|  |  |
| --- | --- |
|  | |
| 3rdMarketPlace-forUseOnLight-Background  **VerioContact**  **Functional Requirements Specifications** | |
| **Release Date:** | 15 November 2012 |
| **Author:** | Verio Inc. |
| **Approver:** | Verio Inc. |
|  |  |
| User of the document is responsible to ensure use of current version of the document.  This document is for restricted circulation. No part of this document shall be reproduced, stored in retrieval system or transmitted in any form or by any means – recording, photocopying, electronic and mechanical without prior permission of Verio Inc. | |
|  | |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Release Date | Author | Nature of Change |
| 0.2 | 11/15/2012 | Verio Inc. | Second Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1 Introduction 4](#_Toc339555017)

[1.1 Overview 4](#_Toc339555018)

[1.2 Purpose 4](#_Toc339555019)

[2 Functional Components Overview 4](#_Toc339555020)

[2.1 Role Based Access 4](#_Toc339555021)

[2.2 VerioContact Components 6](#_Toc339555022)

[2.2.1 Portal Framework Components 6](#_Toc339555023)

[2.2.2 VerioContact GUI 8](#_Toc339555024)

[2.2.3 VerioContact Message Instantiation Service 9](#_Toc339555025)

[3 Use case Elaboration 10](#_Toc339555026)

[3.1 Portal Framework Components 10](#_Toc339555027)

[3.1.1 Template Management 10](#_Toc339555028)

[3.1.2.2 Message Template Form Fields 18](#_Toc339555029)

[3.1.2 Token Management 18](#_Toc339555030)

[3.1.2.2 Token Form Fields 24](#_Toc339555031)

[3.1.3 User Role Management 24](#_Toc339555032)

[3.1.4 Branding repository Management 26](#_Toc339555033)

[3.1.5 Globalization Management 27](#_Toc339555034)

[3.2 VerioContact GUI 31](#_Toc339555035)

[3.2.1 Configuration Management 31](#_Toc339555036)

[3.2.2 Test Message Instantiation 37](#_Toc339555037)

[3.2.3 Audit Logs 41](#_Toc339555038)

[3.3 VerioContact Message Instantiation Service 41](#_Toc339555039)

[3.3.1 Sending Message Instantiation requests 41](#_Toc339555040)

[3.3.2 Requesting Status Service for requests 42](#_Toc339555041)

# Introduction

## Overview

The VerioContact project targets development of a modular application providing outbound communication services to Verio software applications. In doing so, the application will offer the ability to apply customized styling & branding, reporting, administration & statistics in relation to the outbound communications.

VerioContact will provide a services-based interaction model for other Verio applications to invoke. The medium of communications is initially focused on E-mail to be further expanded to include SMS as well.

## Purpose

The Purpose of this Document is to define requirements for the VerioContact application, its interactions with consumer applications. These descriptions will define the scope of work for VerioContact Project - Phase II.

This document takes into consideration the initial list of features, capabilities and prioritization covered in the pre-SOW and SOW discussions.

# Functional Components Overview

## Role Based Access

Roles are broadly categorized at two access levels:

* Users having access to all accounts within their hierarchy.
* Users having access within their own account.

Following are the roles relevant to VerioContact:

|  |  |
| --- | --- |
| **Role Name** | **Description** |
| ContactConfigMaster | Users with this role have access to override application parameters, update the existing parameters at profile level for all accounts within account hierarchy. |
| ContactConfigAdmin | Users with this role have access to override application parameters, update the existing parameters at profile level for their own account. |
| ContactTemplateMaster | Users with this role have access to customize, edit, delete, search, view customized message templates for all accounts existing within account hierarchy.  (Customization refers to customizing message template body with specific tokens, profile, locale, and subject). |
| ContactTemplateAdmin | Users with this role have access to customize, edit, delete, search and view the customized message templates for their own account.  (Customization refers to customizing message template body with specific tokens, profile, locale, and subject). |
| ContactRoleMaster | Users with this role have access to map users to roles of all accounts existing within his/her account hierarchy. |
| ContactRoleAdmin | Users with this role have access to map users to roles existing within their own account. |
| ContactProfileMaster | User with this role have access to configure date format and currency format settings at profile level based on locale, update, reset to default settings for all accounts within account hierarchy. |
| ContactProfileAdmin | User with this role have access to configure date format and currency format settings at profile level based on locale, update, reset to default settings for their own account. |
| ContactSystemTemplateMaster | User with this role have access to root account for creating new templates ,deleting templates associated with the default profile(white labelled) for the VerioContact application. |
| ContactMessageRequestor | Users with this role have access to send message requests via the consumer application for outbound communication services, view the status service of the sent request. |
| ContactSystemConfigMaster | Users with this role have access to configure system level application parameters (Dispatch Rate, Transmitter Status, Default Email/SMS gateway settings). |

## VerioContact Components

### Portal Framework Components

| S. No. | Component Name | Description | Business Role | Priority |
| --- | --- | --- | --- | --- |
|  | Message Template Management | * Message Template management involves creating new message templates both Email and SMS type, update, delete, search, view message templates within the application. * A message Template defines the layout for a message based on account profile and locale language. * It consists of static text and optionally "tokens" (i.e. dynamic placeholder text). | ContactSystemTemplateMaster | High |
|  |  | * Message Template associated with default profile can be customized at profile level based on locale. * Customization refers to message Template content, subject line and tokens within the template. | ContactTemplateMaster,  ContactTemplateAdmin | High |
|  | Token Management | * Token Management involves create new tokens, update, view, delete, search tokens within the application. * Tokens are dynamic placeholder values in the message Template to be replaced at the time of message instantiation by the Consumer Application. | ContactTemplateMaster,  ContactTemplateAdmin | High |
| 3. | User Role Management | * User Role Management involves associating, disassociating existing users with roles. * It includes viewing roles of the existing users. | ContactRoleAdmin,  ContactRoleMaster | High |
| 4. | Globalization Management | * Globalization Management involves locale settings management at a global level for the VerioContact application. * It includes creating, editing, deleting, viewing profile specific locale settings within the application. | ContactProfileAdmin,  ContactProfileMaster | High |
| 5. | Branding Repository Management | 1. It will decide the styling (look and feel) and logos/images to be used for any communication. Branding repository will be common between VMP portal framework and VerioContact. 2. It will include:  * A folder for each profile with respective custom images and CSS (colours, fonts) files within it. * Style sheets consisting of brand able and non-brand able content. |  | Low |

