

Upgrade Verimatrix Platform

HE 4.3 Release

Design Specification

|  |  |
| --- | --- |
| Version: | 0.2 Draft Version |
| Author: | Rajesh Karna (RSI) |
| Contributors: |  |

The document comprises 41 pages, all pages have version no 0.2

This issue was last saved on 02.12.2014 13:52.

This document was edited with MS WinWord version 11.0.

The filename of the main document file was D:\Project Document\Upgrade Verimatrix platform\D-Spec\3\DSpec\_Upgrade\_Verimatrix\_Platform\_V0.2.docx.

Copyright © 2014 Accenture

All rights reserved.

Confidential and Proprietary.

The reproduction, transmission or use of this code/document or its contents

is not permitted without prior express written consent of Accenture.

Offenders will be liable for damages. All rights, including but not

limited to rights created by patent grant or registration of a utility

model or design, are reserved.

Accenture reserves the right to modify technical specifications and features.

Technical specifications and features are binding only insofar as they

are specifically and expressly agreed upon in a written contract.

[0 General Information 5](#_Toc405291675)

[0.1 History 5](#_Toc405291676)

[0.2 References 5](#_Toc405291677)

[0.3 Glossary and Abbreviations 6](#_Toc405291678)

[0.4 Keyword/Descriptor 6](#_Toc405291679)

[0.5 List of Figures and Tables 6](#_Toc405291680)

[1 Introduction 8](#_Toc405291681)

[1.1 Document organization 8](#_Toc405291682)

[2 Related FR 9](#_Toc405291683)

[2.1 Upgrade Verimatrix Platform 9](#_Toc405291684)

[2.1.1 Scope 9](#_Toc405291685)

[2.2 Assumptions and Restrictions 9](#_Toc405291686)

[2.3 Technical Data 9](#_Toc405291687)

[2.3.1 Backwards compatibility 9](#_Toc405291688)

[2.3.2 Performance requirements 10](#_Toc405291689)

[2.3.3 Technical Solution Description 10](#_Toc405291690)

[3 Overview 11](#_Toc405291691)

[3.1 Component Diagram 11](#_Toc405291692)

[4 Object Model 12](#_Toc405291693)

[4.1 Use case Modeling 12](#_Toc405291694)

[4.1.1 Configure TM for interaction with OMI interface 12](#_Toc405291695)

[4.1.2 Notification to Verimatrix on assignment of STB 13](#_Toc405291696)

[4.1.3 Notification to Verimatrix on un-assignment of STB 15](#_Toc405291697)

[4.1.4 Retransmission of Notifications to Verimatrix 18](#_Toc405291698)

[4.1.5 Verimatrix License management from TM GUI 19](#_Toc405291699)

[4.2 Class Modeling 20](#_Toc405291700)

[4.2.1 Class Diagram – Upgrade Vermatrix Platform 20](#_Toc405291701)

[5 Implementation 21](#_Toc405291702)

[5.1 Configure TM for interaction with OMI interface 21](#_Toc405291703)

[5.1.1 Sequence Diagram 21](#_Toc405291704)

[5.1.2 Business Logic 21](#_Toc405291705)

[5.2 Notification to Verimatrix on assignment of STB 23](#_Toc405291706)

[5.2.1 Sequence Diagram 23](#_Toc405291707)

[5.2.2 Business Logic 23](#_Toc405291708)

[5.3 Notification to Verimatrix on un-assignment of STB 26](#_Toc405291709)

[5.3.1 Sequence Diagram 26](#_Toc405291710)

[5.3.2 Business Logic 27](#_Toc405291711)

[5.4 Retransmission of Notifications to Verimatrix 27](#_Toc405291712)

[5.4.1 Sequence Diagram 27](#_Toc405291713)

[5.4.2 Business Logic 28](#_Toc405291714)

[5.5 Verimatrix License management from TM GUI 29](#_Toc405291715)

[5.5.1 Sequence Diagram 29](#_Toc405291716)

[5.5.2 Business Logic 30](#_Toc405291717)

[6 Flow Diagram 31](#_Toc405291718)

[6.1 Configure TM for interaction with OMI interface 31](#_Toc405291719)

[6.2 Notification to Verimatrix on assignment of STB 32](#_Toc405291720)

[6.3 Notification to Verimatrix on un-assignment of STB 32](#_Toc405291721)

[6.4 Retransmission of Notifications to Verimatrix 32](#_Toc405291722)

[6.5 Verimatrix License management from TM GUI 32](#_Toc405291723)

[7 Exported Interfaces 34](#_Toc405291724)

[7.1 Total Manage 34](#_Toc405291725)

[7.1.1 Distributed Interfaces 34](#_Toc405291726)

[7.1.2 Notifications 34](#_Toc405291727)

[7.1.3 GUI Windows 34](#_Toc405291728)

[8 Configuration 36](#_Toc405291729)

[9 Imported Interfaces 37](#_Toc405291730)

[10 Database Migration/Upgrade for Upgrade Vermatrix Platform 38](#_Toc405291731)

[10.1 Required SQL script 38](#_Toc405291732)

[11 Installation 39](#_Toc405291733)

[12 Unit Tests 40](#_Toc405291734)

[14 Design Specification Reviewers List & Template 41](#_Toc405291735)

# General Information

## History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Reason for Changes** |
| 0.1 | 28-11-2014 | Initial Draft. |
| 0.2 | 02-12-2014 | For incorporating review comments. |

Table 1: History

## References

|  |  |
| --- | --- |
| **No.** | **Reference** |
| [1] RS | Requirement Specification Version V0.2 |
| [2] FS | FSpec Upgrade Verimatrix Platform Draft 0.1.docx |

## Glossary and Abbreviations

|  |  |
| --- | --- |
| API | Application Programming Interface |
| CRL | Certificate Revocation List |
| DRM | Digital Right Management |
| OCI | Open Client Interface |
| OMI | Operator Management Interface |
| SE | System Engineering |
| STB | Set Top Box |
| TM | Total Manage |
| UI | User Interface |
| VCAS | Verimatrix Video Content Authority System |
| VMX | Verimatrix |

Table 2: Abbreviations

## Keyword/Descriptor

|  |  |
| --- | --- |
|  |  |

## List of Figures and Tables

Figure 1: Component Digram – (TM, URH, VCAS 3.6, DB) 11

Figure 2: Use Case – Configure TM for interaction with OMI interface 12

Figure 3: Use Case – Notification to Verimatrix on assignment of STB 13

Figure 4: Use Case – Notification to Verimatrix on un-assignment of STB 15

Figure 5: Use Case – Retransmission of Notifications to Verimatrix 18

Figure 6: Use Case – Verimatrix License management from TM GUI 19

Figure 7: Class Diagram – Upgrade Verimatrix Platform 20

Figure 8: Sequence Diagram – Configure TM for interaction with OMI interface 21

Figure 9: Sequence Diagram – Notification to Verimatrix on assignment of STB 23

Figure 10: Sequence Diagram – Notification to Verimatrix on un-assignment of STB 26

Figure 11: Sequence Diagram – Retransmission of Notifications to Verimatrix (Authorization) 27

Figure 12: Sequence Diagram – Retransmission of Notifications to Verimatrix (Revocation) 28

Figure 13: Sequence Diagram – Verimatrix License management from TM GUI (Authorization) 29

Figure 14: Sequence Diagram – Verimatrix License management from TM GUI (Revocation) 30

Figure 15: Flow Digram – Configure TM for interaction with OMI interface 31

Figure 16: Flow Digram – Upgrade Verimatrix Platform 32

Figure 17: TM UI screen - Setup 🡪 Company 🡪 Servers 🡪 Verimatrix Encryption Server 34

[Table 1: History 5](#_Toc405291753)

[Table 2: Abbreviations 6](#_Toc405291754)

[Table 3 : Configure TM for interaction with OMI interface 12](#_Toc405291755)

[Table 4 : Notification to Verimatrix on assignment of STB 15](#_Toc405291756)

[Table 5 : Notification to Verimatrix on un-assignment of STB 17](#_Toc405291757)

[Table 6 : Retransmission of Notifications to Verimatrix 19](#_Toc405291758)

[Table 7 : Verimatrix License management from TM GUI 20](#_Toc405291759)

[Table 8:CCompanyServerDetailsPanel 22](#_Toc405291760)

[Table 9: CVerimatrixServerFeature 22](#_Toc405291761)

[Table 10: AdminMgmtService 23](#_Toc405291762)

[Table 11: AdminMgmtServiceLocator 23](#_Toc405291763)

[Table 12: AdminMgmt 24](#_Toc405291764)

[Table 13: AdminMgmtProxy 24](#_Toc405291765)

[Table 14: AdminMgmtStub 24](#_Toc405291766)

[Table 15: DeviceMgmtService 24](#_Toc405291767)

[Table 16: DeviceMgmtServiceLocator 24](#_Toc405291768)

[Table 17: DeviceMgmt 25](#_Toc405291769)

[Table 18: DeviceMgmtProxy 25](#_Toc405291770)

[Table 19: DeviceMgmtStub 25](#_Toc405291771)

[Table 20: VMXCommunicationClient 26](#_Toc405291772)

[Table 21: Controls - Setup 🡪 Company 🡪 Servers 🡪 Verimatrix Encryption Server 34](#_Toc405291773)

[Table 22 : DB Table VMXNETWORKCONFIG for storing NetworkID. 38](#_Toc405291774)

[*Table 23 Unit Tests – Upgrade Verimatrix Platform* 40](#_Toc405291775)

# Introduction

This D-Spce describes the process to reclaim and reactivate the licenses on VMX server for dismissed and reintroduced STB through Operator Management Interface (OMI). This process will bring the actual license count of the KPN DRM system to the effective number of STB’s by reducing stale data in the VMX DB.

