

**Document of Understanding**

**<< SIMQ / NOKIA CARE >>**

Authors

| Name | Location | Phone | Email |
| --- | --- | --- | --- |
| Raghunath Kukkala | Espoo, Finland | +358468985360 | ext-raghunath.kukkala@nokia.com |
| Murugan | Espoo, Finland | +358468985561 | ext-murugan.gomathinathan@nokia.com |

Revision History

| Date | Author | Description |
| --- | --- | --- |
| 02-Apr-2013 | Lakshmy | Template for SIMQ |

Abbreviations

| Abbreviation | Description |
| --- | --- |
| EDI | Electronic Data Interchange |
| PGI | Post Goods Issued |
| DP | Delivery Prepared |
| WP7 | Windows Phone 7 |
| WP8 | Windows Phone 8 & BB5V3 security enabled devices |
| EM | Equipment Master |
| SS | Security Server |
| NAIP | Nokia Application Integration Platform |
| PMDM | Product Model Data Management |
| RDP | Reference Data Platform |
| DNN | Delivery Notification Number |
| PSN | Production Serial Number |
| ASN | Advance Shipment Notice |

Table of Contents

[A – Functional Overview 5](#_Toc357181767)

[1. Purpose of the Application 5](#_Toc357181768)

[2. Business Process flow diagrams 5](#_Toc357181769)

[3. Sub application details / Key business transaction 5](#_Toc357181770)

[4. Application User Information 8](#_Toc357181771)

[5. Application usage pattern & seasonality and special processes 8](#_Toc357181772)

[6. Support and Coverage requirements 8](#_Toc357181773)

[7. Overview of Functionality and brief descriptions of key process 9](#_Toc357181774)

[8. Online – Screen Inventory (for custom built) 9](#_Toc357181775)

[9. Dependencies 9](#_Toc357181776)

[10. History Related With the application 9](#_Toc357181777)

[B – Technical Overview 10](#_Toc357181778)

[11. Technology Stack 10](#_Toc357181779)

[Minimum setup for a single host: 10](#_Toc357181780)

[12. License Requirements 11](#_Toc357181781)

[13. System architecture diagram 11](#_Toc357181782)

[14. Database Architecture 11](#_Toc357181783)

[Physical data model & Logical data model with mapping to the transactions 11](#_Toc357181784)

[15. Backup, Recovery & Archival 11](#_Toc357181785)

[16. Infrastructure Architecture Diagram for all Environments 11](#_Toc357181786)

[17. Number of Instances 12](#_Toc357181787)

[18. Setup Procedures 12](#_Toc357181788)

[19. Interfaces 12](#_Toc357181789)

[20. Critical Jobs / Integration needs / Schedules 13](#_Toc357181790)

[C – Operational Overview 14](#_Toc357181791)

[21. Ticketing System 14](#_Toc357181792)

[a. Tools 14](#_Toc357181793)

[b. Reports 14](#_Toc357181794)

[22. Quantitative Assessment of Ticket data. 14](#_Toc357181795)

[a. Analyze Application Incident data. 14](#_Toc357181796)

[b. Trend Analysis 14](#_Toc357181797)

[23. Operating Procedures 14](#_Toc357181798)

[a. Incident Management (all types) until Ticket closure. 14](#_Toc357181799)

[b. Incident Reporting & Recording Mechanism. 14](#_Toc357181800)

[c. Planned Activity Resolution Process. 14](#_Toc357181801)

[d. Unplanned activity Resolution Process. 14](#_Toc357181802)

[e. Disaster Recovery process 14](#_Toc357181803)

[f. Change Management Process. 18](#_Toc357181804)

[g. Deployment Process / Rollback Management 18](#_Toc357181805)

[h. Log Locations / Enabling / De bugging Process 20](#_Toc357181806)

[i. Release management frequency & process 20](#_Toc357181807)

[j. Installation Process. 20](#_Toc357181808)

[k. Third party Support including Helpdesk 20](#_Toc357181809)

[l. After hours support procedure 20](#_Toc357181810)

[24. Configuration Management. 21](#_Toc357181811)

[a. Configuration Management Process. 21](#_Toc357181812)

[b. Configuration Mgmt (People, Hardware, Software,other assets). 21](#_Toc357181813)

[25. Communication Process. 21](#_Toc357181814)

[a. Communication within the Team and Client. 21](#_Toc357181815)

[b. Escalation Process and Touch Points. 21](#_Toc357181816)

[c. Customer Communication Framework for Jobs, Monitoring. 21](#_Toc357181817)

[26. Verification and Validation process. 21](#_Toc357181818)

[27. Process templates 26](#_Toc357181819)

[D – Known Issues, Bugs, Knowledgebase, Dependencies 26](#_Toc357181820)

# A – Functional Overview

## Purpose of the Application

Key functionalities of the application are

1. Delivering Unlock & Unblock codes of devices to operators
2. Tracking the Smart Card delivery
3. Admin tool helps the operators to generate codes for the devices

## Business Process flow diagrams



Application deals with 2 major functionalities like Delivery of Unlock & Unblock codes and tracking the Smart Card delivery. We have 4 processes of delivering the unlock & unblock codes

