VOSS Automate NBI Technical Description and Deployment Guide

Release 24.1

Jul 15, 2024

Legal Information

DOCUMENT ID: 20240715225351

Contents

1. Overview

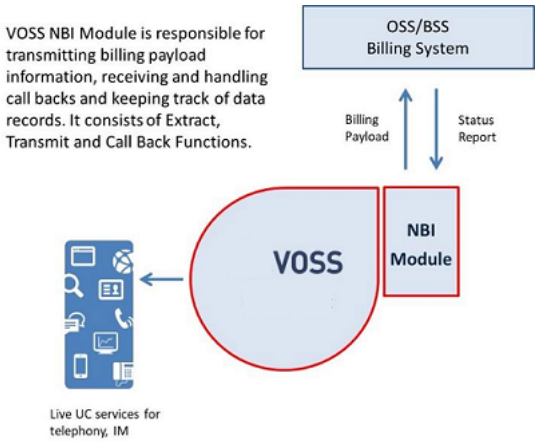
This document is the technical specification and deployment guide for the VOSS Northbound Integration Module (NBI) for VOSS Automate. This specification describes the core functionality of NBI and its interfacing into provider northbound systems. This document may be adapted as required to meet the specific needs of individual service providers and their respective OSS/BSS systems.

2. Introduction

2.1. Introduction to NBI

VOSS Automate comes with a portfolio of service integration modules to support close working with associated provider OSS and BSS systems. Within this portfolio, VOSS Northbound Integration Module (NBI) offers integration with northbound systems for billing and reporting purposes. NBI is a module for VOSS Automate and provides functions to control the billing flow during initial customer provisioning, to monitor for changes to subscribers or their services once the customer is live, and issue real-time billing updates (payloads) northbound for charging.

NBI runs as a separate server appliance and provides this integration with northbound systems through a near real-time RESTful API. This interface is used to transfer a billing payload that contains information on subscribers and associated services and devices whenever a change is made to the live service.



Importantly, NBI can be tailored to the needs of individual providers and their OSS/BSS systems through configuration of this billing payload.

Note: Compatibility

VOSS NBI is compatible with VOSS Automate version 19.3x and above.

2.2. Business Benefits

VOSS NBI offers the following business benefits:

- Reduces time to cash and increases the speed and scale of customer on-boarding activity through full automation of the provisioning to billing information workflow.
- Provides the ability for customers to self-provision a wider range of (chargeable) services.
- Reduces costs and removes errors from the billing process (manual and multiple data entry points).
- Reduces revenue leakage by ensuring correct charging for services at the point of consumption.
- Improves asset utilization and capacity planning through full visibility of service usage.

2.3. Operational Benefits

VOSS NBI offers the following operational benefits:

- Near real-time integration for rapid action where required.
- Small appliance footprint for easy and fast integration into operating environment (add-on).
- Server separation – does not load or interfere with live service.
- Packaged integration service to optimize integration with OSS/BSS.
- Configurable payload to meet the needs of individual northbound billing systems.
- ‘Day 1’ and ‘Day 2’ operating modes – to support on-boarding and on-going billing needs.

2.4. Related Documentation

The following additional documents are available:

- NBI Install Guide

2.5. Provider Responsibilities

As part of the deployment and operation of the NBI, the service provider is expected to provide:

- Information on billing requirements (particularly with respect to the payload content)
- Resources for the installation of NBI – compute, storage and networking
- IT changes as required to integrate NBI into the provider’s billing system
- Testing of the end-to-end billing process and related systems
- Deployment into production and monitoring during business operation

2.6. Glossary of Terms

Term	Definition
API	Application Programming Interface
Billing Payload	Packet of information sent by NBI to northbound billing system
BSS	Provider Business Support System (for example customer management)
CLI	The NBI appliance command interface (used for configuration and diagnostics)
DNS / SRV	Domain Name Server / Service Record
HCS	Cisco Hosted Collaboration Solution
MACD	Moves, Adds, Changes and Deletes
NOC	Provider Network Operations Centre for alarm monitoring
NBI	VOSS Northbound Integration for billing and reporting (also known as the billing-data-extract application)
Northbound System	Provider OSS or BSS, which NBI integrates with for billing purposes
OSS	Provider Operating Support System (for example, asset management)
RBAC	Role Based Access Control
Sandbox	A sub process that can be configured and used to simulate a northbound system
SFTP	Secure File Transfer Protocol
SNMP	Simple Network Management Protocol
SSH	Secure Shell
Subscriber	Consumer of UC, telephony and collaboration services
Tracking Table	Contains a history of all billable events and associated payloads
Trigger Event	The list of VOSS Automate Transaction Codes that cause NBI to create a payload

3. NBI Technical Description

NBI provides functionality to extract subscriber billing related information from VOSS Automate whenever a subscriber change occurs, and to send this information (in a payload) to a northbound billing system.

NBI includes a call back service, which is utilized by the northbound system to provide status updates for each billing message sent. Billing-related changes include changes made to both subscribers directly or to their associated devices and services.

NBI monitors for changes by connecting to one of the VOSS Automate nodes and observes transactions within the VOSS Automate Transaction Log. Connectivity details are discussed below and are defined in the NBI configuration file.

Access to NBI for administration functions is through the NBI Admin GUI (Web Portal).

3.1. Major Functions and Features

This section describes NBI in terms of its major functions and features:

- Voss NBI Module
- NBI Administration Portal (GUI)
- Access Control to NBI GUI
- Real-time Billing Payload
- Billing Reconciliation Support (SDE Report)
- Extract Function (EXTRACTOR PROCESS)
- Transmit Function (NOTIFIER PROCESS)
- Call Back Function (CALLBACK PROCESS)
- Tracking Table
- Connectivity
- Provisioning Control (Site Transition flag)
- Configuration
- Diagnostics
- Provisioning Control (User State) and managing Go-Live Dates
- Managing Site and Subscriber Information

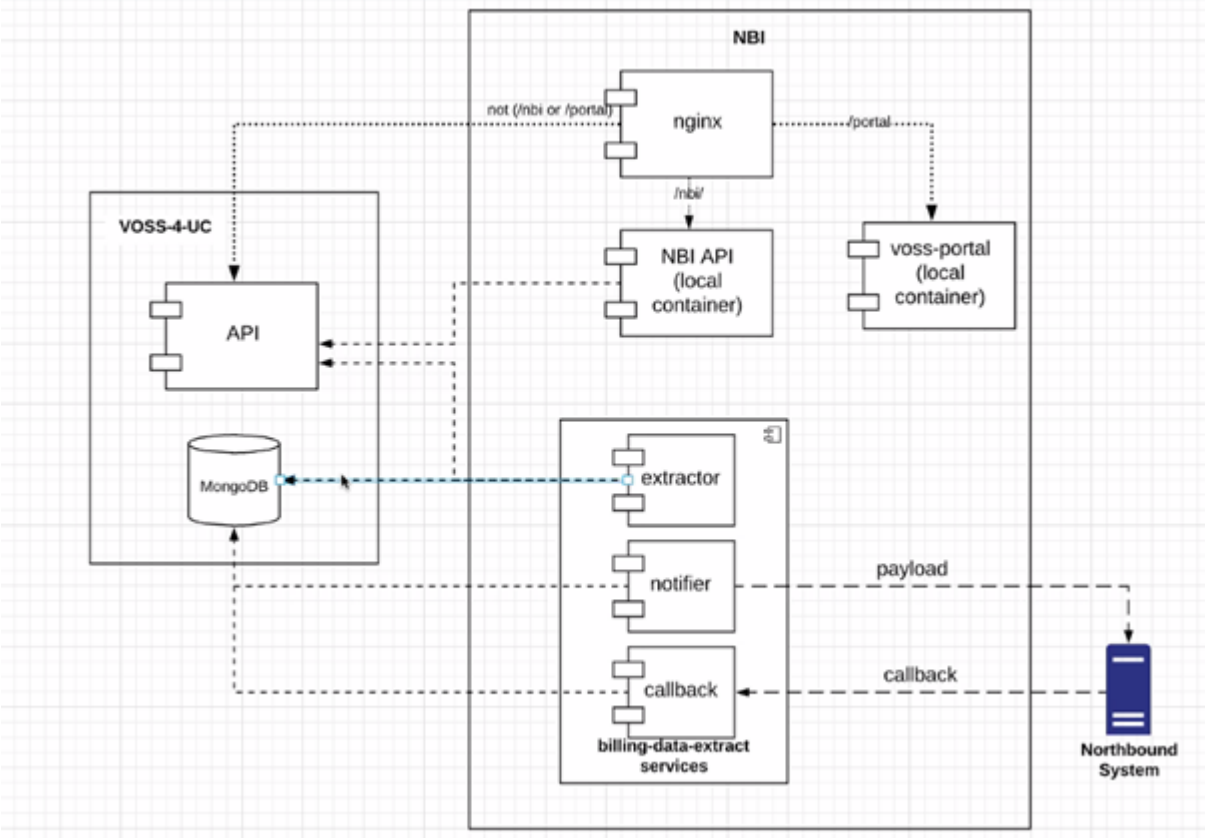
3.1.1. VOSS NBI Module

NBI is a single server appliance that runs on the Ubuntu operating system:

- Small footprint - limited disruption to existing service (this is an add-on)
- Server separation - reducing load on the live VOSS Automate application

NBI consists of the billing payload and three functions to Extract changes, Transmit payloads northbound and, to receive status updates back (Call Back).