This D-Spec explains about the process which is to be implemented in VMX version 3.6. The existing CRL interface is to be replaced with OMI interface. It will be, therefore, requird to AVS IPTV Backend to adapt with respect to the new interface.

## Document organization

The feature requirements expected by the development will be restricted to include the project FRs mentioned in RS and FS.

Regarding the document contributions, it is expected from development that the features to be realized be distributed considering the design phase:

- Analysis, consisting in the "Functional Specification IUS version 1.0" contributions, meaning chapters:  
 General Information  
 Introduction  
 Analysis  
 Affected Components  
 Appendix  
  
- Design, consisting in the remaining "Design Specification" contributions, meaning chapters:

General Information  
 Introduction

Overview  
 Object Model  
 Implementation

Flow Diagram

Exported Interfaces  
 Configuration  
 Imported Interfaces

Database Migration/Upgrade

Installation

Unit Tests

Design Specification Reviewers List & Template

# Related FR

## Upgrade Verimatrix Platform

This chapter describes in brief how the FR is interpreted and what are impacts on the involved sub-systems.

This document describes about the “Upgrade Verimatrix Platform” featrure and its interaction with VMX version 3.6.

### Scope

This document defines the Detailed Design for the requirements and use cases defined in the Requirement Spec and Functional Spec for “Upgrade Verimatrix Platform”.

## Assumptions and Restrictions

1. Description of overall DRM solution and Entitlement process is out of scope of this document.
2. On migration of VMX to version 3.6, it will be the responsibility of TM operator to alter the configurations in order to target OMI interface instead of CRL interface.
3. The new STBs will not be registered in VMX by AVS IPTV Back end. IPTV Backend is only responsible for authorization and revocation of device licenses if they already exist in VMX DB.
4. The solution is based on VCAS 3.6 and use the Operator Management Interface (OMI) Device Management Web Service deployed on VCAS Server.
5. The solution supposes that at least a VCAS 3.x is already integrated in the platform and a version upgrade will be performed.
6. Assign a license to a STB (first time) is out of scope for this document.
7. It is expected that the existing devices would already be migrated to VMX and this feature is responsible only for the further authorization/revocation requests.
8. It will be the responsibility of operator to register the network in VMX. Also the same smsNetworkId should be stored in TM database as well in VMXNETWORKCONFIG table that can be further used by TM to interact with OMI interface.

## Technical Data

### Backwards compatibility

This must be considered as enhancement of the current solution already provided from VMX 3.x that use CRL web service to complete the physical license assignment/removal.

In VCAS 3.6 the CRL is not more available and for this reason an enhancement through OMI SOAP WS is required. For backward compatibility the CRL interface will be used from the backend until VMX will be migrated. After VMX migration to version 3.6, the backend will be able to switch from CRL to OMI interface depending upon the configurations in TM.

TM UI would provide TM operator the option to either choose OMI or CRL interface.

### Performance requirements

The solution must be integrated on platform and it inheriting the same scalability feature of URHs servers. Anyway the response time of interfaces involved (autoConfig, subscriber API, etc...) will increase as per the VMX response time (or timeout configured).

In case of license re-activation the user should not have impact caused by OMI (if VMX is available) in terms of usability of system.

### Technical Solution Description

This document provides an overview of interaction with VMX for authorization and revocation of VMX licenses of STBs through Operator Management Interface (OMI). The solution is based on VCAS 3.6 and use the SOAP call to OMI Device Management Service Web Service deployed on VCAS Server.

OMI has provided a list of SOAP interface to enable this process. It is an enhancement over the implementation of integration with CRL interfaces for authorization and revocation of STB devices.

For the sake of backward compatibility, a drop down box is added to VMX server definition in TM GUI to indicate whether the system has to interact with CRL or OMI interface. The other configurable parameter at TM UI would behave as before. It is the responsibility of the TM operator to switch over from CRL to OMI when VMX upgrade takes place.

# Overview

## Component Diagram

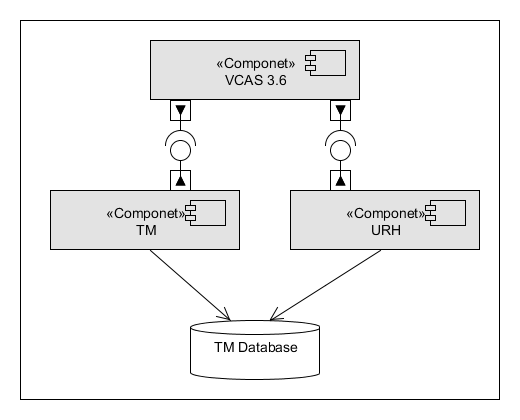


Figure 1: Component Digram – (TM, URH, VCAS 3.6, DB)

# Object Model

## Use case Modeling

### Configure TM for interaction with OMI interface

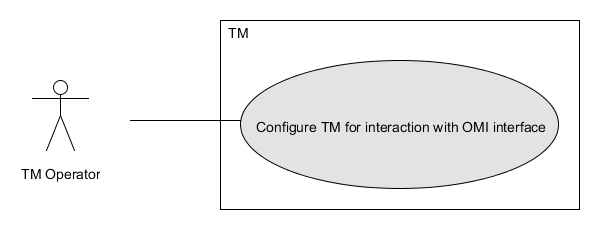


Figure 2: Use Case – Configure TM for interaction with OMI interface

|  |  |
| --- | --- |
| **Actor** | TM Operator |
| **Information** | This use case describes the selection of Verimatrix interface to be used for authorization and revocation process. A newly added field in TM UI under Verimatrix server definition for interface selection will be responsible to achieve this functionality. |
| **Pre-condition** | * TM is up and running. |
| **Flow of events** | 1. TM operator navigates to ‘*Setup🡪Company🡪 Server’* TAB. 2. The operator selects the ‘Verimatrix Encryption Server’ from the server list. 3. The Verimatrix Interface server list contains the values ‘CRL’ and ‘OMI’. 4. TM *operator chooses the* Verimatrix Interface as OMI. 5. Then provides rest of the information required for server definition. 6. On pressing the submit button, the server information is saved into DB. 7. The data is saved into CompanyServerProperties file for server type ‘VES’. The value ‘OMI’ is saved for field ‘value’ and ‘verimatrixInterface’ for field ‘name’. |
| **Post-condition** | * The VMX server has been configured to connect via OMI interface. |
| **Alternative Flow of Events** | * On pressing the cancel button, none of the server information would be saved to DB. |
| **Exception** | * None. |
| **Reference** | *[2]* |