### VerioContact GUI

| S. No. | Component Name | Description | Business Role | Priority |
| --- | --- | --- | --- | --- |
| 1. | Configuration Management | * Configuration Management involves configure, update the application parameters :  1. Email gateway settings 2. SMS gateway settings 3. Transmitter Status 4. Dispatch Rate. | ContactConfigMaster,  ContactConfigAdmin | High |
| 2. | Test Message Instantiation | * Test Message Instantiation allows the user to send test Email/SMS messages by the VerioContact application. * Allows the User to send test Email/SMS messages via the profile specific gateway settings by specifying the necessary message parameters. | Accessible to all the Verio Contact roles | High |
| 3. | Audit logs | * It will maintain Technical audit logs in database to keep tracks of various changes done via application GUI by users. These changes would include Additional/Deletions/Updates made in templates, tokens, configuration settings, locale settings. |  | Low |
| 4. | Report Management | * It will be handled by WSO2 BAM to generate reports for all outbound Email/SMS communications. * No separate UI to be displayed for managing reports within VerioContact application. |  | Medium |

### VerioContact Message Instantiation Service

| S. No. | Component Name | Description |
| --- | --- | --- |
| 1. | Message Instantiation Request Handler. | * Message instantiation Request Handler provides Restful web service interface for “Consumer Applications” to request a communication to be sent out. |
| 2. | Message Validator | * It validates all assembled messages from assembler for ensuring no token placeholder is left in the assembled messages. |
| 3. | Message Transmitter | * Responsibility of Message Transmitter is to receive requests for Email/SMS dispatch and listen to “Dispatch Queues” and forward these to appropriate Email/SMS gateway. * Message transmitter will make use of the following configuration:  1. Profile to Email/SMS gateway mapping. 2. Throttling rate. |
| 4. | Bounce Handler | * Any bounced communication will land up in the bounce email box specified at profile level. Bounce handler will poll the email box and route it to “Error Router”. |
| 5. | Status Service | * A restful web service, which will help “Consumer Applications” retrieve the status of any communication requested providing “Transmission ID” received as acknowledgement during request, as input. * Consumer applications would need to specifically   Invoke status service to retrieve the status of any communication. |
| 6. | Message Listener | * Message listener will receive the messages from Request Queue. * It makes the entry for the same in database and passes it on to message assembler for assembly. |

# Use case Elaboration

### Portal Framework Components

### Template Management

* A message Template defines the content of communication messages based on account profile, locale and type. It will consist of static text and optionally "tokens" (i.e. dynamic placeholder text). Such tokens would be replaced when the message is instantiated (triggered) based upon actual parameterize values passed in at the time of instantiation.
* Initially, message Templates will be defined for a default profile (white labelled profile) consisting of static text and dynamic token placeholders. This template can be further be customized for different profiles and locales as needed.



#### Create Template

|  |  |
| --- | --- |
| Use Case ID | Create Template |
| Summary: | This use case allows the actor to create new templates associated with the default profile(white labeled profile) |
| Pre-condition: | The user must be logged in. |
| Post-condition: | * System saves the template for the white labeled profile in the database * System displays success message. |
| Includes: |  |
| Actor(s): | ContactSystemTemplateMaster |
| Trigger: | Actor clicks Create template tab. |
| Normal Flow: | 1. Actor clicks Create Template tab. 2. System displays Message Template form. Please refer Message Template form fields. 3. Actor enters Message Template form fields:  * Actor enters Template Name * Actor enters Template description * Actor selects message type from dropdown. * Actor selects locale from dropdown. * Profile dropdown shows default profile (white labeled). * Actor enters message Subject line. * Actor enters message Body content.  1. Actor clicks save button. 2. System prompts confirmation message. 3. Actor confirms creation of new template. 4. System saves the template. |
| Alternate Flow: | None |
| Exceptions: | In case of step 3 of normal flow, if template name is not unique then   * System will display an error message “Template Name Exists”. * System will not save the template. |
| Frequency of Use: |  |
| Business Rules: |  |
| Assumptions: | * All message templates created are associated with the default profile (white labeled profile). |
| Notes & Issues: |  |

#### Edit Template

|  |  |
| --- | --- |
| Use Case ID | Edit Template |
| Summary: | This use case allows the actor to edit or modify a previously created template, its content, information. |
| Pre-condition: | 1. At least one Message Template should exist for the user.  2. User should be logged in. |
| Post-condition: | 1. **On Save -**  · Saves the message Template with its associated profile ID and Locale in the database.  · Displays success message.  · Is visible to the role.  2. **On Publish -**  · Publishes the message Template.  · Visible to the Consumer Application.  - Can be used by the Consumer Application at the time of message instantiation.  **3. On Preview -**  · Displays the preview of the template via browser window.(actual form of the template) |
| Includes: | Customize Template |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster, ContactSystemTemplateMaster. |
| Trigger: | Actor clicks Message Template tab. |
| Normal Flow: | 1. Actor clicks the Message Template Tab. 2. If user has role “ContactSystemTemplateMaster” ,then  * System displays List of templates associated with the default profile. * Actor selects a Template to edit. * System displays the Message Template form with the populated values of the existing Template. * Actor updates the form field attributes as required. Please refer Message Template form field attributes.   -Profile dropdown contains white labelled profile (default profile) only.   * Actor clicks save button. * System prompts confirmation message. * Actor confirms the operation. * System updates the Template with captured form attributes and is reflected back at all usages.  1. Actor clicks publish Template button. Modified Template is published. |
| Alternate Flow: | 1. In case of step 2 of normal flow, if user has role “ContactTemplateMaster”,then  * System displays list of templates within his account hierarchy other than the default profile templates. * Actor selects a Template to edit. * System displays the Message Template form with the populated values of the existing Template. * Actor updates the form field attributes as required. Please refer Message Template form field attributes.   (Profile dropdown contains profiles within his account hierarchy).   * Actor clicks save button. * System prompts confirmation message. * Actor confirms the operation. * System updates the Template with captured form attributes and is reflected back at all usages.  1. In case of step 2 of normal flow, if user has role ”ContactTemplateAdmin”,then  * System displays list of templates within his own account other than the default profile templates. * Actor selects a Template to edit. * System displays the Message Template form with the populated values of the existing Template. * Actor updates the form field attributes as required. Please refer Message Template form field attributes.   (Profile dropdown contains profiles within his account.   * Actor clicks save button). * System prompts confirmation message. * Actor confirms the operation. * System updates the Template with captured form attributes |
| Exceptions: | 1. Branch after step 3 of normal flow.  * If the selected template is a Published Template, then * Actor first un publishes the Template. * Actor updates the Template form fields. * Actor clicks save Button. * System saves the updated Template in the database. * Actor clicks the publish Template button. Modified Template is published. |
| Frequency of Use: |  |
| Business Rules: | * Modification Date will be stored in the database against the Templates. * A published template updated will be reflected back to all its existing usages within the application |
| Assumptions: | Message Templates associated with the default profile (white labeled profile) are edited by the ContactSystemTemplateMaster. |
| Notes & Issues: | Template Name is not editable. |

