement terminated notification from product perspective

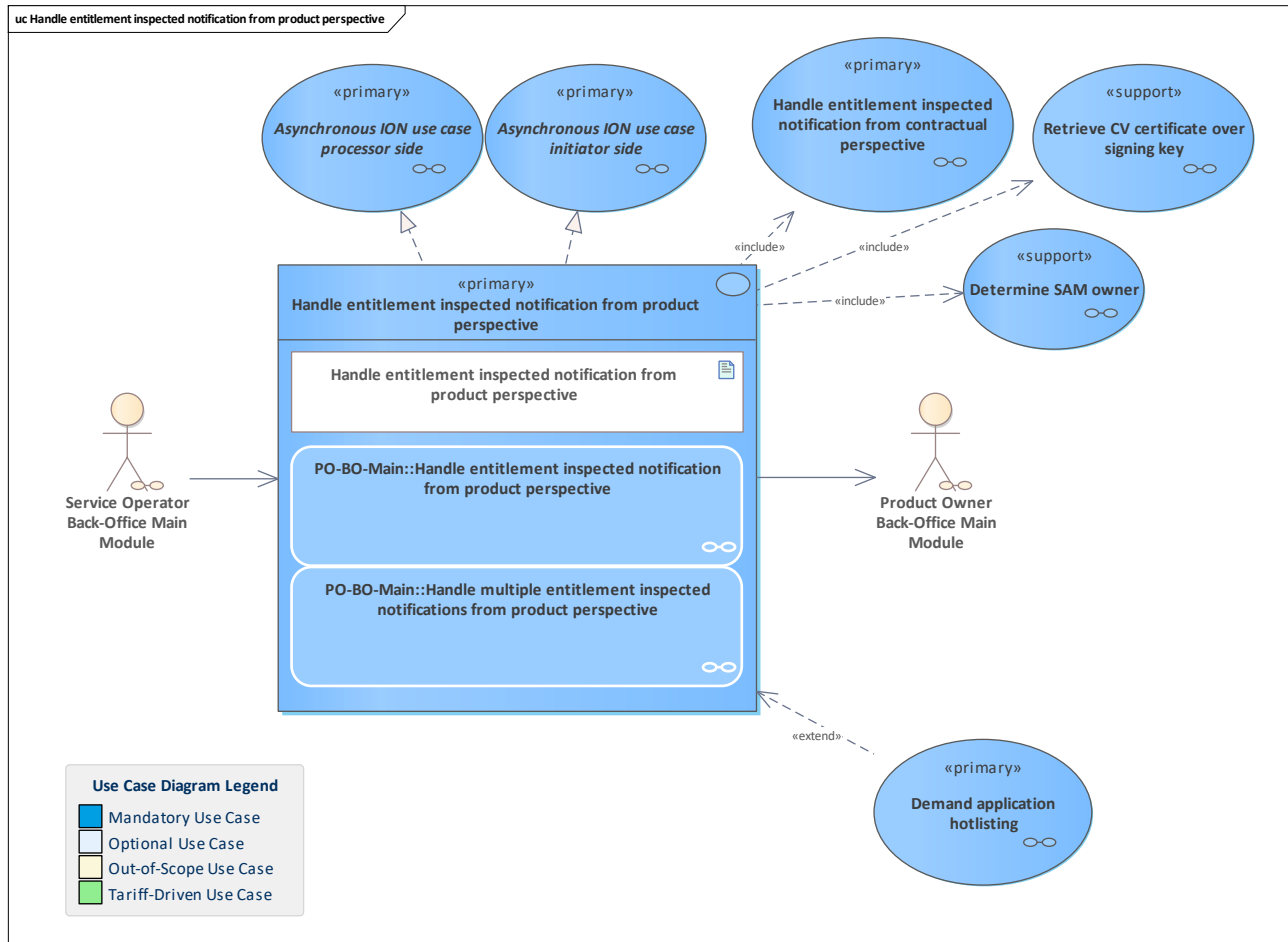


Use Case	Handle entitlement terminated notification from product perspective
Description	<p>Handle an entitlement terminated notification from the product perspective.</p> <p>The entitlement terminated notification is received by the PO. The PO registers the entitlement terminated notification and performs its checks and monitoring from the product perspective regarding the correct execution of the termination. In this context, the signature of the termination attestation is verified and the SAM owner of the SAM that performed the termination is determined.</p> <ul style="list-style-type: none"> if the sender is a sCCP: forward the notification to the pCCP if the sender is the pCCP, do not forward the notification. In this case, the current use case is not an asynchronous ION use case as initiator (Asynchronous ION use case initiator side). <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case</p>



	as initiator due to the asynchronous call to the pCCP.
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle entitlement terminated notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement terminated : notifyEntitlementTerminated
Outputs	Notify entitlement terminated response : notifyEntitlementTerminatedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement terminated exception : notifyEntitlementTerminatedException
Activity Diagram	PO-BO-Main::Handle entitlement terminated notification from product perspective

8.2.3 Handle entitlement inspected notification from product perspective



Use Case	Handle entitlement inspected notification from product perspective
Description	This use case describes the processing of the notification of an inspected entitlement in the PO back-office system. The SO sends the notification, the PO registers it and does its monitoring checks. After these checks, the notification is forwarded to the CCP back-office system. Furthermore, entitlement inspected notifications can be collected in the SO system and then sent as a list in a scheduled process to the PO.
Initiating Actor	Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle entitlement inspected notification from contractual perspective / Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side /



	<u>Asynchronous ION use case processor side</u>
Base Activity	
Inputs	<u>Notify entitlement inspected : notifyEntitlementInspected</u>
Outputs	<u>Notify entitlement inspected response : notifyEntitlementInspectedResponse</u>
Error Cases	<u>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</u> <u>E CO BINARY STRUCTURE CANNOT BE PROCESSED</u> <u>E CO WRONG ATTESTATION IN NOTIFICATION</u> <u>E CO WRONG SECURITY LEVEL</u> <u>E CO DUPLICATE UM ATTESTATION</u> <u>Notify entitlement inspected exception : notifyEntitlementInspectedException</u>
Activity Diagram	<u>PO-BO-Main::Handle entitlement inspected notification from product perspective</u>
Alternative 1	
Inputs	<u>Process entitlement inspected notification list : processEntitlementInspectedNotificationList</u>
Outputs	<u>Process entitlement inspected notification list response : processEntitlementInspectedNotificationListResponse</u>
Error Cases	<u>E CO CANNOT DECOMPRESS DATA</u> <u>E CO CANNOT DECOMPRESS DATA</u> <u>E CO WRONG ELEMENT IN COMPRESSED DATA</u> <u>Process entitlement inspected notification list exception : processEntitlementInspectedNotificationListException</u>
Activity Diagram	<u>PO-BO-Main::Handle multiple entitlement inspected notifications from product perspective</u>



9 Stored-Value Payment Bundle PO-System

Functionality bundle that covers the use cases for a PO back-office system forming the basis for using the stored-value payment method.

9.1 Overview

Handle entitlement issued notification from product perspective

Handle entitlement terminated notification from product perspective

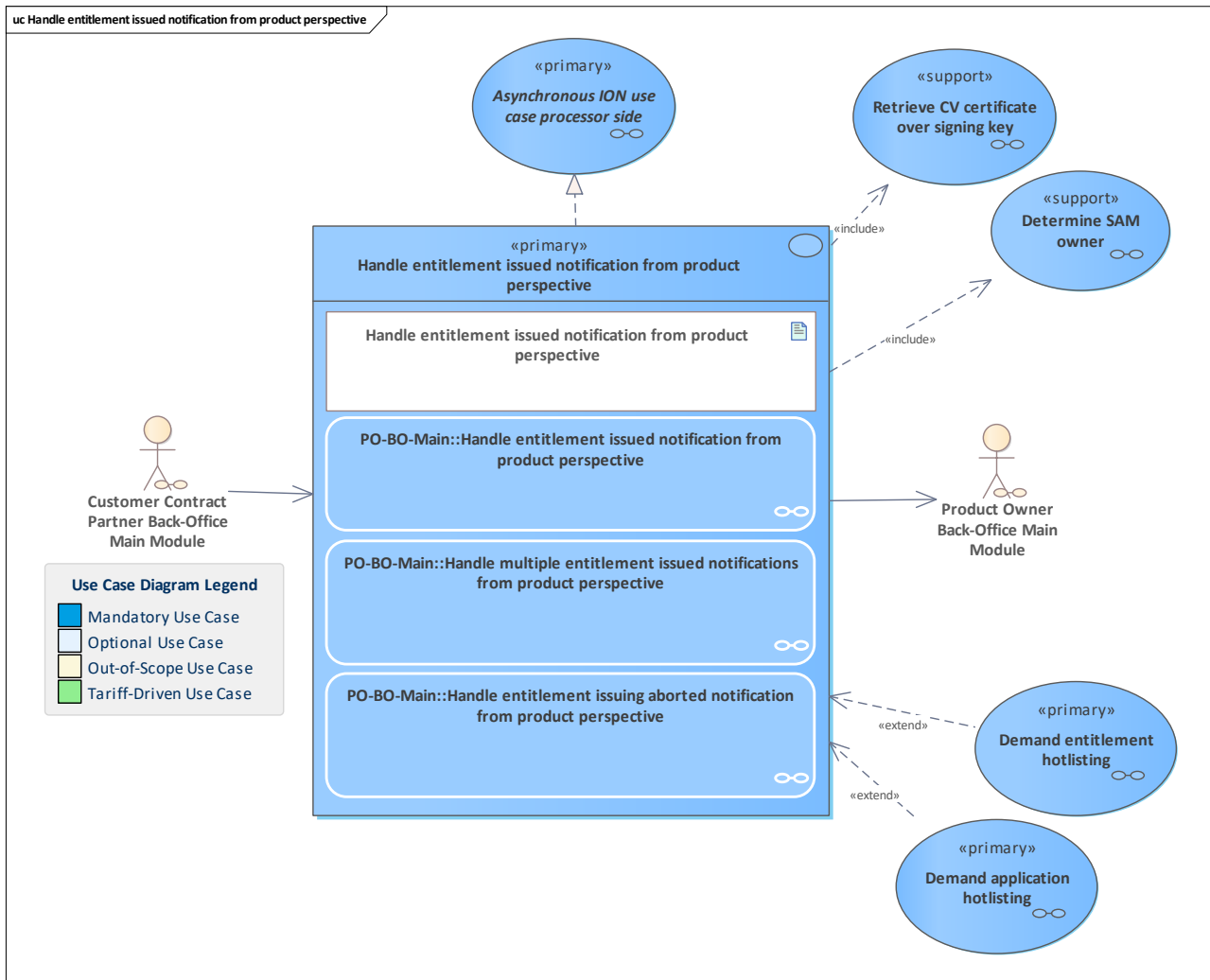
Handle entitlement inspected notification from product perspective

Handle stored-value payment method recharged notification from product perspective

Handle stored-value payment method reimbursed notification from product perspective

9.2 Use Cases

9.2.1 Handle entitlement issued notification from product perspective

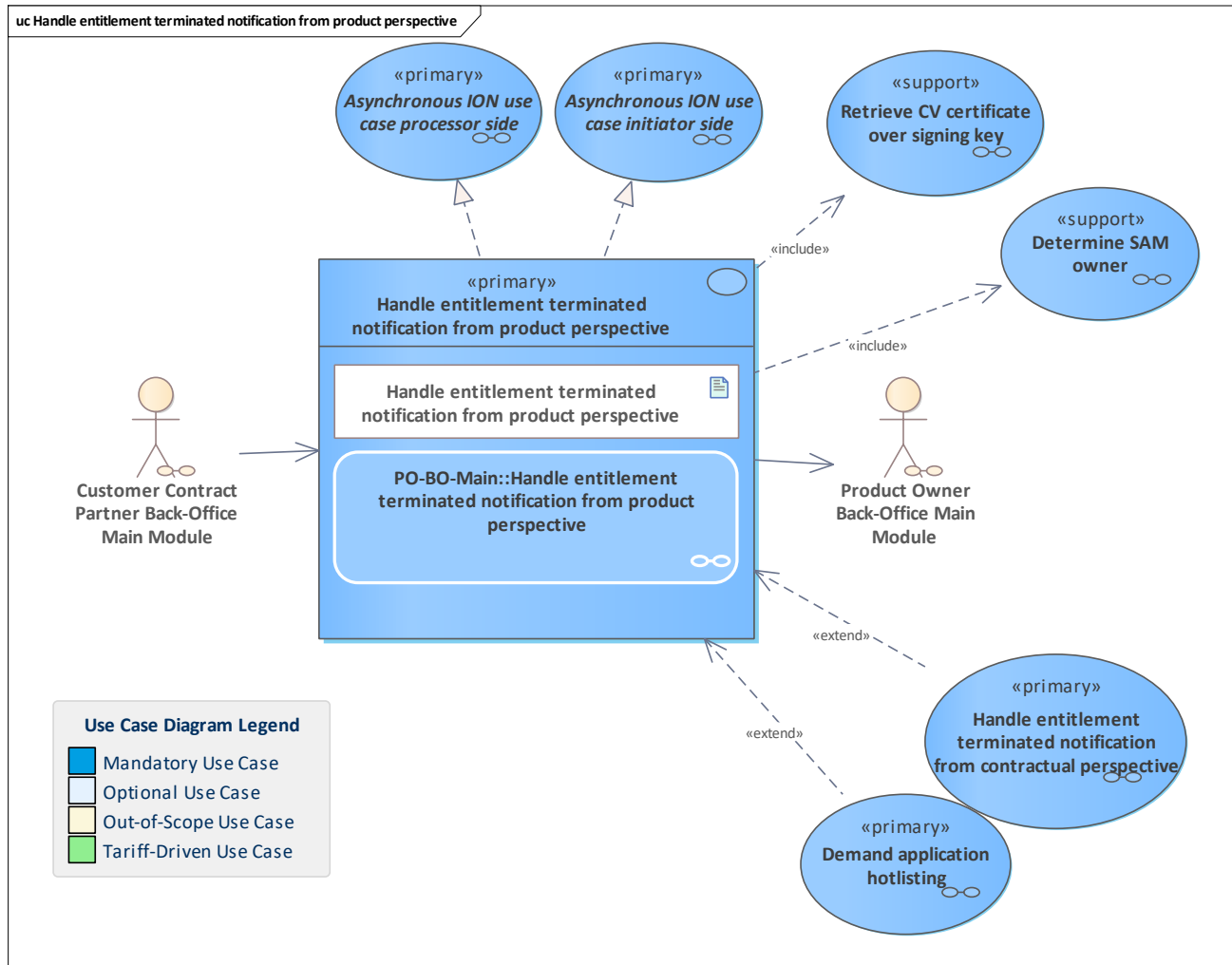


Use Case	<u>Handle entitlement issued notification from product perspective</u>
Description	<p>The PO back-office system receives the notification about an entitlement issuance and handles it from the product perspective. It registers the notification and does checks and monitoring. In the case of a list of notifications, this list is processed instead of a single notification. In the case of an abortion notification, the PO system has to register the SAM and product issuance counter for consistent monitoring.</p> <p>Note: the application instance ID of the user medium the entitlement was issued to can be uniquely identified via the request field <i>umAppInstanceId</i>, which is part of the SignedEntitlementIssuedAttestation that is contained in the EntitlementIssuedNotification.</p> <p>Note: for this use case, the issuance was not triggered by an action order.</p>
Initiating Actor	<u>Customer Contract Partner Back-Office Main Module</u>
Reacting Actor	<u>Product Owner Back-Office Main Module</u>
Preconditions	
Postconditions	



Linked Use Cases (Extended By)	Demand application hotlisting / Demand entitlement hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement issued : notifyEntitlementIssued
Outputs	Notify entitlement issued response : notifyEntitlementIssuedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement issued exception : notifyEntitlementIssuedException
Activity Diagram	PO-BO-Main::Handle entitlement issued notification from product perspective
Alternative 1	
Inputs	Process entitlement issued notification list : processEntitlementIssuedNotificationList
Outputs	Process entitlement issued notification list response : processEntitlementIssuedNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process entitlement issued notification list exception : processEntitlementIssuedNotificationListException
Activity Diagram	PO-BO-Main::Handle multiple entitlement issued notifications from product perspective
Alternative 2	
Inputs	Notify entitlement issuing aborted : notifyEntitlementIssuingAborted
Outputs	Notify entitlement issuing aborted response : notifyEntitlementIssuingAbortedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO WRONG SECURITY LEVEL Notify entitlement issuing aborted exception : notifyEntitlementIssuingAbortedException
Activity Diagram	PO-BO-Main::Handle entitlement issuing aborted notification from product perspective

9.2.2 Handle entitlement terminated notification from product perspective

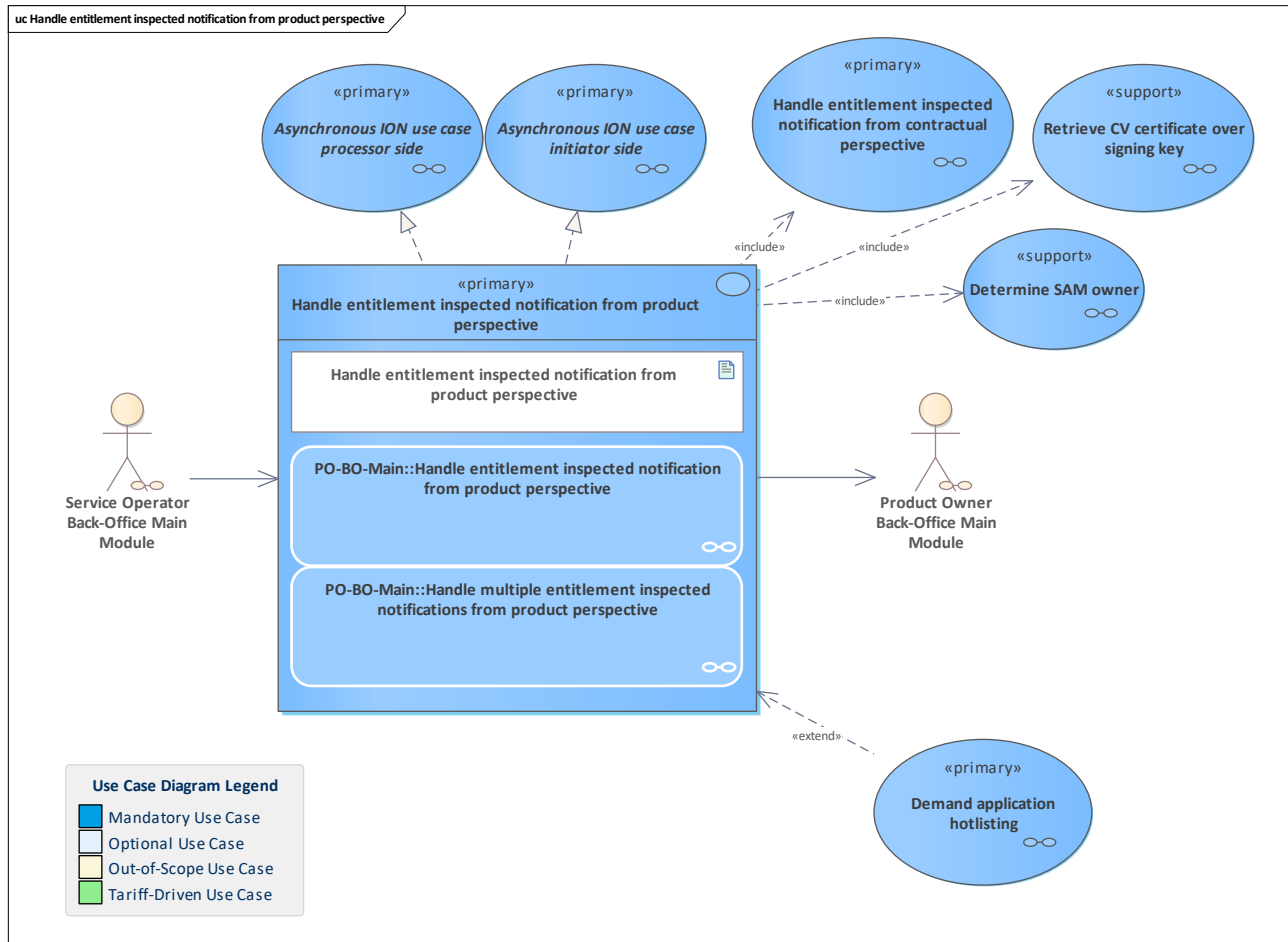


Use Case	Handle entitlement terminated notification from product perspective
Description	<p>Handle an entitlement terminated notification from the product perspective.</p> <p>The entitlement terminated notification is received by the PO. The PO registers the entitlement terminated notification and performs its checks and monitoring from the product perspective regarding the correct execution of the termination. In this context, the signature of the termination attestation is verified and the SAM owner of the SAM that performed the termination is determined.</p> <ul style="list-style-type: none"> if the sender is a sCCP: forward the notification to the pCCP if the sender is the pCCP, do not forward the notification. In this case, the current use case is not an asynchronous ION use case as initiator (Asynchronous ION use case initiator side). <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case</p>



	as initiator due to the asynchronous call to the pCCP.
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle entitlement terminated notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement terminated : notifyEntitlementTerminated
Outputs	Notify entitlement terminated response : notifyEntitlementTerminatedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement terminated exception : notifyEntitlementTerminatedException
Activity Diagram	PO-BO-Main::Handle entitlement terminated notification from product perspective

9.2.3 Handle entitlement inspected notification from product perspective

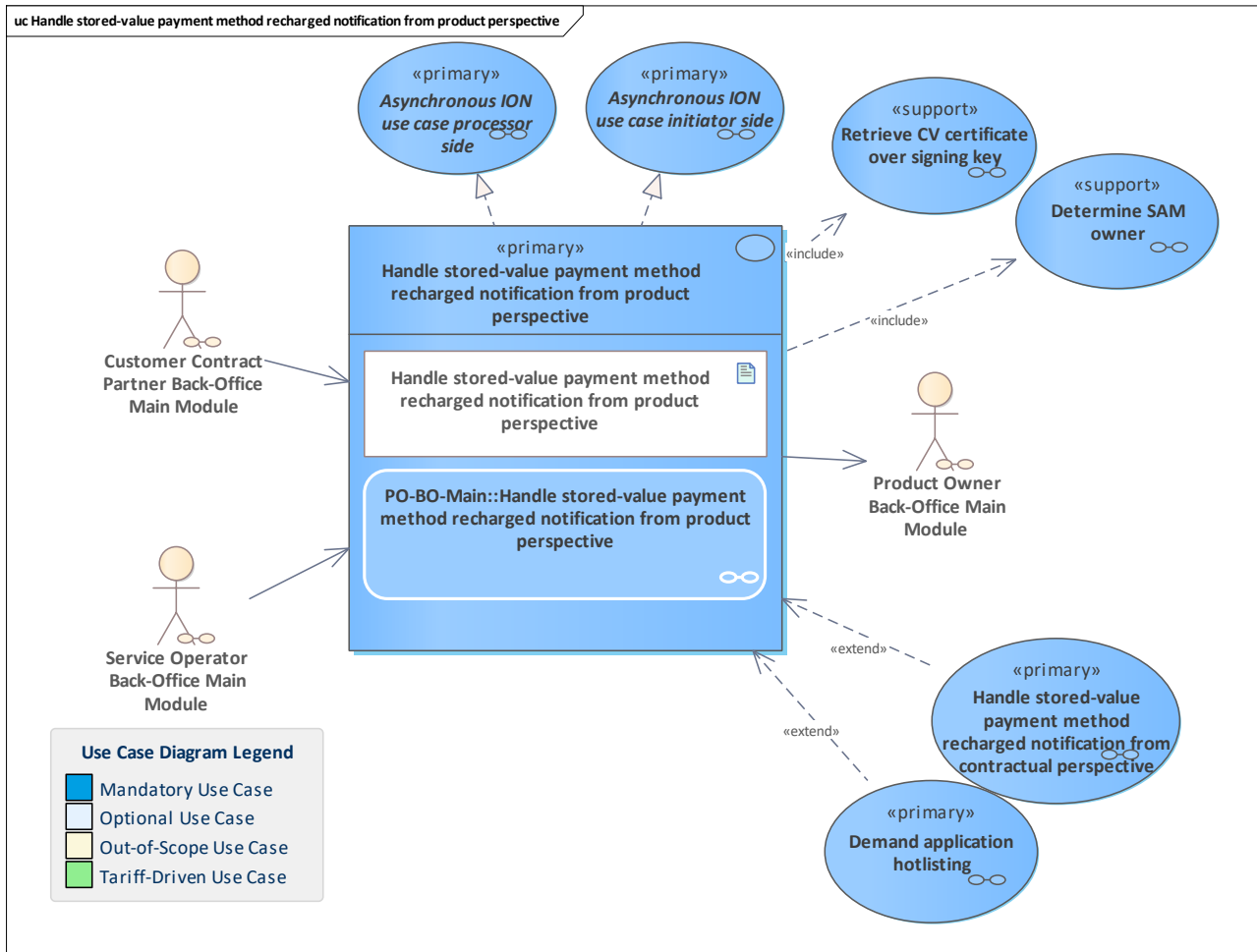


Use Case	Handle entitlement inspected notification from product perspective
Description	This use case describes the processing of the notification of an inspected entitlement in the PO back-office system. The SO sends the notification, the PO registers it and does its monitoring checks. After these checks, the notification is forwarded to the CCP back-office system. Furthermore, entitlement inspected notifications can be collected in the SO system and then sent as a list in a scheduled process to the PO.
Initiating Actor	Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle entitlement inspected notification from contractual perspective / Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side /