Table 3 : Configure TM for interaction with OMI interface

### Notification to Verimatrix on assignment of STB

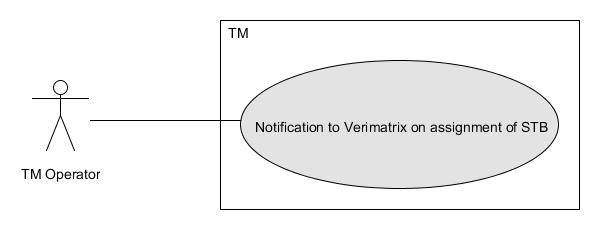


Figure 3: Use Case – Notification to Verimatrix on assignment of STB

|  |  |
| --- | --- |
| **Actor** | TM Operator |
| **Information** | On assignment of a Set top box to a subscriber, a license authorization notification should be sent to the VMX server via OMI. This use case describes the mechanism to send the authorize notification and the various alternates in which it could be done. |
| **Pre-condition** | * TM is up and running. * Verimatrix server is configured in TM and is in active state. * Revocation and Reauthorization process is activated from TM. * TM operator assigns a STB to a subscriber from TM GUI (‘*Setup🡪Set-top Boxes🡪STB Assignment*’ OR ‘*Subscribers🡪Subscriber🡪STB Assignment*’ TAB). * VMX’s OMI interface is opted in TM. |
| **Flow of events** | 1. TM operator assigns a STB to a subscriber. 2. It is checked if the reauthorization process should be run as an asynchronous or a synchronous process. 3. The following steps will be done:    * A SOAP request should be sent to VMX server using AdminMgmtProxy and DeviceMgmtProxy to retrieve the STB related information from VMX DB.    * The end points are configurable at path ‘\opt\local\java\config\path.properties’ with name vmxOmiAdminEndPoint and vmxOmiDeviceEndPoint for AdminMgmtService and DeviceMgmtService respectively.    * If end points are not found in path.properties file they will be considered as “<https://omi.verimatrix.com:8090/services/AdminMgmtService>” and “<https://omi.verimatrix.com:8090/services/DeviceMgmtService>” for AdminMgmtService and DeviceMgmtService respectively.    * The AdminMgmtProxy uses AdminMgmtServiceLocator and AdminMgmtStub for the interaction with VMX server.    * The DeviceMgmtProxy uses DeviceMgmtServiceLocator and DeviceMgmtStub for the interaction with VMX server.    * The signOn method is called using AdminMgmtProxy for starting the session.    * In signOn method UserLoginAttributes, SessionHandleHolder and ResultHolder are passed as parameters.    * The UserLoginAttributes keeps the information of userId and password.    * The getNetworkDeviceIdList method is called using DeviceMgmtProxy for getting the information regarding STB.    * In getNetworkDeviceIdList request NetworkDeviceIdListQuery and sessionHandle are passed as parameters.    * The NetworkDeviceIdListQuery contains the information of macaddress and smsNetworkId. The smsNetworkId is retreived from VMXNETWORKCONFIG table.    * The value of smsNetworkId should be cached as the value of smsNetworkId will be fixed.    * If the STB is not present in VMX DB or it is present in authorized state then no request would be sent to VMX. Assignment of STB is completed by TM.    * If the STB is present in the VMX DB, although in unauthorized state, the request enableDevices should be sent by BE using DeviceMgmtProxy for authorization of this STB and thereafter assignment of STB is completed by TM.    * In the enableDevices request Devices and sessionHandle are passed as parameters.    * The Device is retreived from networkDeviceIdData and networkDeviceIdData is retrieved from networkDeviceIdResult.    * After the completion of this request, the signOff method is called using AdminMgmtProxy to end the session.    * On successful notification to verimatrix server, the data is removed from VMXNOTIFICATIONS table on the basis of macAddress. 4. If the selected process mode is synchronous, the following steps will be done:    * If there is a communication failure with VMX server or it returns an error, the operator will be shown an error message stating ‘STB could not be Re-authorized on Verimatrix Server.’, and the box registration would fail. 5. If the selected process mode is asynchronous then following steps will be followed:    * If there is a communication failure with VMX server or it returns an error and if retransmission mechanism is activated, the message shall be stored in the retransmission table ‘VMXNOTIFICATIONS’ to be dealt with later by retransmission background.    * The STB will remain registered to the subscriber in any case. |
| **Post-condition** | * On receiving the enableDevices request from DeviceMgmtProxy would authorize the STB and respond back with a success response. * SETTOPBOXES table in TM DB will be updated with the status of the field VMXLICENSESTATUS as ‘Authorized’. Whenever this field is updated, an audit log should be created in SETTOPBOXES\_AUDIT table. |
| **Alternative Flow of Events** | 1. STB Assignment via Subscriber API: The flow of events described above would also be applicable on assignment of STBs via subscriber API in TM. 2. STB Assignment via AutoConfig or AutoInstall unicast OCI request from STB: Similarly the URH would also follow the process of sending the synchronous and asynchronous OMI request on registration of STB via ‘autoConfig’ and ‘autoInstall’ OCI interfaces.   However, the difference being that in case of synchronous communication, if an error is encountered then the process will be stalled and STB assignment shall fail with error code ‘-27’ and error description stating ‘Set Top Box Register Failed’. |
| **Exception** | 1. If TM/URH is not able to connect to the VMX’s OMI server because of any reason then the configured number of retries would determine, how many times TM/URH would retry to connect to VMX. It should be more than 0 to enable the retry mechanism. |
| **Reference** | *[2]* |

Table 4 : Notification to Verimatrix on assignment of STB

### Notification to Verimatrix on un-assignment of STB

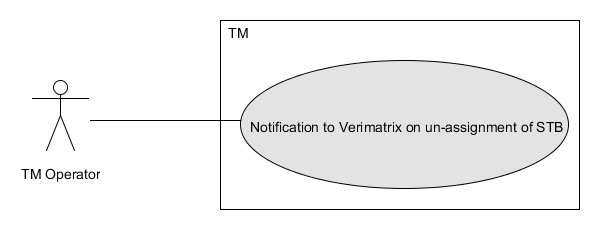


Figure 4: Use Case – Notification to Verimatrix on un-assignment of STB

|  |  |
| --- | --- |
| **Actor** | TM Operator |
| **Information** | On un-assignment of a Set top box to a subscriber, a license revocation request should be sent to the VMX server via OMI interface. This use case describes the mechanism of sending the revoke notification and the various alternates in which it could be done. |
| **Pre-condition** | * TM is up and running. * Verimatrix server is configured in TM and is in active state. * Revocation and Reauthorization process is activated from TM. * TM operator un-assigns a STB to a subscriber from TM GUI (‘*Setup🡪Set-top Boxes🡪STB Assignment*’ OR ‘*Subscribers🡪Subscriber🡪STB Assignment*’ TAB). |
| **Flow of events** | 1. TM operator assigns a STB to a subscriber. 2. It is checked if the reauthorization process should be run as an asynchronous or a synchronous process. 3. The following steps will be done:    * A SOAP request should be sent to VMX server using AdminMgmtProxy and DeviceMgmtProxy to retrieve the STB related information from VMX DB.    * The end points are configurable at path ‘\opt\local\java\config\path.properties’ with name vmxOmiAdminEndPoint and vmxOmiDeviceEndPoint for AdminMgmtService and DeviceMgmtService respectively.    * If end points are not found in path.properties file they will be considered as “<https://omi.verimatrix.com:8090/services/AdminMgmtService>” and “<https://omi.verimatrix.com:8090/services/DeviceMgmtService>” for AdminMgmtService and DeviceMgmtService respectively.    * The AdminMgmtProxy uses AdminMgmtServiceLocator and AdminMgmtStub for the interaction with VMX server.    * The DeviceMgmtProxy uses DeviceMgmtServiceLocator and DeviceMgmtStub for the interaction with VMX server.    * The signOn method is called using AdminMgmtProxy for starting the session.    * In signOn method UserLoginAttributes, SessionHandleHolder and ResultHolder are passed as parameters.    * The UserLoginAttributes keeps the information of userId and password.    * The getNetworkDeviceIdList method is called using DeviceMgmtProxy for getting the information regarding STB.    * In getNetworkDeviceIdList request NetworkDeviceIdListQuery and sessionHandle are passed as parameters.    * The NetworkDeviceIdListQuery contains the information of macaddress and smsNetworkId. The smsNetworkId is retreived from VMXNETWORKCONFIG table.    * The value of smsNetworkId should be cached as the value of smsNetworkId will be fixed.    * If the STB is not present in VMX DB or it is present in revoked state then no request would be required to be sent to VMX and thereafter Un-assignment of STB is completed by TM.    * If the STB is present in the VMX DB, although in authorized state, the request “disableDevices” should be sent by BE to VMX for revoking the license of this STB. Un-assignment of STB is completed by TM.    * In the disableDevices request Devices and sessionHandle are passed as parameters.    * The Device is retreived from networkDeviceIdData and networkDeviceIdData is retrieved from networkDeviceIdResult.    * After the completion of this request, the signOff method is called using AdminMgmtProxy to end the session.    * On successful notification to verimatrix server, the data is removed from VMXNOTIFICATIONS table on the basis of macAddress. 4. If the selected process mode is synchronous, the following steps will be done:    * If there is a communication failure with VMX server or it returns an error, then the operator will be shown an error message that ‘STB could not be revoked from Verimatrix Server. 5. If the selected process mode is asynchronous then following steps will be followed:    * If there is a communication failure with VMX server or it returns an error and if retransmission mechanism is activated, the message shall be stored in the retransmission table ‘VMXNOTIFICATIONS’ to be dealt with later by retransmission background.    * The STB will remain registered to the subscriber in any case. |
| **Post-condition** | * On receiving the disableDevices request from BE, the DeviceMgmtProxy would revoke the STB and respond back with a success response. * SETTOPBOXES table in TM DB will be updated with the status of the field VMXLICENSESTATUS as ‘Revoked’. Whenever this field is updated, an audit log should be created in SETTOPBOXES\_AUDIT table. |
| **Alternative Flow of Events** | 1. STB Un-assignment via Subscriber API: The flow of events described above would also be applicable on un-assignment of STBs via subscriber API in TM. 2. STB un-assignment via unassignSTBunicast OCI request from STB: Similarly the URH would also follow the process of sending the synchronous and asynchronous OMI request on deregistration of STB via unassignSTB OCI interface. |
| **Exception** | 1. If TM/URH is not able to connect to the VMX server via OMI because of any reason then the configured number of retries would determine, how many times TM/URH would retry to connect to VMX. It should be more than 0 to enable the retry mechanism. |
| **Reference** | *[2]* |

Table 5 : Notification to Verimatrix on un-assignment of STB

### Retransmission of Notifications to Verimatrix

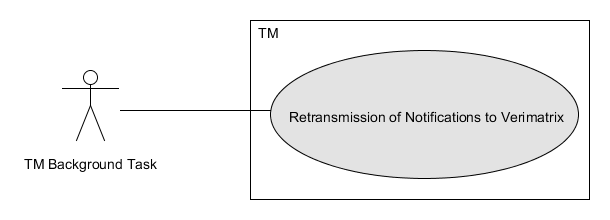


Figure 5: Use Case – Retransmission of Notifications to Verimatrix

|  |  |
| --- | --- |
| **Actor** | TM Backgound Task |
| **Information** | A failsafe mechanism is to be implemented to resend the VMX requests to OMI which fail due to any reason. All the fail requests will be stored in a table in TM DB. A background process would run in TM to resend the failed requests stored in DB at configurable regular intervals. |
| **Pre-condition** | * TM is up and running. * Verimatrix server is configured in TM and is in active state. * Retransmission mechanism is enabled on TM. * Reauthorization process mode is asynchronous. * VMX’s OMI interface is opted in TM. |
| **Flow of events** | 1. The background process of VMXBackgroundTask, in order to retransmit the failed requests would query the DB for any failed notifications. 2. The data are fetched from findAllNotifications() method of VMXNotificationDAO in chronological order. 3. On finding such notifications in DB, the process would repeat the cycle of revocation or authorization of VMX licenses again for all the requests in DB. 4. The background process calls the notifyVerimatrixServer() method of VMXNotificationSender. 5. A asynchronous call is sent to VMXNotificationMDB from sendMessage() method of VMXNotificationSender. 6. On VMXNotificationMDB the method authorizeSTBLicense() or revokeSTBLicense() is called for authorization or revocation respectively. 7. The process of authorization of VMX license is done as given in use case [4.1.1.2](#_Notification_to_Verimatrix). 8. The process of revocation of VMX license is done as given in use case [4.1.1.3](#_Notification_to_Verimatrix_1). |
| **Post-condition** | * All the failed notifications are sent to VMX using OMI interface and updates made to the VMX DB by the VMX application. * The successful retransmissions are removed from the retransmission table VMXNOTIFICATIONS. |
| **Alternative Flow of Events** | * None |
| **Exception** | 1. If resending of requests fail again then the corresponding records should not be removed from TM DB. |
| **Reference** | *[2]* |