1. Fresh data sequence – e-mail delivery
2. Resend data sequence - e-mail delivery
3. EDI sequence 856 – Advance ship notice – EDI delivery
4. EDI sequence 869 – Order status inquiry – EDI delivery

## Sub application details / Key business transaction

WP7 Processing:

* There are 5 different levels of locking the WP7 phones
  + Level 1 – NW(Network)=MCC + MNC
  + Level 2 – NS(Network Subset) =IMSI
  + Level 3 – SP(Service Provider)=GID1
  + Level 4 – CP(Corporate)=GID2
  + Level 5 – SB(SIM Based)
* Unlock and unblock codes are available for each level.
* Unlock codes allow users to decouple the device from its operator subscription. The maximum attempt of this key is 10 for each level.
* If Unlock code is inserted 10 times wrongly, Unblock code should be provided.
* User will have 32 attempts to unblock.
* Codes for unlocking and unblocking will be available in EM

WP8 Processing:

* With single Unlock code, it is possible to perform unlocking and unblocking for all levels.
* Code is generated at Security Server on a need basis*.*

**SIMQ Normal Delivery Process: Fresh data sequence**

1. SIMQ request new data from EM, on a periodic time.
2. EM prepares new data (WP7 + WP8 based on Operator list & Product Type) and provides that data to SIMQ (one DNN at a time)
3. SIMQ splits the data based on Operator and then based on Component (WP7, WP8)
4. SIMQ sends request to SS
   1. If data contains WP8 devices, SIMQ request for unlock code calculation to SS
   2. If data contains WP7 devices, SIMQ request for unlock decryption/re-encryption to SS
5. SS sends response to SIMQ
   1. If data contains WP8 devices, SS calculate unlock code, encrypt the file if needed and returns file to SIMQ
   2. If data contains WP7 devices, SS Decrypt unlock/unblock data, encrypts the file if needed and returns file to SIMQ
6. SIMQ prepares files for transfer and delivers files to Operator based on configuration

**Simple Resend Process:**

1. SIMQ creates list of SN’s for resend
2. SIMQ requests EM for data
3. EM prepares that data and provides that data to SIMQ
4. SIMQ splits the data based on Component (WP7, WP8)
5. SIMQ sends request to SS
   1. If data contains WP8 devices, SIMQ request for unlock code calculation to SS
   2. If data contains WP7 devices, SIMQ request for unlock decryption/re-encryption to SS
6. SS sends response to SIMQ
   1. If data contains WP8 devices; SS calculate unlock code, encrypt the file if needed and returns file to SIMQ
   2. If data contains WP7 devices; SS Decrypt unlock/unblock data, encrypts the file if needed and returns file to SIMQ
7. SIMQ prepares files for transfer and delivers files to Operator based on configuration

**DNN Resend Process**

1. SIMQ creates entry with DNN to be resend
2. SIMQ requests EM for DNN resend data
3. EM prepares that data and provides that data to SIMQ
4. SIMQ splits the data based on Operator and then based on Component (WP7, WP8)
5. SIMQ sends request to SS
   1. If data contains WP8 devices, SIMQ request for unlock code calculation to SS
   2. If data contains WP7 devices, SIMQ request for unlock decryption/re-encryption to SS
6. SS sends response to SIMQ
   1. If data contains WP8 devices, SS calculate unlock code, encrypt the file if needed and returns file to SIMQ
   2. If data contains WP7 devices, SS Decrypt unlock/unblock data, encrypts the file if needed and returns file to SIMQ
7. SIMQ prepares files for transfer and delivers files to Operator based on configuration

**US Electronic Data Interchange – USEDI856 Flow**

1. iMES issues delivery prepared notice to SAP
2. SAP sends PGI to NAIP
3. NAIP process the iDOC and requests for unlock data (list of PSN and SN) from SIMQ
4. SIMQ creates list of PSN’s and sends that list to EM and Request for EDI unlock data
5. EM prepares data and sends that data to SIMQ
6. SIMQ splits the data based on Component (WP7, WP8)
7. SIMQ sends request to SS
   1. If data contains WP8 devices, SIMQ request for unlock code calculation to SS
   2. If data contains WP7 devices, SIMQ request for unlock re-encryption to SS
8. SS sends response to SIMQ
   1. If data contains WP8 devices, SS calculate unlock code and returns clear text file to SIMQ
   2. If data contains WP7 devices, SS decrypt unlock/unblock data and returns clear text file to SIMQ
9. SIMQ combines the data received from SS and send as one file to NAIP
10. NAIP creates EDI856 messages and sends those to Operator
    1. SIMQ periodically poll EDI Message Delivery Status
    2. NAIP sends the corresponding response to periodically poll
11. Operator gives EDI997 message to NAIP
    1. SIMQ periodically poll EDI856 confirmed delivery request
    2. NAIP sends the corresponding response to periodically poll