A single NBI (only) is connected to a VOSS Automate cluster.



3.1.2. NBI Administration Portal (GUI)

The NBI Web-based Administration Portal (GUI) provides the ability to:

- View details on all sent messages - information is extracted from the Tracking Table (including message status)
- Control default Subscriber States (new Subscriber detected)
- Set Subscriber State (Pending, Live), and manage Go-Live dates
- Manage Subscribers individually, or through Bulk Load
- Manage overall Site Transition status (on/off)

Additionally, flexible filters provide the ability to easily search and locate in the different views. It is also possible to export (XLS, JSON, CSV) and import data.

3.1.3. Access Control to the NBI GUI

Secure credentials are required to access the NBI Admin GUI. The same administrator login details are used for the main VOSS Automate application.

3.1.4. Real-time Billing Payload

Whenever a subscriber is changed, a billing information payload is generated in near real-time and is sent northbound. This billing information payload, which may be customized according to the requirements of the receiving system, contains information about:

- The Subscriber
- The action taken (Add/Change/Delete)
- The set of associated devices and services that have been provisioned or changed

3.1.5. Billing Reconciliation (SDE Report)

NBI may be configured to generate a *billing friendly* report export, which may be used by the northbound system to run a lower frequency reconciliation process. This JSON format report has a configurable schedule, and can be made available to all customers or to a selection of customers. The content is configurable based on the requirements of the northbound system, and the output is deposited to a local NBI media directory.

3.1.6. Extract Function (Extractor Process)

NBI detects changes to the live service by polling for new entries in the *VOSS Automate Transaction Log*. Once a change is detected (known as a *Trigger Event*), NBI uses the standard VOSS Automate APIs to extract subscriber and service information and builds a standard, JSON format billing payload. Payloads are stored in the *Tracking Table*, and are marked as *READY* for sending by the Transmit Function.

Note: The full list of Trigger Events that cause NBI to create a payload are documented in the appendix.

To maintain the chronology of events, NBI processes top level transactions sequentially from the VOSS Automate database. The system includes mechanisms to prevent processing being blocked by large bulk load and data sync jobs that may be running on the transaction log.

If NBI is taken offline or loses the connection with VOSS Automate, it retains a record of the last transaction processed and *catches up* once it is brought back online.

NBI typically connects to Secondary Database Nodes so that undue load is not placed on the main Primary Database Node in the VOSS Automate cluster (although this is configurable).

NBI automatically connects to the Primary Database Node for write operations and replication-lag-sensitive read operations.

In a typical installation, to provide redundancy and load distribution, a list of Secondary Database Nodes is configured in addition to the Primary Database Node.

The VOSS Automate *Transaction Log* is polled to detect changes to the live system. To reduce load on the node, the polling period defaults to 5 minutes. A regular, scheduled job ('nbi_sync'), keeps the NBI synchronized with VOSS Automate (specifically, for Customer, Site and Subscriber information). This job may run overnight, and provides a key mechanism to get new Subscribers loaded into NBI.

3.1.7. Transmit Function (Notifier Process)

The Transmit Function regularly searches for payload messages in the Ready state, and then sends these northbound (via HTTP, or secure HTTPS), over a RESTful API service to a well-known destination URL.

Payload transmission may be buffered or re-sent in case of underlying network issues or if the northbound system is busy. Once the northbound system receives the messages, they are marked as *UserInProgress*. If network failures persists, messages are marked as *SendFailed*.

Note: The Transit Function automatically retries sending a payload message several times, in case of intermittent issues on the underlying network.

3.1.8. Call Back Function (Callback Process)

NBI supports a Call Back function and tracking mechanism, which the northbound system can utilize to communicate various status updates or error reports back to NBI, against each billing payload message sent.

The Call Back Function marks messages with these statuses:

- *UserProcessed* - completed successfully
- *UserFailed* - in this case, the northbound system indicates an associated Error Status

3.1.9. Tracking Table

NBI tracks all billing events and messages in its *Tracking Table*, which contains records representing billable events and their associated payloads (including status information and error details, if any).

Note: The contents of the Tracking Table and associated fields are documented in the appendix.

The *Tracking Table* is held on the VOSS Automate node(s), and so benefits from the redundancy (multiple nodes) and backup mechanisms found on these nodes. The NBI appliances itself holds no data billing payloads and their history.

3.1.10. Connectivity

NBI connects to VOSS Automate using a combination of the VOSS Automate API and direct access to the mongo database (where all NBI data is kept – there is none on the NBI appliance itself).

- Access to the VOSS Automate API is made using a service account (configured on VOSS Automate and at the Provider Level in the hierarchy), and through the VOSS Automate Web Proxy Node.
- Access to the mongodb is made through direct access to the VOSS Automate Unified Nodes, with Secondary Nodes being preferred (with the ability to move to another node if the selected node is not available).

NBI connects to the northbound system (for sending payloads) over HTTP or HTTPS. All connectivity details are defined in the NBI configuration file.

3.1.11. Provisioning Control (Site Transition Flag)

Sites may be set into a *Transition* state from the NBI GUI, whereby sending of billing payloads for that particular site is fully disabled, regardless of whether the Subscriber is in a *Live* state.

This feature is provided as a fail-safe, at a site level, and may be deployed when transitioning an existing customer onto the VOSS Automate service or during the initial provisioning of new customers.

3.1.12. Configuration

A configuration file on the NBI appliance manages the configuration of NBI operational parameters. This file is initially configured on NBI installation, and may be updated from the command line (see later commands).

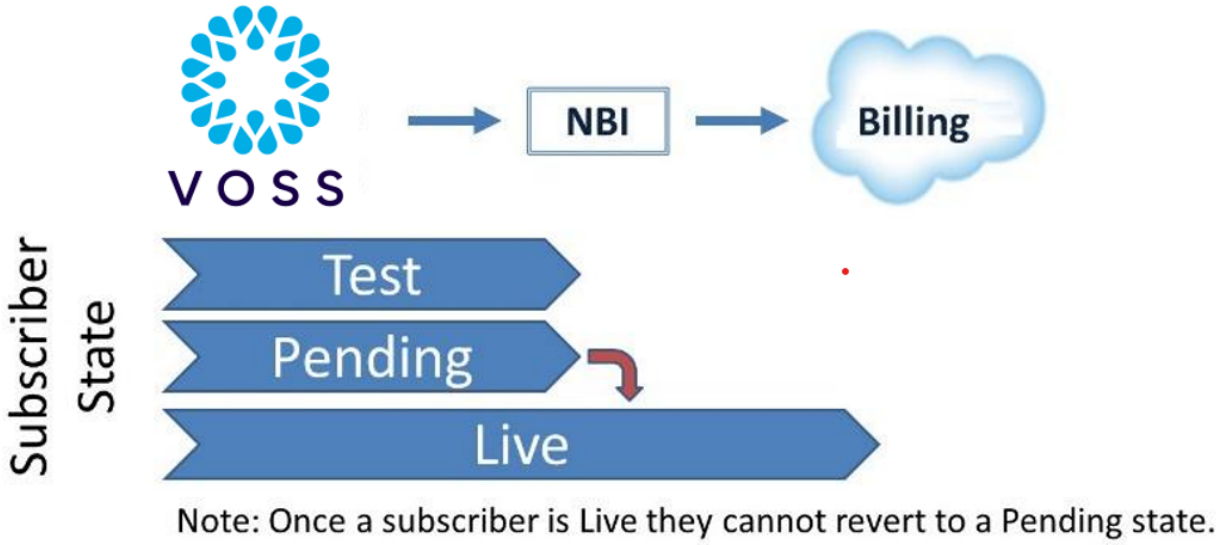
3.1.13. Diagnostics

- Message reporting screens to track billing payloads and status
- Integration and alarming with NOC systems (SNMP)
- Log files are maintained and can be accessed from the command line interface (for each of the main functions, and are date/time stamped)
- Commands to check on status and general health of the server appliance

3.1.14. Provisioning Control (User State) and managing Go-Live Dates

Subscribers (as seen by NBI) are assigned one of the following states:

- Test
- Pending
- Live



In a *Test* or *Pending* state, any changes made against the subscriber do not initiate a billing payload, and no entry is made into the Tracking Table.