Table 6 : Retransmission of Notifications to Verimatrix

### Verimatrix License management from TM GUI

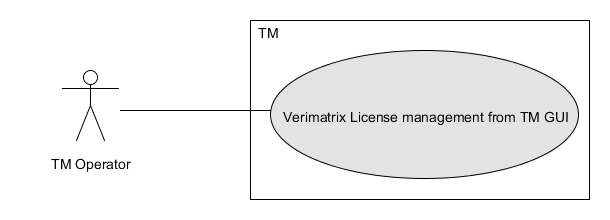


Figure 6: Use Case – Verimatrix License management from TM GUI

|  |  |
| --- | --- |
| **Actor** | TM Operator |
| **Information** | The STB module of TM already have the two buttons to explicitly send authorize or revoke request to VMX for assigned or un-assigned STBs respectively. Same buttons will be used to authorize or revoke request to VMX’s OMI Interface. The selection of VMX’s OMI is to be done in Verimatrix server definition. |
| **Pre-condition** | * TM is up and running. * TM operator un-assigns the STB for the subscriber from TM GUI (‘*Setup🡪Set-top Boxes🡪STB Assignment*’ TAB). * VMX’s OMI interface is opted in TM. |
| **Flow of events** | 1. TM operator navigates to ‘*Setup🡪Set-top Boxes🡪STB Assignment*’ TAB. 2. On pressing ‘Revoke VMX License’ button, the proceesing will be done accoring to **point 3 and 4 of flow of events** of use case [4.1.1.3](#_Notification_to_Verimatrix_1). |
| **Post-condition** | * The license of the particular STB would be revoked from VMX. * A success message would be displayed to the operator stating ‘Revocation of STB from VMX is successful.’ |
| **Alternative Flow of Events** | 1. On pressing ‘Authorize VMX License’ button, the proceesing will be done accoring to **point 3 and 4 of flow of events** of use case [4.1.1.2](#_Notification_to_Verimatrix). 2. A success message would be displayed to the operator stating ‘Authorization of STB in VMX is successful.’ |
| **Exception** | 1. If VMX responds back with some failure response or the BE fails to connect with VMX via OMI then an error message shall be displayed to the operator stating ‘Revocation/Reauthorization of VMX license for STB has failed.’ 2. If retransmission mechanism is activated in TM then this notification would temporarily be stored in TM DB. A background process running in TM would pick these notifications at regular intervals and resend them to VMX via OMI for authorization of STB. The pending notification would be cleared from DB once the notification is resent. 3. If TM/URH is not able to connect to the VMX via OMI server because of any reason then the configured number of retries would determine, how many times TM/URH would retry to connect to VMX. It should be more than 0 to enable the retry mechanism. |
| **Reference** | *[2]* |

Table 7 : Verimatrix License management from TM GUI

## Class Modeling

### Class Diagram – Upgrade Vermatrix Platform

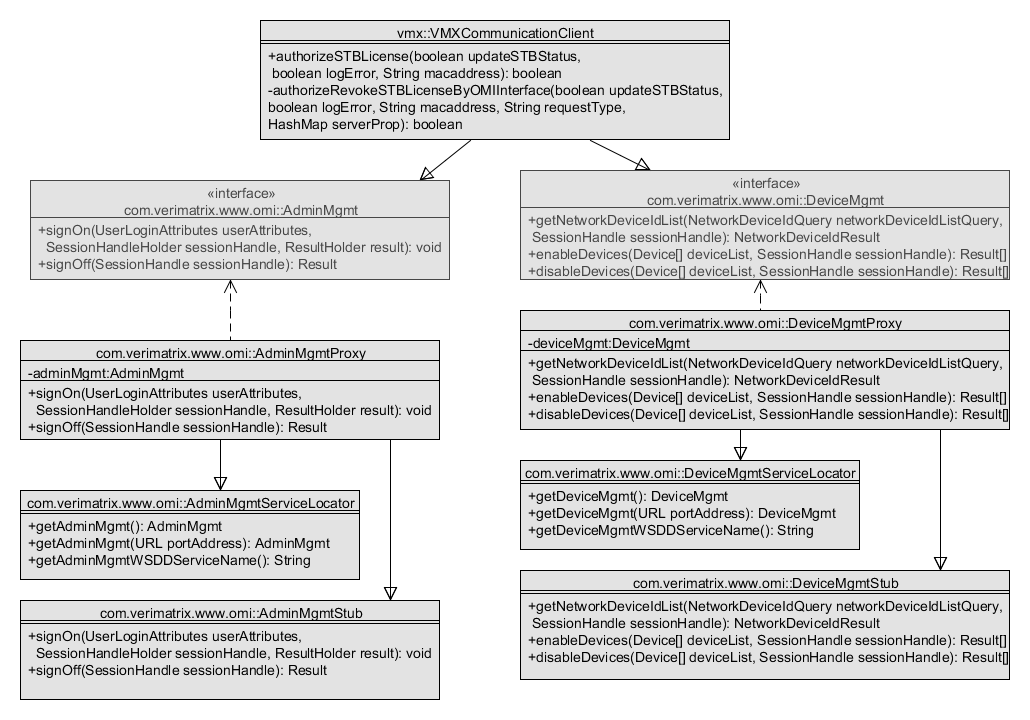


Figure 7: Class Diagram – Upgrade Verimatrix Platform

# Implementation

## Configure TM for interaction with OMI interface

### Sequence Diagram

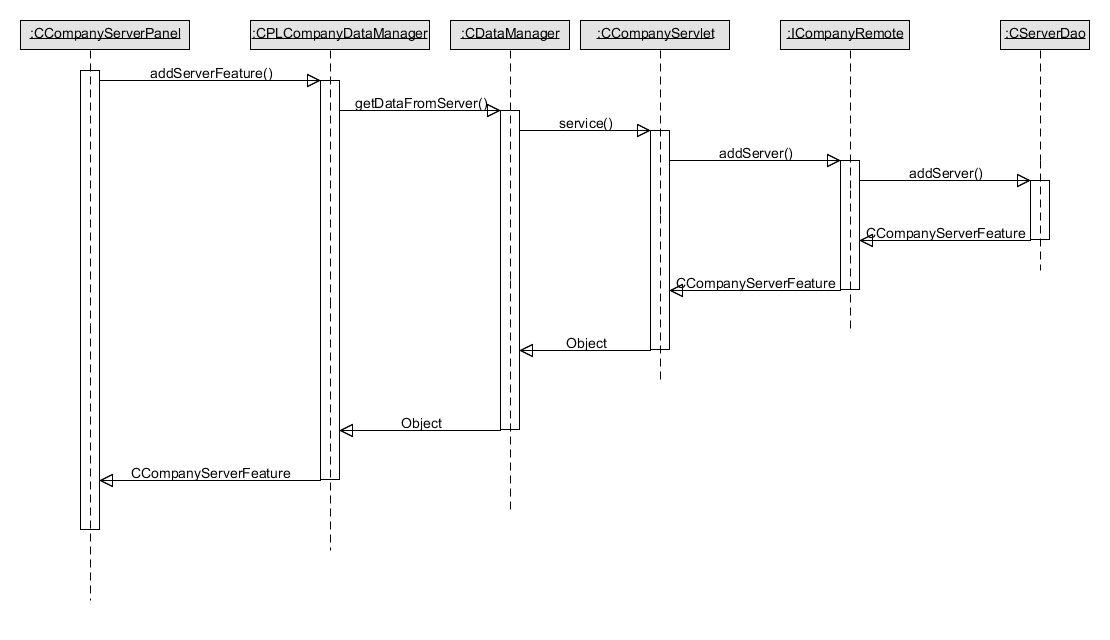


Figure 8: Sequence Diagram – Configure TM for interaction with OMI interface

### Business Logic

#### com.myrio.tm.company.pl.CCompanyServerDetailsPanel

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| Public void initializeComponents() | Method is used to initialize the component. | NA |
| public void layoutServers() | Creates the layout for servers types. | NA |
| public CCompanyServerFeature getServerFeature() | Method returns the server feature data. | * CMandatoryFieldNotFoundException * CException * CPLException |
| private void hideAllServer() | Method is used to hide the server. | NA |
| public void showVEServer() | Method is used to shows VES server fileds. | NA |
| private void disableVeServerControls() | Method is ussed to disables VES Server fields. | NA |
| private void enableVeServerControls() | Method is ussed to enable VES server controls. | NA |
| public void setServerFeature() | Method is ussed to set the current selected server data into the panel fields. | NA |

Table 8:CCompanyServerDetailsPanel

#### com.myrio.tm.company.util.CVerimatrixServerFeature

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| Public String getVerimatrixInterface () | Method to get verimatrix interface. | NA |
| Public void setVerimatrixInterface (CElementText) | Method to set verimatrix interface. | NA |
| Public void setVerimatrixInterface (String) | Method to set verimatrix interface. | NA |