	<u>Asynchronous ION use case processor side</u>
Base Activity	
Inputs	<u>Notify entitlement inspected : notifyEntitlementInspected</u>
Outputs	<u>Notify entitlement inspected response : notifyEntitlementInspectedResponse</u>
Error Cases	<u>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</u> <u>E CO BINARY STRUCTURE CANNOT BE PROCESSED</u> <u>E CO WRONG ATTESTATION IN NOTIFICATION</u> <u>E CO WRONG SECURITY LEVEL</u> <u>E CO DUPLICATE UM ATTESTATION</u> <u>Notify entitlement inspected exception : notifyEntitlementInspectedException</u>
Activity Diagram	<u>PO-BO-Main::Handle entitlement inspected notification from product perspective</u>
Alternative 1	
Inputs	<u>Process entitlement inspected notification list : processEntitlementInspectedNotificationList</u>
Outputs	<u>Process entitlement inspected notification list response : processEntitlementInspectedNotificationListResponse</u>
Error Cases	<u>E CO CANNOT DECOMPRESS DATA</u> <u>E CO CANNOT DECOMPRESS DATA</u> <u>E CO WRONG ELEMENT IN COMPRESSED DATA</u> <u>Process entitlement inspected notification list exception : processEntitlementInspectedNotificationListException</u>
Activity Diagram	<u>PO-BO-Main::Handle multiple entitlement inspected notifications from product perspective</u>

9.2.4 Handle stored-value payment method recharged notification from product perspective

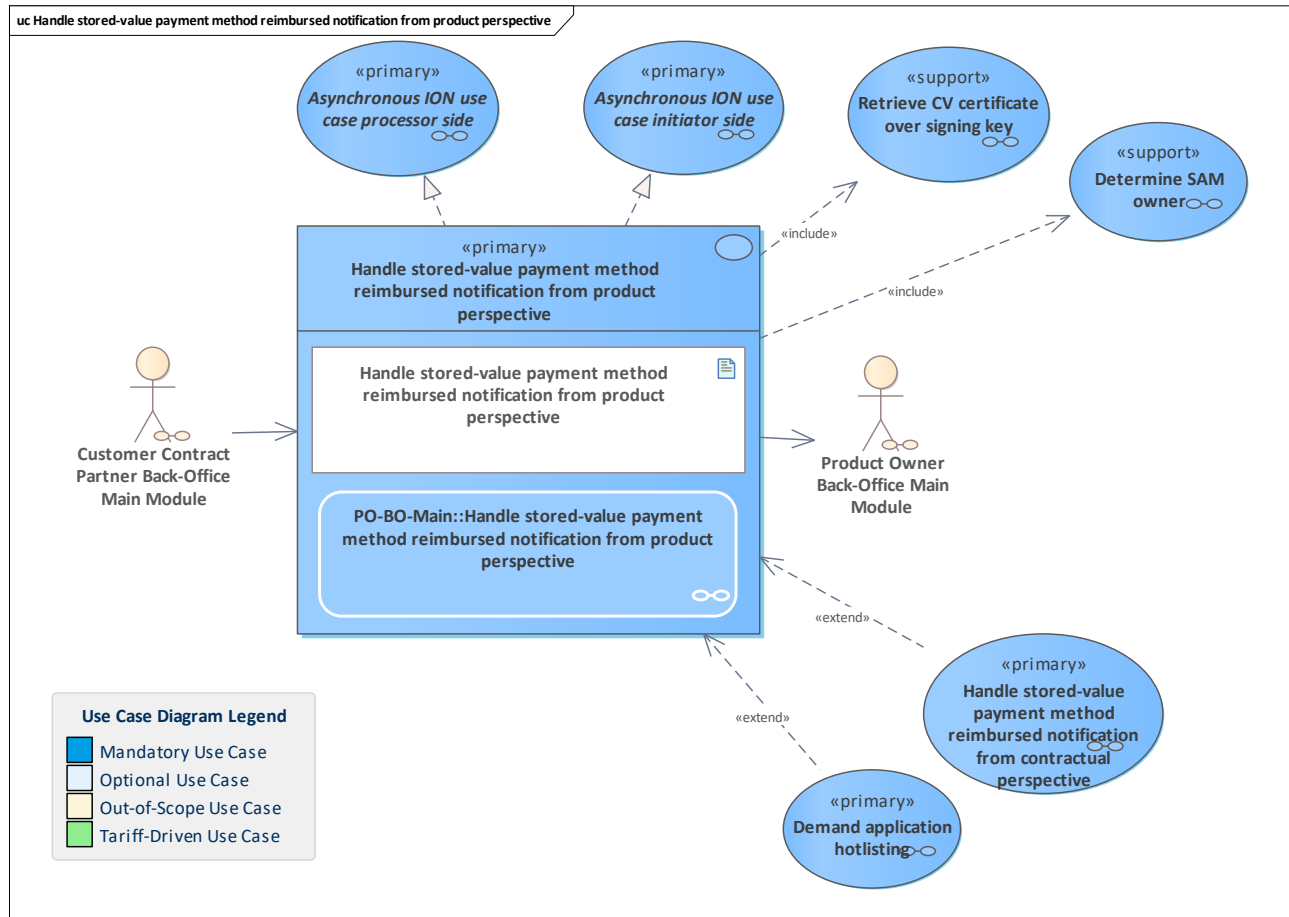


Use Case	Handle stored-value payment method recharged notification from product perspective
Description	<p>Handle a stored-value payment method recharged notification from the product owner perspective.</p> <p>The PO back-office system receives from the CCP system and registers the notification about a performed recharge with a stored-value payment method.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle stored-value payment method recharged notification from contractual perspective / Demand application hotlisting / Demand application hotlisting



Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify stored-value payment method recharged : notifyStoredValuePaymentMethodRecharged
Outputs	Notify stored-value payment method recharged response : notifyStoredValuePaymentMethodRechargedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify stored-value payment method recharged exception : notifyStoredValuePaymentMethodRechargedException
Activity Diagram	PO-BO-Main::Handle stored-value payment method recharged notification from product perspective

9.2.5 Handle stored-value payment method reimbursed notification from product perspective



Use Case	Handle stored-value payment method reimbursed notification from product perspective
Description	<p>Handle a stored-value payment method reimbursed notification from the product owner perspective.</p> <p>The PO back-office system receives from the CCP system and registers the notification about a performed reimburse with a stored-value payment method.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle stored-value payment method reimbursed notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases	Asynchronous ION use case initiator side / Asynchronous ION use



(Realises)	case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify stored-value payment method reimbursed : notifyStoredValuePaymentMethodReimbursed
Outputs	Notify stored-value payment method reimbursed response : notifyStoredValuePaymentMethodReimbursedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify stored-value payment method reimbursed exception : notifyStoredValuePaymentMethodReimbursedException
Activity Diagram	PO-BO-Main::Handle stored-value payment method reimbursed notification from product perspective

10 Sale Electronic Ticket via Account-Based Payment Bundle PO-System

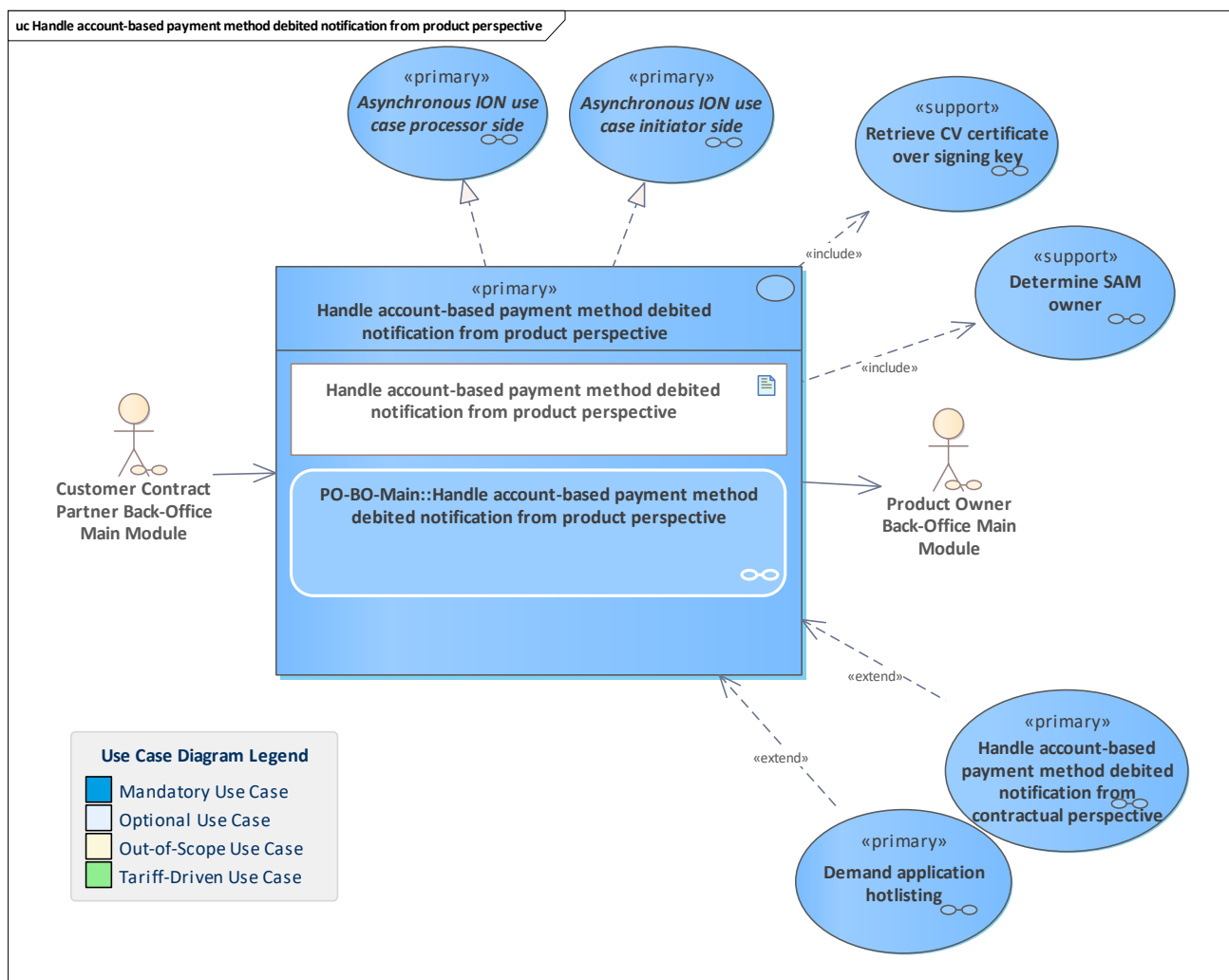
Functionality bundle that provides PO back-office system use cases for selling or purchasing electronic tickets via an account-based payment method on a user medium.

10.1 Overview

Handle account-based payment method debited notification from product perspective
Handle account-based payment method credited notification from product perspective

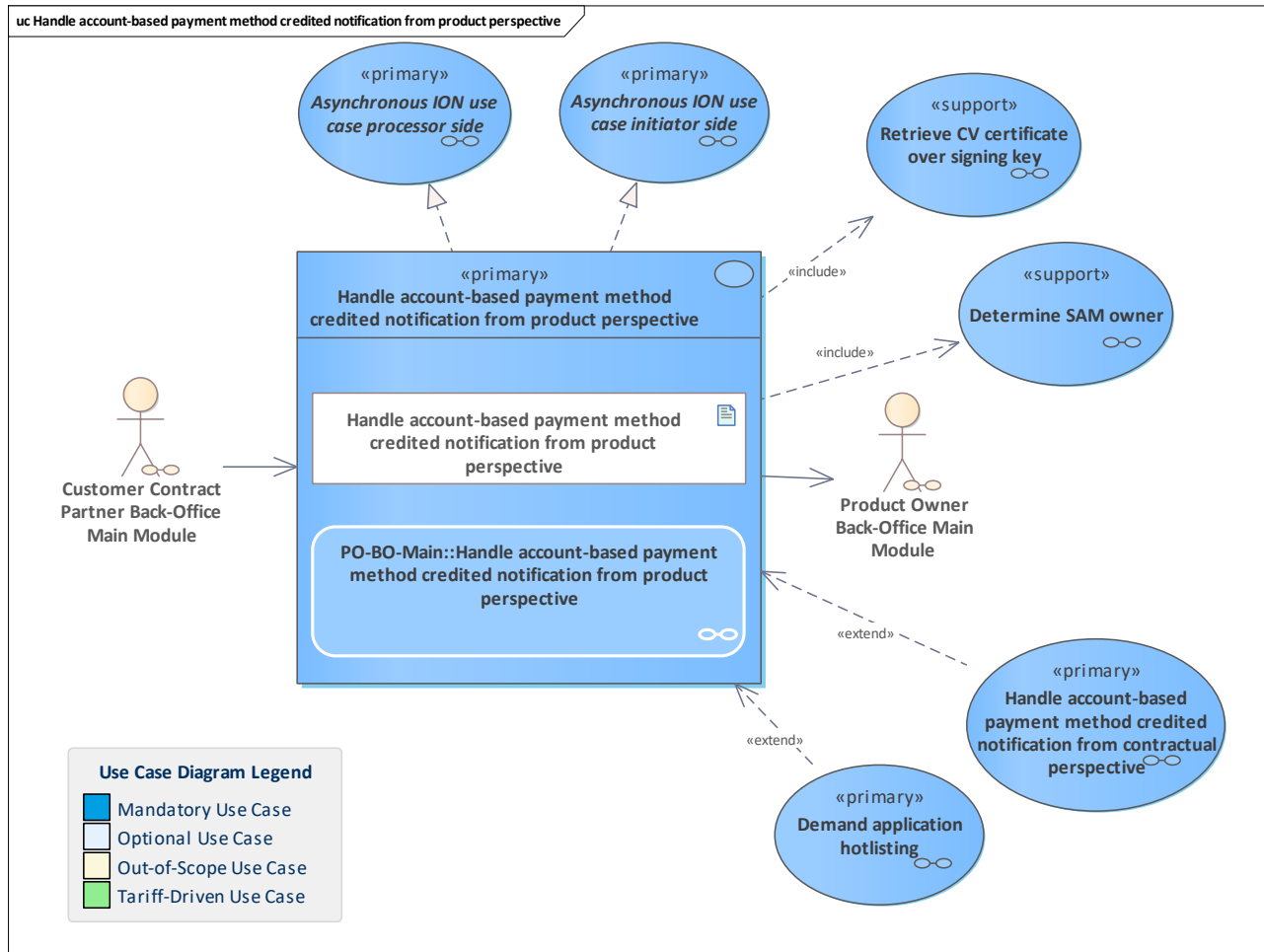
10.2 Use Cases

10.2.1 Handle account-based payment method debited notification from product perspective



Use Case	Handle account-based payment method debited notification from product perspective
Description	<p>Handle a notification about an account-based payment method debiting from the product perspective.</p> <p>The PO back-office system receives and registers the notification about a performed debit with an account-based payment method from the CCP system.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle account-based payment method debited notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side
Base Activity	
Inputs	Notify account-based payment method debited : notifyAccountBasedPaymentMethodDebited
Outputs	Notify account-based payment method debited response : notifyAccountBasedPaymentMethodDebitedResponse
Error Cases	<p>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</p> <p>E CO BINARY STRUCTURE CANNOT BE PROCESSED</p> <p>E CO WRONG ATTESTATION IN NOTIFICATION</p> <p>E CO WRONG SECURITY LEVEL</p> <p>E CO DUPLICATE UM ATTESTATION</p> <p>Notify account-based payment method debited exception : notifyAccountBasedPaymentMethodDebitedException</p>
Activity Diagram	PO-BO-Main::Handle account-based payment method debited notification from product perspective

10.2.2 Handle account-based payment method credited notification from product perspective



Use Case	Handle account-based payment method credited notification from product perspective
Description	<p>Handle an account-based payment method credited notification from the product perspective.</p> <p>The PO back-office system receives and registers the notification about a performed credit with an account-based payment method from the CCP system.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle account-based payment method credited notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases	Retrieve CV certificate over signing key / Determine SAM owner /



(Includes)	Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify account-based payment method credited : notifyAccountBasedPaymentMethodCredited
Outputs	Notify account-based payment method credited response : notifyAccountBasedPaymentMethodCreditedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify account-based payment method credited exception : notifyAccountBasedPaymentMethodCreditedException
Activity Diagram	PO-BO-Main::Handle account-based payment method credited notification from product perspective

11 Sale Electronic Ticket via Stored-Value Payment Bundle PO-System

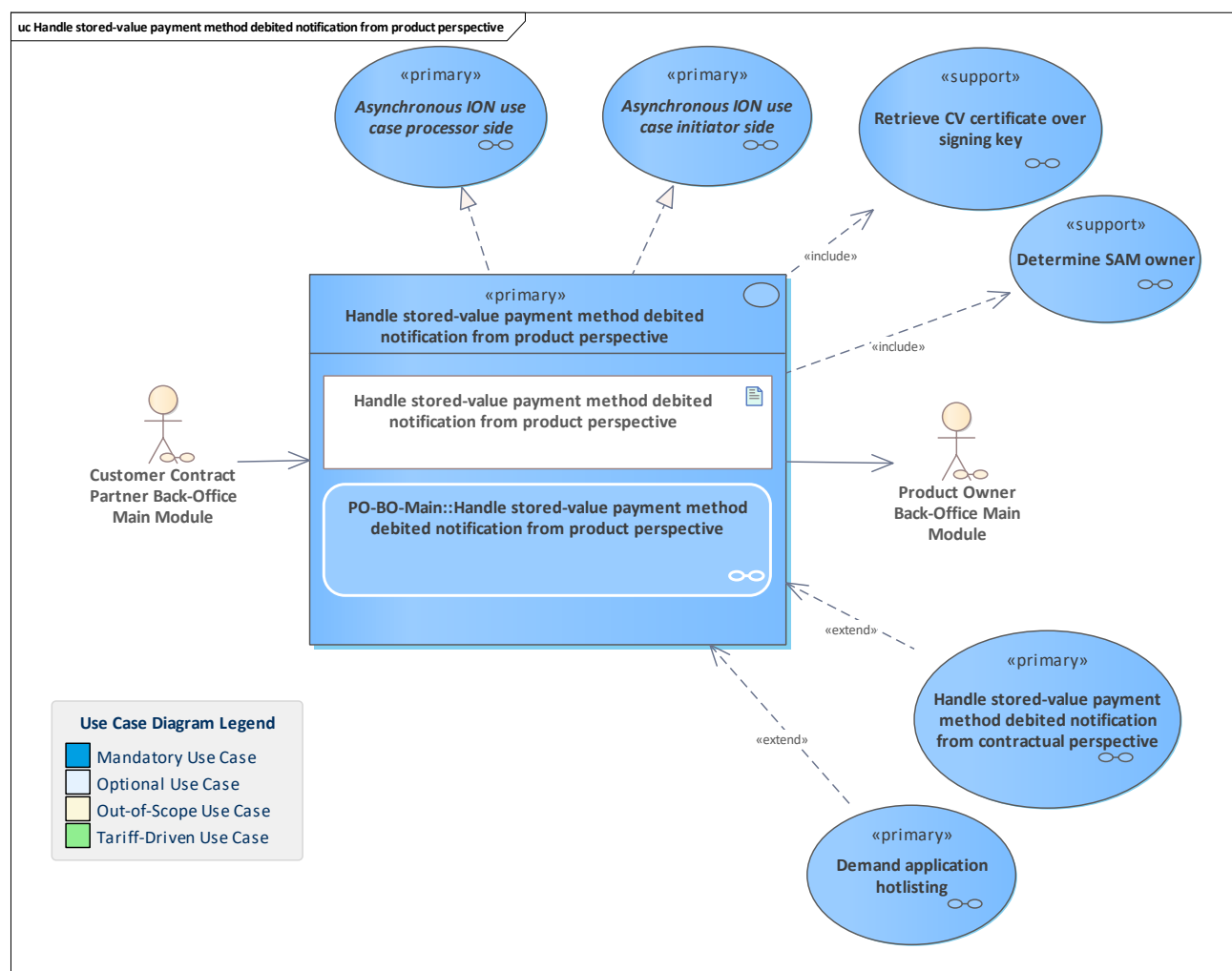
Functionality bundle that provides PO back-office system use cases for selling or purchasing electronic tickets via a stored-value payment method on a user medium.