**US Electronic Data Interchange – USEDI869 Flow**

1. Operator requests unlock codes (EDI869) through NAIP
2. NAIP Retrieve PSNs for list of provided SNs from GMES
3. NAIP process the PGI’s and sends Request unlock data (list of PSN and SN) to SIMQ
4. SIMQ creates list of PSN’s and sends that list to EM and Request for EDI unlock data
5. EM prepares data and sends that data to SIMQ
6. SIMQ splits the data based on Component (WP7, WP8)
7. SIMQ sends request to SS
   1. If data contains WP8 devices, SIMQ request for unlock code calculation to SS
   2. If data contains WP7 devices, SIMQ request for unlock decryption to SS
8. SS sends response to SIMQ
   1. If data contains WP8 devices, SS calculate unlock code and returns clear text file to SIMQ
   2. If data contains WP7 devices, SS Decrypt unlock/unblock data and returns clear text file to SIMQ
9. SIMQ combines WP7 and WP8 devices EDI unlock data and send that to NAIP
10. NAIP creates EDI870 messages and sends those to Operator
    1. SIMQ periodically poll EDI Message Delivery Status
    2. NAIP sends the corresponding response to periodically poll
11. Operator gives EDI997 message to NAIP
    1. SIMQ periodically poll EDI870 conformed delivery request
    2. NAIP sends the corresponding response to periodically poll

## Application User Information

Key users for the application will be from SIMQ operation team. They raise request on behalf of operator in the application. And the team will have following roles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Department/User Group | Key area/functionality of the application used | Number of Users | | IT Site Contact Person (if needed) |
| Total | Concurrent\* |
| Admin | SIMQ Support team | 16 | 16 | ext-mahalakshmi.gopinath@nokia.com |
| Super Admin | SIM Operations team | 10 | 10 | [mika.t.jarvenpaa@nokia.com](mailto:mika.t.jarvenpaa@nokia.com) |
| Dongle Admin | RND Dongle from SharePoint to SIMQ | 5 | 5 | aigi.alapuranen@nokia.com |

## Application usage pattern & seasonality and special processes

SIMQ application is part of mission critical system, its needs to be available 24X7 and able to deliver codes to Operator at any time.

Exceptional peaks of activity are caused just by exceptional situations and requirements.

For E.g.: Mass DNN resend if required will generate great and unusual amount of activity in the application.

The number of elements processed by unit of time will increase dramatically in this context (e.g. in mass resend of all the existing DNNs belonging to a single operator)

## Support and Coverage requirements

T2 support is available 24X7 to cover any possible incidents.

T3 support scope is more technical and its available generally during standard working hours (8X5) the tasks mainly consists of adopting solutions for the technical problems that may happen in production, revising the state of background processes and database related issues.

SIMQ is been monitored by SiteScope in order to detect any possible anomaly happing in the application, servers or background processes. Additionally SIMQ has an archiving mechanism working both in QA & Production server’s in order to audit the activity in the application and in the servers.

## Overview of Functionality and brief descriptions of key process

Customer interacts with operator for ordering a new device. Operator in-turn make sales agreement with Account sales department and then an order is getting placed from sales department as a variant. Now this variant is being shared with Variant tool and SAP. Variant tool will send this information to variant DB. EM interacts with variant DB to get the details.

As per the order from sales department, SIM locked devices will be packed and transport loaded. Transport loading sends CDF info to operator and shipment info to EM.



## Online – Screen Inventory (for custom built)

**

## Dependencies

Availability of following applications is very important without them SIMQ will not be able to work properly, especially if EM or SS are not available.

1. EM
2. SS (Encryption)
3. NAIP (US EDI)
4. MFT (SFTP)
5. SMTP

## History Related With the application

* Inception of the application is in Nov 2011.
* It has been a common policy to update Rails to the last version available.
* Major later enhancement was the last architectural change, when SIMQ delegated encryption on SS and started to communicate directly with EM without using Tucson as intermediary.
* In USEDI it is important that deliveries happen inside a given period of time (8 hours) because in some cases exceeding these time limits may cause penalties from our customer.

# B – Technical Overview

## Technology Stack

## Minimum setup for a single host:

* Large virtual server: 4vCPU, 8GB memory, 30GB storage
* <https://simq.nokia.com> in which simq.nokia.com is a DNS CNAME pointing to the active production site, by default simq-fin.nokia.com.
* Access to SIMQ Production service interface is through NAIP
* All service traffic will be encrypted over SSL through the VIP
* Two virtual servers load balanced by one VIP
  + 80 GB Storage preferred
  + 8 GB Memory
  + Ruby (1.9.3 or more recent) + RVM (Ruby Version Manager)
  + RubyGems
  + RVM will take care of the majority of version issues related to Ruby and related libraries and components.
  + SSL Enabled (Port 443), Port 80 disabled
* One COP database – High Availability
  + 100GB
* Intranet (CARE) connection will be SSL

Operating system

* Red Hat Enterprise Linux (RHEL) release 6.3 (Santiago)

Database Servers

* Nokia standard COP servers
* Single instance for IT/SI
* High Availability instances for QA and PROD

Software / Language

* Apache 2.2.15
* Ruby (1.9.3 or more recent) + RVM (Ruby Version Manager)
* Ruby Gems 1.8.24
* Phusion Passenger 3.0.18
* RVM will take care of the majority of version issues related to Ruby and related libraries and components.
* Oracle 11g DB
* Oracle 11g Client
* PL/SQL Developer or Oracle SQL Developer

The tools and any Integration technologies used

* MFT

Major and minor technologies / frameworks the application is built with.