Billing payloads are only sent for subscribers that are in *Live* stage.

The use cases are as follows:

a) During initial provisioning for a customer, when subscribers being on-boarded, the billing flow may be disabled. Subscribers are automatically discovered as they are built into VOSS Automate, and are placed in a *Pending* state (no payloads sent)

b) Once a *Go-Live* billing date is agreed with the customer, sites and associated users may be promoted into a *Live* state. All subsequent changes will then automatically flow into billing (payloads sent). This promotion may be made manually or pre-set through configuring a go-live date (schedule) against each subscriber.

The default subscriber state (the initial subscriber state) is configured at a site level and may be set to any of Test, Pending or Live:

Test	Subscribers with a Test State are for NBI testing purposes only; payloads are never sent northbound. Once a Subscriber is placed in a Test State, this state can never be changed.
Pending	A Subscriber in a Pending State is staged during the initial onboarding of a customer, and can then be promoted into a Live State (either manually or on a Go-Live schedule).
Live	Subscribers in a Live State are processed and billing payloads are sent automatically to the northbound system.

3.1.15. Managing Site and Subscriber Information

- Use the **Sites** menu to manage sites (associated default subscriber state, transition flag).
- Use the **Subscribers** menu to manage subscribers (associated activation state, go-live date, send new payload). Subscribers can be freely moved between *Pending* and *Live* states (and back again).

3.2. Run Time Configuration Parameters

The parameters described in this section can be configured on the NBI and are contained within the NBI configuration file.

Note: Most of these parameters may be updated from the utility found on the command line interface. Where this is not possible, contact VOSS Support for assistance (these parameters require command line root access).

Parameter	Configuration File Extract (bdeconf.json)
Poll Interval	`"pollInterval": 5000  "sets poll interval in milliseconds to check for new transactions in the VOSS Automate Transaction Log. Default is 5 minutes`
Batch Size	`"batchSize": 100    "number of records to process in a single batch when doing bulk processing`
Location of VOSS Automate application and access credentials	`"vossApiRootURL": "http://<login>:<IP>/api",` `"Administration credentials for NBI to login into the VOSS Automate application. This should be created as a service account with no password expiry and at Provider Level (e.g. nbiadmin)"`
	`"vossDbDSN": "mongodb://<login>@<IP1>:27020, <IP2>:27020, <IP3>:27020, <IP4>:27020/VOSS?authSource=admin&ssl=true&replicaSet=DEVICEAPI&readPreference=secondaryPreferred"` `"Access details for the VOSS Automate database - note each unified node is specified for redundancy with the secondary being preferred"`

Parameter	Configuration File Extract (bdeconf.json)
Call Back function	`"callback": { "address": "0.0.0.0", "port": 5009, "username": "voss", "password": "callbackpass", "url": http://<IP>:5009/callback }`
Sandbox	`"sandbox": { "address": "127.0.0.1", "port": 5006 }`
Location of the northbound service (URL) and access credentials	`"notifier": { "url": "https://<IP>:<PORT>", "useCerts": true, "indicates if certs are deployed (or not)" "key": "/opt/billing-data-extract/certs/<name>.key", "ca": "/opt/billing-data-extract/certs/ca.crt", "cert": "/opt/billing-data-extract/certs/<name>.crt" }`

3.3. Hardware and Network Specification (Single Node Server Appliance)

Resource	Specification
CPU	2 vCPU @ 2GHz with no reservation
Memory	4GB
Disk	80GB
Network	1 Gbit/s minimum

Source (Function)	Destination	Destination Port
NBI (Extract Function)	VOSS Automate	HTTPS TCP 443/8433
NBI (Extract Function)	VOSS Automate	TCP 27020
NBI (Extract Function)	VOSS Automate	SSH TCP 22
NBI (Transmit Function)	Northbound CRM	HTTPS TCP 443/8443
Northbound CRM	NBI (Call Back Function)	TCP 5009
Command Line Management	NBI	SSH/SFTP TCP 22
NBI (Services)	DNS	TCP/UDP 53
NBI (Services)	NTP	UDP 123
NBI (Services)	SNMP Server / NOC	SNMP TCP/UDP 161, 162
SNMP Server / NOC	NBI (Services)	SNMP TCP/UDP 161, 162

Network Bandwidth Requirement	
Flow - NBI to / from VOSS Automate	350 KB/s bandwidth
Flow - NBI to / from northbound system	150 KB/s bandwidth

3.4. Command Interface

Maintenance is carried out from a platform user login application command line, either via SSH or from the VM console command line. The password is configured during installation and can be changed using *system password*. On initial login, the system displays a banner indicating the general system health.

```
login as: platform
platform@              password:
You have new mail.
Last login: Fri Feb 26 17:38:27 UTC 2021 from

host: NBI-1, role: generic, load: 0.04
date: 2021-02-26 17:54:02 +00:00, up: 26 days, 1:31
network: 192.168.100.26, ntp: 192.168.100.26


    mail - local mail management          keys - ssh/sftp credentials
 network - network management           backup - manage backups
billing-data-extract - billing data extract manag     log - manage system logs
  notify - notifications control      schedule - scheduling commands
    diag - system diagnostic tools      system - system administration
    snmp - snmp configuration             user - manage users
 cluster - cluster management           drives - manage disk drives
     web - web server management            app - manage applications
security - security update tools
platform@NBI-1:~$
```

Appliance Commands

The table describes the appliance commands:

network - network management	keys - ssh/sftp credentials
notify - notifications control	log - manage system logs
diag - system diagnostic tools	schedule - scheduling commands
snmp - snmp configuration	system - system administration
drives - manage disk drives	security - security update tools
app - manage applications	user - manage users

Logs are provided for the three main NBI functions:

- Extractor
- Notifier
- CallBack processes

NBI-specific Commands (billing-data-extract)

The table describes the NBI-specific commands (billing-data-extract):

config	View and Edit billing-data-extract configuration file
run_sde	Method for running SDE to produce reconciliation files. This command can be put on schedule to execute nightly.
syncnbi	Ensure that NBI is in sync with VOSS Automate database
syncnbi-dryrun	Dry run to report on sync status between NBI and VOSS Automate database
test_connectior	Test remote VOSS Automate database and VOSS Automate API connection
message-tracking	Manage and prune message tracking database

3.5. Security Considerations (Certificates)

The upstream connection and passage of billing payloads may be secured, where required, to use HTTPS over the northbound connection. In this case, a suitably signed certificate should be produced and then installed onto the NBI – into a folder location and then this recorded into the NBI configuration file.

Note: These certificates need to be copied to, set up, and configured correctly on the northbound system.

3.6. Installation Requirements

The virtual machine is deployed using the OVA template, which is supplied by VOSS.

Installation of the NBI follows this sequence:

1. Configure VOSS Automate for NBI:
   - Open firewall ports for access to mongodb (NBI data collections) on Unified Nodes.
   - Configure NBI service account (limited access profile, password without expiry, at Provider Level).
2. Install NBI appliance OVA and set network configuration parameters.
3. Install billing-data-extract application on NBI appliance.
4. Install certificates (as required).
5. Configure and connect billing-data-extract application to VOSS Automate and northbound system.
   - Location of Unified Nodes (IPs) and Web Proxy
   - URL for northbound system
   - Service access account for VOSS Automate
   - mongodb access credentials
   - Certificates
6. Provision and restart NBI appliance for operation.

Note: Refer to the NBI Install Guide for full instructions.

4. Billing Payload Definition

NBI delivers information to northbound systems by generating a payload of information whenever an add/delete subscriber or service change is detected.

4.1. JSON Schema (Payload) for ADD / MOD Subscriber

This section provides a sample payload, representing the default payload schema. This sample can be customized to meet the specific billing requirements captured during the design stage, with regard to feeding billing information to the northbound system.