Table 9: CVerimatrixServerFeature

## Notification to Verimatrix on assignment of STB

### Sequence Diagram

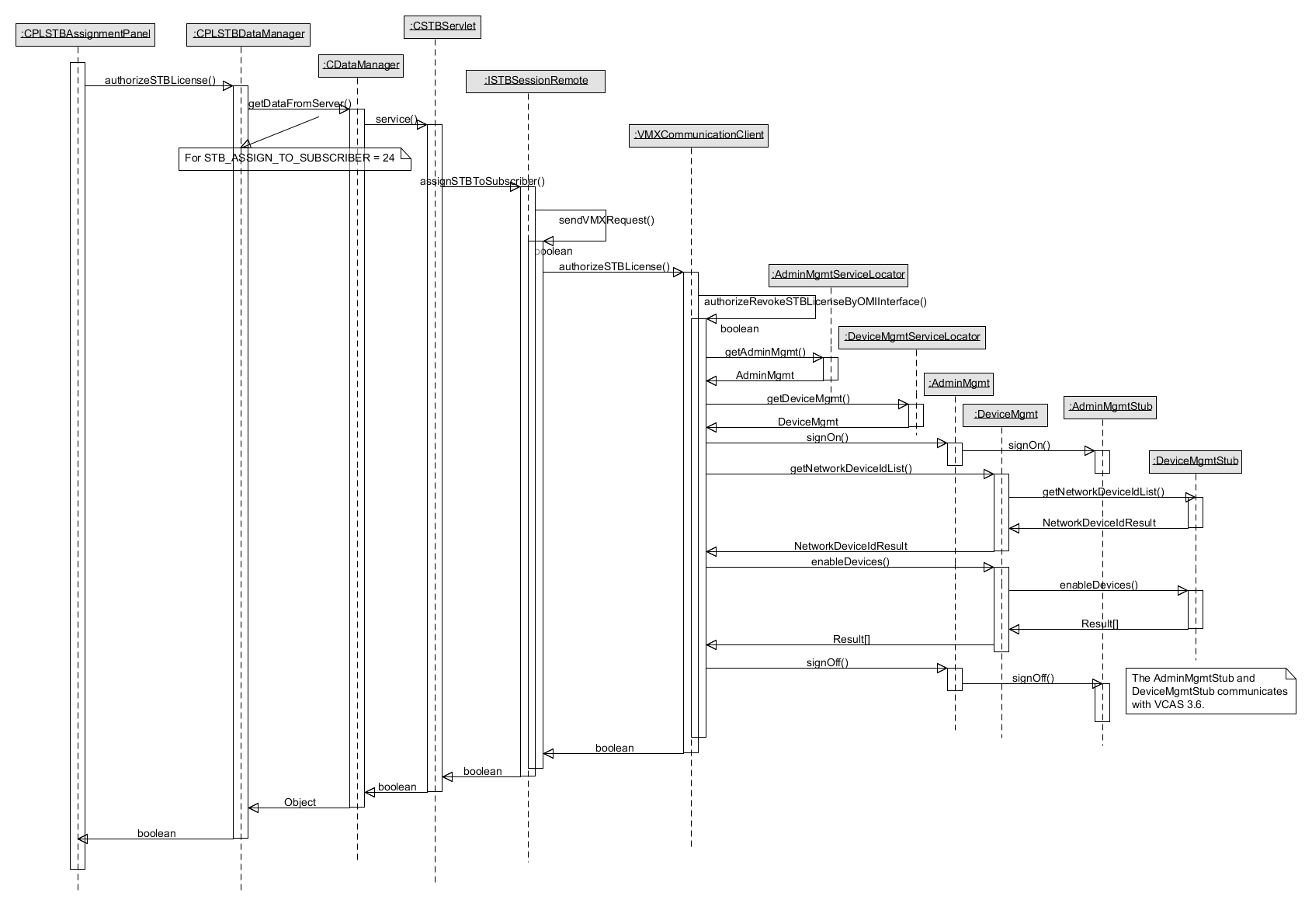


Figure 9: Sequence Diagram – Notification to Verimatrix on assignment of STB

### Business Logic

#### com.verimatrix.www.omi.AdminMgmtService.java

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public AdminMgmt getAdminMgmt() | Method is used to get AdminMgmt. | ServiceException |

Table 10: AdminMgmtService

#### com.verimatrix.www.omi.AdminMgmtServiceLocator.java

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public AdminMgmt getAdminMgmt() | Method is used to get AdminMgmt. | ServiceException |

Table 11: AdminMgmtServiceLocator

#### com.verimatrix.www.omi.AdminMgmt

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public void signOn(UserLoginAttributes, SessionHandleHolder, ResultHolder) | Method is used to sign on. | RemoteException |
| public Result signOff(SessionHandle) | Method is used to sign off. | RemoteException |

Table 12: AdminMgmt

#### com.verimatrix.www.omi.AdminMgmtProxy

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public void signOn(UserLoginAttributes, SessionHandleHolder, ResultHolder) | Method is used to sign on. | RemoteException |
| public Result signOff(SessionHandle) | Method is used to sign off. | RemoteException |

Table 13: AdminMgmtProxy

#### com.verimatrix.www.omi.AdminMgmtStub.java

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public void signOn(UserLoginAttributes, SessionHandleHolder, ResultHolder) | Method is used to sign on. | RemoteException |
| public Result signOff(SessionHandle) | Method is used to sign off. | RemoteException |

Table 14: AdminMgmtStub

#### com.verimatrix.www.omi.DeviceMgmtService.java

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public DeviceMgmt getDeviceMgmt() | Method is used to get DeviceMgmt. | ServiceException |

Table 15: DeviceMgmtService

#### com.verimatrix.www.omi.DeviceMgmtServiceLocator.java

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public DeviceMgmt getDeviceMgmt() | Method is used to get DeviceMgmt. | ServiceException |

Table 16: DeviceMgmtServiceLocator

#### com.verimatrix.www.omi.DeviceMgmt

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public NetworkDeviceIdResult getNetworkDeviceIdList(NetworkDeviceIdQuery, SessionHandle) | Method is used to get the info of STB. | RemoteException |
| public Result[] enableDevices(Device[], SessionHandle) | Method is used to authorize the STB. | RemoteException |
| public Result[] disableDevices(Device[], SessionHandle) | Method is used to revoke the STB. | RemoteException |

Table 17: DeviceMgmt

#### com.verimatrix.www.omi.DeviceMgmtProxy

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public NetworkDeviceIdResult getNetworkDeviceIdList(NetworkDeviceIdQuery, SessionHandle) | Method is used to get the info of STB. | RemoteException |
| public Result[] enableDevices(Device[], SessionHandle) | Method is used to authorize the STB. | RemoteException |
| public Result[] disableDevices(Device[], SessionHandle) | Method is used to revoke the STB. | RemoteException |

Table 18: DeviceMgmtProxy

#### com.verimatrix.www.omi.DeviceMgmtStub

|  |  |  |
| --- | --- | --- |
| ***New Method*** | ***Description*** | ***Exception*** |
| public NetworkDeviceIdResult getNetworkDeviceIdList(NetworkDeviceIdQuery, SessionHandle) | Method is used to get the info of STB. | RemoteException |
| public Result[] enableDevices(Device[], SessionHandle) | Method is used to authorize the STB. | RemoteException |
| public Result[] disableDevices(Device[], SessionHandle) | Method is used to revoke the STB. | RemoteException |

Table 19: DeviceMgmtStub

#### vmx.VMXCommunicationClient

|  |  |  |
| --- | --- | --- |
| ***New/Existing Method*** | ***Description*** | ***Exception*** |
| private boolean authorizeRevokeSTBLicenseByOMIInterface(boolean, boolean, String, String, HashMap) | Method is used to authorize or revoke the STB license through OMI Interface. | NA |
| private String getSMSNetworkId() | Method is used to retrieve the smsNetworkId from database. | SQLException |
| public boolean authorizeSTBLicense(boolean, boolean, String) | Method is used to authorize the STB license. | NA |
| public boolean revokeSTBLicense(boolean, boolean, String) | Method is used to revoke the STB license. | NA |
| public HashMap retreiveVMXServerProperties() | Method is used to retrieve the vmx server properties from database. | SQLException |