#### Customize Template

|  |  |
| --- | --- |
| Use Case ID | Customize Template |
| Summary: | This use case allows the actor to customize existing Message templates for a different profile and locale. |
| Pre-condition: | * The user must be logged in. * Templates associated with the default profile must exist. |
| Post-condition: | * System saves the template with associated profile and locale. * System displays success message. |
| Includes: |  |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster. |
| Trigger: |  |
| Normal Flow: | 1. Actor clicks Message Templates tab. 2. If the actor has role “ContactTemplateMaster”,then 3. System displays the List of Message Templates available to the user within his account hierarchy. 4. Actor selects a Message Templates associated with the default profile. 5. System displays the message template form with the populated values of the existing message template for default profile. 6. Actor updates the form field attributes. Please refer Message form field attributes.   -Profile dropdown lists all profiles within the user’s account hierarchy.   1. System saves the new customized template with the captured form fields. 2. Template is saved with the associated profile and locale. |
| Alternate Flow: | In case of step 2 of normal flow, if user has role “ContactTemplateAdmin”, then actor can customize templates for profile of his own account. |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: | An existing Template can be associated to a different profile or locale setting in customize mode. |
| Assumptions: |  |
| Notes & Issues: |  |

#### Delete Template

|  |  |
| --- | --- |
| Use Case ID | Delete Template |
| Summary: | This use case allows the actor to delete an existing template of the application created by the user. A deleted  Template is no longer in use by the application. |
| Pre-condition: | The user must be logged in. |
| Post-condition: | Template is deleted from the database and does not exist in the Application.  · System displays success message on deletion.  · System displays error message on failure of deletion. |
| Includes: |  |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster,  ContactSystemTemplateMaster. |
| Trigger: | Actor clicks delete Template tab. |
| Normal Flow: | 1. Actor clicks Message Templates tab.  2. System displays the List of Message Templates available to the user.  3. Actor selects a Message Templates.  4. System displays the message template form.  5. Actor clicks delete button.  6. System prompts confirmation message.  7. Actor confirms deletion.  8. System deletes the Message templates from the database.  9. Deleted Templates no more exists in the application. |
| Alternate Flow: | None. |
| Exceptions: | 1. If a message instantiation request exists in the request queue using the deleted Template, then  * The message requests will be cancelled out.  1. If no message Templates for the logged user exists, then  * System will display a message "No Templates Exists". |
| Frequency of Use: |  |
| Business Rules: | 1. All Templates accessible at account level to the user can be deleted.  2. Delete operations should have confirmation messages.  3. Message Requests pending in request queue using the deleted Templates will cancel out automatically.  4. Customized Templates deleted being used will be overridden by the template defined for the default profile (white labeled). |
| Assumptions: | 1. ContactSystemTemplateMaster role can delete templates associated with default profile. 2. ContactTemplateMaster role can delete customized templates associated with all accounts within his account hierarchy. 3. ContactTemplateAdmin role can delete customized templates associated with his account. |
| Notes & Issues: |  |

#### Search Template

|  |  |
| --- | --- |
| Use Case ID | Search Template |
| Summary: | This use case allows the actor to search message templates on a keyword based search on the following fields:   * Template Name * Template Description. |
| Pre-condition: | · The user should be logged in.  · Templates should exist. |
| Post-condition: | · All existing Templates accessible to the user are listed out with the necessary details. |
| Includes: |  |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster, ContactSystemTemplateMaster |
| Trigger: | Actor clicks the Search Message Templates Tab. |
| Normal Flow: | 1. Search Template Tab.  2. Actor can search the template on the basis of:  · Template Name  · Template Description  3. System generates the List of templates with the list table attributes.  4. Actor selects a template from the list.  5. System displays the template in edit mode. |
| Alternate Flow: | If no templates are found on the search criteria, then the system displays the message "No templates found" to the user. |
| Exceptions: | If no field is selected or entered by the user for the search, then the system displays an error message "Select at least one field". |
| Frequency of Use: |  |
| Business Rules: | · Approximately 20 templates are listed out on a page. |
| Assumptions: | 1. Templates accessible to the role at account level will be displayed.  2. System lists all published and unpublished Templates. |
| Notes & Issues: |  |

**List Table attributes:**

|  |  |
| --- | --- |
| Field Name | Description |
| Template Name | Unique Identifier for Templates. |
| Template Description | Describes the Template |
| Template ID | System generated Unique Template ID |
| Last Modified Date | Template last Updated date |
| Last Modified by | Template last Updated By |
| Profile Name | Template associated account/profile |
| Locale | Template associated locale language |
| Published/Unpublished | Actionable item. |
| Template Type | Email/SMS, Plain/Html. |

### 3.1.2.2 Message Template Form Fields

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Field Type | Description | Validation | Mandatory |
| Template Name | Dropdown/  Text Field | Unique Identifier for Templates. | Contains only Alphanumeric text. | Yes |
| Template Description | Text Area | Describes the Template |  | Yes |
| Profile Name | Dropdown | The profile associated with the template |  | Yes |
| Locale | Dropdown | The locale associated with the template. |  |  |
| Subject | Text Field | The subject line describing the message in brief. | Should not be more than 30 characters | Yes |
| Body | Text Area | Defines the Message body to be sent. | Should not be more than 50 characters in case of SMS templates. | Yes |
| Message Type | Dropdown | The message can be of SMS, Email type, Plain Text, HTML,  All type. |  | Yes |
| Save | Button | System saves the message template in database |  | No |
| Cancel | Button | System will not save the template. |  | No |

### Token Management

* Token Management involves creating, updating, deleting, listing of tokens in the application.
* Tokens are dynamic placeholder values in the message Template to be replaced at the time of message instantiation by the Consumer Application.
* Tokens or nested Templates can be referenced in one or more Message Templates of the application.
* Tokens are replaced in the message Template based upon the actual parameterize values passed at the time of message instantiation.