11.1 Overview

Handle stored-value payment method debited notification from product perspective
Handle stored-value payment method credited notification from product perspective

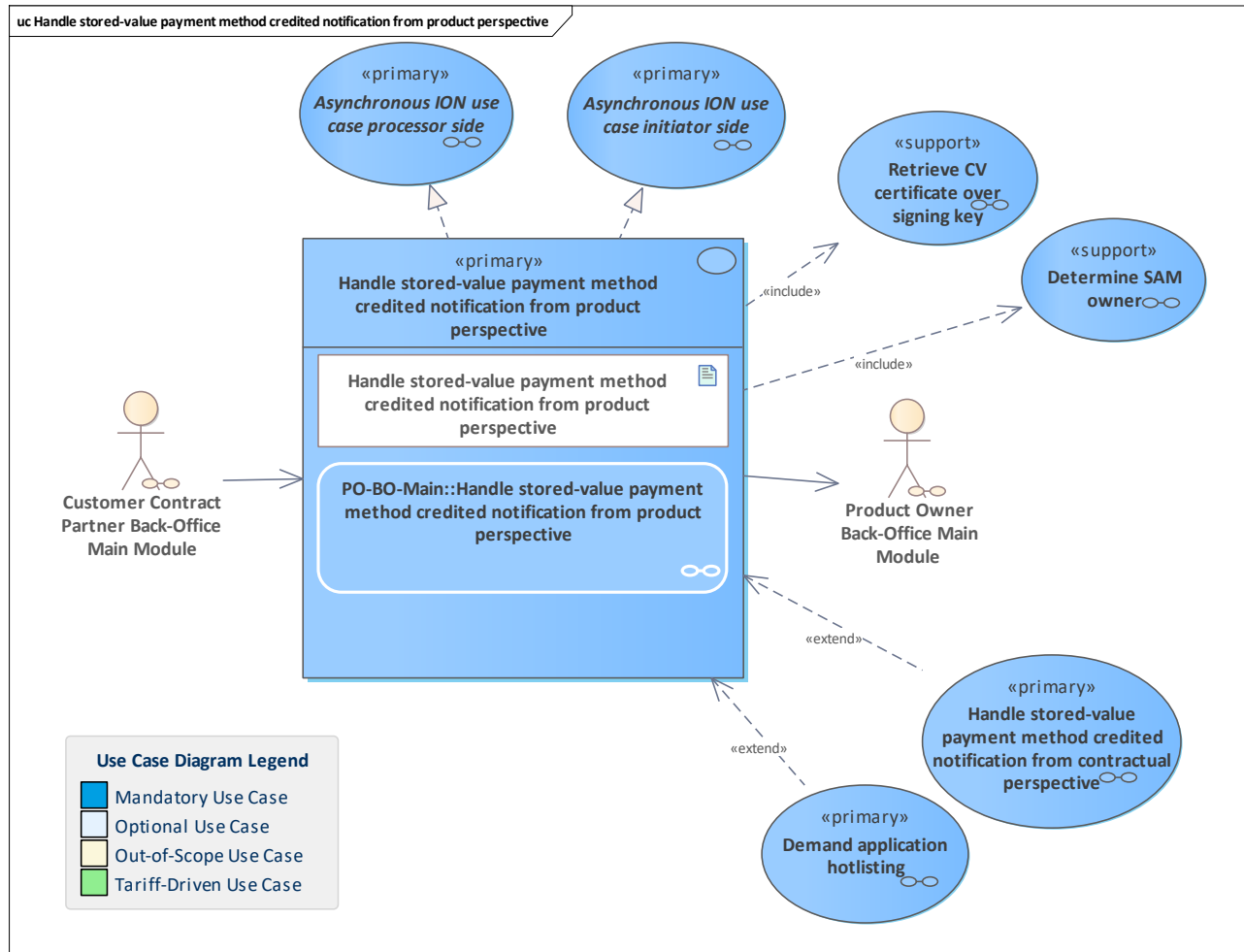
11.2 Use Cases

11.2.1 Handle stored-value payment method debited notification from product perspective



Use Case	Handle stored-value payment method debited notification from product perspective
Description	<p>Handle a notification about a stored-value payment method debiting from the product owner perspective.</p> <p>The PO back-office system receives from the CCP system and registers the notification about a performed debit with a stored-value payment method.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle stored-value payment method debited notification from contractual perspective / Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify stored-value payment method debited : notifyStoredValuePaymentMethodDebited
Outputs	Notify stored-value payment method debited response : notifyStoredValuePaymentMethodDebitedResponse
Error Cases	<p>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</p> <p>E CO BINARY STRUCTURE CANNOT BE PROCESSED</p> <p>E CO WRONG ATTESTATION IN NOTIFICATION</p> <p>E CO WRONG SECURITY LEVEL</p> <p>E CO DUPLICATE UM ATTESTATION</p> <p>Notify stored-value payment method debited exception : notifyStoredValuePaymentMethodDebitedException</p>
Activity Diagram	PO-BO-Main::Handle stored-value payment method debited notification from product perspective

11.2.2 Handle stored-value payment method credited notification from product perspective



Use Case	Handle stored-value payment method credited notification from product perspective
Description	<p>Handle a stored-value payment method credited notification from the product owner perspective.</p> <p>The PO back-office system receives the notification from the CCP system and registers the notification about a performed credit with a stored-value payment method.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system. Then, this use case is also an Asynchronous ION use case initiator side.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle stored-value payment method credited notification from contractual perspective / Demand application hotlisting / Demand application hotlisting



Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify stored-value payment method credited : notifyStoredValuePaymentMethodCredited
Outputs	Notify stored-value payment method credited response : notifyStoredValuePaymentMethodCreditedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify stored-value payment method credited exception : notifyStoredValuePaymentMethodCreditedException
Activity Diagram	PO-BO-Main::Handle stored-value payment method credited notification from product perspective

12 IN-OUT Bundle PO-System

Functionality bundle that provides PO back-office system use cases for IN-OUT functionality. The term CICO is also used.

12.1 Overview

Handle check-in notification from product perspective

Handle check-out notification from product perspective

Handle account-based payment method debited notification from product perspective

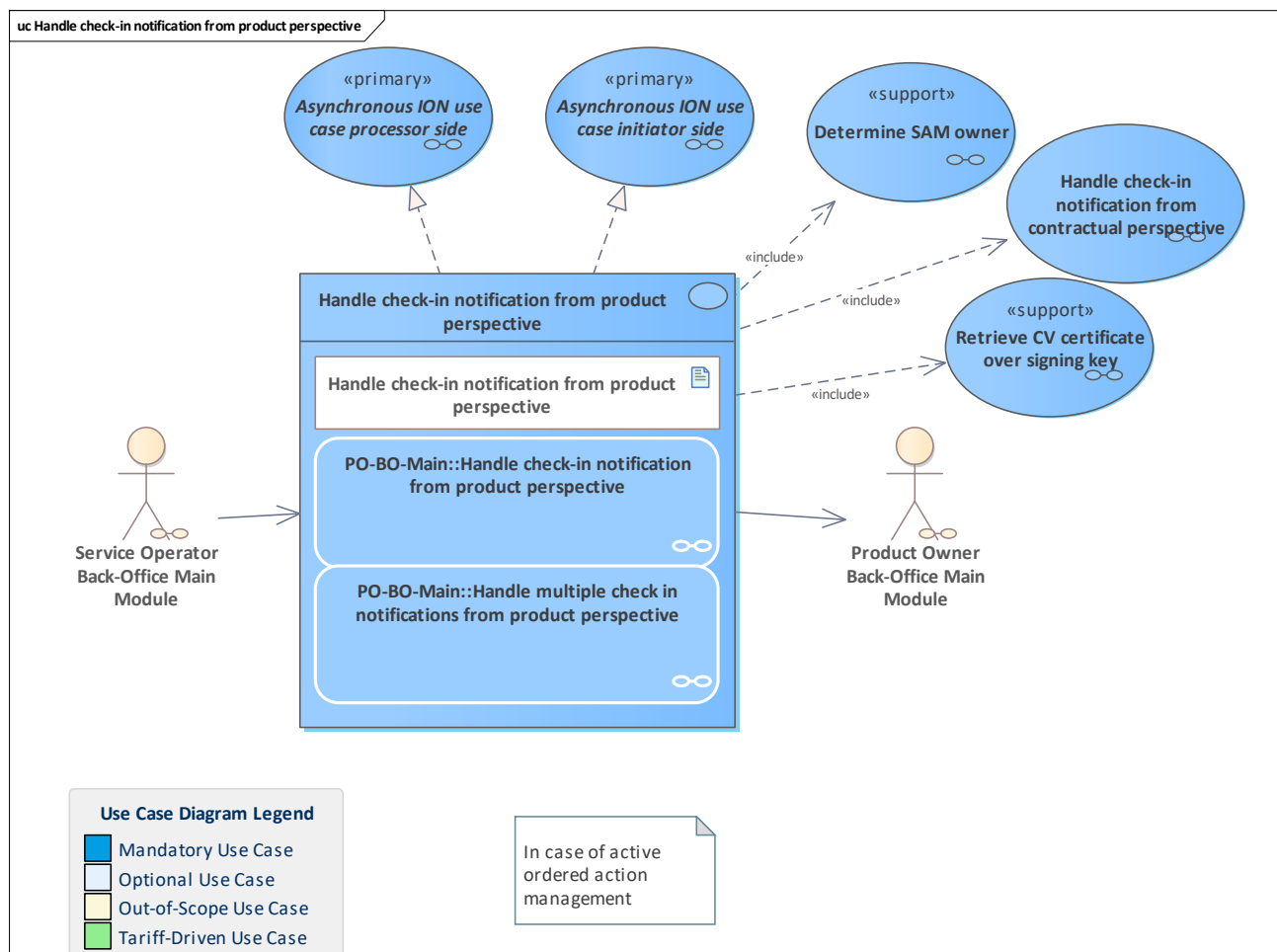
Handle stored-value payment method debited notification from product perspective

Handle user tariff parameters changed notification from product perspective

Demand account-based payment method charging

12.2 Use Cases

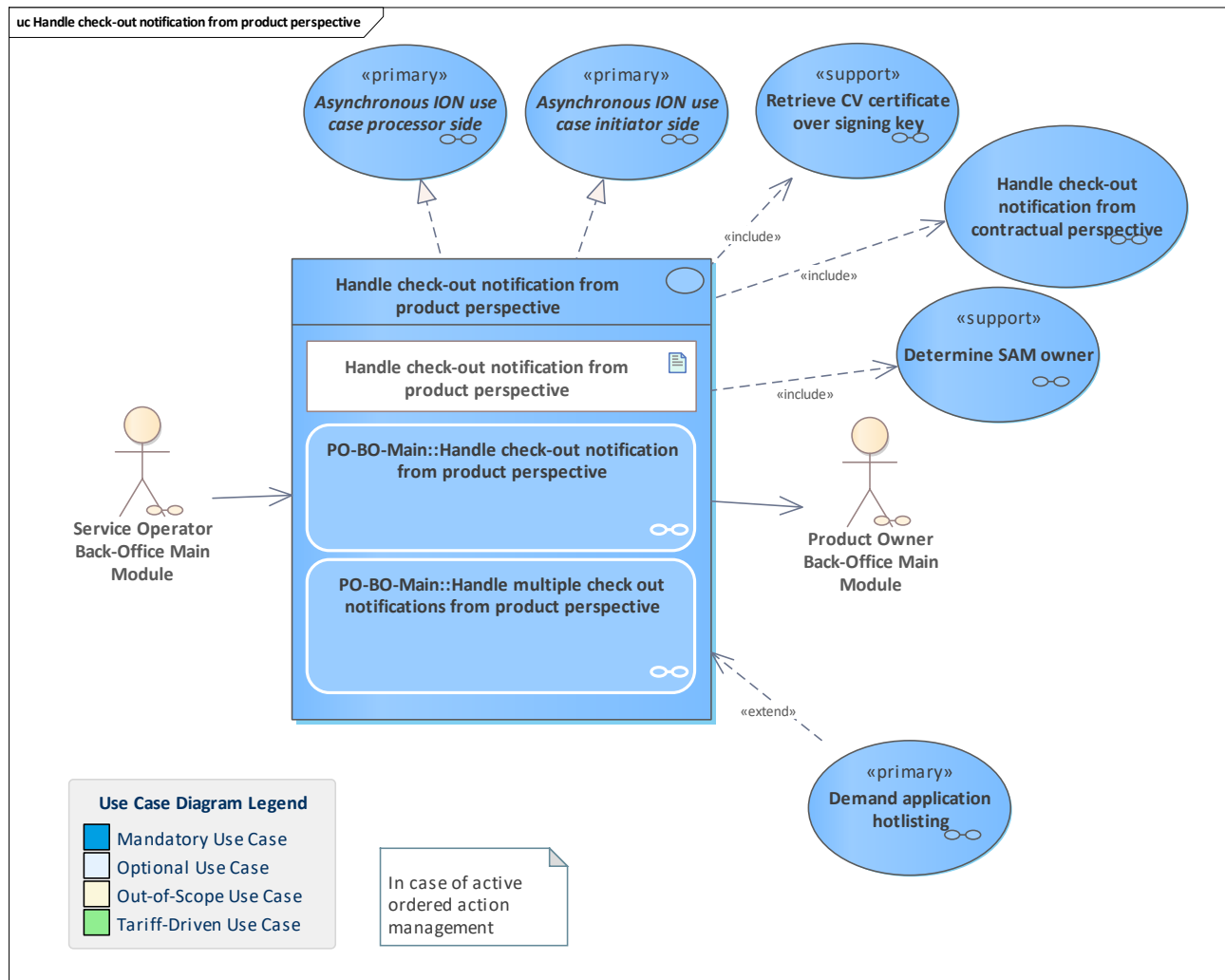
12.2.1 Handle check-in notification from product perspective





Use Case	Handle check-in notification from product perspective
Description	<p>Handle a check-in operation from the product perspective.</p> <p>This use case describes the processing of the check-in notification in the PO back-office system.</p> <p>The SO sends the notification, the PO registers it and does its monitoring checks. After these checks, the notification is forwarded to the CCP back-office system.</p> <p>Furthermore, check-in notifications can be collected in the SO system and then sent as a list in a scheduled process to the PO.</p>
Initiating Actor	Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle check-in notification from contractual perspective / Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify check in : notifyCheckin
Outputs	Notify check in response : notifyCheckinResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify check in exception : notifyCheckinException
Activity Diagram	PO-BO-Main::Handle check-in notification from product perspective
Alternative 1	
Inputs	Process check in notification list : processCheckinNotificationList
Outputs	Process check in notification list response : processCheckinNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process check in notification list exception : processCheckinNotificationListException
Activity Diagram	PO-BO-Main::Handle multiple check in notifications from product perspective

12.2.2 Handle check-out notification from product perspective

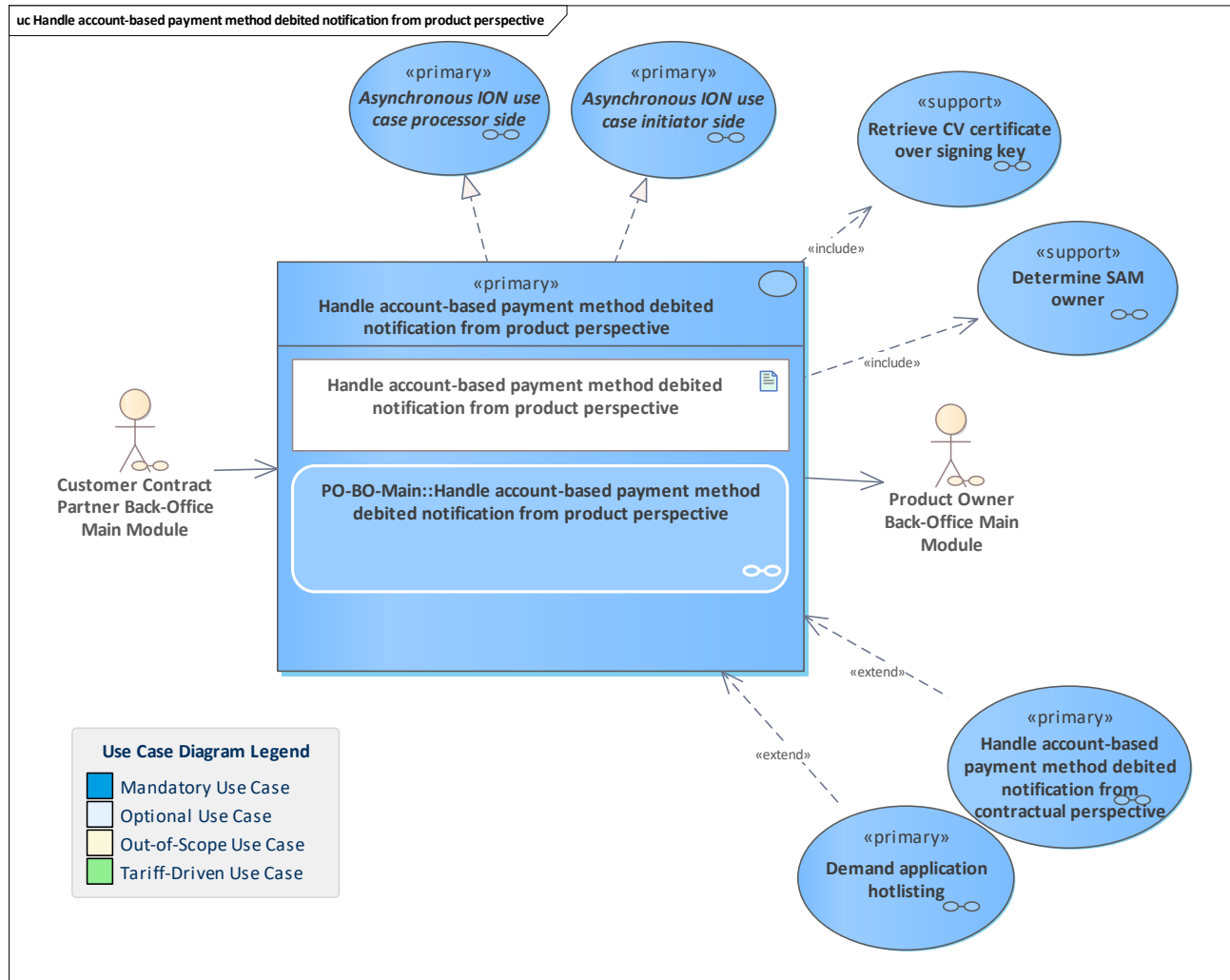


Use Case	Handle check-out notification from product perspective
Description	Handle a check-out operation from the product perspective. This use case describes the processing of the check-out notification in the PO back-office system. The SO sends the notification, the PO registers it and does its monitoring checks. After these checks, the notification is forwarded to the CCP back-office system. Furthermore, check-out notifications can be collected in the SO system and then sent as a list in a scheduled process to the PO.
Initiating Actor	Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle check-out notification from contractual perspective / Determine SAM owner / Retrieve CV certificate over signing key /



	Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify check out : notifyCheckout
Outputs	Notify check out response : notifyCheckoutResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify check out exception : notifyCheckoutException
Activity Diagram	PO-BO-Main::Handle check-out notification from product perspective
Alternative 1	
Inputs	Process check out notification list : processCheckoutNotificationList
Outputs	Process check out notification list response : processCheckoutNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process check out notification list exception : processCheckoutNotificationListException
Activity Diagram	PO-BO-Main::Handle multiple check out notifications from product perspective

12.2.3 Handle account-based payment method debited notification from product perspective

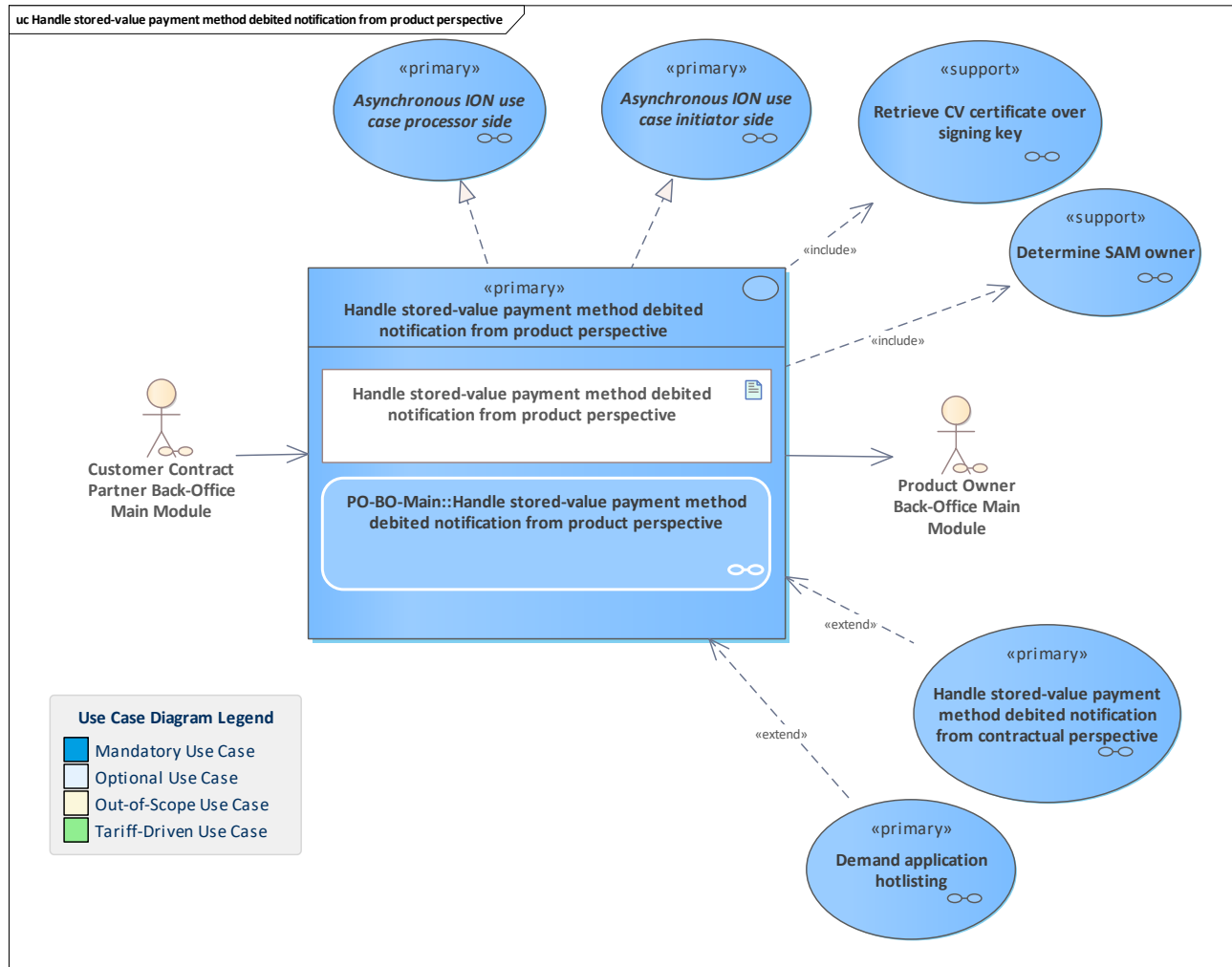


Use Case	Handle account-based payment method debited notification from product perspective
Description	<p>Handle a notification about an account-based payment method debiting from the product perspective.</p> <p>The PO back-office system receives and registers the notification about a performed debit with an account-based payment method from the CCP system.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle account-based payment method debited notification from contractual perspective / Demand application hotlisting / Demand



	application hotlisting
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side
Base Activity	
Inputs	Notify account-based payment method debited : notifyAccountBasedPaymentMethodDebited
Outputs	Notify account-based payment method debited response : notifyAccountBasedPaymentMethodDebitedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify account-based payment method debited exception : notifyAccountBasedPaymentMethodDebitedException
Activity Diagram	PO-BO-Main::Handle account-based payment method debited notification from product perspective