Note: Refer to the appendix for a full definition of the billing payload format (sample schema).

```
{
    "$schema":"http://json-schema.org/release-01/schema#",
     "title":"Payload Add/Change message v0.12",
    "type":"object",
    "required":["Order","User"],
    "properties":{
        "Order":{
            "type":"object",
                "required":[
                    "CallbackURL",  "MessageID",
                    "Timestamp",    "CallingSystem",
                    "UserID",            "Operation",
                    "Customer",       "Location",
                    "HardwareGroup"
                ],
        },
        "User":{
            "type":"array",
            "minItems":1,
            "items":{
                "required":[
                    "Username",
                    "ContactPhone",
                    "ExtensionNumber",
                    "MobilePhone",
                    "FirstName",
```

(continues on next page)

(continued from previous page)

```
                    "LastName",
                    "Email"
                ],
            },
            "Lines": "type":"array", "items": {"required":["ExtensionNumber",
                     "ShortNumber", DDI"]},
            "Devices": "type":"array", "items": { "required":["Model",
                     "Name"]},
            "Mobility Profile": "type":"array", "items":
                     {"required":["Model", "Name"]},
        }
    }
}
```

5. Integration Considerations and Message Sequencing

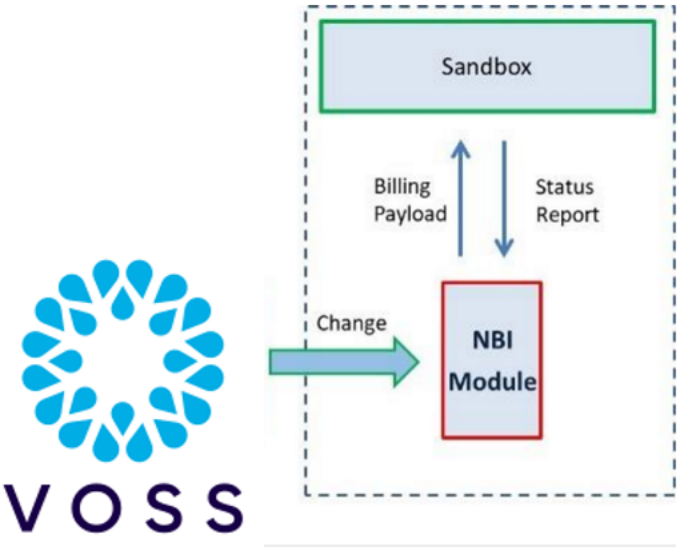
This section provides examples on integration with northbound systems.

5.1. NBI Sandbox for Lab Testing and Payload Validation

Initial implementation of the NBI in the lab can be supported with an inbuilt stub (known as the *Sandbox*), to represent the northbound system. This is a separate process that can be configured to run on the same machine as the main NBI processes.

The Sandbox:

- Receipts each billing payload message received from the NBI Notifier Function, responding with a 200 Ok (NBI marks the message as UserInProgress)
- Then performs a Call Back function, marking the field Response.Status in the call back JSON as “Success” (NBI marks the message as UserProcessed)



The Sandbox can be used to review and validate the payload sent under various use cases.

Contact VOSS Support for configuration of the NBI Sandbox.

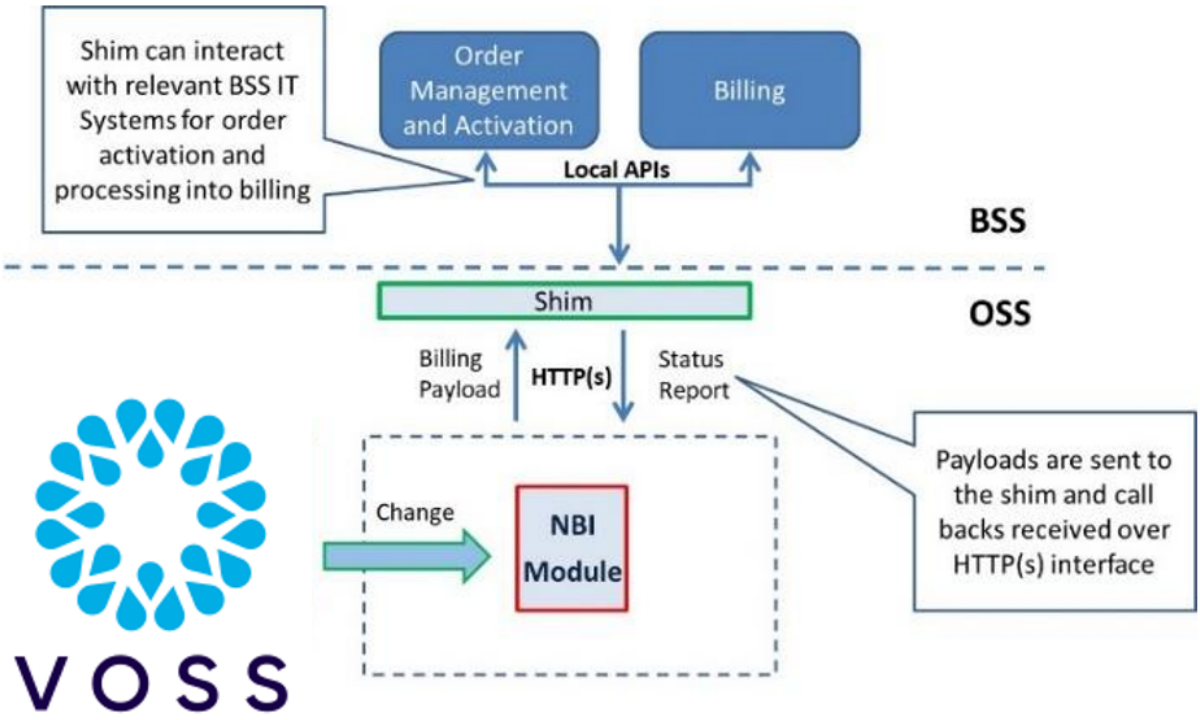
5.2. Integration into Billing

Whereas the Sandbox presents a simple integration use case, a typical integration into a northbound system is more complex because although APIs exist, they are not modern REST-based APIs. Typically, the billing system is often installed and has evolved over time, and as a result, is somewhat rigid in terms of how it can be engaged.

In this integration the choice is to deploy a thin shim to receipt the NBI payload and then make the necessary local API calls or engagement with the billing system. Once this operation is complete, the shim is then responsible for initiating the call back with NBI.

In this scenario a single call back is made, with *response.status* set to “Success” or “Error”. For the latter, a suitable error response can then be carried in the *response.ResponseText* field.

The shim itself requires some design from a provider’s IT team, with regard to its design and deployment.

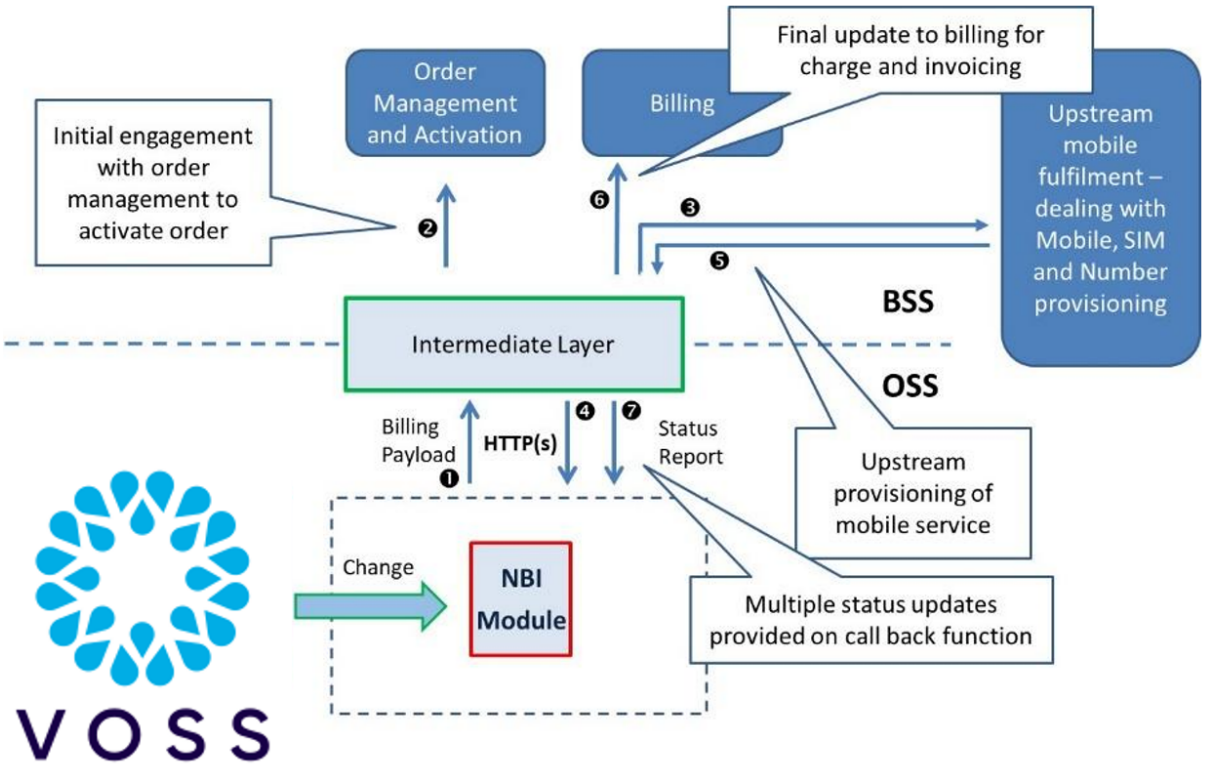


5.3. Integration and Upstream Provisioning

NBI also provides flexibility to integrate into more complex environments, where further upstream provisioning activities take place on receipt of a payload. For example, a use case where a subscriber is configured with additional services, such as a Fixed / Mobile package (FMC) that requires provisioning in upstream systems (outside of the HCS domain, and in this case with the provider’s mobile fulfilment system).