#### Create Token

|  |  |
| --- | --- |
| Use Case ID | Create Token |
| Summary: | This use case allows the actor to create new tokens for the application at system level associated with the default profile (white labeled). |
| Pre-condition: | 1. The user must be logged in. |
| Post-condition: | 1. System saves the tokens in the database. 2. System displays success message. |
| Includes: |  |
| Actor(s): | ContactSystemTemplateMaster |
| Trigger: | Actor clicks Create token tab. |
| Normal Flow: | 1. Actor clicks Create Token tab. 2. System displays token form. 3. Actor enters token form fields. Please refer token form fields. 4. Actor clicks save button. 5. System validates form fields. 6. System saves token in the database with captured field values. |
| Alternate Flow: | None. |
| Exceptions: | In case of step 3 of normal flow, if token name already exists in the database then the   * System will not save the token. * System displays an error message. |
| Frequency of Use: |  |
| Business Rules: |  |
| Assumptions: | * All tokens are associated with the default profile (white labeled).   Profile dropdown will list only the white labeled profile. |
| Notes & Issues: |  |

#### Edit Token

|  |  |
| --- | --- |
| Use Case ID | Edit Token |
| Summary: | This use case allows the actor to update the defined tokens for the application. |
| Pre-condition: | 1. The user must be logged in.  2. At least a token should exist in the database. |
| Post-condition: | **On Edit:**   * System updates/modifies the token in the database. * System displays success message. * Is visible to the role. |
| Includes: | Customize token |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster, ContactSystemTemplateMaster |
| Trigger: | Actor clicks Edit token Tab. |
| Normal Flow: | 1. Actor clicks edit token Tab.  2. System displays list of tokens  3. Actor clicks a specific token.  4. System displays token details.  5. Actor modifies token form attributes. Please refer token attributes section.  6. System prompts confirmation message.  7. Actor confirms updating the token.  8. System updates token in the database.  9. Updated tokens automatically reflected in all the message templates. |
| Alternate Flow: | In case of step 7 of normal flow, if actor chooses not to update the token, then the system will not update the token. |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: | 1. Updated tokens are reflected at all usages and references in the application. 2. Token name updated is always unique. 3. Tokens accessible to the user can be edited/ updated. |
| Assumptions: | 1. System Template Master can edit the master list of tokens associated with the default profile. 2. ContactTemplateAdmin can edit customized tokens associated with the default profile for his/her account profile. 3. ContactTemplateMaster can edit customized tokens associated with default profile for all accounts within the account hierarchy. |
| Notes & Issues: |  |

#### Customize Token

|  |  |
| --- | --- |
| Use Case ID | Customize Token |
| Summary: | This use case allows the actor to customize the tokens for the specific accounts in the application. The actor can customize tokens with different values for a specific profile and locale. |
| Pre-condition: | 1. The user must be logged in. 2. At least a token should exist in the database. |
| Post-condition: | 1. System saves the token in the database. 2. System displays success message. 3. Is visible to the role. |
| Includes: |  |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster. |
| Trigger: | Actor clicks Edit token Tab. |
| Normal Flow: | 1. Actor clicks edit token Tab. 2. Actor selects customize token option. 3. System displays master list of tokens associated with default profile. 4. Actor clicks a specific token. 5. System displays token details. 6. Actor modifies token form attributes. Please refer token attributes section. 7. System prompts confirmation message. 8. Actor confirms the operation. 9. System saves token in the database. 10. Tokens automatically reflected in all the message templates. |
| Alternate Flow: |  |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: | Token name cannot be customized. |
| Assumptions: | 1. ContactTemplateMaster can customize the master list of tokens associated with the default profile for all accounts within his account hierarchy. 2. ContactTemplateAdmin can customize the master list of tokens associated with the default profile for his account. |
| Notes & Issues: |  |

#### Delete Token

|  |  |
| --- | --- |
| Use Case ID | Delete token |
| Summary: | This use case allows actor to delete tokens. A deleted token will not exist and is no longer in use by the application. |
| Pre-condition: | The user must be logged in.  At least one token should exist in the database.  Token is not in use. |
| Post-condition: | 1. System deletes token from the database. 2. System displays success message. 3. Is not visible to the application. |
| Includes: |  |
| Actor(s): | ContactSystemTemplateMaster |
| Trigger: |  |
| Normal Flow: | 1. Actor clicks delete token tab. 2. System displays list of tokens. 3. Actor views list of Tokens. 4. Actor checks the tokens to be deleted. 5. Actor clicks delete button. 6. System displays confirmation message. 7. Actor confirms token deletion. 8. System deletes token from the database. 9. Token does not exist. |
| Alternate Flow: | In case of step 7 of normal flow, if the actor chooses not to delete, then system will not delete the token. |
| Exceptions: | In case of step 8 of normal flow, if the token is in use by the consumer application, then the token will not be deleted.   * System generate error message. * The use case is terminated. |
| Frequency of Use: |  |
| Business Rules: | 1. Token once deleted have no longer any reference or existence in the application. 2. Tokens in use by the Consumer Application cannot be deleted. 3. System checks Token references and use before all delete operations. 4. System always displays confirmation message for all delete operations. |
| Assumptions: |  |
| Notes & Issues: |  |

#### Search Token

|  |  |
| --- | --- |
| Use Case ID | Search Token |
| Summary: | This use case allows the actor to search the list of existing tokens specific to his profile on a keyword based search on token name. |
| Pre-condition: | 1. The user must be logged in. 2. At least one profile specific token should exist in the database. |
| Post-condition: | **On success:**  System displays the token list.  **On failure:**  System displays the message “No token Found”. |
| Includes: |  |
| Actor(s): | ContactTemplateAdmin, ContactTemplateMaster, ContactSystemTemplateMaster. |
| Trigger: | Actor clicks Search Tokens tab. |
| Normal Flow: | 1. Actor clicks search tokens tab. 2. Actor enters the search criteria i.e. Token name. 3. System displays tokens list with the following fields: 4. Token Name. 5. Token ID. 6. Token Type. 7. Token Value 8. Locale |
| Alternate Flow: | None |
| Exceptions: | If no token exists in the database. In that case, system will display an error message "No token exists". |
| Frequency of Use: |  |
| Business Rules: | All tokens specific to the users profile and the default profile (White labeled profile) are visible. |
| Assumptions: |  |
| Notes & Issues: |  |

### Token Form Fields

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Description** | **Validation** | **Mandatory** |
| Token Name | Text Field | Unique Token Identifier Name defining the location.·  Ex-account.users.details.AccountName | Should consist of at least 6 characters. | Yes |
| Token Type | Dropdown | Type of Token: Plain, HTML. etc. |  | Yes |
| Token value | Text Field | Value of the token |  | No |
| Profile | Dropdown | Profile associated with the token |  | Yes |
| Locale | Dropdown | Locale associated with the token |  | Yes |

### User Role Management

* This web component is independently deployed and becomes accessible to applications like VerioContact and others via HTTP(S) URL reference. This component includes multiple roles, listed based upon the portal invoking it.
* This package involves assigning, de-assigning existing users with roles. It includes viewing assigned and de-assigned roles of a user.