12.2.4 Handle stored-value payment method debited notification from product perspective

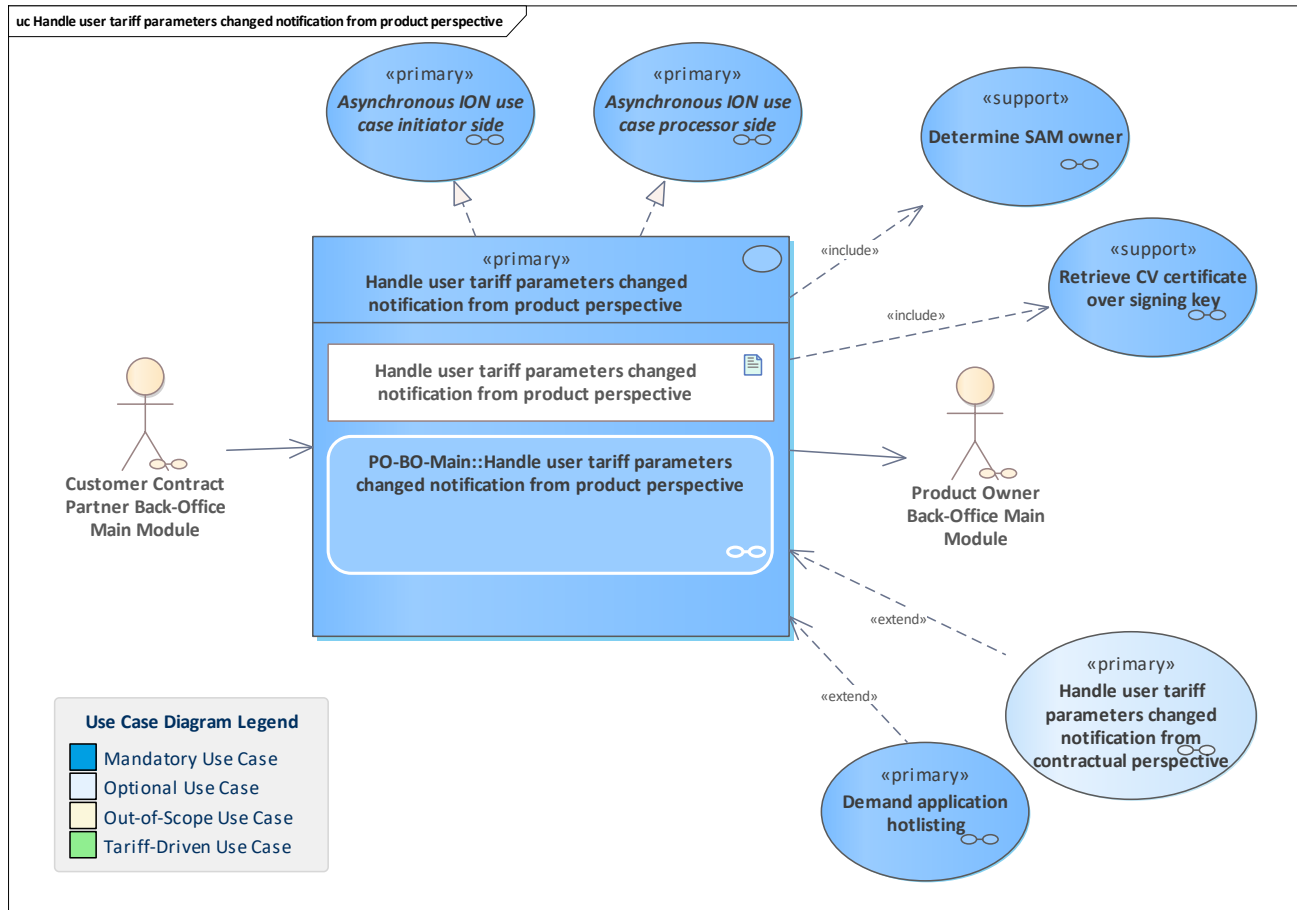


Use Case	Handle stored-value payment method debited notification from product perspective
Description	<p>Handle a notification about a stored-value payment method debiting from the product owner perspective.</p> <p>The PO back-office system receives from the CCP system and registers the notification about a performed debit with a stored-value payment method.</p> <p>It does the checks and monitoring from the product owner perspective.</p> <p>If the sending CCP was not the pCCP, the notification is forwarded to the responsible pCCP system.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle stored-value payment method debited notification from contractual perspective / Demand application hotlisting / Demand application hotlisting



Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Determine SAM owner / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify stored-value payment method debited : notifyStoredValuePaymentMethodDebited
Outputs	Notify stored-value payment method debited response : notifyStoredValuePaymentMethodDebitedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify stored-value payment method debited exception : notifyStoredValuePaymentMethodDebitedException
Activity Diagram	PO-BO-Main::Handle stored-value payment method debited notification from product perspective

12.2.5 Handle user tariff parameters changed notification from product perspective

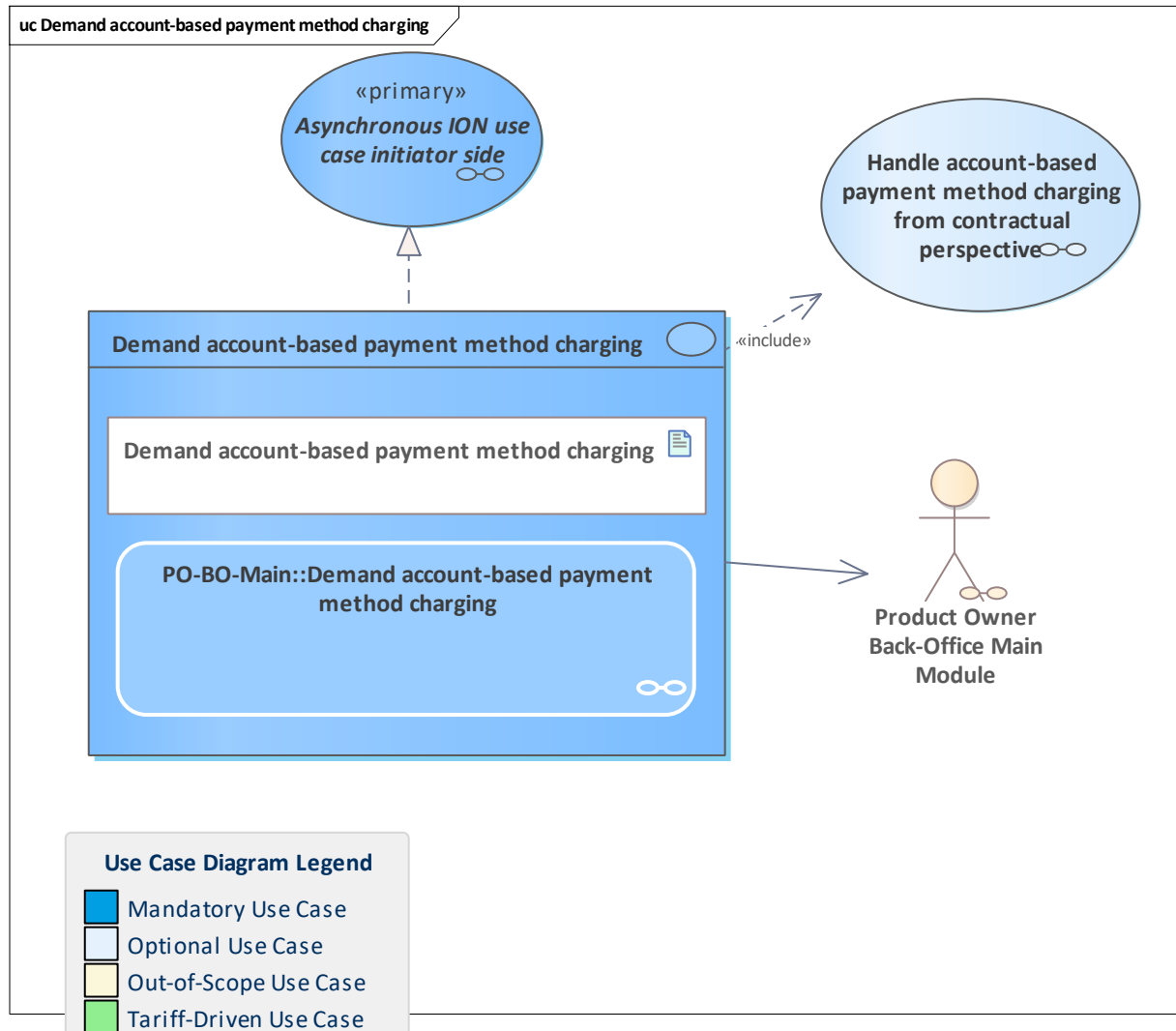


Use Case	Handle user tariff parameters changed notification from product perspective
Description	<p>Handle an changed user tariff parameters notification from the product owner perspective.</p> <p>The entitlement changed user tariff parameters notification is received by the PO.</p> <p>The PO registers the changed user tariff parameters notification and does its checks and monitoring from the product perspective regarding the correct execution. In this context, the signature of the embedded attestation is verified and the SAM owner of the SAM that performed the changing of the user tariff parameters is determined.</p> <ul style="list-style-type: none"> if the sender is a sCCP: forward the notification to the pCCP if the sender is the pCCP: do not forward the notification. In this case, the current use case is not an asynchronous ION use case as an initiator. <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case as initiator due to the asynchronous call to the pCCP.</p>



Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting / Handle user tariff parameters changed notification from contractual perspective
Linked Use Cases (Includes)	Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify user tariff parameters changed : notifyUserTariffParametersChanged
Outputs	Notify user tariff parameters changed response : notifyUserTariffParametersChangedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify user tariff parameters changed exception : notifyUserTariffParametersChangedException
Activity Diagram	PO-BO-Main::Handle user tariff parameters changed notification from product perspective

12.2.6 Demand account-based payment method charging



Use Case	Demand account-based payment method charging
Description	When using an account-based payment system in the CICO context: The PO determines the price of all journeys based on received CICO messages and informs the pCCP about these journeys. This is done in a scheduled process within a defined time interval. The process details are out of scope.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle account-based payment method charging from contractual perspective
Linked Use Cases (Realises)	Asynchronous ION use case initiator side



Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-Main::Demand account-based payment method charging

13 Ordered Action Management Bundle

PO-System

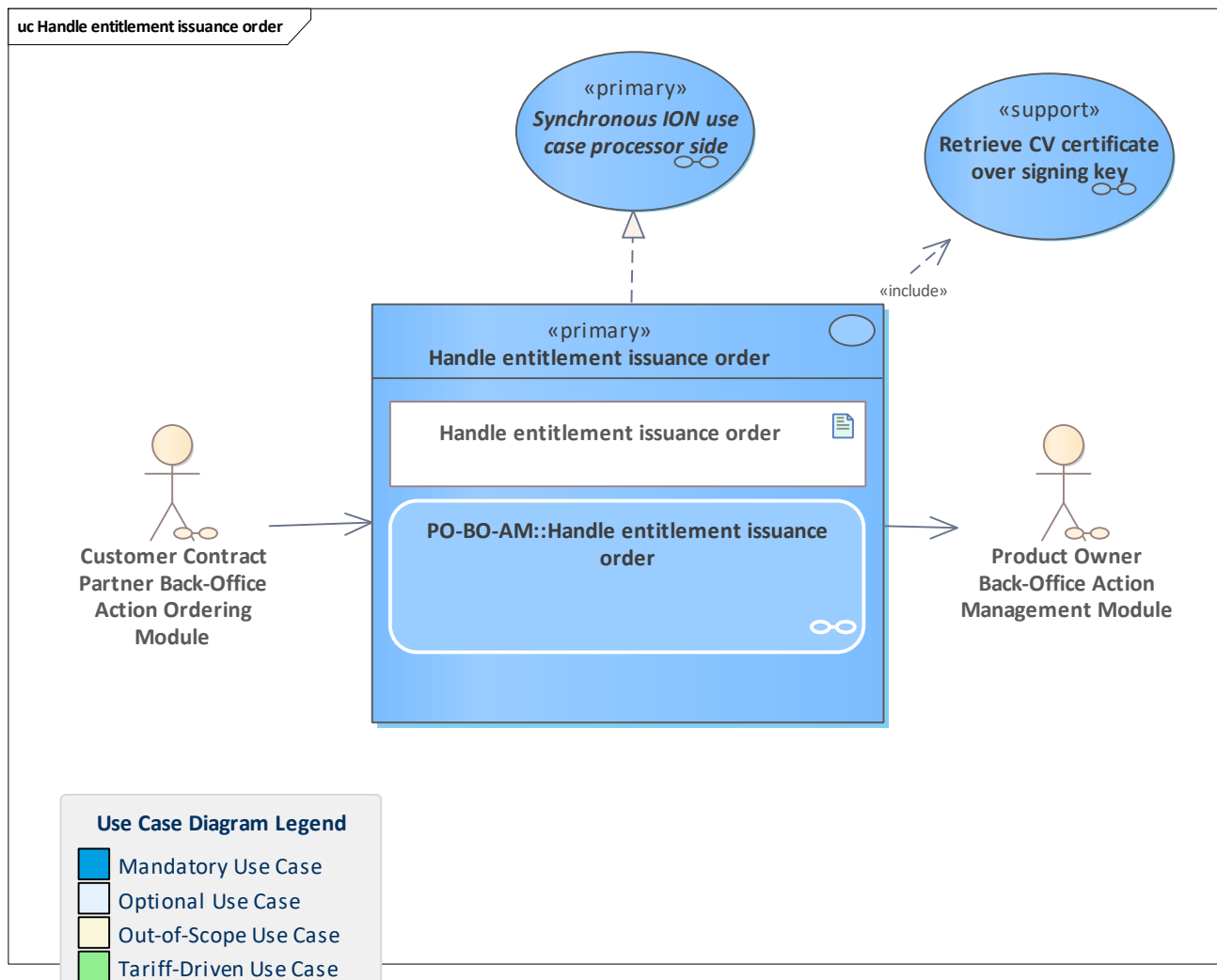
Functionality bundle that covers the use cases to implement the PO back-office system functionality for working with action lists. This could include ordering or cancelling actions, provisioning of action lists, distribution of retrieval configuration, etc.

13.1 Overview

[Handle entitlement issuance order](#)
[Handle entitlement unblocking order](#)
[Handle entitlement termination order](#)
[Handle entitlement blocking order](#)
[Handle order group](#)
[Handle order cancellation](#)
[Handle ordered entitlement issued notification from product perspective](#)
[Handle ordered entitlement unblocked notification from product perspective](#)
[Handle ordered entitlement terminated notification from product perspective](#)
[Handle ordered entitlement blocked notification from product perspective](#)
[Handle retrieval request for action list](#)
[Handle retrieval request for incremental action list](#)
[Handle verification request for action list updated via increments](#)
[Generate current action list](#)
[Put action list retrieval configuration](#)
[Analyse order history from product perspective](#)
[Check for order obsolescence](#)

13.2 Use Cases

13.2.1 Handle entitlement issuance order

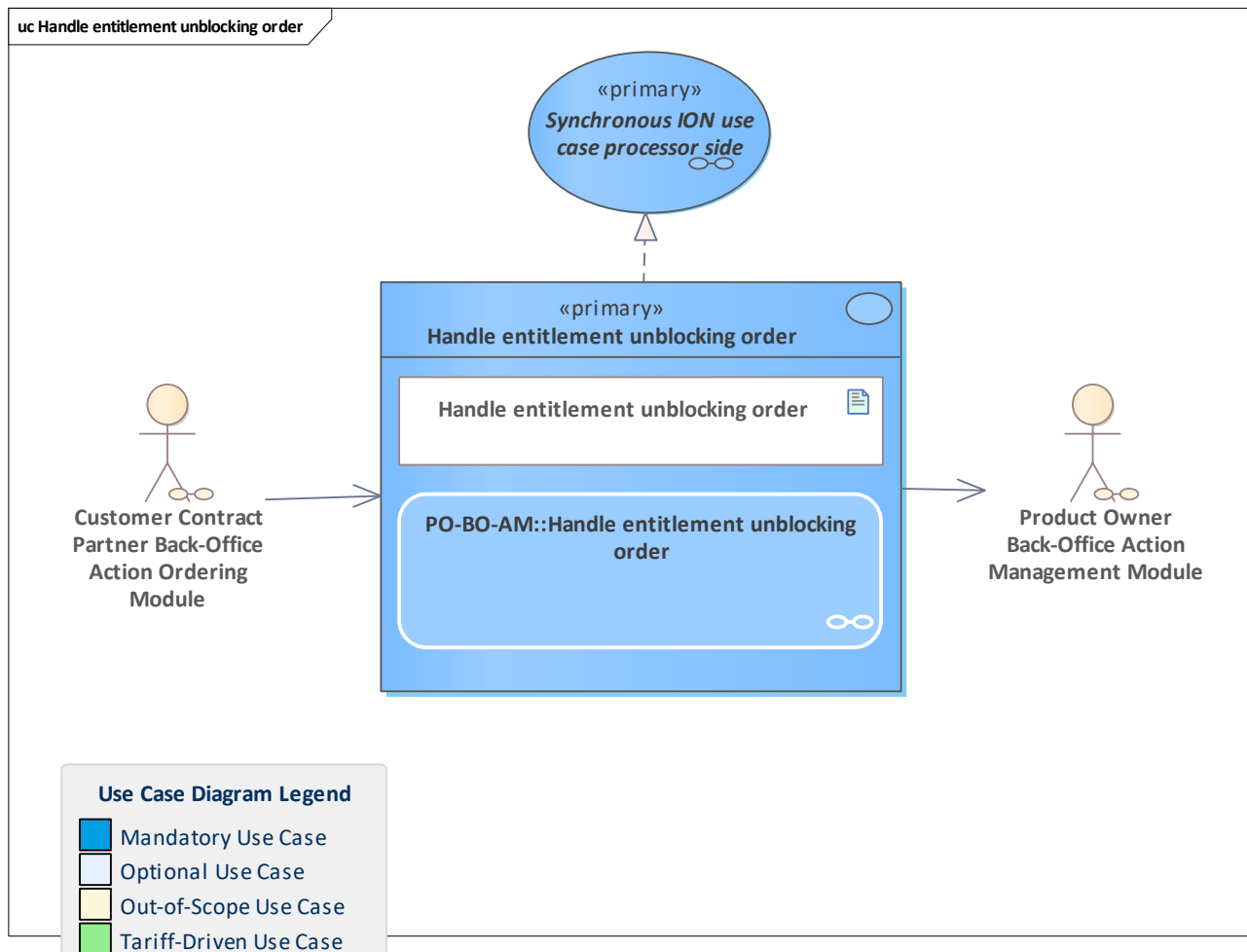


Use Case	Handle entitlement issuance order
Description	The PO back-office system with an action management extension handles an entitlement issuance order. If the order passes all checks, it is added to the order inventory and may be considered for the next action list.
Initiating Actor	Customer Contract Partner Back-Office Action Ordering Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Order entitlement issuance : orderEntitlementIssuance
Outputs	Order entitlement issuance response : orderEntitlementIssuanceResponse
Error Cases	E_POOAM_DUPLICATE_ORDER_ID E_POOAM_SENDER_TO_ORDER_ID_MISMATCH



	E POOAM INVALID PRODUCT PARAMETERS E POOAM ACTION ORDER EXPIRED E POOAM INVALID METADATA E POOAM INVALID ISSUANCE PARAMETERS E POOAM INVALID MAXIMUM BALANCE E POOAM ENTITLEMENT ALREADY REPLACED E CO APP INSTANCE ID UNKNOWN E CO THIRD PARTY SYSTEM ERROR E CO WRONG SECURITY LEVEL Order entitlement issuance exception : orderEntitlementIssuanceException
Activity Diagram	PO-BO-AM::Handle entitlement issuance order

13.2.2 Handle entitlement unblocking order

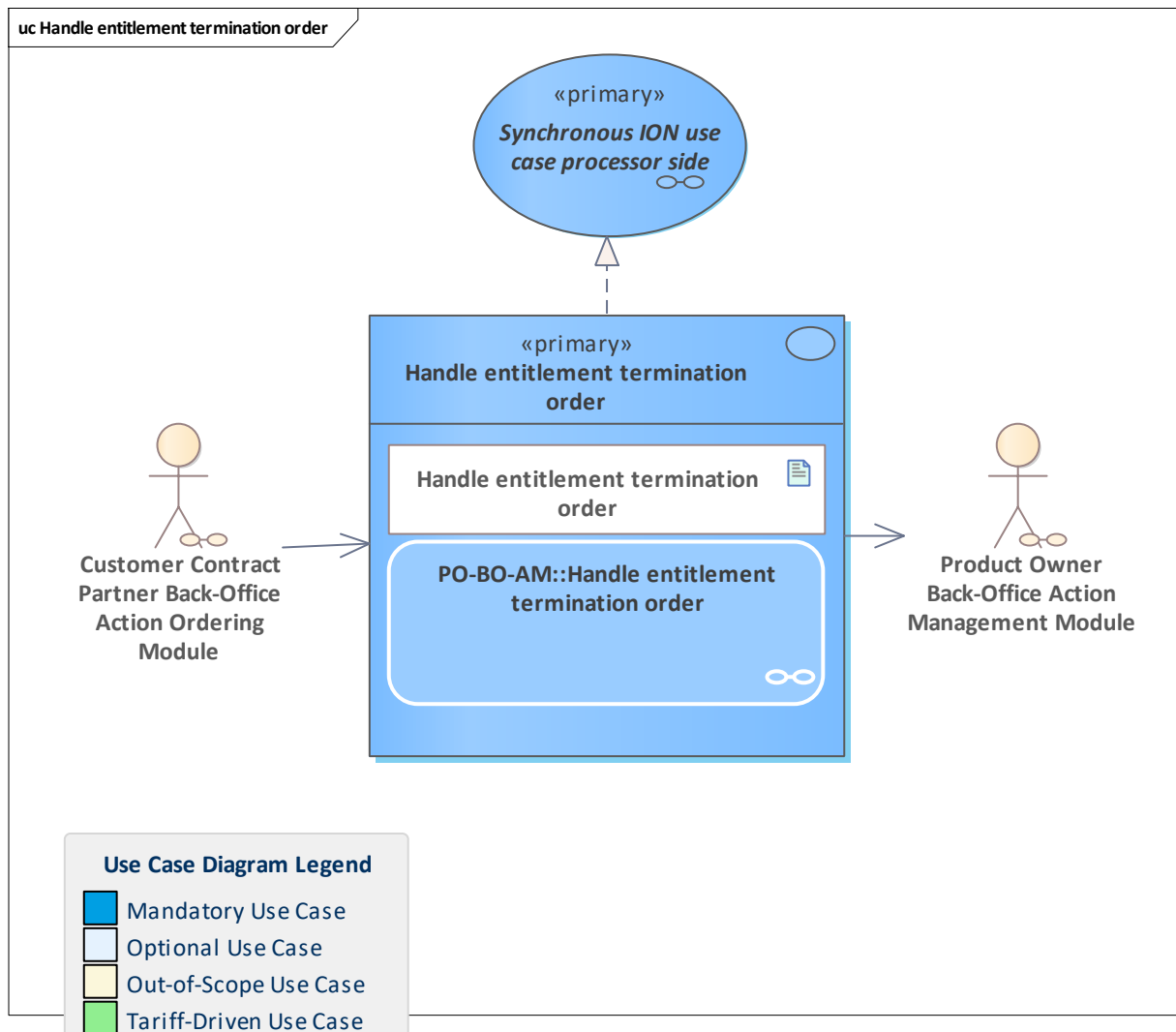


Use Case	Handle entitlement unblocking order
Description	The Product Owner Back-Office Action Management Module handles an entitlement unblocking order. If the order passes all checks, it is added to the order inventory and may be considered for the next action list.
Initiating Actor	Customer Contract Partner Back-Office Action Ordering Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Order entitlement unblocking : orderEntitlementUnblocking
Outputs	Order entitlement unblocking response : orderEntitlementUnblockingResponse



Error Cases	<u>E POOAM SENDER TO ORDER ID MISMATCH</u> <u>E POOAM DUPLICATE ORDER ID</u> <u>E POOAM ACTION ORDER EXPIRED</u> <u>E POOAM CONFLICTING ACTION ORDER EXISTS</u> <u>E POOAM SENDER NOT ENTITLEMENT OWNER</u> <u>E CO WRONG SECURITY LEVEL</u> <u>E CO WRONG SECURITY LEVEL</u> <u>Order entitlement unblocking exception :</u> <u>orderEntitlementUnblockingException</u>
Activity Diagram	<u>PO-BO-AM::Handle entitlement unblocking order</u>