In these more complex deployment scenarios, there is a requirement for an intermediate layer that can receipt the NBI payload and then orchestrate the various northbound systems.

The steps for the use case example above is as follows:



1. The NBI sends the payload northbound.
2. The payload is receipted by the intermediate layer, which then activates order management.
3. On activation, the mobile provisioning system is contacted to provision the mobile (FMC).
4. NBI is updated, by call back, as to the progress of the operation (UserInProgressMob).
5. Mobile provisioning completes and the intermediate layer pushes the change into billing.
6. NBI is updated, by call back, of the final state of the operation (UserProcessedMob).

In this more complex example, the call back function is initiated more than once by the intermediate layer to update NBI on progress in the northbound systems.

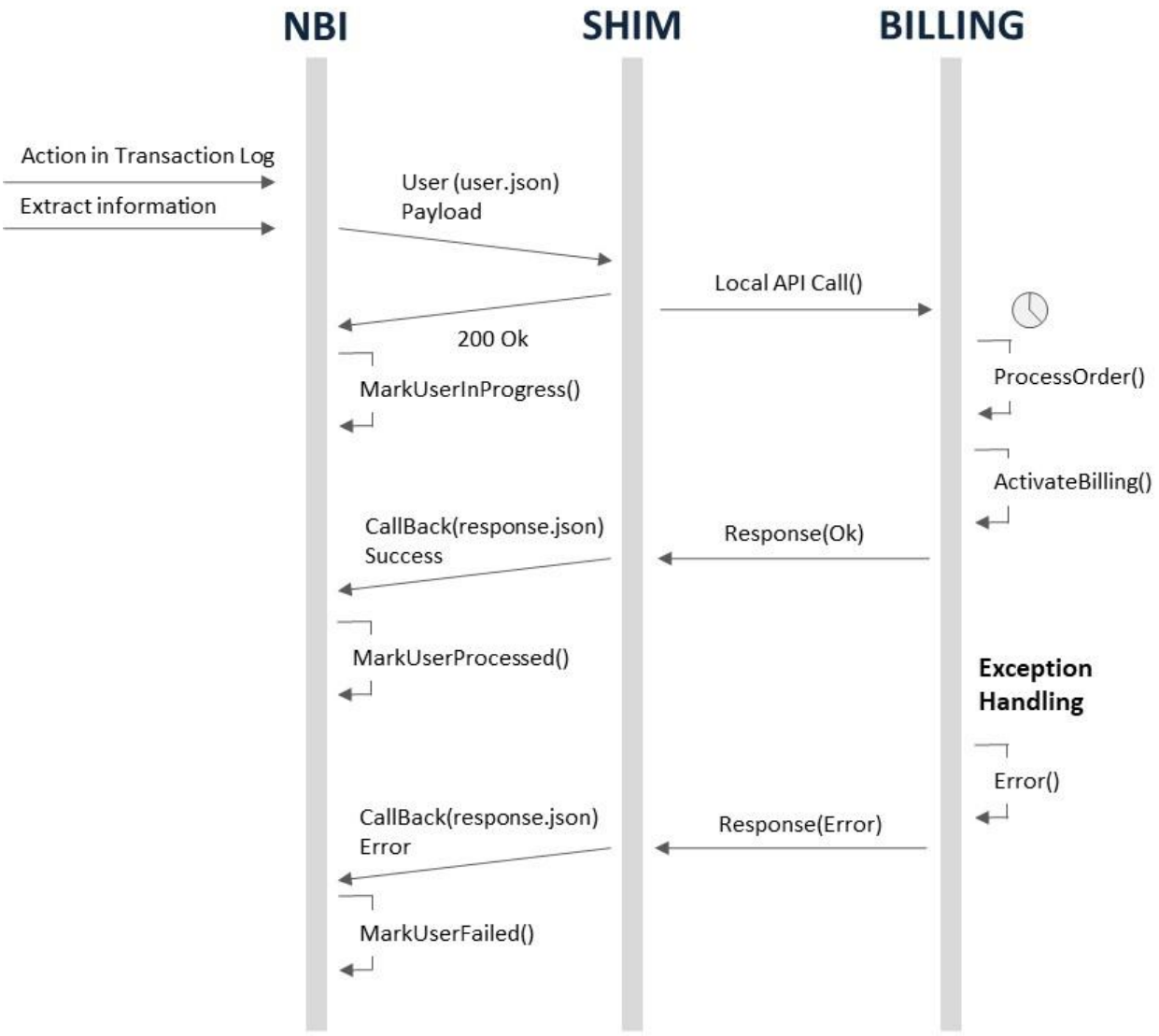
Note: Above user states are examples – specifics to be discussed with VOSS Solutions.

Reference Case:

NBI has been successfully implemented in a complex OSS/BSS estate – interfacing into an intermediate layer constructed on Software AG's WebMethods. In this deployment customer and order management was performed by SalesForce.com and charging and billing using Convergys's Geneva billing system.

5.4. Typical Sequence Diagram

The sequence diagram below illustrates the sequence of events and messages for the case where NBI is integrated with the billing system, through a shim layer.



6. Operational Matters and Production Hardening

6.1. Operating Parameters

NBI runs as a standalone Ubuntu appliance, as a virtual machine, and as such it is relatively autonomous and low maintenance. Certain operational and support arrangements are required and these are put into place as part of the go-live process.

On sign-in to the console, a health report indicates the system status.

The table describes the arrangements for operation in a production environment:

Component	Description
Support	Support services are provided directly from the VOSS Service Desk, where calls can be logged during normal business hours, either over a telephone or through the VOSS Customer Portal [https://voss.portalshape.com]. All calls are logged and assigned an Incident Number (VSR-XXXXX).
Access control	Access to NBI is protected and requires a user to enter their VOSS Automate login credentials.
Service Monitoring and Troubleshooting	NBI continually monitors a number of conditions and generates events (SNMP Traps) whenever a condition is detected. Events are categorized into three levels (Error, Warn and Info) with the default level being set at Error. Events are configured to be sent to the provider's Network Operations Centre. The SNMP alarm definitions are included in the Appendix. Detailed logs are also provided for troubleshooting and are available from the CLI.
Payload Monitoring	Message tracking displays are configured on both NBI and VOSS Automate to provide display and tracking of billing payloads being sent to the northbound system. Status information is provided on send and processing failures.
Backup and Restore	Any information required for NBI is backed up as part of the normal VOSS Automate backup process. A backup copy of the NBI configuration information should be taken after the initial installation.

6.2. Monitoring from the NBI GUI

All payload activity is logged in the Tracking Table and this may be viewed from the NBI GUI. This provides a quick and easy mechanism to check activity on the NBI.

6.3. Early Life Support

On completion of a provider's operational readiness testing, NBI will be formally handed over by VOSS under change control, along with updated documentation ('as built'). Knowledge transfer sessions are held as required.

Early life project support is then provided for a further four (4) weeks.

7. Appendix A: Trigger Events and Tracking Table

7.1. Billing Trigger Events (VOSS Automate Transaction Log)

The VOSS Automate Transaction is monitored by NBI (polled every five seconds) for bill affecting events. This section describes the *Events* deemed as bill affecting, and which cause NBI to raise a billing payload.