Table 20: VMXCommunicationClient

## Notification to Verimatrix on un-assignment of STB

### Sequence Diagram

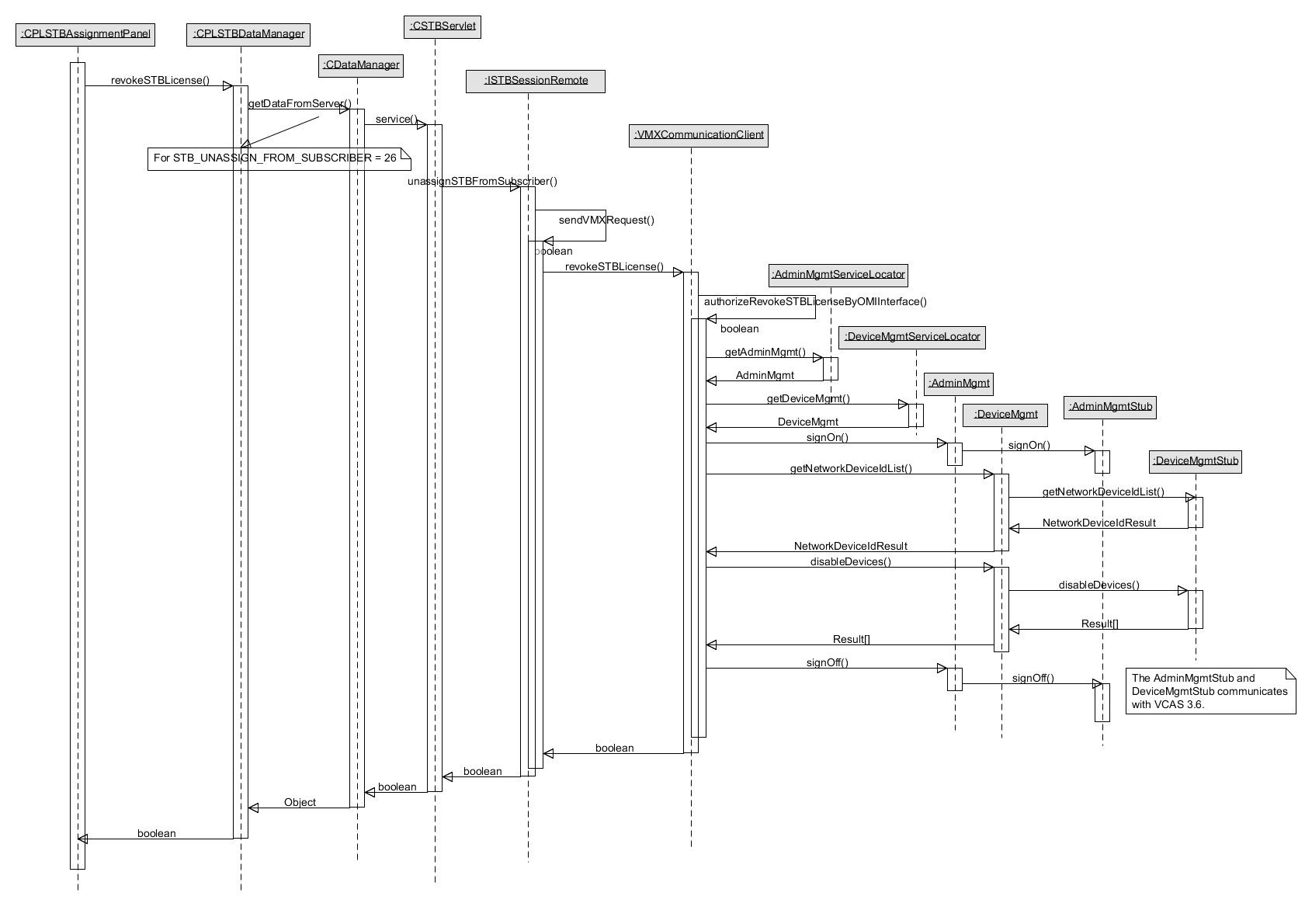


Figure 10: Sequence Diagram – Notification to Verimatrix on un-assignment of STB

### Business Logic

Please refer the section [5.2.2](#_Business_Logic).

## Retransmission of Notifications to Verimatrix

### Sequence Diagram

#### Authorization

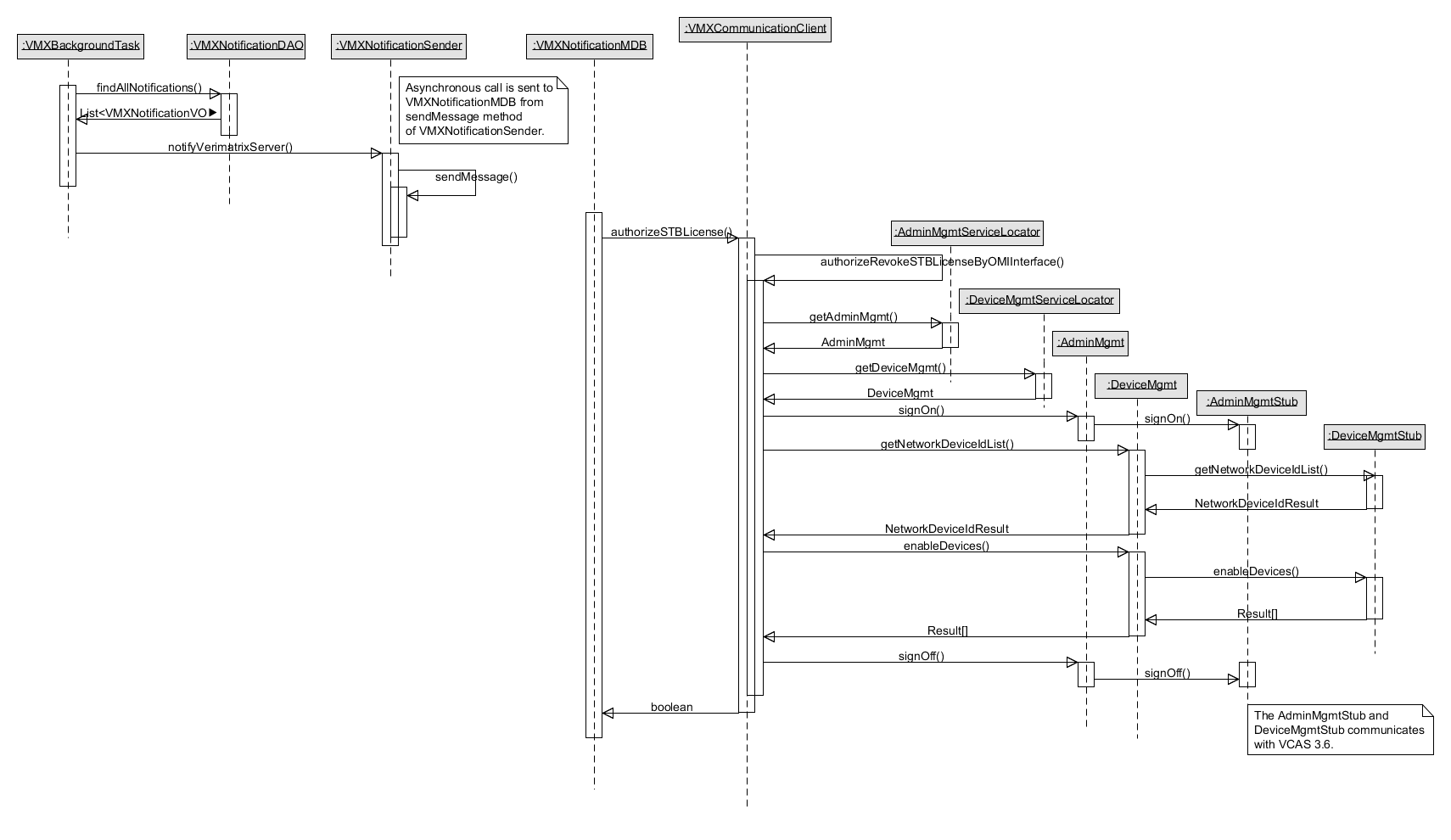


Figure 11: Sequence Diagram – Retransmission of Notifications to Verimatrix (Authorization)

#### Revocation

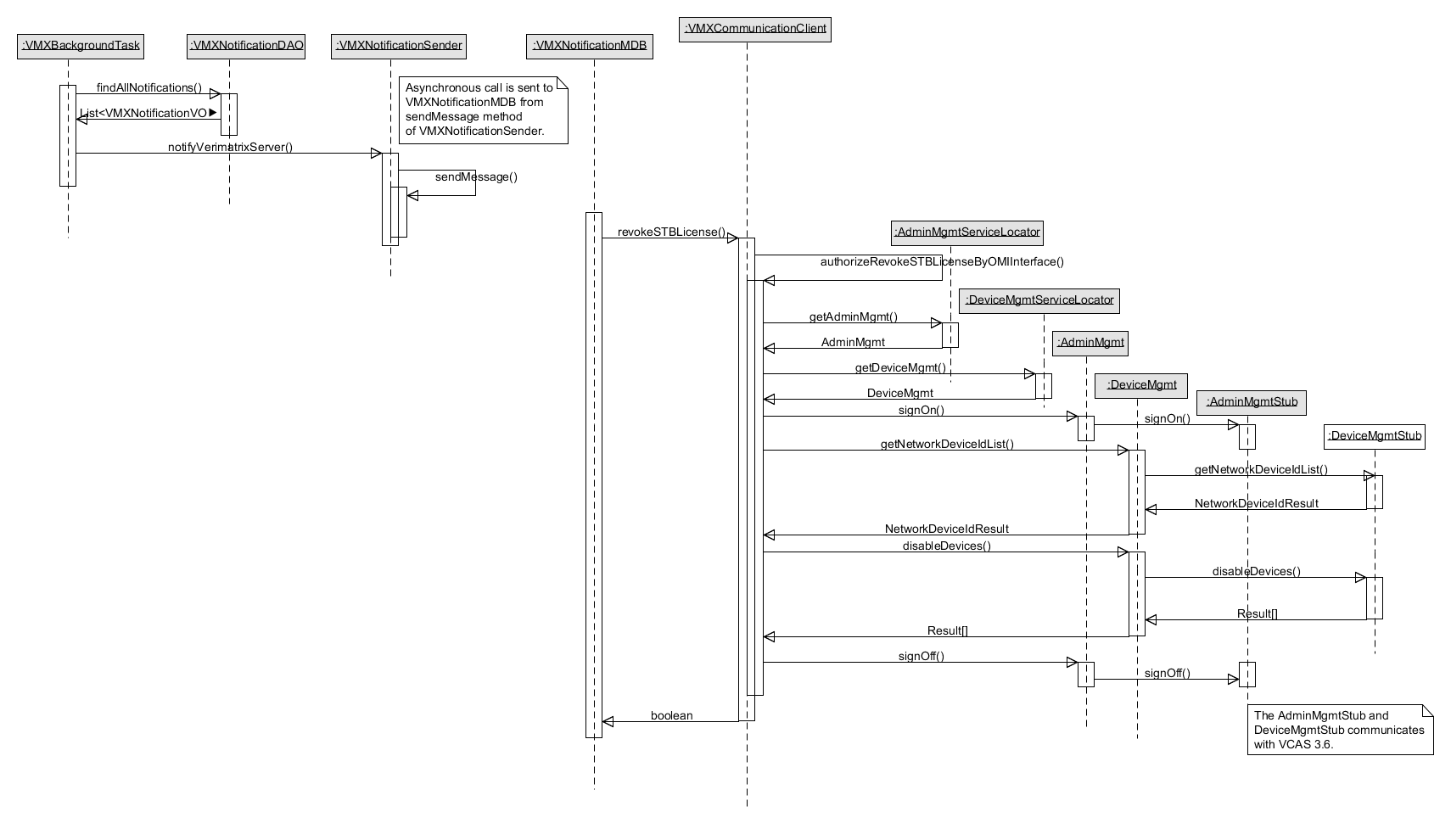


Figure 12: Sequence Diagram – Retransmission of Notifications to Verimatrix (Revocation)

### Business Logic

Please refer the section [5.2.2](#_Business_Logic).