13.2.3 Handle entitlement termination order

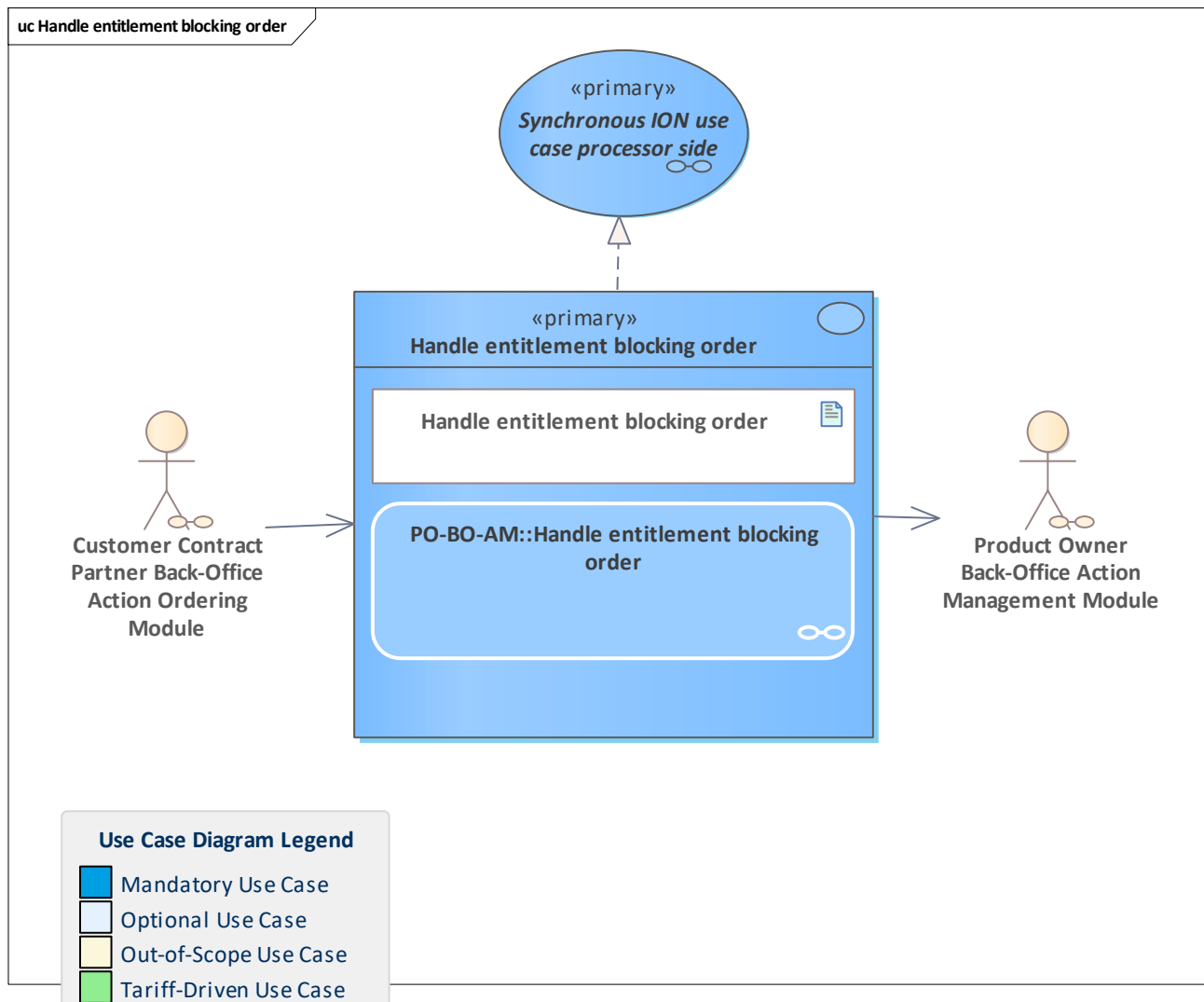


Use Case	Handle entitlement termination order
Description	The PO back-office system with an action management extension handles an entitlement termination order. If the order passes all checks, it is added to the order inventory and may be considered for the next action list.
Initiating Actor	Customer Contract Partner Back-Office Action Ordering Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Order entitlement termination : orderEntitlementTermination



Outputs	<u>Order entitlement termination response :</u> <u>orderEntitlementTerminationResponse</u>
Error Cases	<u>E POOAM DUPLICATE ORDER ID</u> <u>E POOAM SENDER TO ORDER ID MISMATCH</u> <u>E POOAM SENDER NOT ENTITLEMENT OWNER</u> <u>E POOAM CONFLICTING ACTION ORDER EXISTS</u> <u>E POOAM ACTION ORDER EXPIRED</u> <u>E CO WRONG SECURITY LEVEL</u> <u>E CO WRONG SECURITY LEVEL</u> <u>Order entitlement termination exception :</u> <u>orderEntitlementTerminationException</u>
Activity Diagram	<u>PO-BO-AM::Handle entitlement termination order</u>

13.2.4 Handle entitlement blocking order

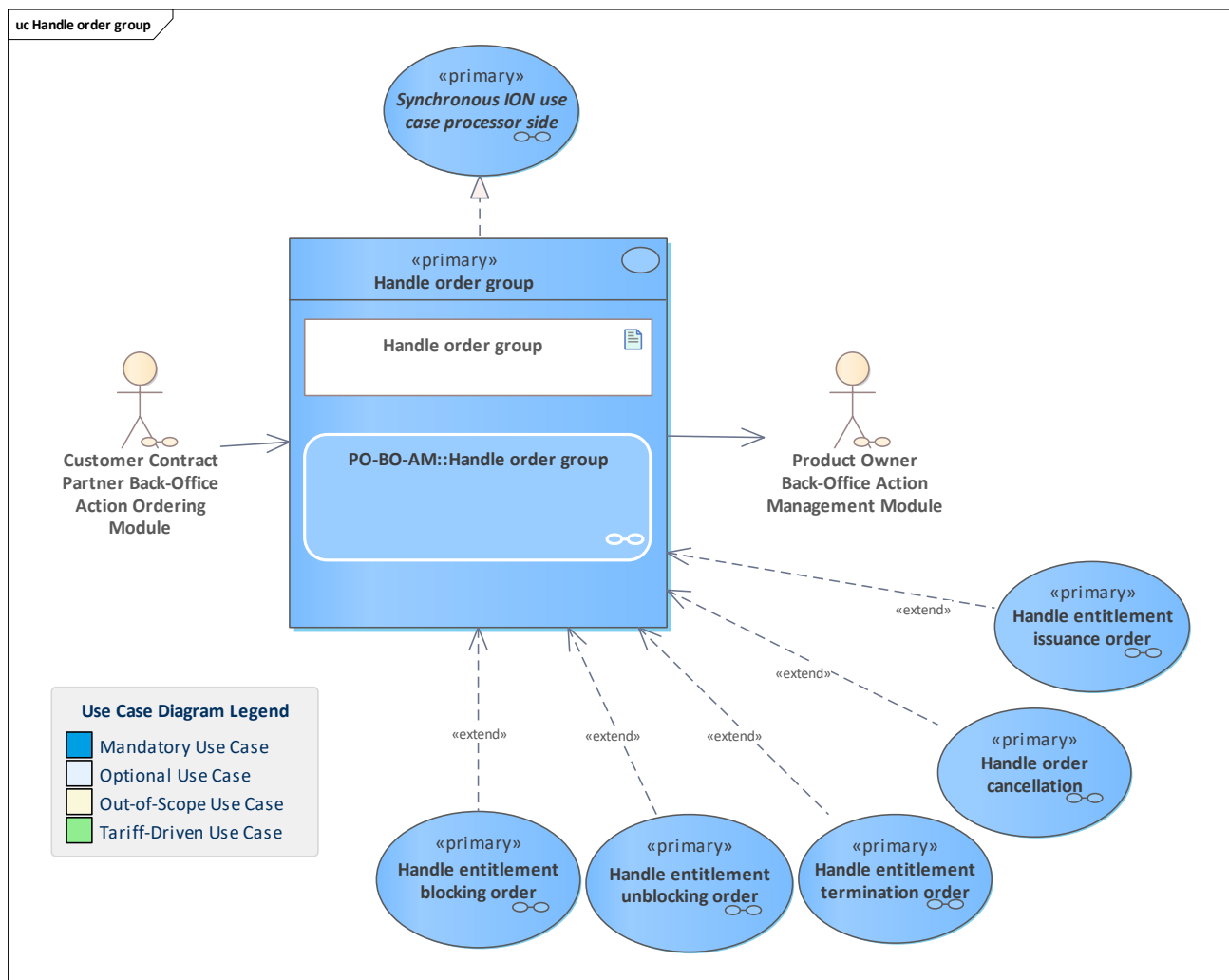


Use Case	Handle entitlement blocking order
Description	The PO back-office system with an action management extension handles an entitlement blocking order. If the order passes all checks, it is added to the order inventory and may be considered for the next action list.
Initiating Actor	Customer Contract Partner Back-Office Action Ordering Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Order entitlement blocking : orderEntitlementBlocking



Outputs	<u>Order entitlement blocking response :</u> <u>orderEntitlementBlockingResponse</u>
Error Cases	<u>E POOAM SENDER TO ORDER ID MISMATCH</u> <u>E POOAM DUPLICATE ORDER ID</u> <u>E POOAM ACTION ORDER EXPIRED</u> <u>E POOAM SENDER TO ORDER ID MISMATCH</u> <u>E CO WRONG SECURITY LEVEL</u> <u>Order entitlement blocking exception :</u> <u>orderEntitlementBlockingException</u>
Activity Diagram	<u>PO-BO-AM::Handle entitlement blocking order</u>

13.2.5 Handle order group

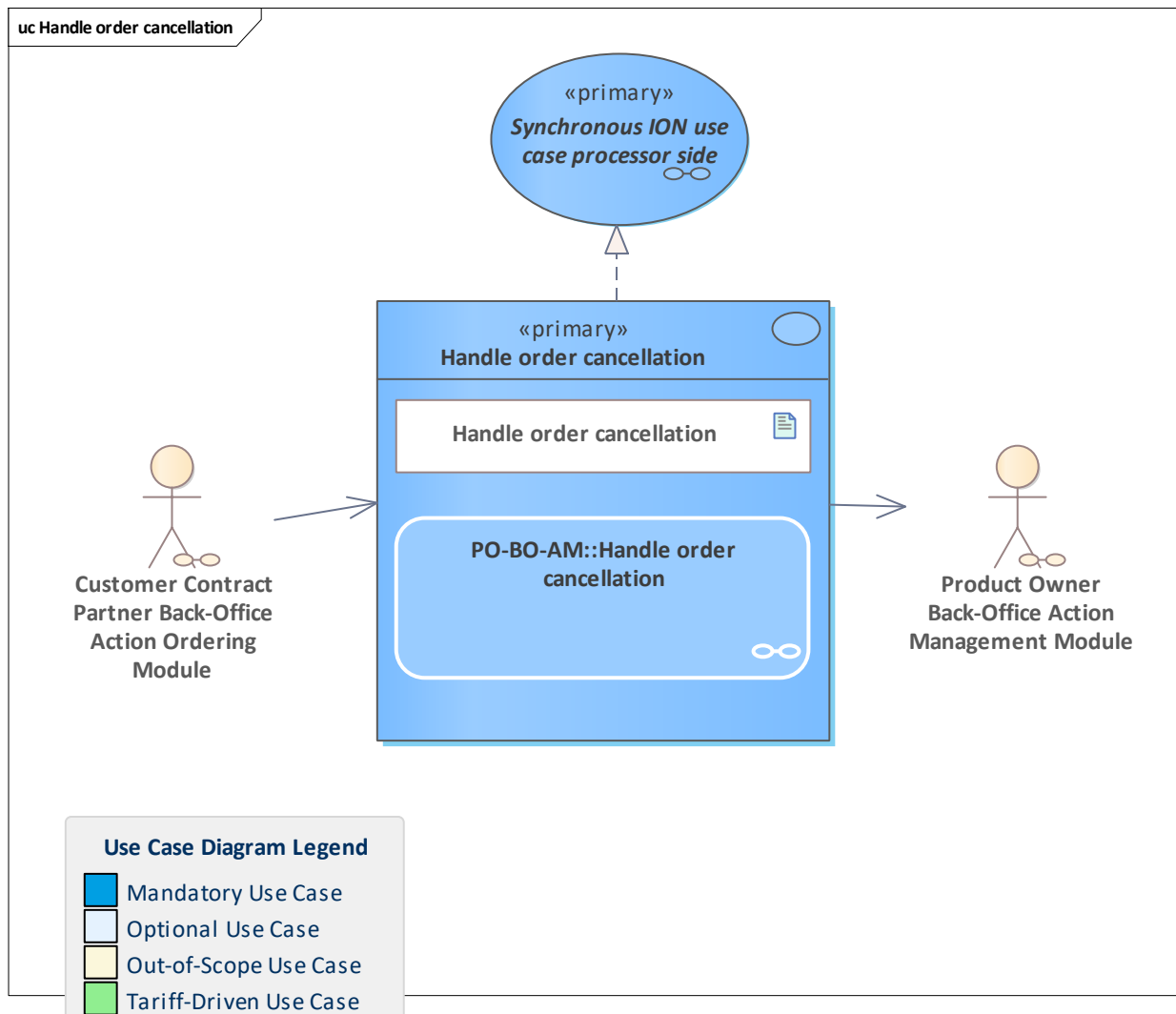


Use Case	<u>Handle order group</u>
Description	The <u>Product Owner Back-Office Action Management Module</u> handles an order group. Grouping the orders guarantees that either all grouped orders will be accepted or all grouped orders will be declined. If the order group passes all checks, the contained orders are added to the order inventory and will be considered for the next action list.
Initiating Actor	<u>Customer Contract Partner Back-Office Action Ordering Module</u>
Reacting Actor	<u>Product Owner Back-Office Action Management Module</u>
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	<u>Handle entitlement issuance order</u> / <u>Handle order cancellation</u> / <u>Handle entitlement termination order</u> / <u>Handle entitlement unblocking order</u> / <u>Handle entitlement blocking order</u>
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	<u>Synchronous ION use case processor side</u> / <u>Synchronous ION use case processor side</u> / <u>Synchronous ION use case processor side</u> /



	Synchronous ION use case processor side / Synchronous ION use case processor side / Synchronous ION use case processor side
Base Activity	
Inputs	Order group : orderGroup
Outputs	Order group response : orderGroupResponse
Error Cases	E POOAM APP INSTANCE ID NOT IDENTICAL OVER GROUP E POOAM GROUP ID NOT IDENTICAL OVER GROUP E POOAM DUPLICATE ORDER ID E POOAM SENDER TO ORDER ID MISMATCH E POOAM INVALID PRODUCT PARAMETERS E POOAM ACTION ORDER EXPIRED E POOAM INVALID METADATA E POOAM INVALID ISSUANCE PARAMETERS E POOAM INVALID MAXIMUM BALANCE E POOAM ENTITLEMENT ALREADY REPLACED E CO APP INSTANCE ID UNKNOWN E CO THIRD PARTY SYSTEM ERROR E CO WRONG SECURITY LEVEL E POOAM SENDER TO ORDER ID MISMATCH E POOAM CONFLICTING ACTION ORDER EXISTS E POOAM SENDER NOT ENTITLEMENT OWNER E CO WRONG SECURITY LEVEL E POOAM ORDER ALREADY EXECUTED E POOAM REFERENCED ACTION ORDER UNKNOWN E POOAM ORDER ALREADY CANCELLED Order group exception : orderGroupException
Activity Diagram	PO-BO-AM::Handle order group

13.2.6 Handle order cancellation

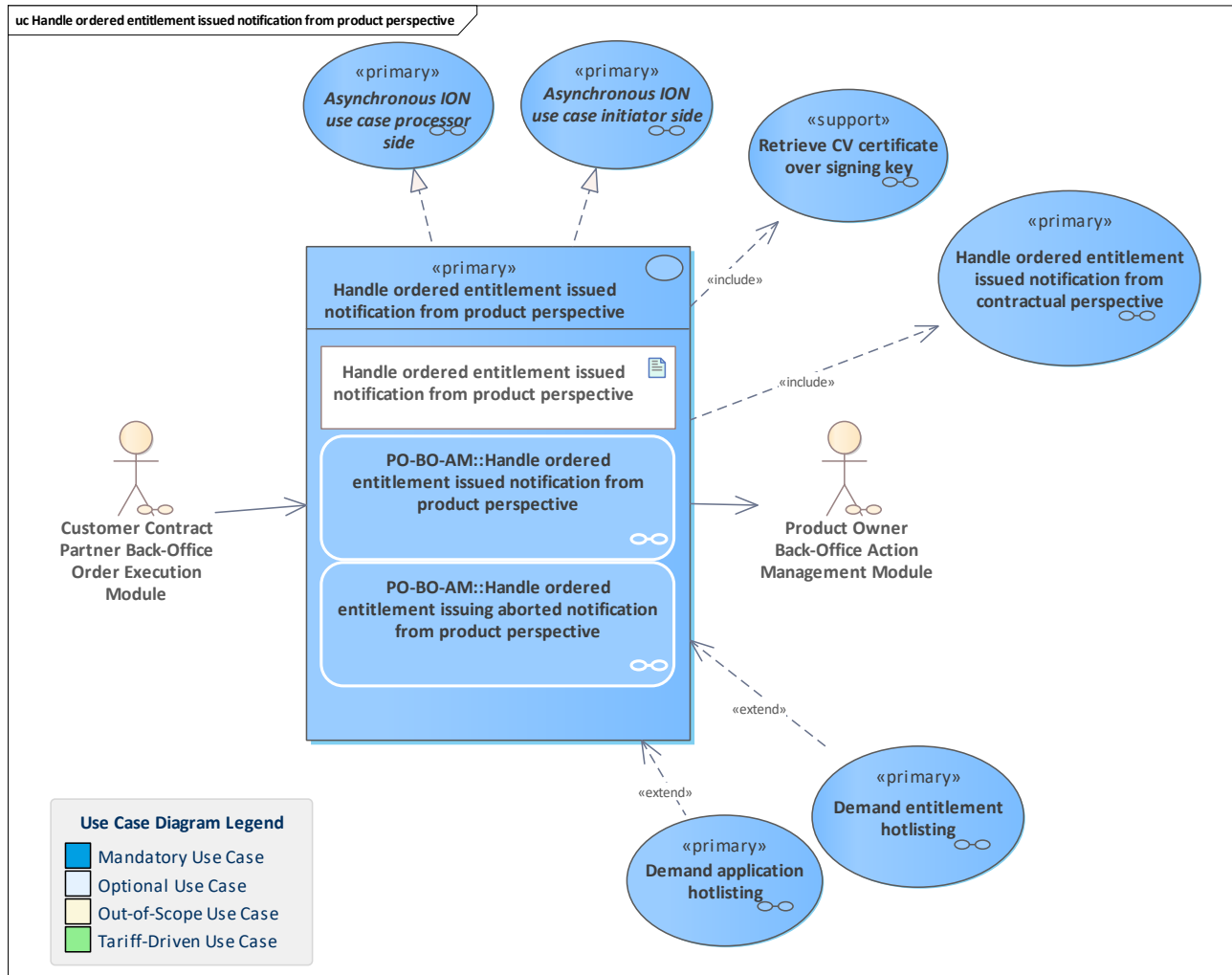


Use Case	Handle order cancellation
Description	The Product Owner Back-Office Action Management Module handles an order cancellation. If the cancellation passes all checks, the referenced order is marked as cancelled in the order inventory and will be removed from the next action list.
Initiating Actor	Customer Contract Partner Back-Office Action Ordering Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	



Inputs	Cancel order : cancelOrder
Outputs	Cancel order response : cancelOrderResponse
Error Cases	E POOAM ACTION ORDER EXPIRED E POOAM ORDER ALREADY EXECUTED E POOAM REFERENCED ACTION ORDER UNKNOWN E POOAM SENDER TO ORDER ID MISMATCH E POOAM ORDER ALREADY CANCELLED Cancel order exception : cancelOrderException
Activity Diagram	PO-BO-AM::Handle order cancellation

13.2.7 Handle ordered entitlement issued notification from product perspective

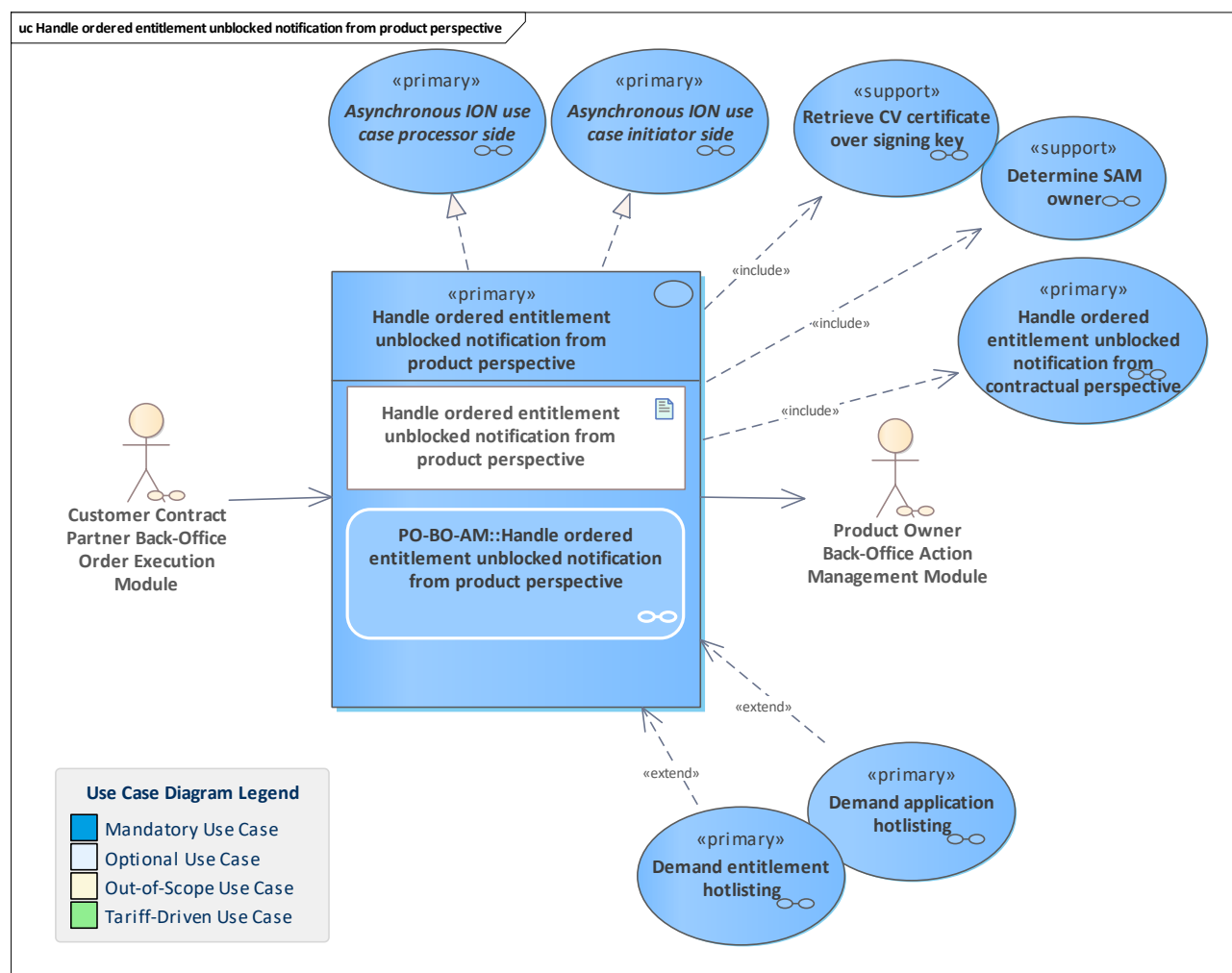


Use Case	Handle ordered entitlement issued notification from product perspective
Description	<p>Handle the notification about a successful execution of an entitlement issuance ordered via action management from the product owner perspective.</p> <p>The PO back-office system receives the notification about an ordered entitlement issuance and handles it from the product perspective. It registers the notification and performs checks and monitoring.</p> <p>Finally, the notification is forwarded to the CCP that initiated the ordered action.</p> <p>In the case of an abortion notification, the PO system has to register the SAM and product issuance counter for consistent monitoring.</p> <p>Note that the application instance ID of the user medium the entitlement was issued to can be uniquely identified via <i>umAppInstanceId</i>, which is part of the SignedEntitlementIssuedAttestation that is contained in the</p>



	OrderedEntitlementIssuedNotification.
Initiating Actor	Customer Contract Partner Back-Office Order Execution Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand entitlement hotlisting / Demand application hotlisting / Demand entitlement hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle ordered entitlement issued notification from contractual perspective / Retrieve CV certificate over signing key / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case initiator side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify ordered entitlement issued : notifyOrderedEntitlementIssued
Outputs	Notify ordered entitlement issued response : notifyOrderedEntitlementIssuedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify ordered entitlement issued exception : notifyOrderedEntitlementIssuedException
Activity Diagram	PO-BO-AM::Handle ordered entitlement issued notification from product perspective
Alternative 1	
Inputs	Notify ordered entitlement issuing aborted : notifyOrderedEntitlementIssuingAborted
Outputs	Notify ordered entitlement issuing aborted response : notifyOrderedEntitlementIssuingAbortedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO WRONG SECURITY LEVEL Notify ordered entitlement issuing aborted exception : notifyOrderedEntitlementIssuingAbortedException
Activity Diagram	PO-BO-AM::Handle ordered entitlement issuing aborted notification from product perspective