* Ruby on Rails
* Phantomjs (Testing)
* Cucumber (Testing)
* Capistrano

The version control tool

* Git 1.7.6

## License Requirements

*[Mention the Software Licences /Requirements/Availability (if pertinent for support requirement / offshore access]*

## System architecture diagram

* *

## Database Architecture

## Physical data model & Logical data model with mapping to the transactions



## Backup, Recovery & Archival

Two Production systems are being maintained now (PROD1, PROD2). In case, if any one Production system has a service break then the other Production system replaces/ handles it.

In case of a long service break in the both PROD environments then the QA environment can be relatively quickly converted to take over the production.

**OFF SITE /ON site locations of backups**

ICO Computing as the hosting service provider is responsible on the back-up arrangements.

When restoration to an alternate server is needed, Computing shall be able to restore the databases and file systems directly to the secondary servers.

Critical locations where copies of these documents are stored:

* Key Managers’ / Team leads’ lap top,
* IT Service Documentation Teamtool <http://www.connecting.nokia.com/itservicedocumentation>, and
* encrypted on a CD/DVD or
* encrypted on memory stick.

## Infrastructure Architecture Diagram for all Environments



## Number of Instances

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Instance  No | Instance type  ( Eg : Production , Development , Testing ) | Instance Location  [ If hosted outside Customer’s Network ] | Synchronization with Production Instance , State Yes / No | Accessibility  Details ( URL , IP Address etc ) | Remarks / Centralized / Decentralized |
| 1 | Integration Testing | Salo | No | https://it.simq.nokia.com/ 147.243.188.108 | Centralized |
| 2 | Integration Testing | Salo | No | https://it.simq.nokia.com/ 147.243.188.109 | Centralized |
| 1 | Quality Assurance | Salo | Yes | https://qa.simq-fin.nokia.com sasimqq001.europe.nokia.com | Centralized |
| 2 | Quality Assurance | Salo | Yes | https://qa.simq-fin.nokia.com sasimqq002.europe.nokia.com | Centralized |
| 1 | Production | Salo | Yes | https://simq-fin.nokia.com sasimqp002.europe.nokia.com | Centralized |
| 2 | Production | Salo | Yes | https://simq-fin.nokia.com sasimqp002.europe.nokia.com | Centralized |
| 1 | Sandbox | Espoo | No | 10.160.109.251 | Centralized |

Detail the ALL access URL and links for different components such as

Version control tools - https://source.nokia.com/

Documents - https://extranet.nokia.com/sites/CareSolutionDelivery/Lifecycle and Security/Shared

## Setup Procedures



## Interfaces

The interfaces related documents are attached here for EM, NAIP, SS

EM



NAIP



SS



## Critical Jobs / Integration needs / Schedules

Background processes are critical part of SIMQ, as the most business critical processes, such as Fresh data load and US-EDI 856, are handled by these automated processes without any human intervention.

Background processing is built on [Delayed::Job](https://github.com/collectiveidea/delayed\_job), and uses database to queue and manage jobs.

SIMQ processes could, in theory, executed on one take by a single thread. Due to number of integrations, it's safer to split the work to number of delayed jobs. For instance, once we get a DNN from EM and split it to SBatches and CodeDeliveries, we don't have to do this process all again if Security Server happens to unreachable for a while.

*Critical Job Information*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Critical Job* | *Server or Site* | *Schedule Name* | *Execution Interval* | *Impacts if job does not run* |
| ExpireSessions | Production Application Server | ExpireSessions | 5 min | Session won’t Expires |
| EmPeriodicFreshDataCall | Production Application Server | EmPeriodicFreshDataCall | 1 min | EM integration won’t work |

# C – Operational Overview

*[ If processes are common across all applications , these could Common processes could be segregated out in a separate document]*

## Ticketing System

## Tools

*[Mention about the tool used by the users to log the issues/Tickets, links, authentication, authorization etc)*

## Reports

*[What reports are generated from this tool , how is it reported to the customer ,, format etc]*

In SIMQ application available reports are

1. SIMQ Reports (Product and Batches, Delivery Statuses)
2. SCF Report (Deliveries, Roles, CD-Keys)

## Quantitative Assessment of Ticket data.

## Analyze Application Incident data.

*[Percentage of ticket data in planned / unplanned, % by criticality,% by data issues , production support issues , breakfix related etc ]*

## Trend Analysis

*[Gather the Ticket data for the application for last 3-4 months. Get the Root Cause analysis document from the present incumbent.]*

## Operating Procedures

## Incident Management (all types) until Ticket closure.

## Incident Reporting & Recording Mechanism.

*[How the incidents are reported, where they are logged, how it is being accessed etc)]*

## Planned Activity Resolution Process.

## Unplanned activity Resolution Process.

## Disaster Recovery process

The entire primary production system is hosted in Salo Datacenter.

*NOTE: Always gain permission from SIMQ team and service manager before starting these steps.*

If it is not possible to bring the failed production environment online in **six** hours then take the steps detailed below.