In the case where such an event creates a payload that is identical to the one sent previously for that subscriber (i.e. a service parameter has been changed against a subscriber but this parameter is not included in the payload), then the payload is marked as *SameAsPrevious* in the message tracking log.

```
"actions": [
     "Create Subscriber",
     "Update Subscriber",
     "Delete Subscriber",
     "Create Cucm User",
     "Update Cucm User",
     "Delete Cucm User",
     "Create Cucm Phone",
     "Update Cucm Phone",
     "Delete Cucm Phone",
     "Create Cucm Line",
     "Update Cucm Line",
     "Delete Cucm Line",
     "Create Cuc User",
     "Update Cuc User",
     "Delete Cuc User",
     "Update Bde Bulk",
     "Create Quick Subscriber View",
     "Create User Extended Dat",
     "Delete User Extended",
     "Create Hierarchy Node",
     "Update Hierarchy Node",
     "Delete Hierarchy Node",
     "Create Hierarchy Delete",
     "Create User Phone Move Users View"
 ]
```

7.2. Tracking Table

The *Tracking Table* contains a history of all billable events and the payloads associated with them.

Payload Status	The current state of the changed entity. See the list of possible states below.
Entity Type	Only needed if the entity type is not always Subscriber
PKID	VOSS Automate Subscriber PKID
Payload	Contains the JSON format event payload data.
ID	A sequential number. A unique identifier for the tracking table.
Transaction ID	The VOSS Automate Transaction ID.
User ID	The user who initiated the transaction on VOSS Automate.
Operation	Create, Update or Delete
Customer	Customer name (of the changed Subscriber)
Site	Site location name (of the changed Location)
Time	Last updated timestamp
Order ID	The order into which this change was included (set on callback from).

7.3. Payload Status

Billing Payloads are marked with one of the following status fields:

Field	Description
Received	A subscriber change on VOSS Automate has been detected (Change Record) and NBI is in the process of creating a Tracking Table record. This is a transient state.
Ready	The Tracking Table Record is populated with all the required information and the billing payload is ready for sending northbound.
UserInProgress	The billing payload has been created, sent northbound and an acknowledgement has been received (200 Ok) – now waiting for an order status update / Call Back.
SendFailed / Resent	The sending of the billing payload northbound has failed – this is most likely due to a network connectivity or credential error (certificates). The option is available for an Administrator to resend a message in this state from the GUI - during this process the message is temporarily marked as *Resent*.
ValidationFailed	The format or content of the payload is not valid (basic payload validation). This is most likely a set-up configuration issue that will need escalation to resolve.
UserProcessed	The Call Back has been received from the northbound system with a successful status (“Success”). This indicates that the billing process is complete.
UserFailed	The Call Back has been received from the northbound system – the operation has not been successful and a status report has been included in *State Update Message*. The latter is a report from the northbound system to indicate the reason for the failure and further investigation is required.
SameAsPrevious	The billing payload was created but not sent to the northbound system. This is because the contents of the billing payload have not changed from the previous payload sent for this subscriber.
UserInProgressMob / UserProcessedMob	Intermediate states where the upstream system is undertaking secondary provisioning activities (such as mobile provisioning) – see Integration and Upstream Provisioning section.

8. Appendix B: Billing Payload Definition

This appendix details the standard payload schema and examples.

8.1. Billing Payload: JSON Schema for ADD / MOD Subscriber

```
 {
   "$schema":"http://json-schema.org/V-04/schema#",
   "title":"CRM User Add/Change message v0.12",
   "description":"A JSON definition of a User MACD message for adds and changes as␣
↪required by the CRM / NORTHBOUND system",
   "type":"object",
   "required":["Order","User"],
   "properties":{
       "Order":{
           "type":"object",
               "required":[
                   "CallbackURL",
                   "MessageID",
                   "Timestamp",
                   "CallingSystem",
                   "UserID",
                   "Operation",
                   "Customer",
                   "Location",
                   "HardwareGroup"
               ],
           "properties":{
               "CallbackURL":{
                   "type":"string",
                   "description":"URL of the callback server"
               },
               "MessageID":{
                   "type":"string",
                   "description":"Unique ID of message from calling system that will be␣
↪used in responses by the CRM system"
               },
               "Timestamp":{
                   "type":"string",
                   "format":"date-time",
```

(continues on next page)

(continued from previous page)

```
                    "description":"Timestamp set by calling system, typically the system␣
↪time at message creation"
                },
                "CallingSystem":{
                    "type":"string",
                    "description":"String identifying calling system, could be logical␣
↪name of system with or without version numbers;
                    used for reports and reconciliation"
                },
                "UserID":{
                    "type":"string",
                    "description":"Login userID of individual making the request on the␣
↪calling system (or system name if triggered
                    automatically); used for reports and reconciliation and␣
↪differentiating between internal (Provider) and external
                    (customer) users"
                },
                "CustomerRef":{
                    "type":"string",
                    "description":"Transaction reference optionally provided by customer␣
↪making the request; used for
                    reports, reconciliation and possibly invoicing"
                },
                "Operation":{
                    "enum":[
                        "Add",
                        "Change"
                    ]
                },
                "Customer":{
                    "type":"string"
                },
                "Location":{
                    "type":"string"
                },
                "HardwareGroup":{
                    "type":"string",
                    "description":"Network Device List in VOSS Automate; Effectively an␣
↪indicator of which VOSS Automate cluster
                    on which this User is configured. A customer can have multiple NDLs,␣
↪but a location will only ever
                    be on a single cluster, and have a single NDL."
                },
                "ExternalCustomerID":{
                    "type":"string"
                }
            }
        },
        "User":{
            "type":"array",
            "minItems":1,
```

(continues on next page)

(continued from previous page)

```
            "items":{
                "required":[
                    "Username",
                    "ContactPhone",
                    "ExtensionNumber",
                    "MobilePhone",
                    "FirstName",
                    "LastName",
                    "Email"
                ],
                "properties":{
                    "Username":{
                        "type":"string",
                        "description":"Unique identifier of the user; usually email␣
↪address, or maybe any other identifier
                        that is meaningful to the Customer"
                    },
                    "ActivationDate":{
                        "type":"string",
                        "format":"date-time"
                    },
                    "ChangeDate":{
                        "type":"string",
                        "format":"date-time"
                    },
                    "DisconnectionDate":{
                        "type":"string",
                        "format":"date-time"
                    },
                    "ContactPhone":{
                        "type":"string"
                    },
                    "ExtensionNumber":{
                        "type":"string"
                    },
                    "MobilePhone":{
                        "type":"string"
                    },
                    "Salutation":{
                        "type":"string"
                    },
                    "FirstName":{
                        "type":"string"
                    },
                    "MiddleName":{
                        "type":"string"
                    },
                    "LastName":{
                        "type":"string"
                    },
                    "Email":{
```

(continues on next page)

(continued from previous page)

```
                        "type":"string"
                    },
                    "Title":{
                        "type":"string"
                    },
                    "EndUserVoicemail":{
                        "type":"boolean"
                    },
                    "Lines":{
                        "type":"array",
                        "minItems":1,
                        "items":{
                            "type":"object",
                            "anyOf":[
                            {
                                "required":[
                                    "ExtensionNumber"
                                ],
                                "properties":{
                                    "ShortNumber":{
                                        "type":"string"
                                    },
                                    "ExtensionNumber":{
                                        "type":"string"
                                    },
                                    "DDI":{
                                        "type":"string"
                                    }
                                }
                            }
                            ]
                        },
                        "uniqueItems":true
                    },
                    "Devices":{
                        "type":[
                            "array",
                            "null"
                        ],
                        "items":{
                            "required":[
                                "Model",
                                "Name"
                            ],
                            "properties":{
                                "Model":{
                                    "type":"string",
                                    "description":"Preconfigured device name within VOSS
↪Automate, e.g. 'Cisco IP Communicator',
                                    '7821 IP Phone', 'One Analogue Gateway port', etc."
                                },
```

(continues on next page)

(continued from previous page)

```
                                "Name":{
                                    "type":"string",
                                    "description":"Unique identifier of the device, e.g.␣
↪for IP Phone 'SEP'+Mac and for Jabber
                                    Android device 'BOT'+unique string"
                                }
                            }
                        }
                    },
                    "MobilityProfiles":{
                        "type":[
                            "array",
                            "null"
                        ],
                        "items":{
                            "required":[
                                "Model",
                                "Name"
                            ],
                            "properties":{
                                "Model":{
                                    "type":"string",
                                    "description":"Preconfigured device name within VOSS␣
↪AutomateRed, e.g. 'Cisco IP Communicator',
                                    '7821 IP Phone', 'One Analogue Gateway port', etc."
                                },
                                "Name":{
                                    "type":"string",
                                    "description":"The Device Profile / Mobility Profile␣
↪name"
                                }
                            }
                        }
                    },
                    "FMC":{
                        "type":[
                            "array",
                            "null"
                        ],
                        "items":{
                            "required":[
                                "MobileNumber",
                                "Extension"
                            ],
                            "properties":{
                                "MobileNumber":{
                                    "type":"string",
                                    "description":"FMC device mobile number"
                                },
                                "Extension":{
                                    "type":"string",
```