13.2.8 Handle ordered entitlement unblocked notification from product perspective



Use Case	Handle ordered entitlement unblocked notification from product perspective
Description	<p>Handle the notification about a successful execution of an entitlement unblocking ordered via action management from the product owner perspective.</p> <p>The ordered entitlement unblocked notification is received by the PO.</p> <p>The PO registers the entitlement unblocked notification and does its checks and monitoring from the product perspective regarding the correct execution of the blocking. In this context, the signature of the embedded attestation is verified and the SAM owner of the SAM that performed the action is determined.</p> <p>Finally, the notification is forwarded to the ordering CCP.</p> <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case as initiator due to the asynchronous call to the ordering CCP.</p>
Initiating Actor	Customer Contract Partner Back-Office Order Execution Module
Reacting Actor	Product Owner Back-Office Action Management Module

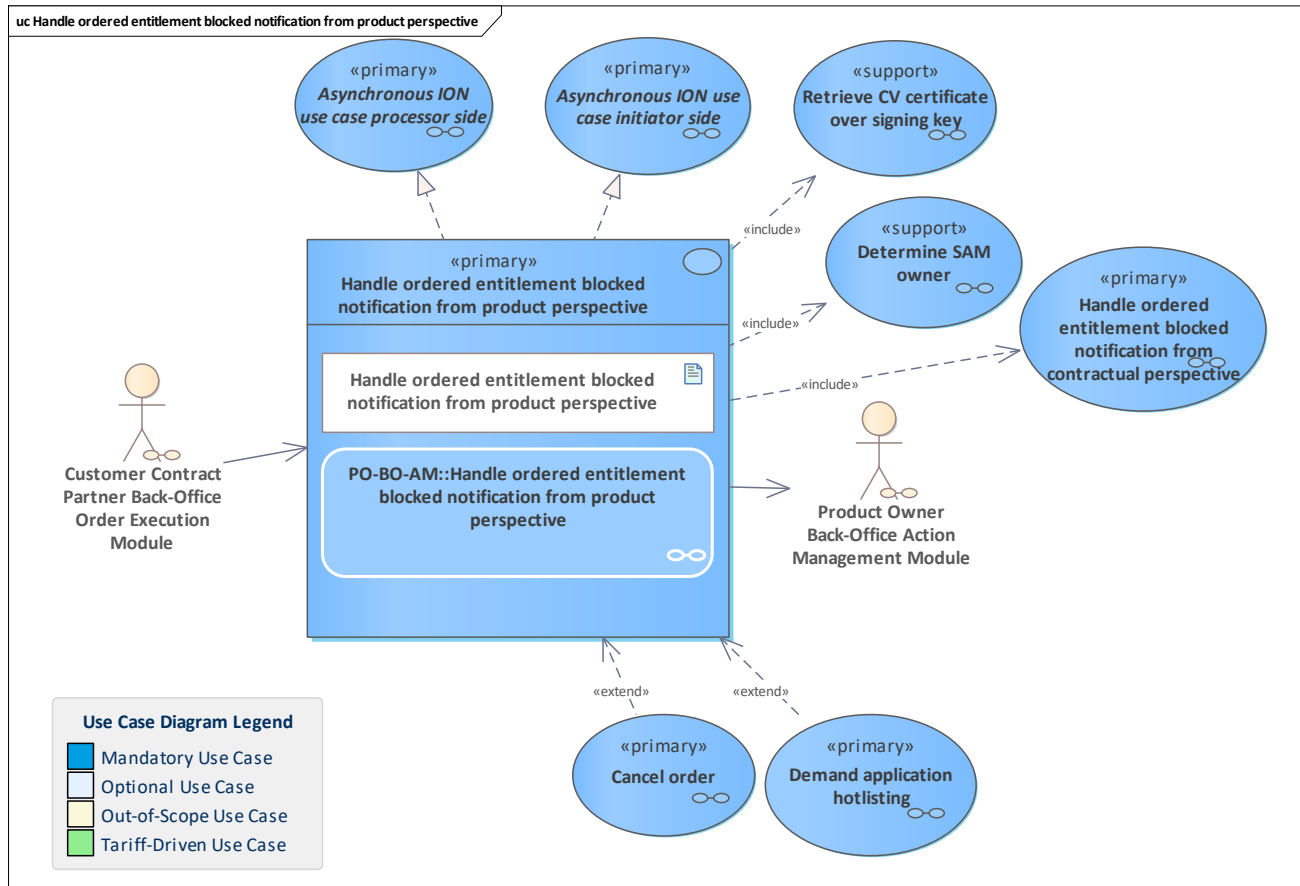


Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand entitlement hotlisting / Demand entitlement hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle ordered entitlement unblocked notification from contractual perspective / Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify ordered entitlement unblocked : notifyOrderedEntitlementUnblocked
Outputs	Notify ordered entitlement unblocked response : notifyOrderedEntitlementUnblockedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify ordered entitlement unblocked exception : notifyOrderedEntitlementUnblockedException
Activity Diagram	PO-BO-AM::Handle ordered entitlement unblocked notification from product perspective



Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting
Linked Use Cases (Includes)	Handle ordered entitlement terminated notification from contractual perspective / Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side
Base Activity	
Inputs	Notify ordered entitlement terminated : notifyOrderedEntitlementTerminated
Outputs	Notify ordered entitlement terminated response : notifyOrderedEntitlementTerminatedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify ordered entitlement terminated exception : notifyOrderedEntitlementTerminatedException
Activity Diagram	PO-BO-AM::Handle ordered entitlement terminated notification from product perspective

13.2.10 Handle ordered entitlement blocked notification from product perspective

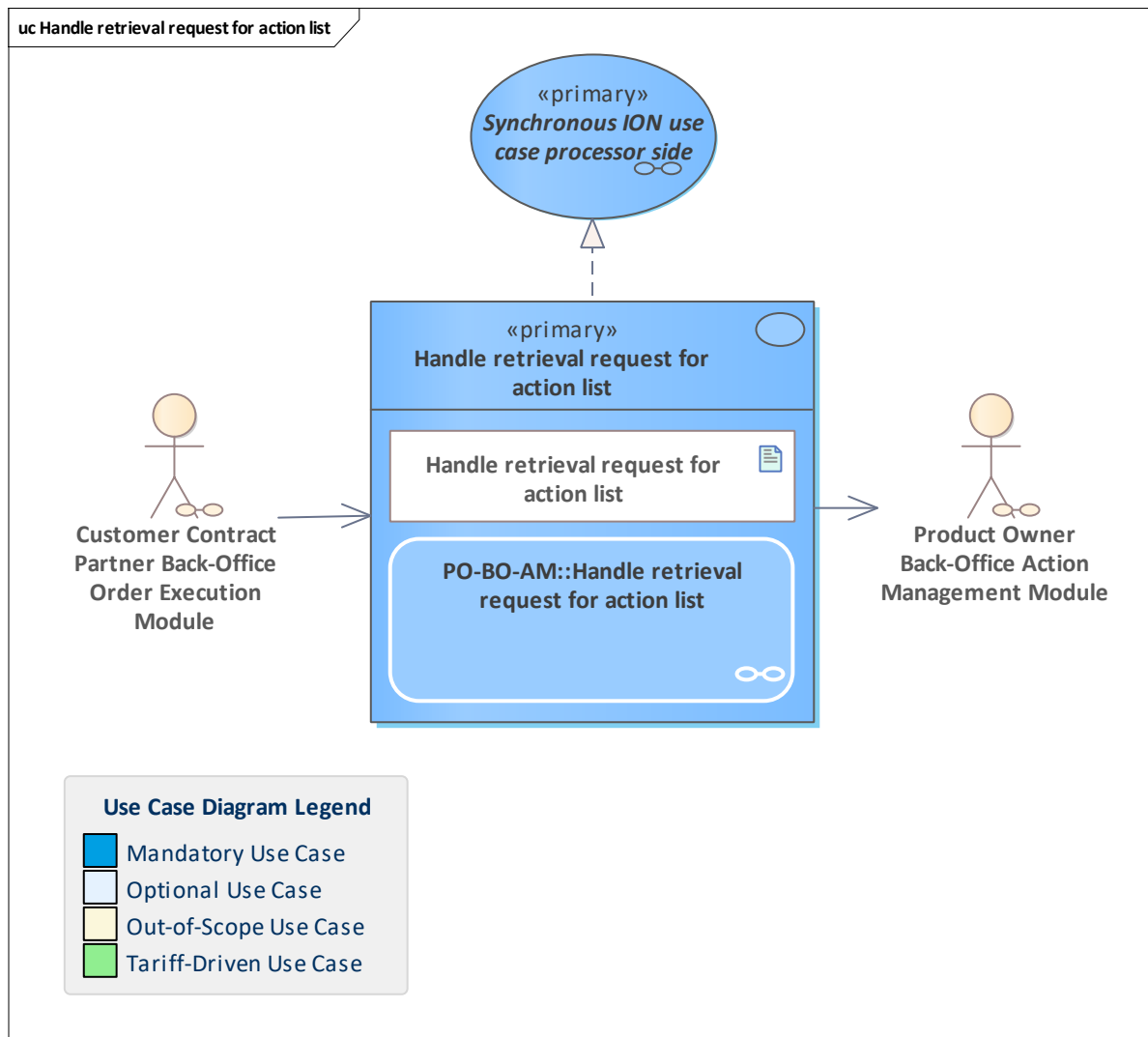


Use Case	Handle ordered entitlement blocked notification from product perspective
Description	<p>Handle the notification about a successful execution of an entitlement blocking ordered via the action management from product owner perspective.</p> <p>The ordered entitlement blocked notification is received by the PO. The PO registers the entitlement blocked notification and does its checks and monitoring from the product perspective regarding the correct execution of the blocking. In this context, the signature of the embedded attestation is verified and the SAM owner of the SAM that performed the action is determined.</p> <p>Finally, the notification is forwarded to the ordering CCP.</p> <p>Note: in the ION context, the use case asynchronous as processor (process the notification) and an asynchronous use case as initiator due to the asynchronous call to the ordering CCP.</p>
Initiating Actor	Customer Contract Partner Back-Office Order Execution Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting / Cancel order
Linked Use Cases	Handle ordered entitlement blocked notification from contractual



(Includes)	perspective / Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify ordered entitlement blocked : notifyOrderedEntitlementBlocked
Outputs	Notify ordered entitlement blocked response : notifyOrderedEntitlementBlockedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify ordered entitlement blocked exception : notifyOrderedEntitlementBlockedException
Activity Diagram	PO-BO-AM::Handle ordered entitlement blocked notification from product perspective

13.2.11 Handle retrieval request for action list

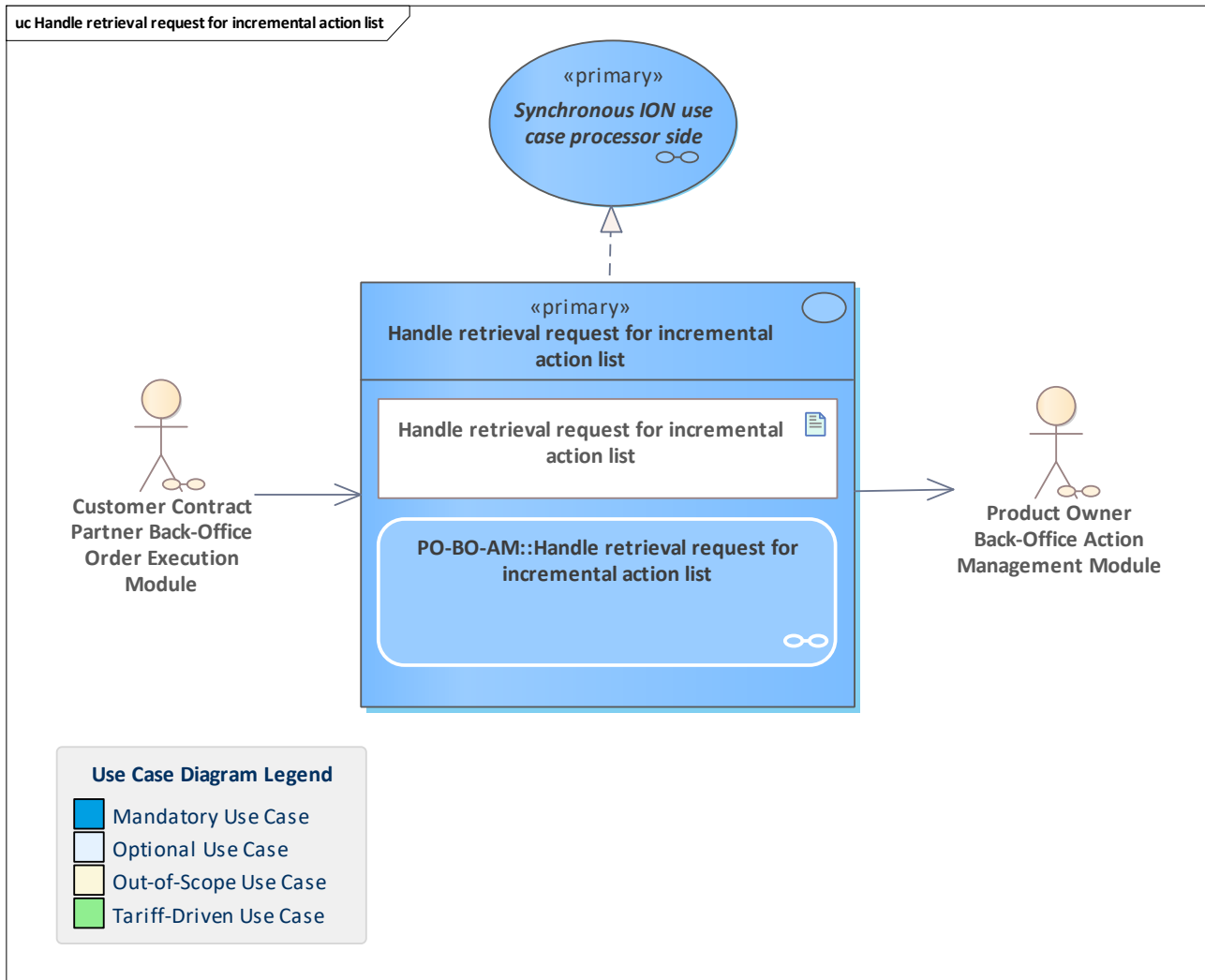


Use Case	Handle retrieval request for action list
Description	The Product Owner Back-Office Action Management Module provides the current full action list for the Customer Contract Partner Back-Office Order Execution Module .
Initiating Actor	Customer Contract Partner Back-Office Order Execution Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Get action list : getActionList



Outputs	Get action list response : getActionListResponse
Error Cases	E_POOAM_RETRIEVAL_PERMISSION_MISSING E_POOAM_STILL_GENERATING_ACTION_LIST Get action list exception : getActionListException
Activity Diagram	PO-BO-AM::Handle retrieval request for action list

13.2.12 Handle retrieval request for incremental action list

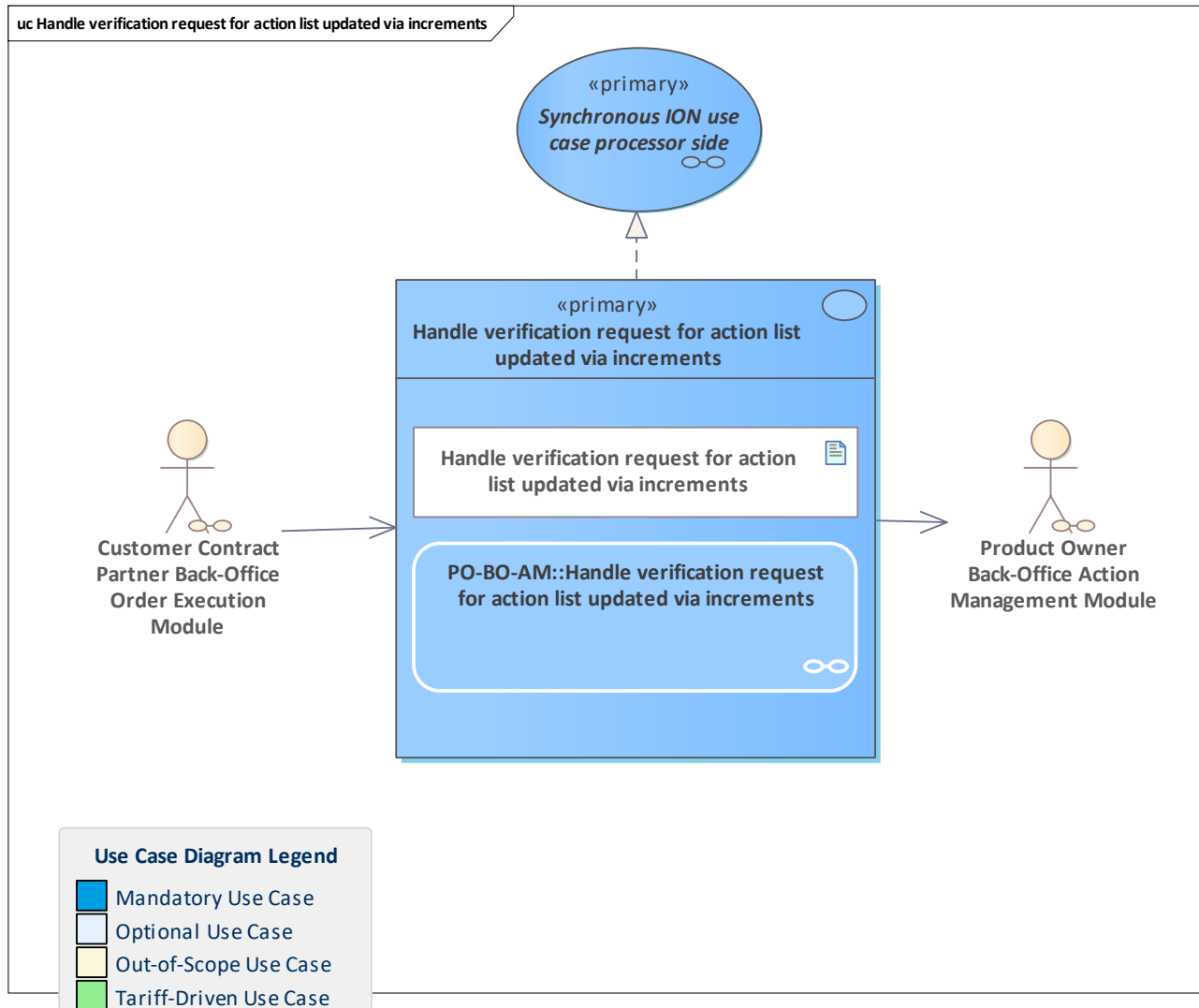


Use Case	Handle retrieval request for incremental action list
Description	The Product Owner Back-Office Action Management Module provides the incremental action list covering the cycles starting with the given cycle up to the latest cycle for the Customer Contract Partner Back-Office Order Execution Module .
Initiating Actor	Customer Contract Partner Back-Office Order Execution Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	



Inputs	Get incremental action list : getIncrementalActionList
Outputs	Get incremental action list response : getIncrementalActionListResponse
Error Cases	E_POOAM_RETRIEVAL_PERMISSION_MISSING E_POOAM_STILL_GENERATING_ACTION_LIST E_CO_CYCLE_NUMBER_UNKNOWN_OR_TOO_OLD Get incremental action list exception : getIncrementalActionListException
Activity Diagram	PO-BO-AM::Handle retrieval request for incremental action list

13.2.13 Handle verification request for action list updated via increments

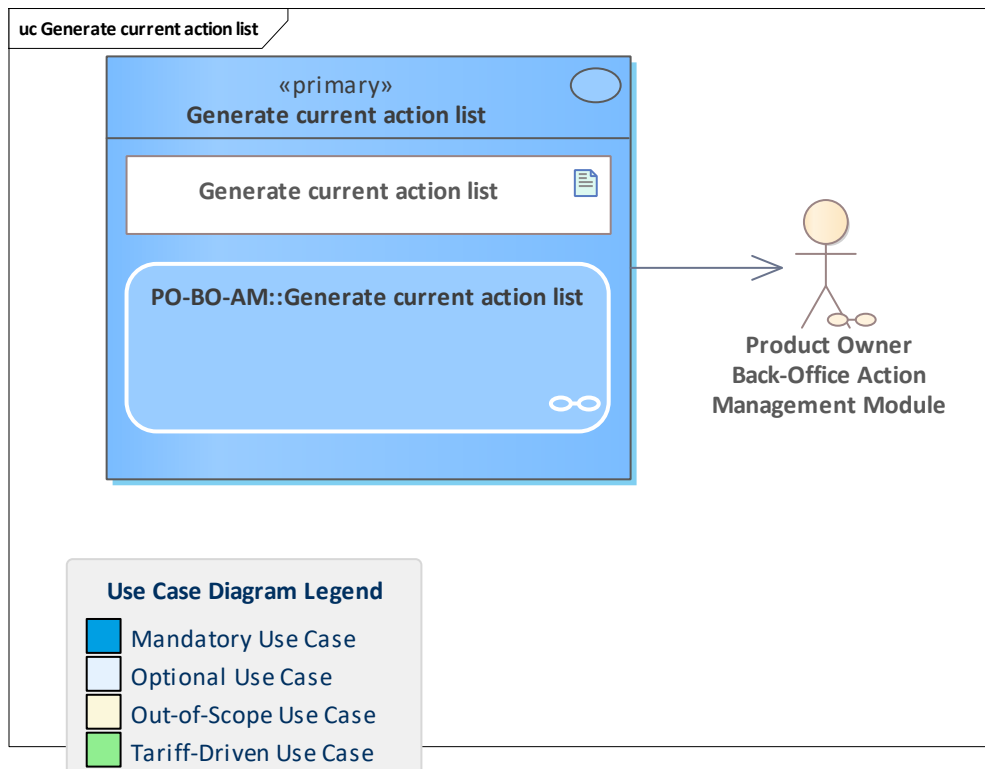


Use Case	Handle verification request for action list updated via increments
Description	<p>The Product Owner Back-Office Action Management Module verifies that the given checksum matches the calculated one of the total list for the given cycle.</p> <p>May be used after an incremental action list was incorporated into a system's inventory of action list entries to verify that this inventory is consistent with the full list that would have been provided by this module.</p> <p>See also Checksum calculation for hotlist and action list verification and Example calculation for an action list inventory.</p>
Initiating Actor	Customer Contract Partner Back-Office Order Execution Module
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	