#### Taking an alternate environment to production use in SIMQ

There are a couple of alternatives to selecting a secondary environment in case the primary SIMQ instance in Salo cannot be recovered within hours.  
The decision which approach to use has to be done separately on case by case basis.

1. Assuming the whole site in Salo is not cut out of the network while only the PROD cell in Salo is lost, using the QA instance as the secondary instance is the desired option.

**Note:**  Since QA and PROD environments are hosted in the same datacenters, there is a risk that anything serious enough to take down PROD instance will also take QA down. Should QA not be available, this alternative cannot be chosen.

1. Putting the IT/SI instance in production use may or may not be an option, but certainly that would require large concensus among the SIMQ core team and with the representatives of the integration peers.  
   Please, understand that this is very much a **LAST RESORT** option since firewall exceptions will be required.
2. In the ideal world there would be a pre-installed and pre-configured hot stand-by instance hosted in a geographically distant enough data center such that any damage disrupting service at the primary site were very unlikely to stretch itself also to the secondary site.  
   Since there is no hot stand-by instance available at the time of this writing, one has to be prepared to setting up a whole new instance in a very speedy manner, if the worst were to happen and both PROD and QA became unavailable simultaneously.

When the secondary instance has been selected, follow the step described below as applicable to the secondary instance selected.  
All of them will not be appropriate independent of which secondary instance is selected.

1. Request the Nokia DNS administrators to point the DNS CNAME record (alias) **simq.nokia.com** to the VIP of the chosen secondary site.  
   Under normal production conditions it will refer to **simq-fin.nokia.com**.
2. Inform all SIMQ Integration peers (see Service managers for details) and SIMQ Admin team by email with high priority that the production environment is not available and **the selected alternate environment** shall be taken into use

Content of email:  
  
Title:   
SIMQ production environment is unavailable – switch to **XXX** environment  
  
Text:   
The SIMQ IT production environment is unavailable.   
The service will be moved to the **XXX** environment.   
Please note that NO TESTING IS ALLOWED until further notice.   
If you have ongoing or planned test activities against the SIMQ **XXX** environment please stop them.   
Prepare to redirect your production connections to SIMQ **XXX** environment.  
If your system refers to SIMQ only using the domain name **simq.nokia.com** the service may recover on its own after a while.

In these sample snippets the place holder **XXX** shall be replaced with the name or description of the secondary instance selected.

1. Inform all key users (*see service managers for details*) by email with high priority that primary SIMQ production environment is not available and which secondary site shall be taken into use.

Content of email:  
  
 Title:  
SIMQ production environment is unavailable – switch to **XXX** environment  
  
Text:   
The SIMQ production environment is unavailable.   
The service will be moved to the **XXX** environment.   
Please note that NO TESTING IS ALLOWED until further notice.   
If you have ongoing or planned test activities against the SIMQ **XXX** environment please stop them.

In these sample snippets the place holder **XXX** shall be replaced with the name or description of the secondary instance selected.

1. Confirm that the primary PROD environment is still down and there is no acceptable ETA for its recovery, and then follow the technical steps explained in detail below.
2. Detailed steps to take the selected secondary instance in PROD use

* Ensure that the application servers in the selected secondary instance contain same applications and release versions that were used in PROD before the disaster.  
  Deploy the versions used in the primary PROD instance, if necessary.
* Delete current application data on the selected secondary instance.  
  (e.g. QA data stored to last snapshot backup)
* When the FW request is implemented: change the application servers in the selected secondary instance to refer to PROD touchpoints.
* Restart the application processes in the selected secondary instance.
* Remember to request system monitoring team to activate the same monitor probes that have been used for PROD also for the secondary instance.
* Unless the archiving service already exports the archive volume to the selected secondary site, request the NFS export be enabled ASAP.
* Unless the firewalls already allow NFS connections from the selected secondary site to the archiving service, request the firewall rules be added ASAP.
* On the secondary site:  
  Unmount /archive, if there is an NFS mount.  
    
  umount /archive

Make sure there is the following mount rule enabled in /etc/fstab.

saecua02:/sfs/gateway\_nfs/simq\_archive /archive nfs rw,soft,nolock 0 0

Also comment out the following, if it was present and enabled in /etc/fstab.

saehnasq10:/qa-simq /archive nfs rw,soft,nolock 0 0

Mount the proper archive volume.  
  
mount /archive

* On the secondary site:  
  In case the selected secondary site was redirecting connection to the primary production site, remember to comment out the following line in the section <VirtualHost \*:81> in the httpd.conf file

**Redirect permanent /** [**https://simq-fin.nokia.com/**](https://simq-fin.nokia.com/)

Since the QA setup has similar hardware and software specifications to PROD, full production functionality and performance should be available immediately once the connectivity between all of the integration peers has been properly redirected to the secondary instance.  
*This assumption will most likely not be true, if the IT/SI environment gets used.*

#### Restore the service to the production environment

Once the primary production environment is back online (fixed or rebuilt), take the steps described below to return to normal operation:

The SIMQ team will take care of notifying the services about the availability of any SIMQ instances *(esp. QA and PROD)* once proven and communicating changes needed to reverse any previous configuration changes.