#### Assign Roles

|  |  |
| --- | --- |
| Use Case ID | Assign Roles |
| Summary: | This use case allows the actor to assign roles to a user within the VerioContact Application |
| Pre-condition: | 1. The user must be logged in. 2. Roles must exist in the system. 3. At least one user should exist in the system |
| Post-condition: | 1. User is assigned the selected roles. 2. System displays success message. |
| Includes: |  |
| Actor(s): | ContactRoleAdmin, ContactRoleMaster |
| Trigger: | Actor clicks Assign Roles tab. |
| Normal Flow: | 1. Actor clicks roles tab. 2. System displays Roles form. 3. Actor enters the following details:   · Actor selects an account.  · Actor selects a user from the list of users.  · Actor checks one or more roles from list of roles.   1. Actor clicks Save button. 2. System prompts a confirmation message. 3. Actor confirms assigning roles. 4. System saves the selected roles for the selected user. |
| Alternate Flow: |  |
| Exceptions: | * In case of step 5 of normal flow, if the user is already assigned the selected roles, then the system will make no change. |
| Frequency of Use: |  |
| Business Rules: | 1. A user can be assigned multiple roles. 2. Roles will be listed down based upon the portal invoking it. 3. Users will be listed based upon account selection. |
| Assumptions: | * ContactRoleMaster can assign roles to all users existing within his account hierarchy. * ContactRoleAdmin can assign roles to all existing users of his account only. |
| Notes & Issues: |  |

#### DE assign Roles

|  |  |
| --- | --- |
| Use Case ID | De-assign roles |
| Summary: | This use case allows the actor to de-assign or remove roles of a user. |
| Pre-condition: | 1. The user must be logged in. 2. At least a user having a role should exist. |
| Post-condition: | 1. System de-assigns/de-associates the roles of a user. 2. System displays a success message |
| Includes: |  |
| Actor(s): | ContactRoleAdmin, ContactRoleMaster |
| Trigger: | Actor clicks De-assign roles tab. |
| Normal Flow: | 1. Actor clicks Roles tab. 2. System displays the Roles form. 3. Actor enters the following details:   · Actor selects an account.  · Actor selects a user from the list of users.  · Actor un-checks one or more roles from list of roles.   1. Actor clicks save button. 2. System prompts a confirmation message. 3. Actor confirms the de-assigning of unchecked roles. 4. System de-assigns/removes the roles of the selected user from the database. |
| Alternate Flow: | * In case of step 9 of normal flow, if the actor chooses not to de-assign, then the system will make no changes to the existing roles of a user. |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: | * All assigned and de-assigned roles are visible to the user. * Roles will be listed down based upon the portal invoking it. * Users will be listed based upon account selection. |
| Assumptions: | * ContactRoleMaster can de-assign roles to all users existing within his account hierarchy. * ContactRoleAdmin can de-assign roles to all existing users of his account only. |

### Branding repository Management

* A file system based content repository accessible over http that is used to decide the styling (look and feel) and logos/images to be used for any communication. Branding repository will be common between VMP portal framework and VerioContact.
* Verio will be developing File Manager System for Branding repository, to be reused in VerioContact as well.
* Profile will make the basis of branding and there will be a folder for each profile with respective custom images and CSS (colors, fonts) files within it.
* Branding theme will be managed based upon the associated profile.

[To be elaborated-POC under progress]

### Globalization Management

* Globalization Management involves locale settings management at a profile level for the VerioContact Application.
* It includes creating, editing, deleting, viewing locale settings for the application.
* Locale represents a set of user interface properties-including Date Format, Currency Format. It determines what language UI text is displayed in it, how dates are formatted, what the language encoding is.
* These settings will be applied during message instantiation step to format certain tokens in the message Templates.



#### Specify new Locale Settings

|  |  |
| --- | --- |
| Use Case ID | Specify new Locale Settings |
| Summary: | This use case allows the actor to create a new Locale-specific profile in the system. This locale profile will be applied during message instantiation step to format certain tokens in the message. |
| Pre-condition: | The user must be logged in. |
| Post-condition: | 1. System saves the locale-specific profiles. 2. System displays success message. |
| Includes: |  |
| Actor(s): | ContactProfileAdmin, ContactProfileMaster. |
| Trigger: | Actor clicks the locale configuration tab. |
| Normal Flow: | 1. Actor clicks locale configuration tab. 2. Actor selects new locale Configuration. 3. System displays the locale settings form. 4. Actor enters the locale settings with the following fields:  * Selects the profile. * Selects the locale Identifier. * Selects the date format from drop down list. * Selects the currency format from the drop down list.  1. Actor clicks save button. 2. System validates the locale form fields. 3. System saves the locale settings. |
| Alternate Flow: |  |
| Exceptions: | * In case of step 6 of normal flow, if locale form fields are invalid then the system will generate an error message. |
| Frequency of Use: |  |
| Business Rules: | 1. The locale-specific profiles settings defined are saved in the system. 2. The locale name is a unique Identifier comprising of language ID and region ID. |
| Assumptions: |  |
| Notes & Issues: |  |

#### Edit Locale Settings

|  |  |
| --- | --- |
| Use Case ID | Edit Locale Settings |
| Summary: | This use case allows an actor to modify an existing Locale-specific profile in the system. The edited locale settings  are reflected back into all the locale profile usages |
| Pre-condition: | 1. The user must be logged in. 2. At least a locale-specific profile must exist in the system. |
| Post-condition: | 1. System saves the edited locale-specific profile settings in the database. 2. System displays success message. 3. The locale settings changes are reflected into the application. 4. Is visible to all account users. |
| Includes: |  |
| Actor(s): | ContactProfileAdmin, ContactProfileMaster. |
| Trigger: | Actor clicks Edit locale configuration tab. |
| Normal Flow: | 1. Actor clicks edit locale settings tab. 2. System displays list of locale profiles. 3. Actor selects a locale profile. 4. System displays locale settings of the selected locale profile in edit mode. 5. Actor edits/updates the locale settings fields. The following fields can be updated:   · The Date format.  · The currency format.   1. Actor clicks Save button. 2. System validates all the fields. 3. System prompts confirmation message. 4. Actor selects "Yes" for confirmation. 5. System saves the updated locale settings. |
| Alternate Flow: | 1. In case of step 7 of normal flow, if updated locale settings fields are invalid.  * System will display an error message. * System does not update the locale settings.  1. In case of step 9 of normal flow, if user chooses not to update the locale profile.  * System does not update the locale profile settings in the database. * The use case is terminated. |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: | 1. Default locale settings will be associated to a profile if no settings are defined for it. 2. All updated locale profile settings will be reflected back into the application. |
| Assumptions: |  |
| Notes & Issues: |  |