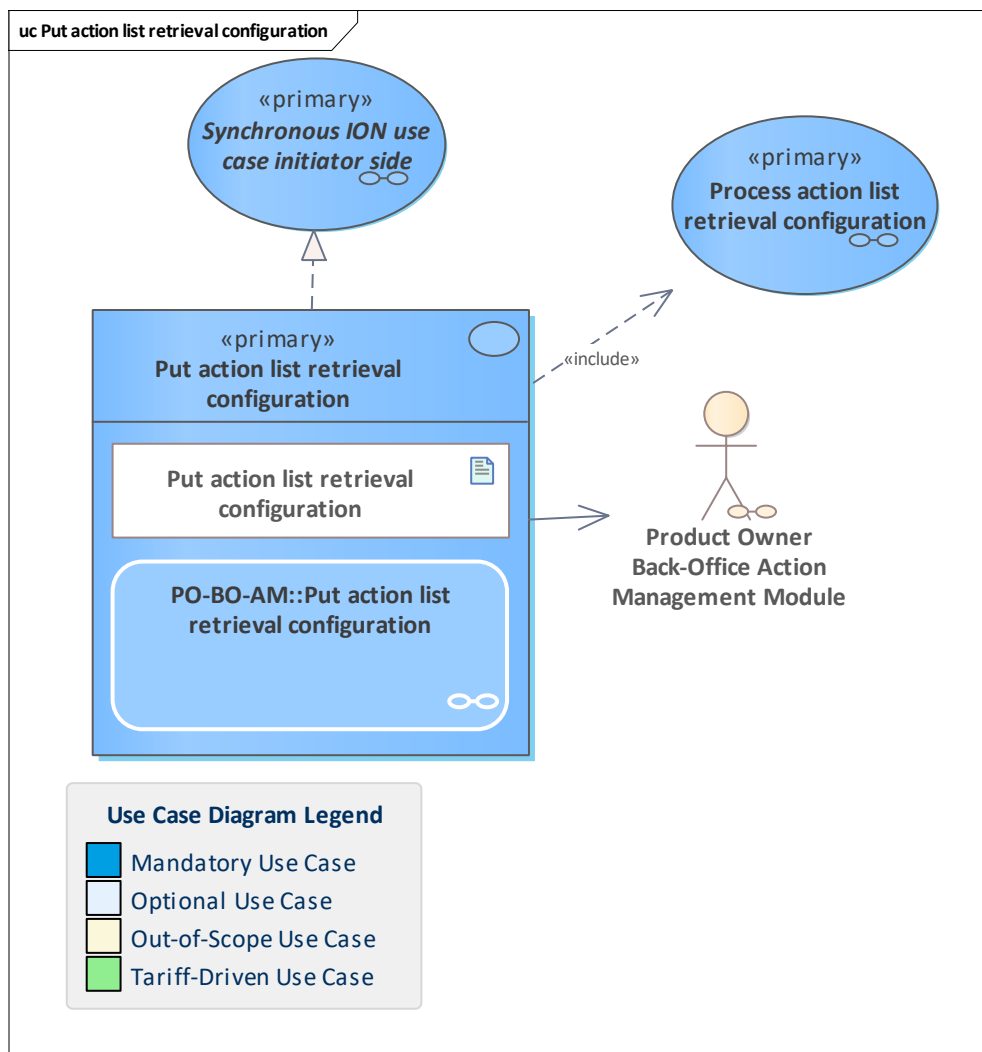
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Verify incremental action list : verifyActionListUpdatedViaIncrements
Outputs	Verify incremental action list response : verifyActionListUpdatedViaIncrementsResponse
Error Cases	E_CO_CYCLE_NUMBER_UNKNOWN_OR_TOO_OLD E_CO_INCORRECT_CHECKSUM Verify incremental action list exception : verifyActionListUpdatedViaIncrementsException
Activity Diagram	PO-BO-AM::Handle verification request for action list updated via increments

13.2.14 Generate current action list



Use Case	Generate current action list
Description	The Product Owner Back-Office Action Management Module generates and stores the action list for a new cycle. This process runs periodically on its own.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-AM::Generate current action list

13.2.15 Put action list retrieval configuration

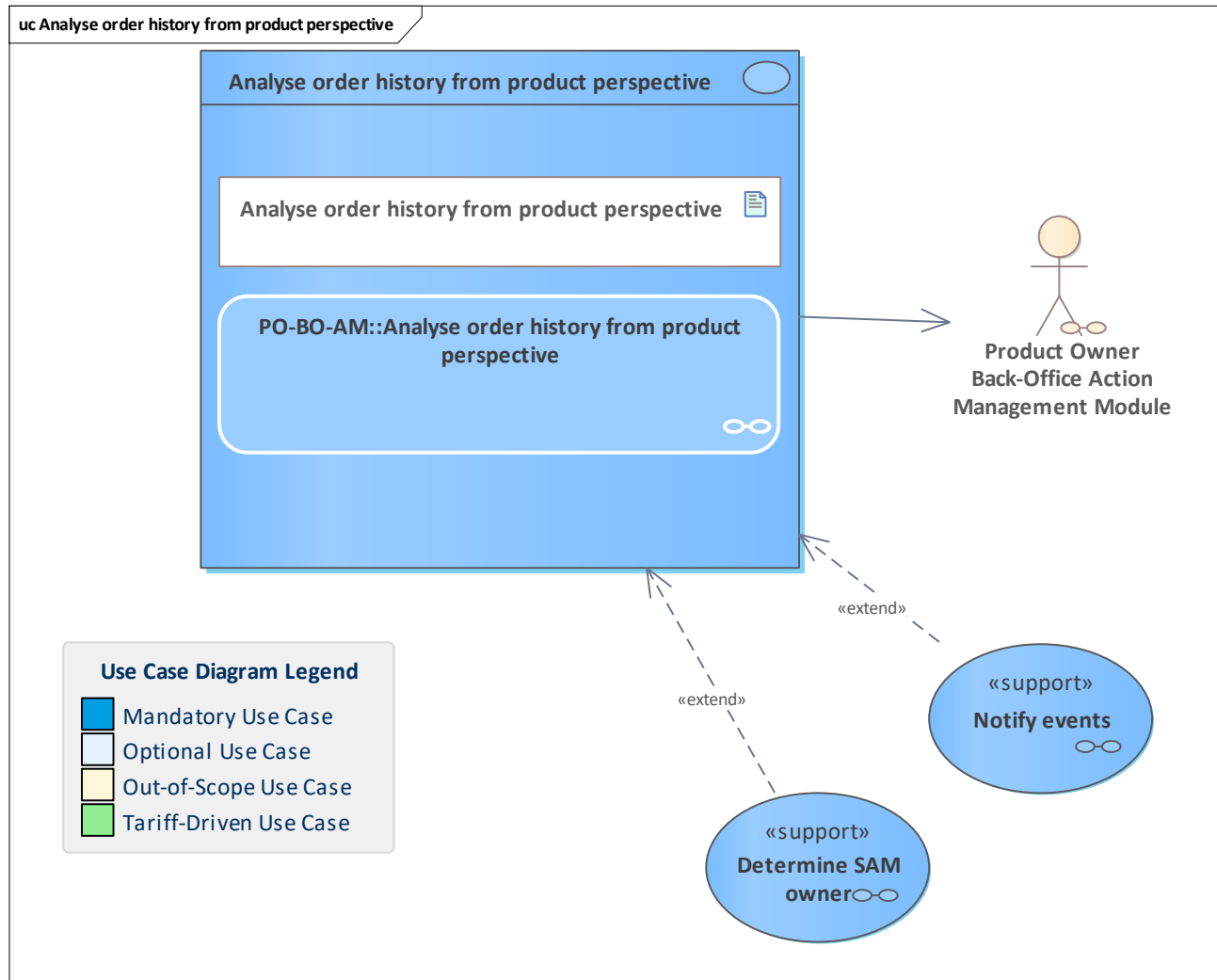


Use Case	Put action list retrieval configuration
Description	The action list retrieval configuration is sent to an/the Customer Contract Partner Back-Office Order Execution Module(s) by the Product Owner Back-Office Action Management Module . The configuration contains information about whether and when the action list is provided in an updated form. It overwrites any previous configurations.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Process action list retrieval configuration
Linked Use Cases (Realises)	Synchronous ION use case initiator side
Base Activity	



Inputs	Action list retrieval configuration : ActionListRetrievalConfiguration CCP organisation ID : OrganisationId
Outputs	
Error Cases	
Activity Diagram	PO-BO-AM::Put action list retrieval configuration

13.2.16 Analyse order history from product perspective

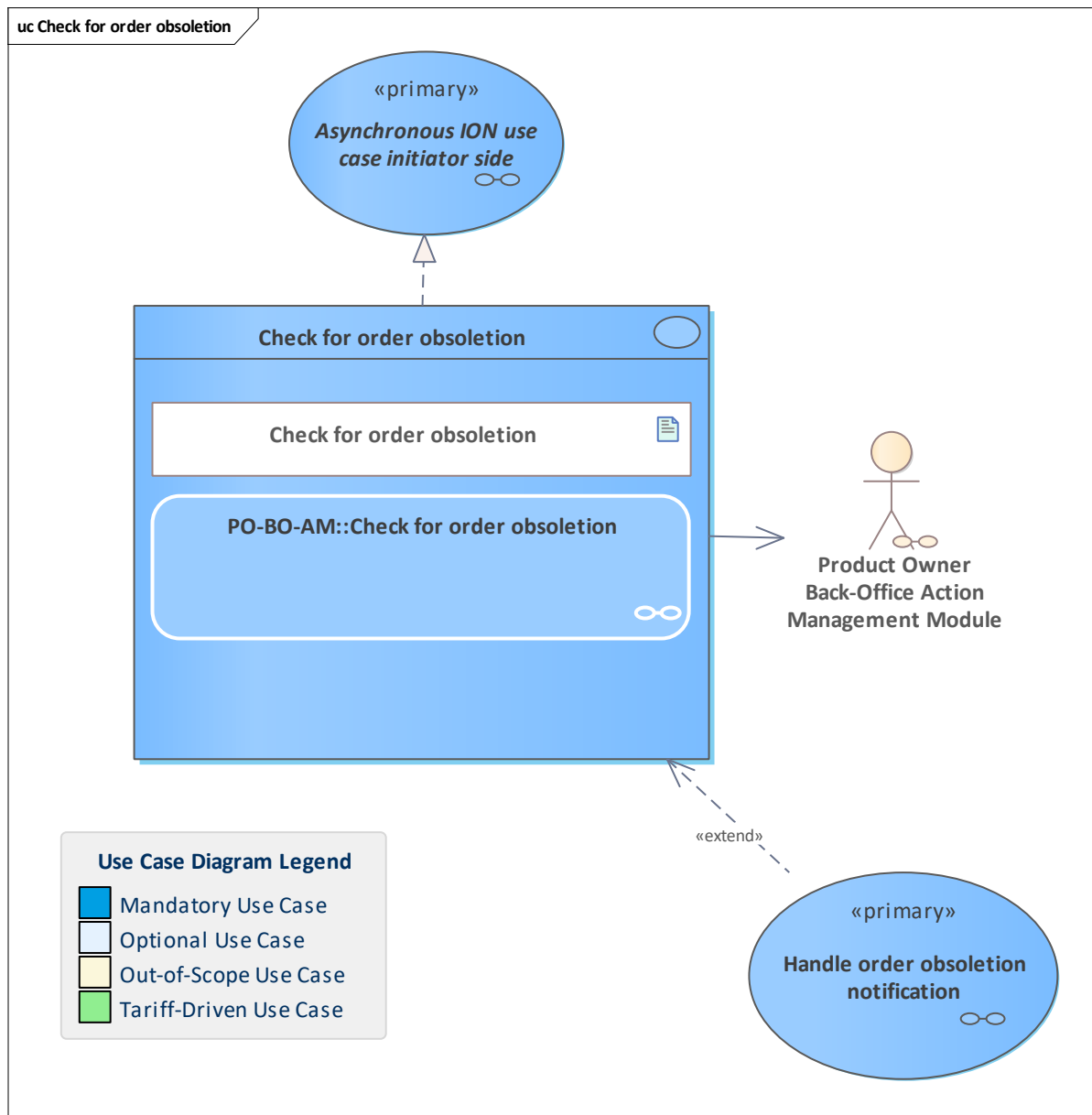


Use Case	Analyse order history from product perspective
Description	The notifications related to action orders are analysed for correctness.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Notify events / Notify events / Determine SAM owner
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	
Base Activity	
Inputs	



Outputs	
Error Cases	
Activity Diagram	<u>PO-BO-AM::Analyse order history from product perspective</u>

13.2.17 Check for order obsolescence



Use Case	Check for order obsolescence
Description	Check if an entitlement notification obsoletes an Active order for unblocking an entitlement. Such an order is marked as Obsolete and the ordering CCP is notified about it. Note that orders in state Obsolete are no longer part of the distributed action lists.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Action Management Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Handle order obsolescence notification
Linked Use Cases (Includes)	



Linked Use Cases (Realises)	Asynchronous ION use case initiator side
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-AM::Check for order obsolescence



14 Static Entitlements Bundle PO-System

Functionality bundle that covers PO back-office system use cases for working with static entitlements.

14.1 Overview

Handle static entitlement issued notification from product perspective

Handle static entitlement inspected notification from product perspective

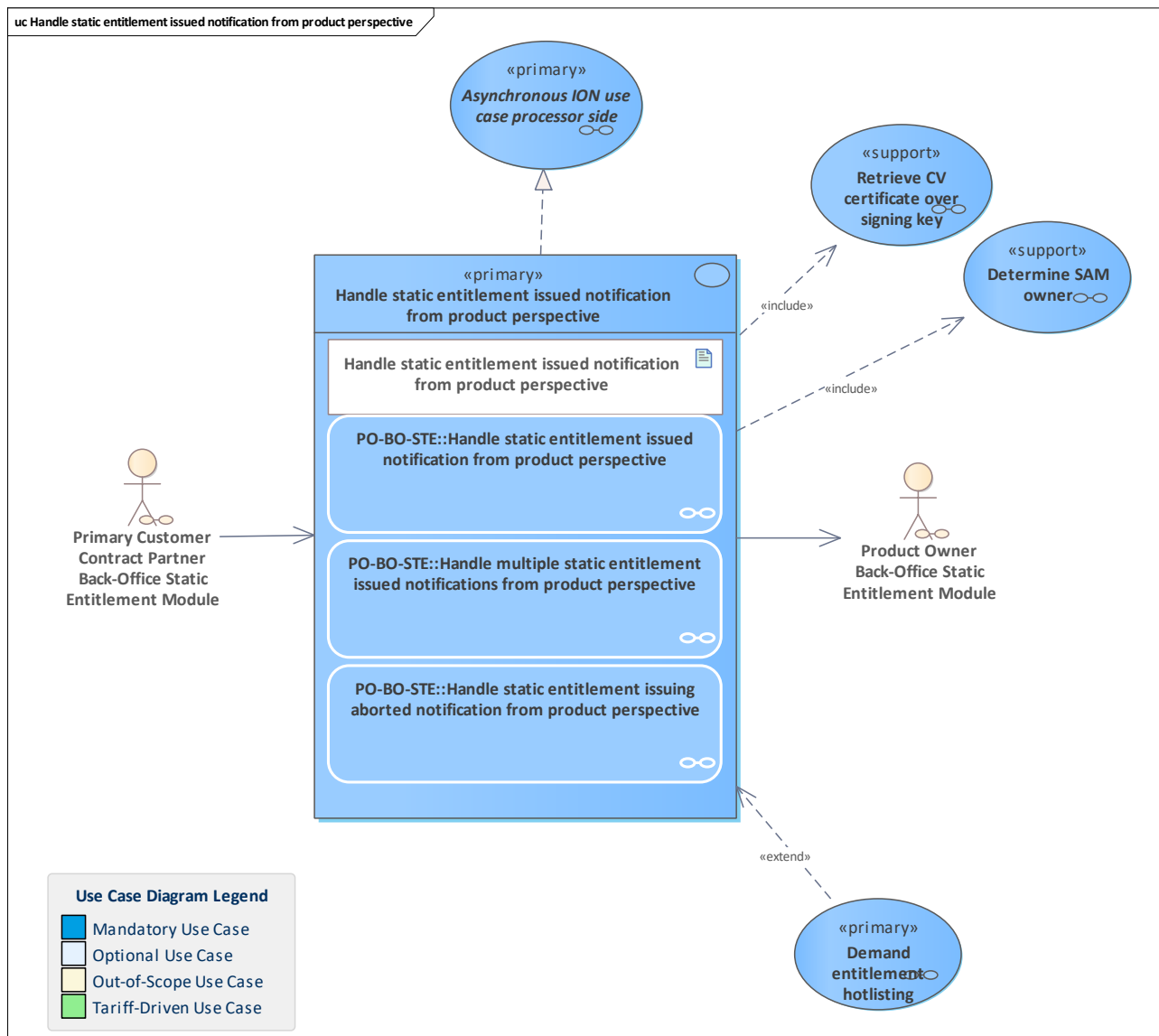
Handle static entitlement terminated notification from product perspective

Check static entitlement notifications against issuance notification from product perspective

Check static entitlement notifications for plausibility

14.2 Use Cases

14.2.1 Handle static entitlement issued notification from product perspective

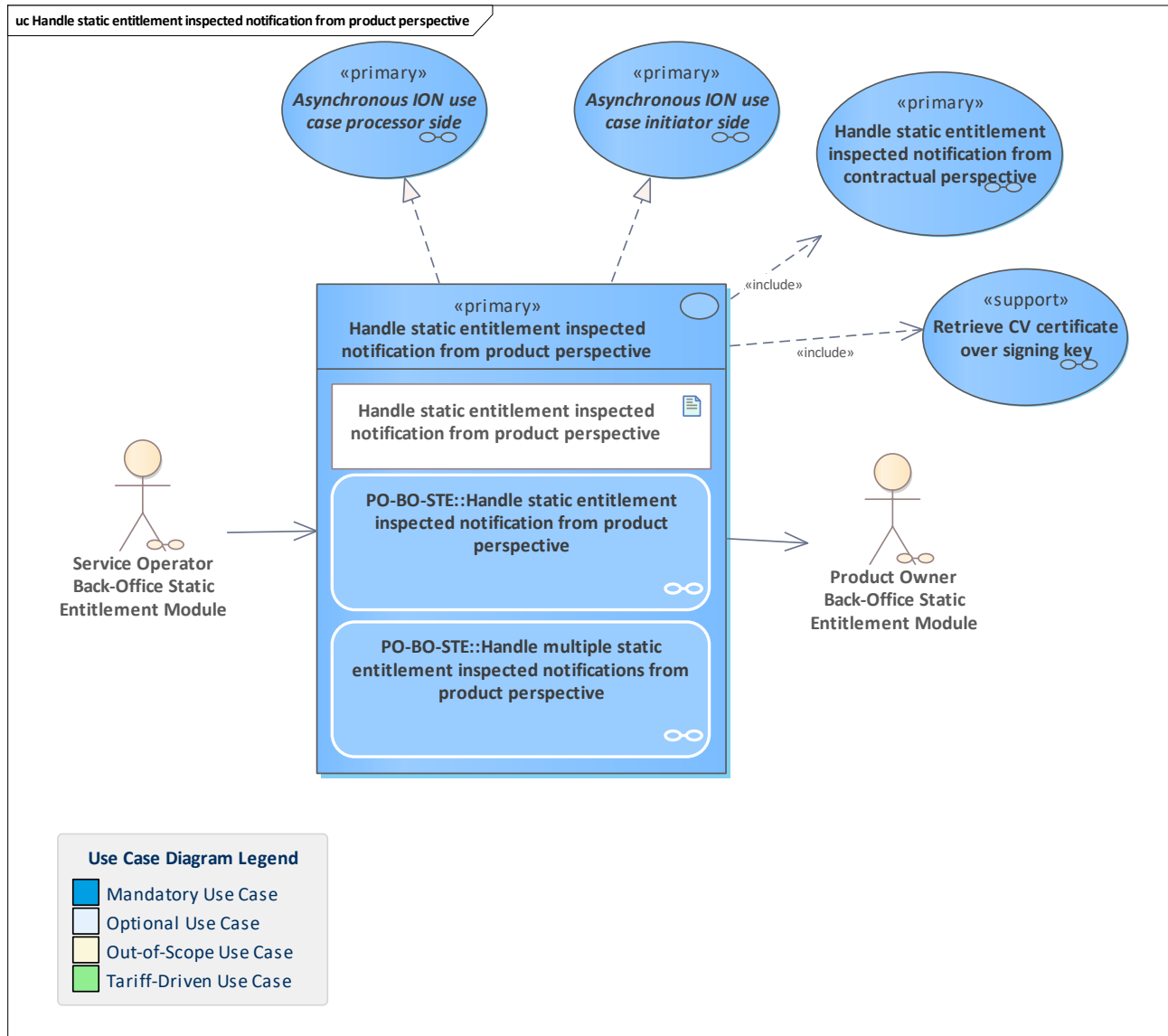


Use Case	Handle static entitlement issued notification from product perspective
Description	Handle a notification about a static entitlement issuance from the product owner perspective. The PO back-office system with a static entitlement extension receives the message coming from the pCCP. The message can be either a single notification or a notification list. The PO registers the notification(s) and does its checks and monitoring from the product owner perspective.
Initiating Actor	Primary Customer Contract Partner Back-Office Static Entitlement Module
Reacting Actor	Product Owner Back-Office Static Entitlement Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand entitlement hotlisting
Linked Use Cases	Retrieve CV certificate over signing key / Determine SAM owner



(Includes)	
Linked Use Cases (Realises)	Asynchronous ION use case processor side
Base Activity	
Inputs	Notify static entitlement issued : notifyStaticEntitlementIssued
Outputs	Notify static entitlement issued response : notifyStaticEntitlementIssuedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E STE DUPLICATE STATIC ENTITLEMENT ISSUANCE E STE DUPLICATE STATIC ENTITLEMENT TERMINATION E CO WRONG SECURITY LEVEL Notify static entitlement issued exception : notifyStaticEntitlementIssuedException
Activity Diagram	PO-BO-STE::Handle static entitlement issued notification from product perspective
Alternative 1	
Inputs	Process static entitlement issued notification list : processStaticEntitlementIssuedNotificationList
Outputs	Process static entitlement issued notification list response : processStaticEntitlementIssuedNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process static entitlement issued notification list exception : processStaticEntitlementIssuedNotificationListException
Activity Diagram	PO-BO-STE::Handle multiple static entitlement issued notifications from product perspective
Alternative 2	
Inputs	Notify static entitlement issuing aborted : notifyStaticEntitlementIssuingAborted
Outputs	Notify static entitlement issuing aborted response : notifyStaticEntitlementIssuingAbortedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO WRONG SECURITY LEVEL Notify static entitlement issuing aborted exception : notifyStaticEntitlementIssuingAbortedException
Activity Diagram	PO-BO-STE::Handle static entitlement issuing aborted notification from product perspective