## Verimatrix License management from TM GUI

### Sequence Diagram

#### Authorization

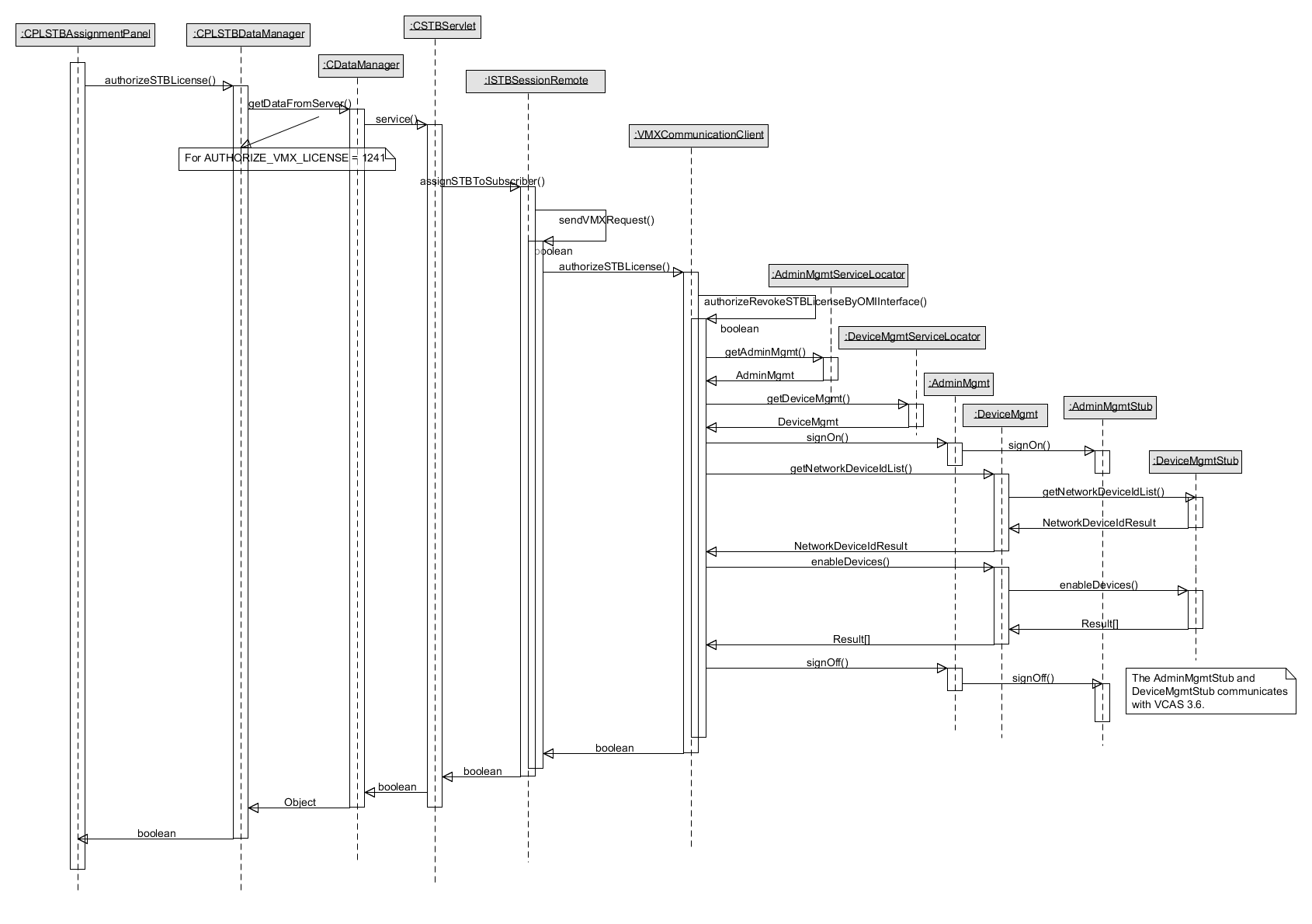


Figure 13: Sequence Diagram – Verimatrix License management from TM GUI (Authorization)

#### Revocation

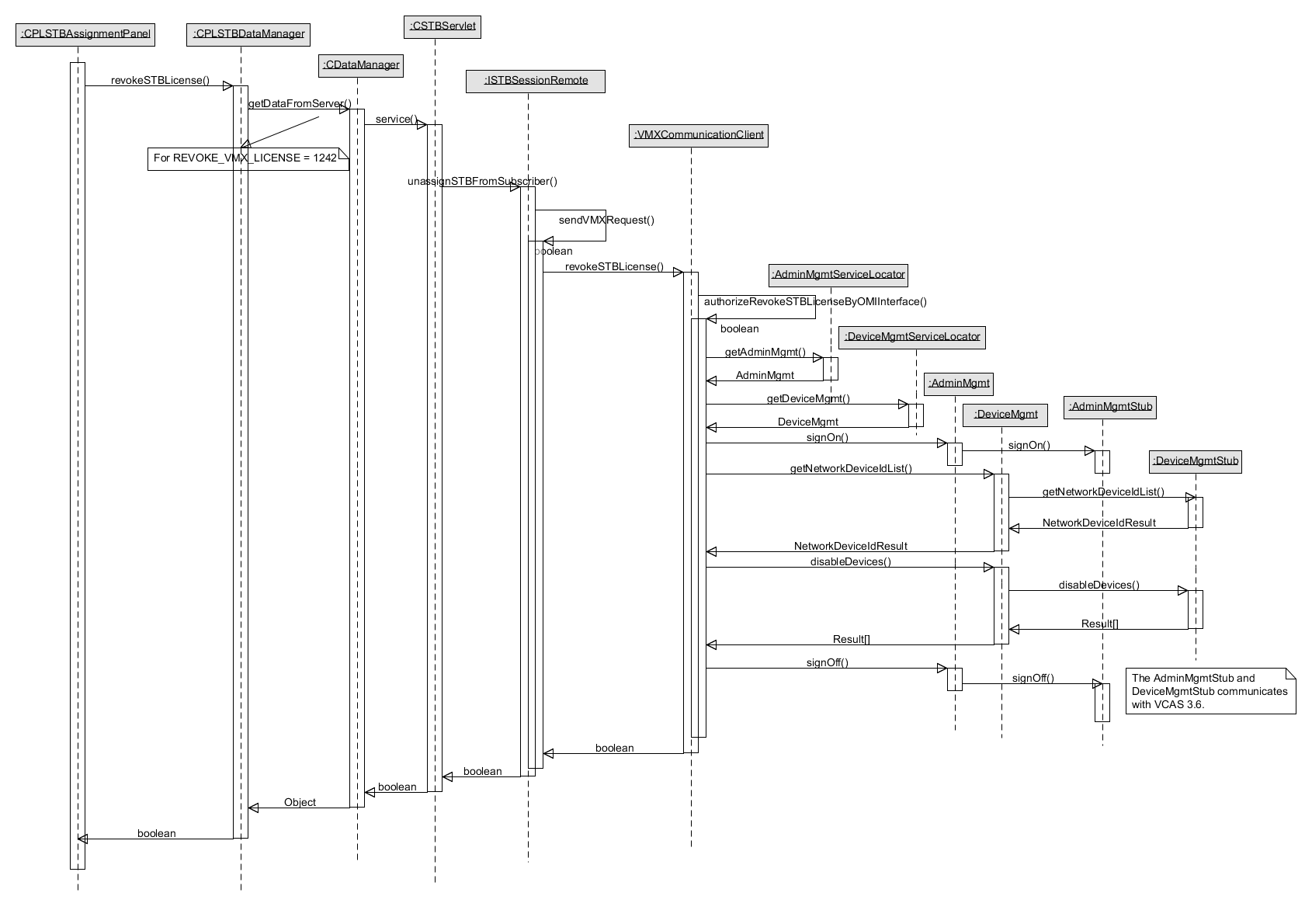


Figure 14: Sequence Diagram – Verimatrix License management from TM GUI (Revocation)

### Business Logic

Please refer the section [5.2.2](#_Business_Logic).