#### Delete Locale Settings

|  |  |
| --- | --- |
| Use Case ID | Delete Locale Settings |
| Summary: | This use case allows an actor to delete an existing Locale-specific profile in the system. Once deleted, the locale profile can no longer be used in a message.  If the locale profile continues to be referenced anywhere, it will be overridden by the default locale profile. |
| Pre-condition: | 1. The user must be logged in. 2. At least one locale-specific profile must exist. |
| Post-condition: | **On success:**   1. System deletes locale-specific profile from the database. 2. System displays success message. 3. The locale no more exists in the application.   **On failure:**   1. The system displays an error message. 2. System does not perform any delete operation. 3. The locale exists in the application. |
| Includes: |  |
| Actor(s): | ContactProfileAdmin, ContactProfileMaster |
| Trigger: | Actor clicks Delete Locale Configurations tab. |
| Normal Flow: | 1. Actor clicks Delete Locale configuration tab.  2. System displays list of existing locale specific profiles for the user.  3. Actor selects a locale-specific profile.  3. Actor clicks the delete button.  4. System prompts a confirmation message.  5. Actor confirms deletion.  6. System deletes the locale-specific profile from the database.  7. All references to it are overridden by the default locale profile. |
| Alternate Flow: | 1. In case of step 8 of normal flow, if actor chooses not to delete the system will display an message and the locale profile will not be deleted. |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: |  |
| Assumptions: |  |
| Notes & Issues: |  |

### Setting Profile Parameters

This involves setting profile specific “From” and “reply to” fields for all message instantiation requests. Based on these profile parameters all message requests will be managed within VerioContact.

#### Specify new profile Settings

|  |  |
| --- | --- |
| Use Case ID | Specify new Profile Settings |
| Summary: | This use case allows the actor to create new profile settings in the system.  Profile settings includes “from” field setting and “reply to” settings specific to profiles. These profile settings will be applied during message instantiation for sending message requests. |
| Pre-condition: | The user must be logged in. |
| Post-condition: | 1. System saves the profiles level settings. 2. System displays success message. |
| Includes: |  |
| Actor(s): | ContactProfileAdmin, ContactProfileMaster. |
| Trigger: | Actor clicks the Profile configuration tab. |
| Normal Flow: | 1. Actor clicks Profile configuration tab. 2. Actor selects new Profile Configuration. 3. System displays the profile settings form. 4. Actor enters the following profile settings fields:  * Selects the profile from dropdown. * Selects the locale from dropdown * Enters the “from” field value. * Enters the “Reply to” field value as Email ID’s.  1. Actor clicks save button. 2. System validates the form fields. 3. System saves the profile settings. |
| Alternate Flow: |  |
| Exceptions: | * In case of step 6 of normal flow, if form fields are invalid then the system will generate an error message. |
| Frequency of Use: |  |
| Business Rules: | * The settings defined are saved in the system at profile level. |
| Assumptions: | * ContactProfileAdmin can specify profile settings for profiles of his account. * ContactProfileMaster can specify profile settings for all profiles within his account hierarchy. |
| Notes & Issues: |  |

## VerioContact GUI

### Configuration Management

* Configuration Management involves configuring application parameters:
* Email gateway settings
* SMS gateway settings
* Transmitter Status
* Dispatch Rate.
* It includes editing/updating, deleting all application gateway settings and parameters.
* It covers all sets of operations related to application level settings and configurations required for the VerioContact application to work.



#### Configure Dispatch Rate

|  |  |
| --- | --- |
| Use Case ID | Configure Dispatch Rate |
| **Summary:** | This use case allows the actor to configure dispatch rates for the outbound messages send per hour. |
| **Pre-condition:** | 1. The user must be logged in. |
| **Post-condition:** | 1. System saves the dispatch rate configuration for all servers. 2. System displays success message. |
| **Includes:** |  |
| **Actor(s):** | ContactSystemConfigMaster |
| **Trigger:** |  |
| **Normal Flow:** | 1. Actor clicks Configure Application tab. 2. System displays list of application parameters. 3. Actor selects dispatch rate parameter. 4. System displays the dispatch rate configuration form. 5. Actor enters the value for Dispatch rate of outbound messages per hour. 6. Actor clicks save button. 7. System saves the dispatch rate parameter in the database. |
| **Alternate Flow:** |  |
| **Exceptions:** |  |
| **Frequency of Use:** |  |
| **Business Rules:** | 1. Dispatch rate is defined at the application level. 2. All outbound messages are sent using the dispatch rate configuration settings. |
| **Assumptions:** |  |
| **Notes & Issues:** |  |

#### Configure Transmitter Status

|  |  |
| --- | --- |
| Use Case ID | Configure Transmitter Status |
| Summary: | This use case allows the actor to configure Transmitter Status for the Email/SMS gateways |
| Pre-condition: | 1. The user must be logged in. 2. Email/SMS gateways specific to profiles must exist. |
| Post-condition: | 1. System saves the transmitter status against the gateway settings. 2. System displays Transmitter status On/off message. |
| Includes: |  |
| Actor(s): | ContactSystemConfigMaster. |
| Trigger: |  |
| Normal Flow: | 1. Actor clicks the Configure Application tab. 2. Actor selects Configure Transmitter Status option. 3. System displays Transmitter Status form. 4. Actor selects the transmitter status value(on/off) for the gateway. 5. Actor clicks submit button. 6. System prompts confirmation message for the transmitter status value. 7. Actor confirms the Transmitter status(on/off). 8. System saves the Transmitter status. |
| Alternate Flow: | In case of step 10 of normal flow, if actor chooses not to set the Transmitter status, then the system will set the transmitter status "off”. |
| Exceptions: | If the actor sets the transmitter status to "off" ,then all the message requests in the request queue will be canceled out. |
| Frequency of Use: |  |
| Business Rules: | 1. Proper warning messages will be displayed for transmitter status setting operations. 2. Transmitter Status value is set at the application/system level. |
| Assumptions: |  |
| Notes & Issues: |  |