14.2.2 Handle static entitlement inspected notification from product perspective

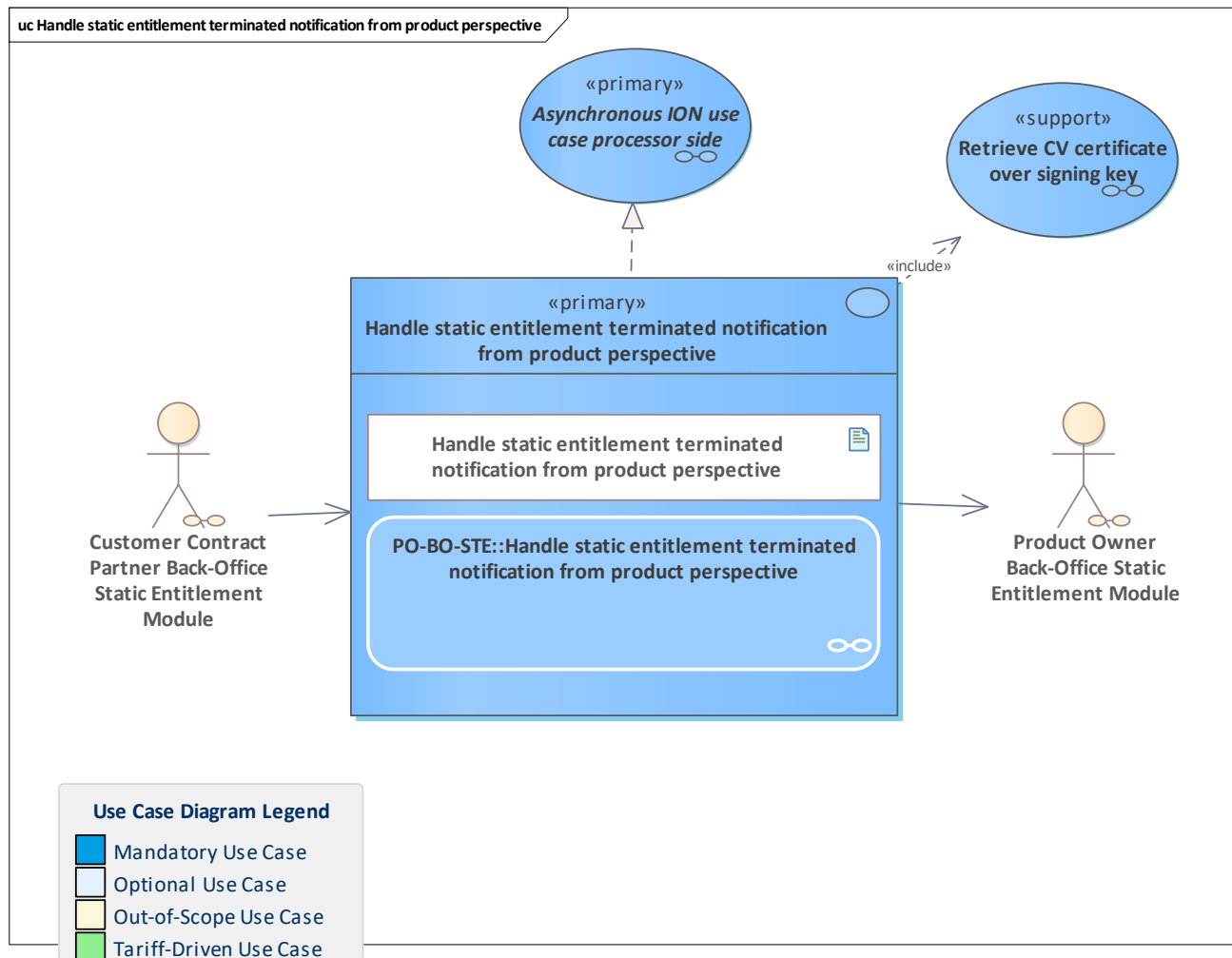


Use Case	Handle static entitlement inspected notification from product perspective
Description	This use case describes the processing of the notification of an inspected static entitlement in the PO back-office system. The SO sends the notification, the PO registers it and does its monitoring checks. After these checks, the notification is forwarded to the CCP back-office system. Furthermore, static entitlement inspected notifications can be collected in the SO system and then sent as a list in a scheduled process to the PO.
Initiating Actor	Service Operator Back-Office Static Entitlement Module
Reacting Actor	Product Owner Back-Office Static Entitlement Module
Preconditions	
Postconditions	



Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Handle static entitlement inspected notification from contractual perspective / Retrieve CV certificate over signing key / Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify static entitlement inspected : notifyStaticEntitlementInspected
Outputs	Notify static entitlements inspected response : notifyStaticEntitlementInspectedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E STE DUPLICATE STATIC ENTITLEMENT ISSUANCE E STE DUPLICATE STATIC ENTITLEMENT TERMINATION E CO WRONG SECURITY LEVEL Notify static entitlements inspected exception : notifyStaticEntitlementInspectedException
Activity Diagram	PO-BO-STE::Handle static entitlement inspected notification from product perspective
Alternative 1	
Inputs	Process static entitlement inspected notification list : processStaticEntitlementInspectedNotificationList
Outputs	Process static entitlement inspected notification list response : processStaticEntitlementInspectedNotificationListResponse
Error Cases	E CO CANNOT DECOMPRESS DATA E CO CANNOT DECOMPRESS DATA E CO WRONG ELEMENT IN COMPRESSED DATA Process static entitlement inspected notification list exception : processStaticEntitlementInspectedNotificationListException
Activity Diagram	PO-BO-STE::Handle multiple static entitlement inspected notifications from product perspective

14.2.3 Handle static entitlement terminated notification from product perspective

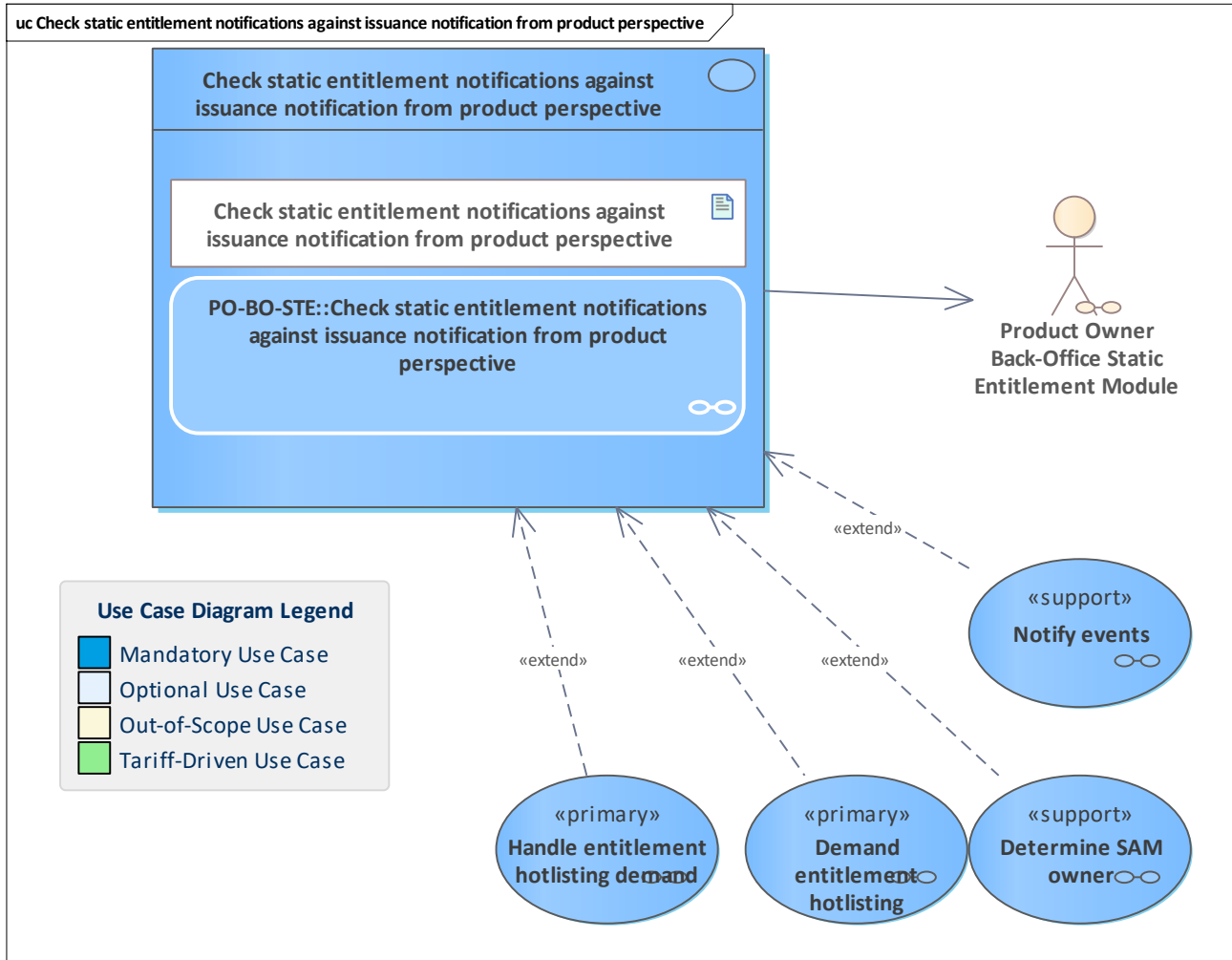


Use Case	Handle static entitlement terminated notification from product perspective
Description	Static entitlement terminated notification is handled from the product owner perspective. The static entitlement terminated notification is received by the PO. The PO registers static entitlement terminated notification and does its checks and monitoring from the product owner perspective.
Initiating Actor	Customer Contract Partner Back-Office Static Entitlement Module
Reacting Actor	Product Owner Back-Office Static Entitlement Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	Retrieve CV certificate over signing key
Linked Use Cases (Realises)	Asynchronous ION use case processor side



Base Activity	
Inputs	<u>Notify static entitlement terminated :</u> <u>notifyStaticEntitlementTerminated</u>
Outputs	<u>Notify static entitlement terminated response :</u> <u>notifyStaticEntitlementTerminatedResponse</u>
Error Cases	<u>E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT</u> <u>E CO BINARY STRUCTURE CANNOT BE PROCESSED</u> <u>E STE DUPLICATE STATIC ENTITLEMENT ISSUANCE</u> <u>E STE DUPLICATE STATIC ENTITLEMENT TERMINATION</u> <u>E CO WRONG SECURITY LEVEL</u> <u>Notify static entitlement terminated exception :</u> <u>notifyStaticEntitlementTerminatedException</u>
Activity Diagram	<u>PO-BO-STE::Handle static entitlement terminated notification from</u> <u>product perspective</u>

14.2.4 Check static entitlement notifications against issuance notification from product perspective

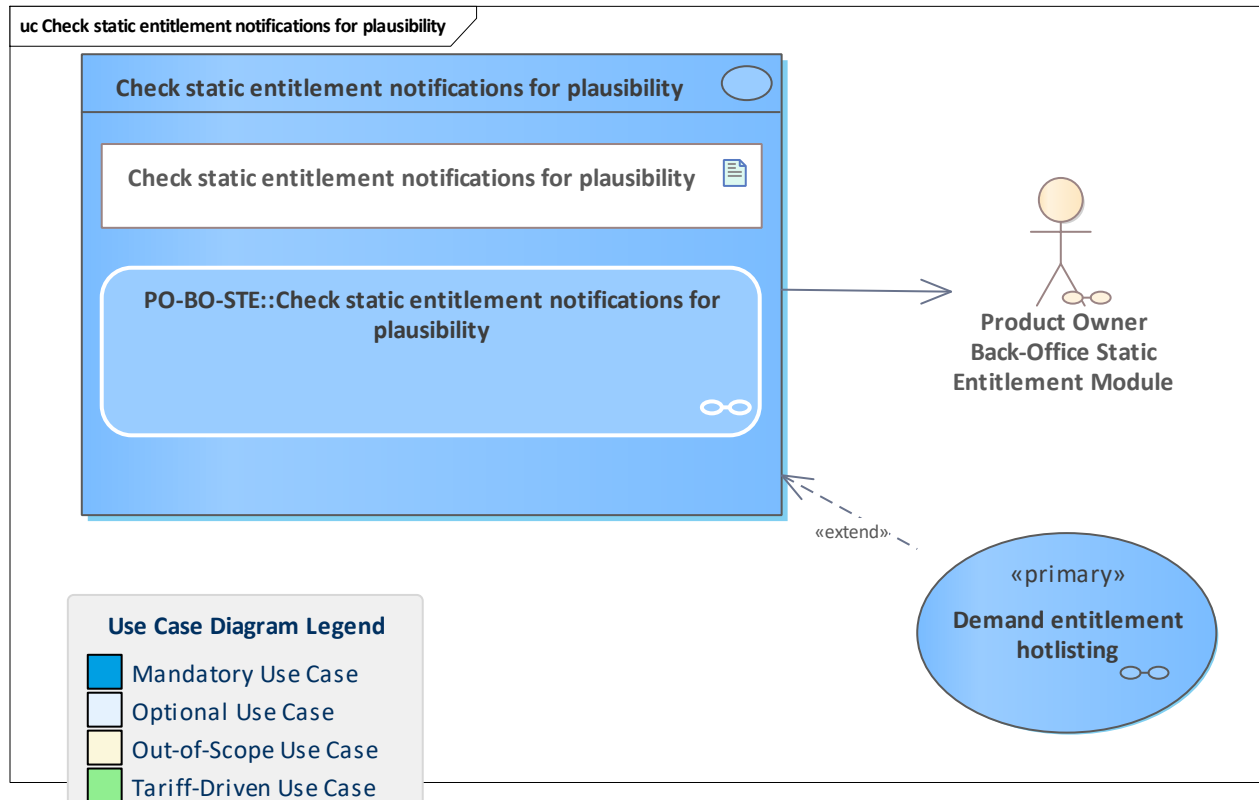


Use Case	Check static entitlement notifications against issuance notification from product perspective
Description	Non-issuing static entitlement notifications need to be checked for consistency with their issuance notification, e.g., the entitlement validity period. This also makes sure that every entitlement seen live is known to the system or has been reported as issued.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Static Entitlement Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand entitlement hotlisting / Notify events / Handle entitlement hotlisting demand / Determine SAM owner
Linked Use Cases (Includes)	Handle entitlement hotlisting demand
Linked Use Cases (Realises)	
Base Activity	



Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-STE::Check static entitlement notifications against issuance notification from product perspective

14.2.5 Check static entitlement notifications for plausibility



Use Case	Check static entitlement notifications for plausibility
Description	Check static entitlement notifications for plausibility to detect illegal copies and similar fraud scenarios.
Initiating Actor	
Reacting Actor	Product Owner Back-Office Static Entitlement Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand entitlement hotlisting
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	
Base Activity	
Inputs	
Outputs	
Error Cases	
Activity Diagram	PO-BO-STE::Check static entitlement notifications for plausibility

15 Miscellaneous Bundle PO-System

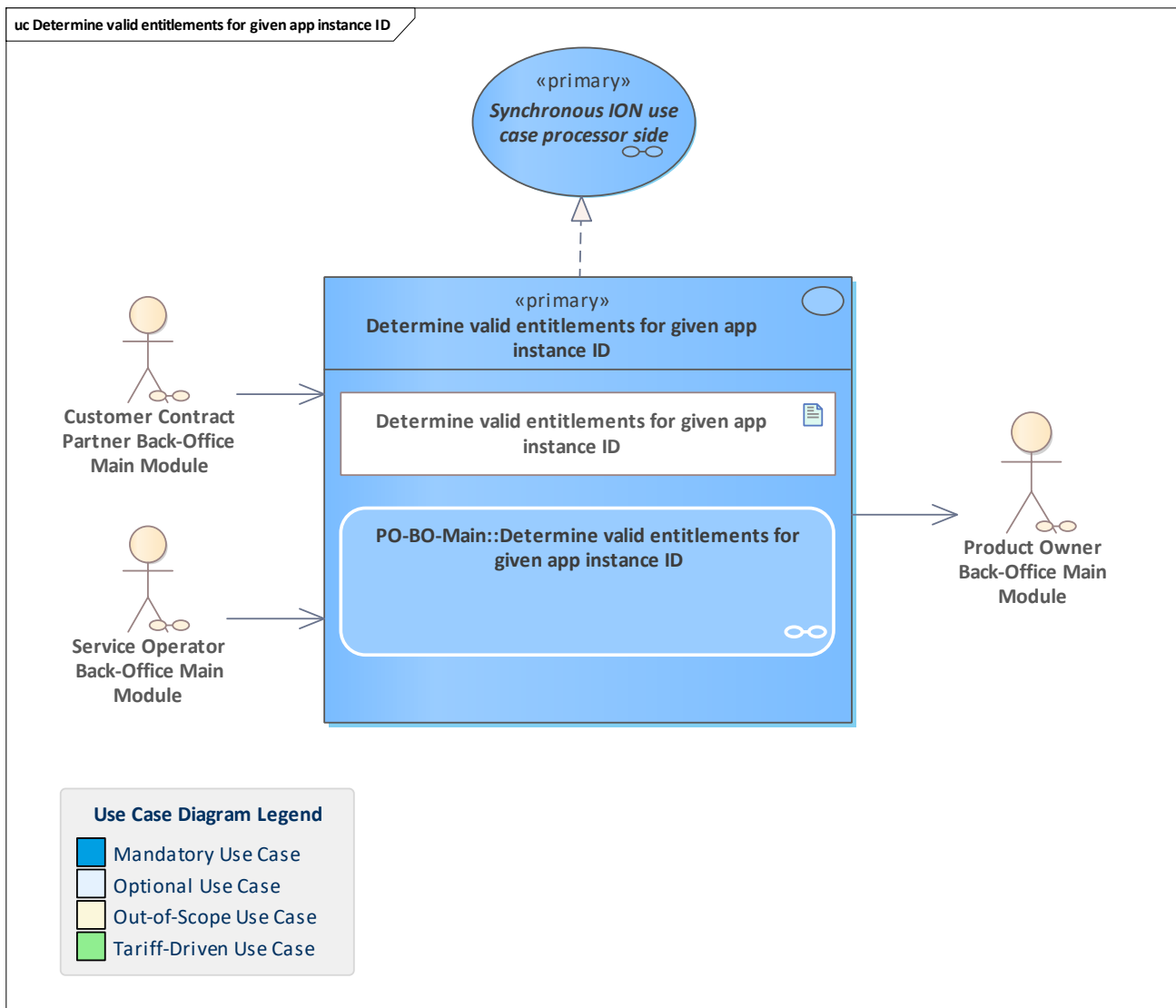
Functionality bundle that covers miscellaneous PO back-office system use cases.

15.1 Overview

Determine valid entitlements for given app instance ID
Handle entitlement validated notification from product perspective

15.2 Use Cases

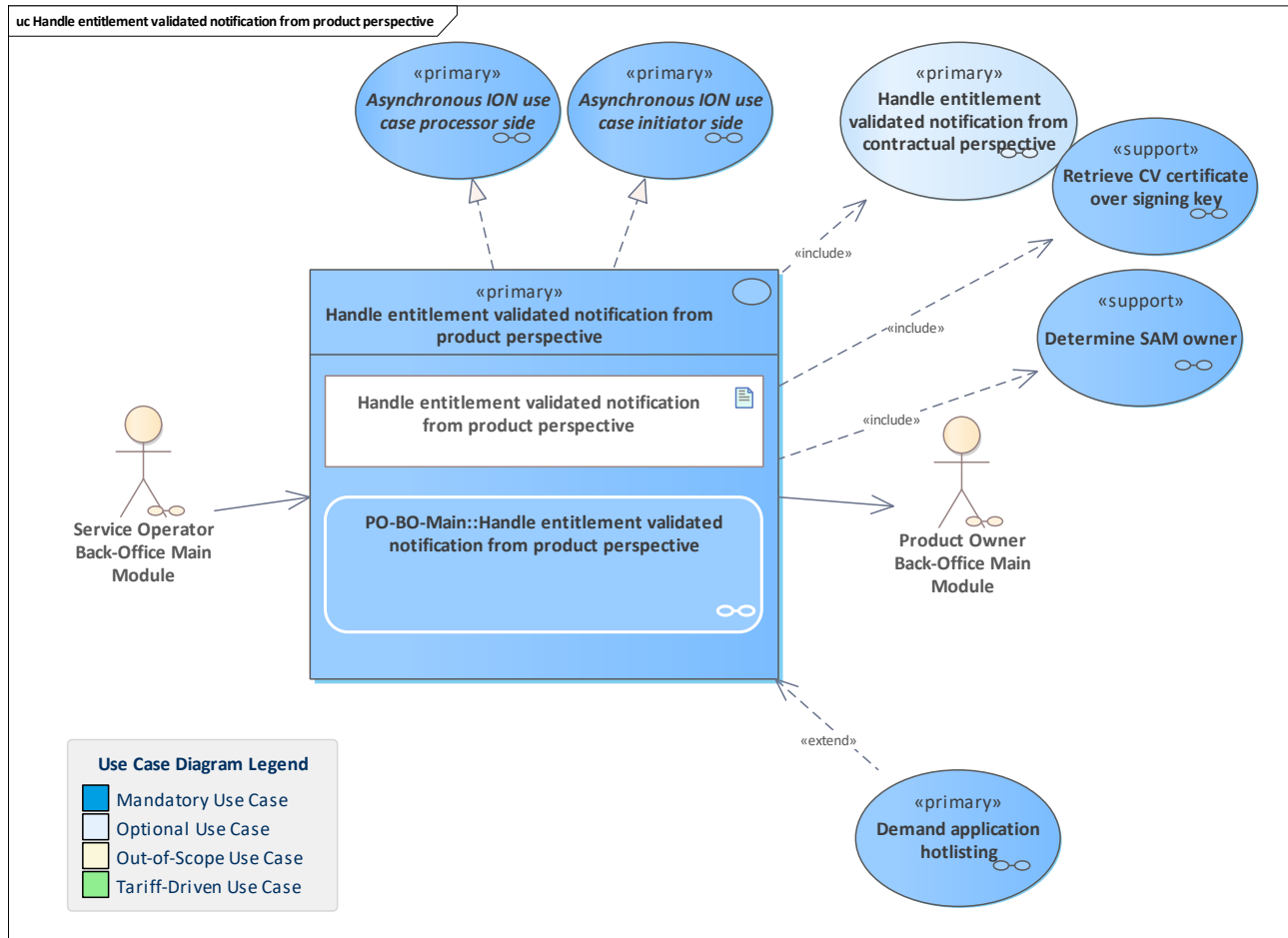
15.2.1 Determine valid entitlements for given app instance ID





Use Case	Determine valid entitlements for given app instance ID
Description	<p>Valid entitlements are determined based on the given application instance ID and given date "<i>validOn</i>".</p> <p>Please note that the <i>validOn</i> has a data type of ZonedDate due to a possible grace period or delays.</p> <p>Valid entitlements means that all entitlements for the requested application instance ID are returned that were valid at the time of <i>validOn</i> and were not on the entitlement hotlist.</p> <p>The PO has no information about entries in the application hotlist or the blocking of the application with the requested application instance ID.</p> <p>If valid entitlements are requested as part of an inspection process, the tariff checks must be executed on the CCP or SO side.</p>
Initiating Actor	Customer Contract Partner Back-Office Main Module
Reacting Actor	Service Operator Back-Office Main Module
Preconditions	Product Owner Back-Office Main Module
Postconditions	
Linked Use Cases (Extended By)	
Linked Use Cases (Includes)	
Linked Use Cases (Realises)	Synchronous ION use case processor side
Base Activity	
Inputs	Get entitlements for given app instance ID : getEntitlementsForGivenAppInstanceId
Outputs	Get entitlements for given app instance ID response : getEntitlementsForGivenAppInstanceIdResponse
Error Cases	E PO GIVEN VALIDITY TIME TOO FAR Get entitlements for given app instance ID exception : getEntitlementsForGivenAppInstanceIdException
Activity Diagram	PO-BO-Main::Determine valid entitlements for given app instance ID

15.2.2 Handle entitlement validated notification from product perspective



Use Case	Handle entitlement validated notification from product perspective
Description	<p>Handle an entitlement validated notification from the product perspective.</p> <p>The entitlement validated notification is received by the PO. The PO registers the entitlement validated notification and does its checks and monitoring from the product perspective regarding the correct execution of validation. In this context, the signature of the validation attestation is verified and the SAM owner of the SAM that performed the validation is determined.</p> <p>Finally, the notification is forwarded to the pCCP</p> <p>Note: in the ION context, the use case is asynchronous as processor (process the notification) and an asynchronous use case as initiator due to the asynchronous call to the pCCP.</p>
Initiating Actor	Service Operator Back-Office Main Module
Reacting Actor	Product Owner Back-Office Main Module
Preconditions	
Postconditions	
Linked Use Cases (Extended By)	Demand application hotlisting / Demand application hotlisting



Linked Use Cases (Includes)	Handle entitlement inspected notification from contractual perspective / Handle entitlement validated notification from contractual perspective / Retrieve CV certificate over signing key / Retrieve CV certificate over signing key / Determine SAM owner
Linked Use Cases (Realises)	Asynchronous ION use case initiator side / Asynchronous ION use case processor side / Asynchronous ION use case initiator side / Asynchronous ION use case processor side
Base Activity	
Inputs	Notify entitlement validated : notifyEntitlementValidated
Outputs	Notify entitlement validated response : notifyEntitlementValidatedResponse
Error Cases	E PO RECEIVER IS NOT PRODUCT OWNER OF OBJECT E CO BINARY STRUCTURE CANNOT BE PROCESSED E CO WRONG ATTESTATION IN NOTIFICATION E CO WRONG SECURITY LEVEL E CO DUPLICATE UM ATTESTATION Notify entitlement validated exception : notifyEntitlementValidatedException
Activity Diagram	PO-BO-Main::Handle entitlement validated notification from product perspective