Detailed steps to restore the production environment back in use:

* Ensure that all primary SIMQ PROD servers are operational.
* If VMWare backup is available: Restore SIMQ servers from VMWare backup images.
* If VMWare backup is not available: Create a fast track ticket for computing to implement PROD server’s installations according ISG document.  
  Deploy correct release versions SIMQ software to PROD servers from source repository. The word “correct” refers to the latest known good release that has been used for production previously.
* Check that application’s configuration is in a sane state and restart the application processes.
* Move the data generated on the selected secondary instance during the period it was used to the primary PROD instance.
* Monitor the primary PROD environment for correct operation.
* If the QA instance was used as the selected secondary instance, return it to its normal state.
  + Stop the application processes on the QA servers
  + Restore QA configuration to use QA touch points
  + Restore QA data from latest snapshot backup
  + Restart QA applications processes
* Tell the system monitoring team to reactivate monitoring probes for PROD.
* On the secondary site:  
  Reverse the archive volume mount done while moving the service to temporary location.   
  First unmount the archive.  
    
  umount /archive  
    
  Then comment out the following mount rule in /etc/fstab.
* saecua02:/sfs/gateway\_nfs/simq\_archive /archive nfs rw,soft,nolock 0 0
* On the secondary site:  
  In case the selected secondary site was redirecting connection to the primary production site, remember to uncomment the following line in the section <VirtualHost \*:81> in the httpd.conf file
  + - **Redirect permanent /** [**https://simq-fin.nokia.com/**](https://simq-fin.nokia.com/)

## 

## Change Management Process.

#### Change Control and Change Approval Board Information.

*[List the steps that should be taken to initiate the change control , to get the approval etc)Documents and templates to be used].*

#### Communication Process

*[Who needs to be contacted for the CR’s and how. List the tools, contacts etc.]*

## Deployment Process / Rollback Management

**Deploying to production**

You must tag the version to be deployed (see the previous section)

You need AppGate access to "NeSe Prodcution Networks"

The more sensitive configuration is not in the version control and the deployment script does more work to support this.

If you need to update e.g. service passwords during the deployment, these have to be done manually.

Deploy using tag instead of branch

You usually deploy:update\_code as a preparation step and deploy:publish\_code during the "official" deployment.

Preparation deploys (NOTICE the big capital -S):

```

cap -S tag=v2.X.X prod001 deploy:update\_code

cap -S tag=v2.X.X prod002 deploy:update\_code

```

Let the users know how long we are down:

```

cap prod001 deploy:web:disable REASON="Updating to release 2.X.X" UNTIL="XX:XX Finnish time"

cap prod002 deploy:web:disable REASON="Updating to release 2.X.X" UNTIL="XX:XX Finnish time"

```

…and/or you can use the Maintenance tab to display a message during the update.

Finally, after everything seems to be in place:

```

cap prod001 deploy:publish\_code

cap prod002 deploy:publish\_code

```

…and possibly:

```

cap prod001 deploy:web:enable

cap prod002 deploy:web:enable

```

**Single file upload to production and hot fixes**

Sometimes you might need to upload a single file to servers (e.g. product migrations). Other times you need to do a hot fix to patch an issue on a HAML view or something.

Normally you should follow the normal deployment procedure by tagging a version and so on. Sometimes there are, however, fixes that are important enough and low risk enough that it's ok to do a hot fix by replacing a single file on the servers. This could e.g. be a SCF view that raises an Exception and the fix is no-risk change on one line in the view template.

NOTE: Do NOT EVER attempt this kind of updates on SIMQ processes or anything that could possibly break something running in the background.

```

cap prod001 deploy:upload FILES=app/views/xxx/xxx.html.haml

cap prod002 deploy:upload FILES=app/views/xxx/xxx.html.haml

```

If you deploy a single template (that e.g. earlier caused a problem), you should not need to restart servers. If the restart needs to happen (which is already suspicious enough for these kinds of hot-fixes), you can do it by:

```

cap prod001 deploy:restart

cap prod002 deploy:restart

```

If something goes so wrong that you even need to restart delayed jobs:

```

cap kill\_djs

cap kill\_djs

```

In the normal deployment procedure the servers and delayed jobs are restarted automatically.

**After the deployment**

The 'master' branch should always keep up-to-date with the latest stuff we have on production. So merge your release branch/tag to master.

If you need to do hot-fixes to production, DO NOT TRUST the master branch? Instead, make your changes to a branch based on the tag that was deployed.

**Rollback**

It is simple just to create a symbolic link (symlink) to the previous version which is working fine

Command to create symlink

ln -s {/path/to/file-name} {link-name}

Example

ln -s /AppacheWWW/simq/releases/2013151030 current

The deployment doc is also added it contains all the environments deployment procedures



## Log Locations / Enabling / De bugging Process

Logs available at ApacheWWW/simq/current/logs

## Release management frequency & process

## Installation Process.



## Third party Support including Helpdesk

*[Document the details of third party involvement for providing support]*

## After hours support procedure

*[Document the afterhours support procedure]*

## Configuration Management.

## Configuration Management Process.

*[Mention the overall Process of doing changes to the source code]*

## Configuration Mgmt (People, Hardware, Software,other assets).

*[List down the key people and their contacts , tool used for CM , software required to access the tool and any special; hardware/software Requirements*.]