#### Configure gateway settings

|  |  |
| --- | --- |
| Use Case ID | Configure gateway settings |
| Summary: | This use case allows the actor to configure SMS, Email gateway settings at profile level. |
| Pre-condition: | 1. The user must be logged in. 2. Profiles must exist. |
| Post-condition: | 1. System saves the gateway configuration details with the associated profile in the database. 2. System displays a success message. |
| Includes: |  |
| Actor(s): | ContactConfigMaster/ContactConfigAdmin |
| Trigger: |  |
| Normal Flow: | 1. Actor clicks Configure Application tab. 2. System displays Configure Email and Configure SMS gateway options. 3. Actor selects one of the options. 4. System displays default gateway and customize gateway options. 5. Actor selects customize gateway option. 6. System displays the configure gateway form. 7. Actor enters the following gateway settings:  * Actor selects the profile from the list. * Actor enters the server name. * Actor enters Server IP. * Actor enters authentication parameters username, password. * Actor enters the port number to be used.  1. Actor clicks save button. 2. System prompts confirmation message. 3. Actor confirms the configuration of the gateway settings. 4. System will save the gateway settings against the selected profile in the database. |
| Alternate Flow: | * In case of step 3 of normal flow, if the actor chooses default gateway settings for the profile then the system will save the default gateway for the profile. |
| Exceptions: | * In case of step 4 of normal flow, if the selected profile has predefined Email/SMS gateway settings then, * System displays a message "Email/SMS gateway configured". |
| Frequency of Use: |  |
| Business Rules: | 1. Each profile can have a Email/SMS gateway configuration. 2. System will maintain a default configuration setting for the profiles having no Email/SMS gateway configured. 3. ContactSystemConfigMaster will specify the default gateway settings for Email/SMS associated with the default profile (white labeled) and default locale (en\_USA). |
| Assumptions: |  |
| Notes & Issues: |  |

#### Edit Configuration

|  |  |
| --- | --- |
| Use Case ID | Edit Configuration |
| Summary: | This use case allows the actor to update the application parameters including Email/SMS gateway settings, dispatch Rate, Transmitter Status. |
| Pre-condition: | 1. The user must be logged in . 2. Application Parameters must exist. |
| Post-condition: | 1. System updates the application parameters. 2. System displays success message. |
| Includes: |  |
| Actor(s): | ContactConfigAdmin, ContactConfigMaster. |
| Trigger: | Actor clicks Edit Configuration link. |
| Normal Flow: | 1. Actor clicks Edit Configuration link. 2. System displays the list of application parameters. 3. Actor selects one of the application parameter options:   · Email gateway settings.  · SMS gateway settings.  · Dispatch Rate.  · Transmitter Status.   1. System displays the selected Application parameter configuration form in edit mode. 2. Actor updates/edits the application parameter settings. 3. Actor clicks save button. 4. System prompts confirmation message. 5. Actor confirms the edit operation. 6. System updates the application parameter in the database. |
| Alternate Flow: | 1. In case of step 3 ,if the actor ContactConfigMaster selects Email/SMS gateway application parameter, then  * System displays the gateway settings form of the logged in user account in edit mode. * System displays the list of Email/SMS gateways configured in his/her account hierarchy.  1. Return to step 5 of normal flow. |
| Exceptions: | * In case of step 3 of normal flow, if the actor chooses to edit the Email/SMS gateway settings parameter and no   gateway has been configured for the logged in user, then  System displays a message "Default gateway settings exist". |
| Frequency of Use: |  |
| Business Rules: | All application parameters can be edited/updated specific to the user. |
| Assumptions: |  |
| Notes & Issues: | Default gateway settings editable? |

#### Delete Configuration

|  |  |
| --- | --- |
| Use Case ID | Delete Configuration |
| Summary: | This use case allows the actor to delete configuration settings defined specific to profiles. |
| Pre-condition: | 1. The user must be logged in. 2. Gateway Configuration settings specific to user account should exist. |
| Post-condition: | 1. System deletes the gateway configuration customized by the user. 2. System displays success message. |
| Includes: |  |
| Actor(s): | ContactConfigAdmin, ContactConfigMaster. |
| Trigger: |  |
| Normal Flow: | 1. Actor clicks configuration Settings tab. 2. System displays application parameters list. 3. Actor selects Gateway configuration parameter. 4. Actor has role “ContactConfigAdmin” then, 5. System displays Gateway configuration form with customized gateway values defined for the user account. 6. Actor clicks delete button. 7. System prompts confirmation message. 8. Actor confirms deletion. 9. System deletes the customized gateway configuration settings. |
| Alternate Flow: | In case of step 4 of normal flow, if actor has role “ContactConfigMaster” then,   1. System will display list of all gateways configured within the users account hierarchy. 2. Actor selects a gateway configuration setting. 3. System displays Gateway configuration form with populated values of the selected gateway configuration. 4. Return to step 6. |
| Exceptions: | * In case if no gateway settings have been customized for the logged in user account then,   System will display a message” No gateway configured”. |
| Frequency of Use: |  |
| Business Rules: | * Default gateway settings will be associated for the user account on successful deletion of the customized gateway settings. |
| Assumptions: |  |
| Notes & Issues: |  |

### Test Message Instantiation

This package allows the user to send test Email/SMS messages by the application.

User can test the sending of Email/SMS messages via the profile specific gateway setting by specifying the necessary message parameters.



#### Send Test Email message

|  |  |
| --- | --- |
| Use Case ID | Send Test Email Message |
| Summary: | This use case allows the actor to send a Test Email message via the application to a specified list of recipients. The actor will specify the required message parameters for sending the test Email message. |
| Pre-condition: | 1. The user must be logged in.  2. Profile specific Email Templates must exist for the user.  3. Profile specific Branding Themes must exist. |
| Post-condition: | Post conditions:  **On success:**   1. System sends the Test message. 2. System displays a success message. 3. Test message is visible to the user. 4. System maintains Test message instantiation results within the application logs with the following details:   · Name of consumer application  · Success status  · Date & time stamp of message  · Recipients List  **On failure:**   1. System will not send the test Message. 2. An error will be returned to the application. 3. System displays the error message. 4. System will maintain the error code logs within the application with the following Details:   · Name of consumer application  · Fail status  · Date & time stamp of message  · Error  · Recipients List |
| Includes: |  |
| Actor(s): | All |
| Trigger: | Actor clicks the Send Test Message link. |
| Normal Flow: | 1. Actor clicks the Test message link. 2. Locates the Email test message tab. 3. System displays Message Instantiation form. 4. Actor enters the following form details:   · Actor selects the profile.  · Actor selects the locale.  · Actor selects the mail message format/type.  · Actor selects the Template from the specified list.  · Actor enters the key content information and meta data information.  · Actor specifies the list of recipients as Email ID’s.  · Actor enters the token values for the selected template.   1. Actor can preview the message Template with the captured form field’s values. 2. Actor clicks the Send Test Message button. 3. System prompts confirmation message. 4. Actor confirms the sending. 5. Test message is sent by the application. 6. System creates a log for the sent Test message with success status. |
| Alternate Flow: | * In case of step 8 of normal flow, if actor chooses not to send the test message, then the system will not trigger the test message instantiation. |
| Exceptions: | 1. In case of step 9 of normal flow, if no Email Gateway is configured for the selected profile, then  * System will return an error code to the application. * System will not send the Test message. * System will maintain a log entry for the error code in the application.  1. In case of step 4 of normal flow, if the transmitter status of the selected profile gateway is "off", then  * System will not send the test message. * System displays error message "Transmitter Status Off". * The use case is terminated. |
| Frequency of Use: |  |
| Business Rules: | 1. Override parameters are applicable to the message template & branding theme at the time of test message instantiation, if not specified then the default parameters will be used. 2. Look and feel settings for the message will be specified by the branding theme applied and the selected message template. 3. Error handling behaviors will be maintained through logs within the application. 4. All Email message instantiation operations will use the gateway settings specific to the profile. |
| Assumptions: |  |
| Notes & Issues: |  |