# Flow Diagram

## Configure TM for interaction with OMI interface

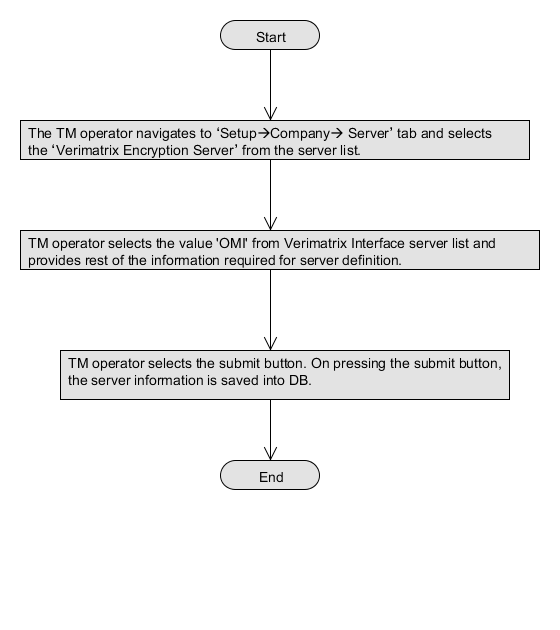


Figure 15: Flow Digram – Configure TM for interaction with OMI interface

## Notification to Verimatrix on assignment of STB

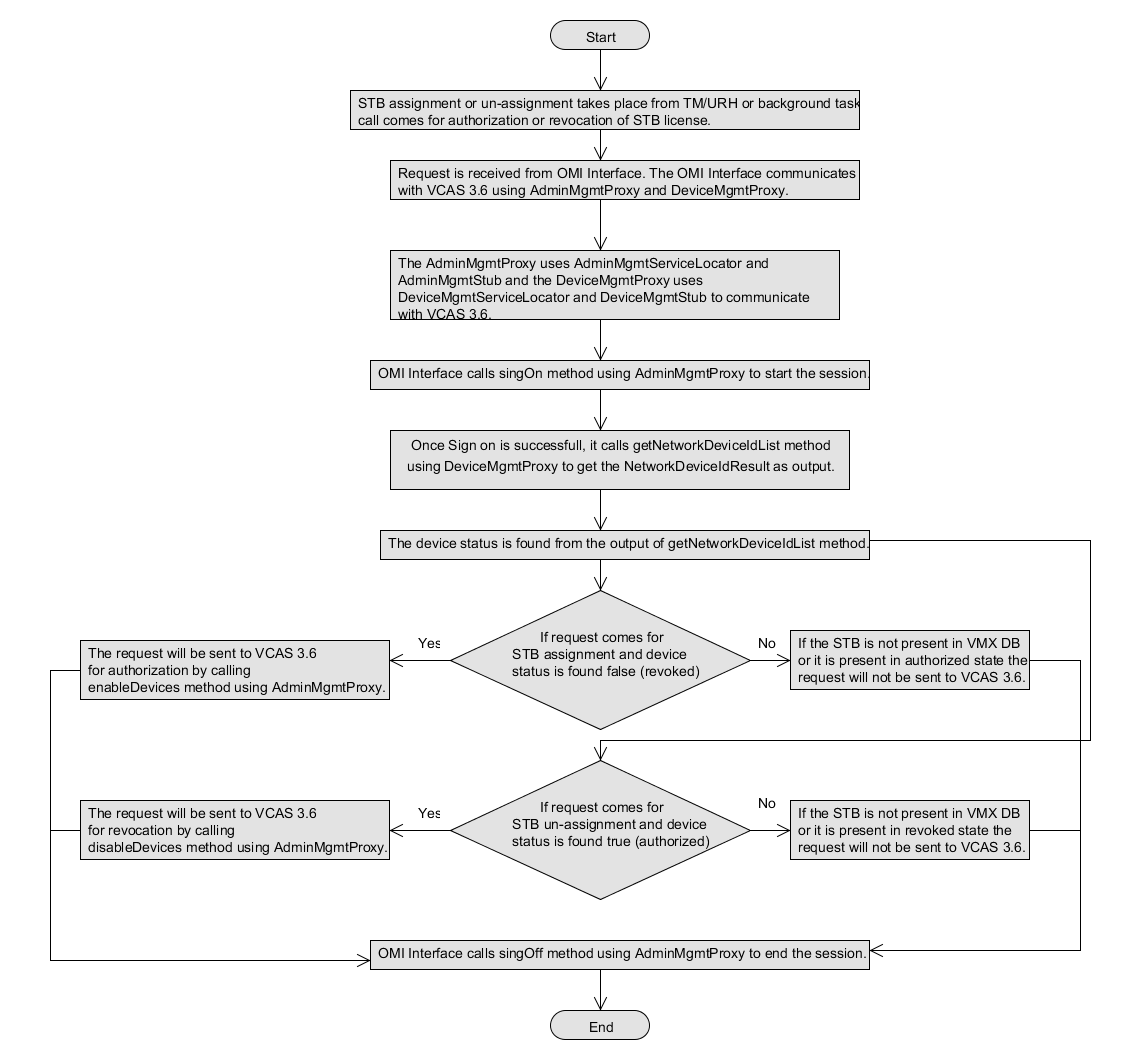


Figure 16: Flow Digram – Upgrade Verimatrix Platform

## Notification to Verimatrix on un-assignment of STB

Please refer the flow diagram of section [6.2](#_Notification_to_Verimatrix_2).

## Retransmission of Notifications to Verimatrix

Please refer the flow diagram of section [6.2](#_Notification_to_Verimatrix_2).

## Verimatrix License management from TM GUI

Please refer the flow diagram of section [6.2](#_Notification_to_Verimatrix_2).

# Exported Interfaces

## Total Manage

### Distributed Interfaces

#### Broker Documents

### Notifications

Not Applicable

### GUI Windows

#### Screen Masks for Setup 🡪 Company 🡪 servers 🡪Verimatrix Encryption Server

The TM GUI is enhanced to support new manageable data.

The Company Servers panel (***Setup – Company – servers – Verimatrix Encryption Server***) is enhanced to include following new setting:

* ‘Verimatrix Interface’, combo box.

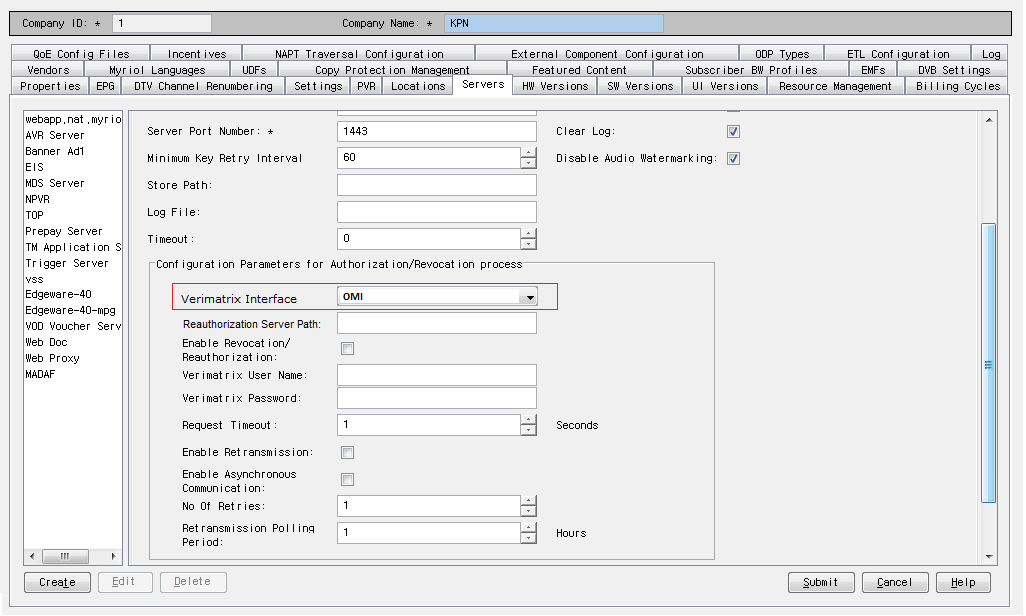


Figure 17: TM UI screen - Setup 🡪 Company 🡪 Servers 🡪 Verimatrix Encryption Server

##### Controls

|  |  |  |
| --- | --- | --- |
| **Caption** | **Control** | **Description** |
| Verimatrix Interface | Combo Box | The combo box keeps the values ‘CRL’ and ‘OMI’. |

Table 21: Controls - Setup 🡪 Company 🡪 Servers 🡪 Verimatrix Encryption Server

# Configuration

In VMXNETWORKCONFIG table COMPANYID and NETWORKID will be inserted through script by operator. The value of COMPANYID and NETWORKID is required to be determined by operator. The value can not be specified in the script as NETWORKID is available on VCAS 3.6 database.

# Imported Interfaces

None

# Database Migration/Upgrade for Upgrade Vermatrix Platform

Creation of new table VMXNETWORKCONFIG

|  |  |  |
| --- | --- | --- |
| **Table Name:** VMXNETWORKCONFIG | | |
| **Field** | **Type** | **Description** |
| COMPANYID | Number(11) | Company Id. NOT NULL |
| NETWORKID | VARCHAR2(25) | ID of AVS IPTV network as configured in verimatrix NOT NULL |

Table 22 : DB Table VMXNETWORKCONFIG for storing NetworkID.

## Required SQL script



# Installation

None.

# Unit Tests

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Class** | **Test Case**  **(method name)** | **Functional Description**  **(what is going to be tested)** | **Sprint number**  **(note: sprint (s) when it’s planned to run the test)** |
| VMXCommunicationClient.java | authorizeSTBLicense() | This method will test the functionality of STB license authorization. | None. |
| VMXCommunicationClient.java | revokeSTBLicense() | This method will test the functionality of STB license revocation. | None. |

*Table 23 Unit Tests – Upgrade Verimatrix Platform*

1. **Design Specification Reviewers List & Template**

**\*\*\*\*\*\*\*\*\*\*\* This chapter is Mandatory \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**The scope of the next file is to identify specific Design Specifications mandatory and optional reviewers.**

****

**Next document refers to Review Report Template.**

****

**Review Comments:**

****

# Design Specification Reviewers List & Template

None.