## Communication Process.

## Communication within the Team and Client.

*[How the status reports are shared , template . Frequency of client meeting ,Team meeting , MOM’s etc .]*

## Escalation Process and Touch Points.

*[Depict the Escalation Process with details like (Name , Contact information)]*

## Customer Communication Framework for Jobs, Monitoring.

*[How the Jobs failure/success is being reported to client. Who needs to be contacted in case of major failure etc].*

## Verification and Validation process.

**Test Phases and Types**

The following types of testing will be conducted in both the environments:

**Point to Point Testing**

The objective of PP Testing is to verify that the integrations between various systems work with the use of stubs to recreate similar data flow between them.

PP Testing is similar to Integration Testing except for the use of stubs to recreate same data flows.

**Point to Point Test Scope**:

* One full cycle of PP Test Cases will be executed by the Test team.
* All defects detected during PP testing will be raised in QC and retested once fixed.
* Failed Unit test cases will be executed in subsequent cycles.

**Regression Testing**

The purpose of regression testing is to ensure that the existing system performs as expected and is not affected when new components are introduced into the system.

Test Scenarios will be developed where the new unit codes will be tested. The old test cases will be run against the new ones to make sure that all the old capabilities still work.

The tests will be carried out in the IT and QA environment as applicable.

Test Team will write test scenarios and test cases with Nokia inputs and Nokia will review them.

**The Scope of Regression Testing includes**:

A round of Smoke Test Cases will be executed prior to the Regression Testing.

All Test Cases should be entered in QC after the required approval process and execution also carried forward using SIMQ QC (details given in coming pages)

Testing team will log all defects found in Mercury Quality Center and specify the appropriate severity and priority for each defect.

The Defects identified would be fixed in Dev environment by the developers and the fixes would be applied in IT and QA environment during the subsequent IT & QA build.

A complete round of Smoke Test Cases will be executed when a new build moved into IT & QA.

**System Integration Testing**

System Integration Testing ensures that the integration between SIMQ application and other systems, including third party applications, functions as expected.

The objective of System Integration Testing is to test the in-scope requirements of SIMQ. The focus of the testing is on business processes.

SIT includes integration testing between SIMQ and other related applications like TUCSON, EM, SS.

The System Integration Test cases will be reviewed by Nokia Concept Team

The System Integration Testing will be carried out in IT Environment

The Responsibility of writing and executing the System integration test cases is with the Test Manager/Testing Team.

Mercury Quality Center will be used for recording, executing test cases (manually), and management of any defects found during this phase.

The System integration Test result will be delivered to Nokia Manager.

**The Scope of System Integration Testing includes**:

A complete round of Smoke Test Cases will be executed prior to the SIT execution and every time new builds are deployed.

One full cycle of System Integration Test Cases will be executed by the Testing team.

Testing team will log all defects found in Mercury Quality Center and specify the appropriate severity and priority for each defect.

The Defects identified would be fixed in Dev environment by the developers and the fixes would be applied in IT environment during the subsequent IT build.

All SIT cases will be executed in subsequent cycles.

**Smoke Testing**

The Smoke Testing is done to verify that the SIMQ application test environment is stable and that the build (code), configuration and all other changes is robust enough for testing before the onset of the actual Testing.

The SIMQ application Test Team conducts Smoke Tests before the onset of the actual Testing for SIMQ application.

These high-level, non-exhaustive tests aim to detect possible installation and configuration problems, to check that the needed components and backend connections are in place, as well as to make sure that the most crucial functions of SIMQ application work.

**The Scope of Smoke Testing includes**:

One full cycle of Smoke Test Cases will be executed by the Testing team with every IT & QA build.

All major functionalities of the application will be tested during the Smoke Testing phase to ensure that the build is stable and ready for testing.

**User Acceptance Testing**

UAT is a process to obtain confirmation from the owner/client of the AUT (Application under Test), through trial, that application meets the mutually agreed-upon requirements.

The focus is on a final verification of the required business function and flow of the application, emulating real-world usage of the application.

Test Manager will plan, manage and report UAT.

UAT ensures that the application is ready for moving to Production.

UAT should ensure that all requirements are working as expected.

Defects found in UAT should be tested in IT and passed and then moved to QA environment for retesting.

**Performance Testing**

Performance Testing -- Currently it is customer scope and customer is using tieto to get it done**.**

Performance Testing will be carried out in QA Environment. Performance test requirement specification and design is done by.