#### Send Test SMS message

|  |  |
| --- | --- |
| Use Case ID | Send Test SMS message |
| Summary: | This use case allows the actor to send a Test SMS message via the application to a specified list of recipients. The actor will specify the required message parameters for sending the test Email message. |
| Pre-condition: | 1. The user must be logged in. 2. SMS templates specific to user profile must exist. |
| Post-condition: | Post conditions:  **On success:**   1. System sends the Test message. 2. System displays a success message. 3. Test message is visible to the user. 4. System maintains Test message instantiation results within the application logs with the following details:  * Name of consumer application * Success status * Date & time stamp of message * Recipients List   **On failure:**   1. System will not send the test Message. 2. An error will be returned to the application. 3. System displays the error message. 4. System will maintain the error code logs within the application with the following Details:    * Name of consumer application    * Fail status    * Date & time stamp of message    * Error    * Recipients List |
| Includes: |  |
| Actor(s): | All |
| Trigger: | Actor clicks send Test message link. |
| Normal Flow: | 1. Actor clicks the Test message link. 2. Locates the SMS test message tab. 3. System displays Message Instantiation form. 4. Actor enters the following form details:  * Actor selects the profile. * Actor selects the locale. * Actor selects the mail message format/type. * Actor selects the Template from the specified list. * Actor enters the key content information and meta data information. * Actor specifies the list of recipients as Email ID’s. * Actor enters the token values for the selected template.  1. Actor can preview the message Template with the captured form field’s values. 2. Actor clicks the Send Test Message button. 3. System prompts confirmation message. 4. Actor confirms the sending. 5. Test message is sent by the application. 6. System creates a log for the sent Test SMS message with success status. |
| Alternate Flow: | * In case of step 8 of normal flow, if actor chooses not to send the test message, then the system will not trigger the test message instantiation |
| Exceptions: | 1. In case of step 9 of normal flow, if no SMS Gateway is configured for the selected profile, then  * System will return an error code to the application. * System will not send the Test message. * System will maintain a log entry for the error code in the application.  1. In case of step 4 of normal flow, if the transmitter status of the selected profile gateway is "off", then  * System will not send the test message. * System displays error message "Transmitter Status Off". * The use case is terminated. |
| Frequency of Use: |  |
| Business Rules: | * The SMS gateway settings configured at profile level will be used for sending the test messages. * Default parameters for message instantiation will be used if not specified. * Token values of the message templates can be overridden at the time of message instantiation. * Profile specific SMS templates will be used for test message instantiation. |
| Assumptions: |  |
| Notes & Issues: |  |

### Audit Logs

This package will be used in VerioContact for maintaining Technical audit logs in the database to keep tracks of various changes. There will be separate audit logs tables maintained with respect to Verio Contact and framework tables to record any change [insertion/edition/deletion] taking place. DB triggers will be used to populate the audit logs tables. The tables will contain the following details:

* Date Timestamp
* User triggering the change
* Action

### Report Management

This package will be handled by WSO2 BAM server for generating reports specific to Email/SMS message communications. There will be no separate UI to display and generate the reports for VerioContact outbound communications. VerioContact will integrate with the interface provided by BAM setup for report Management. It involves generating reports for VerioContact based on the input provided to the BAM interface.

[BAM Setup and Reporting is out of scope of Verio Contact].

## VerioContact Message Instantiation Service

* VerioContact Message instantiation service will provide Restful web service interface to consumer application to request a communication to be sent out. Consumer applications will invoke the appropriate web service, providing all the details needed for it. The service will queue up the request in a JMS queue to be processed asynchronously, and return transmission id to consumer application to use in checking status check later on.
* A restful status service will be provided where consumer application can check the status of the requested communication providing transmission id received as response to communication request.

### Sending Message Instantiation requests

|  |  |
| --- | --- |
| Use Case ID | Sending Message Instantiation requests |
| Summary: | This use case allows the actor to send message requests via the consumer application. |
| Pre-condition: | * The user must be logged in. * All request parameters must be specified. |
| Post-condition: | * Message request will be queued up in the request queue. * Message transmission id will be returned to the consumer application. |
| Includes: |  |
| Actor(s): | ContactMessageRequestor. |
| Trigger: |  |
| Normal Flow: | 1. Actor specifies message request parameters:  * Profile Name * Template Name * Token values * Message content data * Recipients list as Email ID’s. * Mail Message type (Email, SMS).  1. Actor sends the message request via Consumer Application. 2. The request is queued in the request queue with the specified parameters. 3. The message transmission ID is returned to the Consumer Application. |
| Alternate Flow: | None. |
| Exceptions: | If the request parameters are not valid or request is not sent, then   * The request will be passed onto the error router for logging it on the error queue. * The error status will be visible via status service. |
| Frequency of Use: |  |
| Business Rules: |  |
| Assumptions: |  |
| Notes & Issues: |  |

### Requesting Status Service for requests

|  |  |
| --- | --- |
| Use Case ID | Requesting Status Service |
| Summary: | This use case allows the actor to retrieve the Status of the requested message. |
| Pre-condition: | * Transmission ID must be given. * The user must be logged in. |
| Post-condition: | System displays status service for the requested transmission ID. |
| Includes: |  |
| Actor(s): | ContactMessageRequestor. |
| Trigger: |  |
| Normal Flow: | 1. Actor invokes Status service via consumer application. 2. Actor enters transmission ID. 3. System displays status of the requested transmission ID as:  * In Progress * Sending Successful. * Pending. * Failed |
| Alternate Flow: |  |
| Exceptions: |  |
| Frequency of Use: |  |
| Business Rules: |  |
| Assumptions: |  |
| Notes & Issues: |  |