(continues on next page)

(continued from previous page)

```
                                    "description":"Extension number linked to FMC Mobile␣
↪Number"
                                },
                                "MobileBrand":{
                                    "type":"string",
                                    "description":"This is mobile brand - e.g.
↪'Blackberry' or 'Other Mobile Phone'"
                                },
                                "DataOption":{
                                    "type":"string",
                                    "description":""
                                },
                                "RoamingOption":{
                                    "type":"string",
                                    "description":""
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}
```

8.2. Example: JSON Schema for ADD Subscriber

```
{
 "transactionId": "d8d303e5-781f-445e-a304-71bbfc68ef39",
 "transactionSeqId": "2659147",
 "transactionTimestamp": "2018-08-15T13:10:50.484Z",
 "timestamp": "2018-08-15T13:26:07.540Z",
 "state": "UserProcessed",
 "payload": {
   "MessageID": "5b7429ef90d7eb1e45b49eec",
   "Timestamp": "2018-08-15T13:26:07.542Z",
   "CallingSystem": null,
   "UserID": "rsevenster_csp",
   "Operation": "Create",
   "CustomerRef": null,
   "Customer": "GeoLogic",
   "Location": "GLGC-London",
   "Username": "pfelt",
   "HardwareGroup": "GeoLogic-CL2-NDL",
   "ContactPhone": null,
   "ExtensionNumber": "102108",
   "MobilePhone": null,
   "FirstName": "Peter",
```

(continues on next page)

(continued from previous page)

```
   "MiddleName": null,
   "LastName": "Felt",
   "Title": null,
   "Email": "pfelt@geologic.com",
   "EndUserVoicemail": true,
   "Devices": [
     {
       "Model": "Cisco 7945",
       "Name": "SEP794500102108"
     }
   ],
   "Lines": [
     {
       "ExtensionNumber": "102108",
       "DDI": "+442074221008"
...NDL",
     "ExternalCustomerID": "GLGC-LO"
   },
   "User": [
     {
       "Username": "pfelt",
       "ActivationDate": "2018-08-15T00:00:00.000Z",
       "ContactPhone": "",
       "ExtensionNumber": "102108",
       "MobilePhone": "",
       "FirstName": "Peter",
       "MiddleName": "",
       "LastName": "Felt",
       "Email": "pfelt@geologic.com",
       "Title": "",
       "EndUserVoicemail": true,
       "Lines": [
         {
           "ExtensionNumber": "102108",
           "DDI": "+442074221008"
         }
       ],
       "Devices": [
         {
           "Model": "Cisco 7945",
           "Name": "SEP794500102108"
         }
       ],
       "MobilityProfiles": [
         {
           "Model": "Cisco 7945",
           "Name": "pfelt-UDP"
         }
       ]
     }
   ]
```

(continues on next page)

(continued from previous page)

```
},
"callback": {
  "MessageID": "5b7429ef90d7eb1e45b49eec",
  "Timestamp": "2018-08-15T13:26:18.499Z",
  "Stage": "ActiveOrder",
  "Status": "Success"
}
```

8.3. Example: JSON Schema for MOD Subscriber

```
{
"transactionId": "85de4176-1d9d-4776-b4bd-d5e592b06088",
"transactionSeqId": "2680575",
"transactionTimestamp": "2018-08-16T13:48:39.597Z",
"timestamp": "2018-08-16T13:49:27.075Z",
"state": "UserProcessed",
"payload": {
  "MessageID": "5b7580e790d7eb1e45b49efc",
  "Timestamp": "2018-08-16T13:49:27.077Z",
  "CallingSystem": null,
  "UserID": "hbarton_csp",
  "Operation": "Change",
  "CustomerRef": null,
  "Customer": "GeoLogic",
  "Location": "GLGC-London",
  "Username": "pfelt",
  "HardwareGroup": "GeoLogic-CL2-NDL",
  "ContactPhone": null,
  "ExtensionNumber": "102108",
  "MobilePhone": null,
  "FirstName": "Peter",
  "MiddleName": null,
  "LastName": "Felt",
  "Title": null,
  "Email": null,
  "EndUserVoicemail": true,
  "Devices": [
    {
      "Model": "Cisco 7945",
      "Name": "SEP794500102108"
    }
  ],
  "Lines": [
    {
      "ExtensionNumber": "102108",
      "DDI": "+442074221008"
    }
  ],
```

(continues on next page)

(continued from previous page)


```
  "...Group": "GeoLogic-CL2-NDL",
    "ExternalCustomerID": "GLGC-LO"
  },
  "User": [
    {
      "Username": "pfelt",
      "ChangeDate": "2018-08-16T13:48:39.597Z",
      "ContactPhone": "",
      "ExtensionNumber": "102108",
      "MobilePhone": "",
      "FirstName": "Peter",
      "MiddleName": "",
      "LastName": "Felt",
      "Email": "",
      "Title": "",
      "EndUserVoicemail": true,
      "Lines": [
        {
          "ExtensionNumber": "102108",
          "DDI": "+442074221008"
        }
      ],
      "Devices": [
        {
          "Model": "Cisco 7945",
          "Name": "SEP794500102108"
        }
      ],
      "MobilityProfiles": [
        {
          "Model": "Cisco 7945",
          "Name": "pfelt-UDP"
        }
      ]
    }
  ]
},
"callback": {
  "MessageID": "5b7580e790d7eb1e45b49efc",
  "Timestamp": "2018-08-16T13:49:37.661Z",
  "Stage": "ActiveOrder",
  "Status": "Success"
}
```

8.4. Billing Payload: JSON Schema for DEL Subscriber

```
{
  "$schema":"http://json-schema.org/V-04/schema#",
  "title":"CRM User Delete message v0.3",
  "description":"A JSON definition of a MACD message for deletes as required by the CRM␣
↪/ NORTHBOUND system",
  "type":"object",
  "properties":{
     "Order":{
        "type":"object",
        "properties":{
           "CallbackURL":{
                "type":"string",
                "description":"URL of the callback server"
            },
           "MessageID":{
              "type":"string",
              "description":"Unique ID of message from calling system that will be used␣
↪in responses by the CRM system",
              "required":true
           },
           "Timestamp":{
              "type":"string",
              "format":"date-time",
              "description":"Timestamp set by calling system, typically the system time␣
↪at message creation",
              "required":true
           },
           "CallingSystem":{
              "type":"string",
              "description":"String identifying calling system, could be logical name␣
↪of system with or without
              version numbers; used for reports and reconciliation",
              "required":true
           },
           "UserID":{
              "type":"string",
              "description":"Login userID of individual making the request on the␣
↪calling system (or system name
              if triggered automatically); used for reports and reconciliation and␣
↪differentiating between internal
              (Provider) and external (customer) users",
              "required":true
           },
           "CustomerRef":{
              "type":"string",
              "description":"Transaction reference optionally provided by customer␣
↪making the request; used for
              reports, reconciliation and possibly invoicing"
           },
```

(continues on next page)

(continued from previous page)

```
            "Operation":{
               "enum":[
                  "Delete"
               ]
            },
            "Customer":{
               "type":"string",
               "required":true
            },
            "Location":{
               "type":"string",
               "required":true
            },
            "HardwareGroup":{
               "type":"string",
               "required":true,
               "description":"Network Device List in VOSS Automate; Effectively an␣
↪indicator of which VOSS Automate cluster
               on which this User is configured. A customer can have multiple NDLs, but␣
↪a location will only ever be
               on a single cluster, and have a single NDL."
            },
            "ExternalCustomerID":{
               "type":"string",
               "required":false
            }
         }
      },
      "User":{
         "type":"array",
         "minItems":1,
         "items":[
            {
               "Username":{
                  "type":"string",
                  "description":"Unique identifier of the user; usually email address,␣
↪or maybe any other identifier
                  that is meaningful to the Customer",
                  "required":true
               },
               "DisconnectionDate":{
                  "type":"string",
                  "format":"date-time"
               }
            }
         ]
      }
   }
}
```