Performance test Environment set up, verification and validation is a joint responsibility of & Nokia

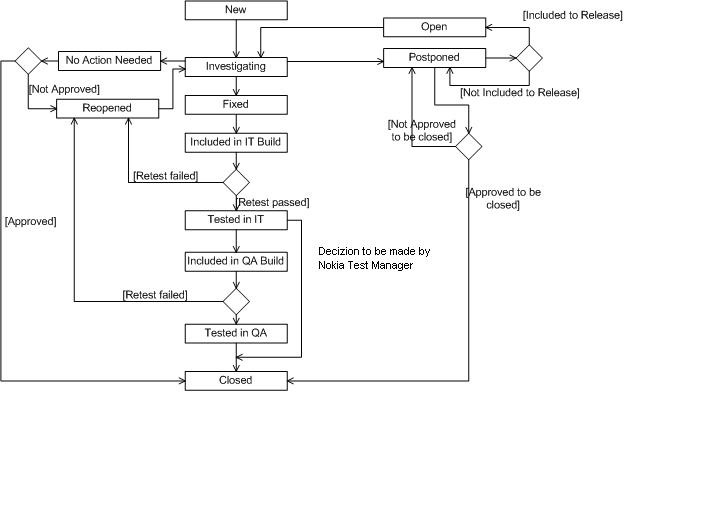
Nokia will do test scenarios.

Performance test script creation and execution is done by Nokia Test Specialist.

Performance Testing will be conducted to ensure that load increases do not affect the performance of the SIMQ and related applications.

**Defect Management**

The defect Life Cycle followed is as in the following figure



Following are the explanations of each of the status of the defects raised

| **Defect Status** | **Description** |
| --- | --- |
| New | When a new defect is created, the status is automatically set to “New”. |
| Investigating | The status “Investigating” is set by the person to whom the defect has been assigned. When person starts working on a “ New” (re “ Reopen”) defect he/she should change the status to “Investigating” |
| Fixed | The correction has been completed and unit tested information from Test Manager is that this mean it is not tested. |
| Included in IT build | The status “Included in IT build” is used to indicate that a correction has been implemented into the IT testing environment and is ready for testing. |
| Tested in IT | When a corrected defect passes the test in the IT environment, the status is set to “Tested in IT”. |
| Included in QA build | The status “Included in QA build” is used to indicate that a correction has been implemented into the QA testing environment and is ready for testing. |
| Tested in QA | When a corrected defect passes the test in the QA environment, the status is set to “Tested in QA”. |
| Closed | The defect has been corrected and verified in all the needed tests. |
| Reopened | If a previously corrected problem re-occurs, the defect can be reopened. A defect is also reopened, if a corrected defect fails the re-test. In this case the defect can be directly assigned to the developer who made the correction. |
| No Action Needed | If investigations show that the reported issue is not a fault, and that no corrective actions are needed information from Test Manager that this means no actions is needed from create team, but there has or can be actions from other parties, the status is set to “No Action Needed”. This is the same as “Rejected”. The person who raised the defect then needs to check the comments and needs to either close or reopen the defect depending on the situation. In some cases, it may be decided that an issue, which is not a fault, is to be handled as a Change Request. In such a case, the “Change Request” is set as “Y”, and the status is changed according to what is appropriate in that particular situation |
| Open | The defects remains not yet resolved and is pending closure |
| Rejected | The defect involved is not valid and thus has been rejected citing valid reasons for the same |
| Postponed | If it is decided that the correction of a defect is postponed to, for example, a later release, the status “Postponed” is used. Business representatives have to approve postponing and including defect to scope for future releases or Maintenance Release. Maintenance Release needs also Production Manager’s approval. |
| Open | The defect has been postponed and will be corrected in the current release or Maintenance Release. Needed approvals are in place and information about planned release is updated at defect information. |

Given below is the categorization of defects priorities:

|  |  |
| --- | --- |
| **Priority** | **Description** |
| P1 | * Total loss of functionality of product. Defect prevents usage of product/system. * Significant loss of functionality. A part of the applications/systems feature does not work, or the whole feature does not work in some situations. |
| P2 | * Level of functionality is degraded, but preventing normal flow of operation (according to use case = Test Scenario definition). Some functionality cannot be utilized. Examples: * Application or system is unstable. * A part of applications/systems feature does not work or whole feature does not work in some situation * User interface doesn’t react as documented or usage is strongly complicated. * Errors during operations, but these errors do not prevent the operation. * Fault in some uncommon situation |
| P3 | * Level of functionality is degraded, but not preventing normal operation (according to use case = Test Scenario definition). * Documentation error * Major User interface errors, e.g. major graphical errors, incoherencies but no functionality is impacted. |
| P4 | * User interface errors, e.g. minor graphical errors, spelling mistakes, incoherencies but no functionality is impacted. |

* Severity category for the defect is equal with level of Priority during testing phase.
* If there are any P3 and P4 defects that are Not Closed, there must be a clear continuation plan for closing these defects.
* P1, P2 defect(s) cannot be in Open status to move the current build in higher environments unless agreed otherwise with Product team.
* All the Defects found in IT will be kept in status “Tested in IT” after they are fixed and verified by the testing team in IT and QA level. Test Manager will move the status of the defects to Closed only after discussion.
* All the P1 & P2 Defects found in IT level will be closed after the defects are tested and verified also in QA.

## Process templates

Obtain templates and relevant formats for Unplanned / Planned activities

# D – Known Issues, Bugs, Knowledgebase, Dependencies

*<<Document the availability and details as applicable>>*