8.5. Example: JSON Schema for DELETE Subscriber


```
{
 "transactionId": "bee89f13-c772-45c6-b28b-f7be6c570632",
 "transactionSeqId": "2659127",
 "transactionTimestamp": "2018-08-15T13:06:57.596Z",
 "timestamp": "2018-08-15T13:26:03.234Z",
 "state": "UserProcessed",
 "payload": {
   "MessageID": "5b7429eb90d7eb1e45b49ee9",
   "Timestamp": "2018-08-15T13:26:03.234Z",
   "CallingSystem": null,
   "UserID": "hbarton_csp",
   "Operation": "Delete",
   "CustomerRef": null,
   "Customer": "GeoLogic",
   "Location": "GLGC-London",
   "Username": "pfelt",
   "HardwareGroup": "GeoLogic-CL2-NDL",
   "ContactPhone": null,
   "ExtensionNumber": "102108",
   "MobilePhone": null,
   "FirstName": "Peter",
   "MiddleName": null,
   "LastName": "Felt",
   "Title": null,
   "Email": "pfelt@geologic.com",
   "EndUserVoicemail": true,
   "Devices": [
     {
       "Model": "Cisco 7945",
       "Name": "SEP794500102108"
     }
   ],
   "Lines": [
     {
       "ExtensionNumber": "102108",
       "DDI": "+442074221008"
   ...ate": "2018-08-15T00:00:00.000Z"
 },
 "userId": "5481e2fd46da02148e770b4c",
 "subscriberId": "5b7421072b2c1e0d2c177928",
 "operation": "Delete",
 "activation_state": "Live",
 "GoLiveDate": "2018-08-15T00:00:00.000Z",
 "resubmitPayload": false,
 "order": {
   "Order": {
     "MessageID": "5b7429eb90d7eb1e45b49ee9",
     "CallbackURL": "http://localhost:5009/callback",
     "Timestamp": "2018-08-15T13:26:03.234Z",
```


(continues on next page)

(continued from previous page)

```
    "CallingSystem": "Default",
    "UserID": "hbarton_csp",
    "Operation": "Delete",
    "Customer": "GeoLogic",
    "Location": "GLGC-London",
    "CustomerRef": "GeoLogic-01",
    "HardwareGroup": "GeoLogic-CL2-NDL",
    "ExternalCustomerID": "GLGC-LO"
  },
  "User": [
    {
      "Username": "pfelt",
      "DisconnectionDate": "2018-08-15T13:06:57.596Z"
    }
  ]
},
"callback": {
  "MessageID": "5b7429eb90d7eb1e45b49ee9",
  "Timestamp": "2018-08-15T13:26:11.937Z",
  "Stage": "ActiveOrder",
  "Status": "Success"
}
```

8.6. Billing Call Back: JSON Schema for Response.xml (Callback Payload)

```
{
  "$schema": "http://json-schema.org/V-04/schema#",
  "title": "Response message v0.3",
  "description": "A JSON definition of a response",
  "type": "object",
  "properties": {
     "Response": {
        "type": "object",
        "required": [
           "MessageID",
           "Timestamp",
           "Stage",
           "Status"
        ],
        "properties": {
           "MessageID": {
              "type": "string",
              "description": "Unique ID of message from calling system to which this␣
↪message is a response"
           },
           "OrderID": {
              "type": "string",
```

(continues on next page)

(continued from previous page)

```
            "description": "The Order within which this message was included"
         },
         "Timestamp": {
            "type": "string",
            "format": "date-time"
         },
         "Stage": {
            "enum": [
               "Parsing",
               "ActiveOrder",
               "MobileMigrated",
               "MobileOrder"
            ]
         },
         "Status": {
            "enum": [
               "Success",
               "Warning",
               "Error"
            ]
         },
         "ResponseText": {
            "type": "string"
         },
         "User": {
            "type": "array",
            "minItems": 0,
            "items":{
               "type":"object",
               "title":"User",
               "required":[
                  "Username",
                  "Status",
                  "ResponseText"
               ],
               "properties": {
                  "Username": {
                     "type": "string"
                  },
                  "Status": {
                     "enum": [
                        "Warning",
                        "Error"
                     ]
                  },
                  "ResponseText": {
                     "type": "string"
                  }
               }
            }
         }
```

(continues on next page)

(continued from previous page)

```
            }
        }
    }
}
```

9. Appendix C: Alarm Definition

SNMP allows a northbound monitoring system (NOC) to receive notifications in the form of traps or informs in response to events, threshold violations, whatever the trap definitions in the loaded MIBs are. Notifications are categorised at three levels (INFORM, WARNING, ERROR) and the NBI may be set to report at a certain level or above. The following event notifications are reported by NBI.

In addition to the events listed here, database and API connection as well as service recovery events are listed in the NBI Troubleshooting Guide.

See the *NBI SNMP Traps* topic in the NBI Troubleshooting Guide.

Event	Entity to monitor	Message	Severity
System Startup	snmpTrapOID	.iso.org.dod.internet.snmpV2.snmpModules.snmpMIB. snmpMIBObjects.snmpTraps.coldStart	Info
System Shutdown	snmpTrapOID	.iso.org.dod.internet.private.enterprises.netSnmp. netSnmpNotificationPrefix.netSnmpNotifications. nsNotifyShutdown	Critical
Service Changes	snmpTrapOID	Process Restart	Info
Service Changes	snmpTrapOID	Process Warning, Process Stop, Process Error	Critical
Disk Status	mteTriggerFired mteHotTrigger.0	DISK ALMOST FULL: Disk <disk> is more than 80 percent full DISK FULL: Disk full	Critical
Disk Status	mteTriggerFired mteHotTrigger.0	DISK STATUS: Disk <disk> is now running below 80 percent	Info
Disk Latency	mteTriggerFired mteHotTrigger.0	ERROR: Disk slow	Critical
Disk Latency	mteTriggerFired mteHotTrigger.0	INFO: The disk latency returned to normal	Info

Event	Entity to monitor	Message	Severity
Health Emails	mteTriggerFired mteHotTrigger.0	ERROR: Trouble sending health email	Minor
Health Emails	mteTriggerFired mteHotTrigger.0	INFO: Health emails is now being sent	Info
Mailbox Status	mteTriggerFired mteHotTrigger.0	INFO: Messages for <server> auto archived as it reached more than 500	Info
Mailbox Status	mteTriggerFired mteHotTrigger.0	INFO: The total local messages for <server> has reached in excess of 200	Warning
Mailbox Status	mteTriggerFired mteHotTrigger.0	INFO: The total local messages for <server> is now under 200	Info
Health Emails	mteTriggerFired mteHotTrigger.0	WARNING: Not all notify levels is configured with an external email address	Minor
Health Emails	mteTriggerFired mteHotTrigger.0	INFO: All notify levels is now configured with an external email address	Info
Large Log Files	mteTriggerFired mteHotTrigger.0	ERROR: Log files larger than 1Gig found in /var/log	Urgent
Large Log Files	mteTriggerFired mteHotTrigger.0	INFO: /var/log rotated	Info
Service Status	mteTriggerFired mteHotTrigger.0	ERROR: Service Failures	Critical
Service Status	mteTriggerFired mteHotTrigger.0	INFO: Services started successfully	Info

Event	Entity to monitor	Message	Severity
Network Status	mteTriggerFired mteHotTrigger.0	ERROR: Network Failures	Critical
Network Status	mteTriggerFired mteHotTrigger.0	INFO: Network failures resolved	Info
Memory Usage	mteTriggerFired mteHotTrigger.0	ERROR: Memory swap error	Critical
Memory Usage	mteTriggerFired mteHotTrigger.0	INFO: Memory usage returned to normal	Info
CPU Usage	mteTriggerFired mteHotTrigger.0	ERROR: Excessive Load	Critical
CPU Usage	mteTriggerFired mteHotTrigger.0	WARNING: High CPU usage	Urgent
CPU Usage	mteTriggerFired mteHotTrigger.0	ERROR: Extremely high CPU usage	Critical
NTP Status	mteTriggerFired mteHotTrigger.0	WARNING: The ntp daemon has stopped on <server>	Critical
NTP Status	mteTriggerFired mteHotTrigger.0	WARNING: The ntp offset exceeds 1 second on <server>	Urgent
NTP Status	mteTriggerFired mteHotTrigger.0	ERROR: No ntp configured for <server>	Urgent
DNS status	mteTriggerFired mteHotTrigger.0	WARNING: No dns configured for <server>	Urgent
Domain Status	mteTriggerFired mteHotTrigger.0	WARNING: No domain configured for <server>	Urgent