



VOSS-4-UC BulkLoader Provisioning Guide

Release 19.3.4

Mar 30, 2021

Legal Information

Please take careful note of the following legal notices:

- Copyright © 2021 VisionOSS Limited.
All rights reserved.
- VOSS, VisionOSS and VOSS-4-UC are trademarks of VisionOSS Limited.
- No part of this document may be reproduced or transmitted in any form without the prior written permission of VOSS.
- VOSS does not guarantee that this document is technically correct, complete, or that the product is free from minor flaws. VOSS endeavors to ensure that the information contained in this document is correct, whilst every effort is made to ensure the accuracy of such information, VOSS accepts no liability for any loss (however caused) sustained as a result of any error or omission in the same.
- This document is used entirely at the users own risk. VOSS cannot be held responsible or liable for any damage to property, loss of income, and or business disruption arising from the use of this document.
- The product capabilities described in this document and the actual capabilities of the product provided by VOSS are subject to change without notice.
- VOSS reserves the right to publish corrections to this document whenever VOSS deems it necessary.
- The terms Cisco, Movius, MeetingPlace, Netwise and all other vendor/product names mentioned in this document are registered trademarks and belong to their respective owners. VOSS does not own, nor is related to, these products and vendors. These terms have been included to showcase the potential of the VOSS solution and to simplify the deployment of these products with VOSS should you select to utilize them.

Security Information

This product may contain cryptographic features that may be subject to state and local country laws that govern the import, export, transfer and use of such features. The provision of this software does not imply that third-party authorization to import, export, distribute or use encryption in your particular region has been obtained. By using this product, you agree to comply with all applicable laws and regulations within your region of operation. If you require further assistance, please contact your dedicated VOSS support person.

Contents

- 1 What's New** **1**
 - 1.1 BulkLoader Provisioning Guide: Release 19.3.4 1
 - 1.2 BulkLoader Provisioning Guide: Release 19.3.4-PB1 1
 - 1.3 BulkLoader Provisioning Guide: Release 19.3.4-PB2 1

- 2 Getting Started** **2**
 - 2.1 Overview 2
 - 2.2 Bulk Loading Tools 2
 - 2.3 VOSS-4-UC Administration Tools Menu 2
 - 2.4 Bulk Load Templates 3
 - 2.5 Bulk Loader Template Reference Set 3

- 3 Administration Tools > Bulk Loader Menu** **4**
 - 3.1 Bulk Load Overview 4
 - 3.2 Data Export 4
 - 3.3 Bulk Load Template Export 5
 - 3.4 Bulk Load Sheets 5
 - 3.5 Bulk Load Template Sheet Layout 6
 - 3.6 Bulk Load Limitations 10
 - 3.7 Sample Bulk Loaders 11
 - 3.8 Export Data Sheet Layout 11
 - 3.9 Template Location and Template Reference Set 12
 - 3.10 Bulk Loading a File 12
 - 3.11 Bulk Load Sheet Macro Evaluation 13
 - 3.12 Carry out a Bulk Load Template Export 14
 - 3.13 Bulk Load Transactions 14
 - 3.14 Bulk Export of Model Data 15

- 4 APIs** **17**
 - 4.1 Bulk Load API 17

1 What's New

1.1. BulkLoader Provisioning Guide: Release 19.3.4

- N/A

1.2. BulkLoader Provisioning Guide: Release 19.3.4-PB1

- N/A

1.3. BulkLoader Provisioning Guide: Release 19.3.4-PB2

- EKB-7466: Bulkloading macros for boolean fields does not evaluate macro. See: *Bulk Load Sheet Macro Evaluation*

2 Getting Started

2.1. Overview

You can use any of the following tools to bulk provision VOSS-4-UC:

- VOSS-4-UC Administration Tools > Bulk Load menu.
- VOSS-4-UC REST API.

The VOSS-4-UC Administration Tools > Bulk Load menu option is available to bulk load information into VOSS-4-UC.

The VOSS-4-UC REST API can be used to feed bulk information one record at a time. This is an effective means to bulk load information from an integrated OSS/BSS system.

For more information about the VOSS-4-UC API, refer to the VOSS-4-UC API Guide.

2.2. Bulk Loading Tools

- VOSS-4-UC Administration Tools Menu
- Bulk Load Templates
- Bulk Loader Template Reference Set

2.3. VOSS-4-UC Administration Tools Menu

In the Administration Tools > Bulk Load menu item, available in the VOSS-4-UC GUI for various hierarchies, the administrator can upload Excel files (Bulk Load Templates) and apply them to the system for bulk provisioning. Partners who choose to maintain their existing operations structure can do so using the VOSS-4-UC Administration Tool Bulk Loader.

Note: VOSS-4-UC bulk loaders are not equivalent to CUCDM 8.x bulk loaders. VOSS-4-UC uses different fields and workflows to accomplish provisioning.

2.4. Bulk Load Templates

A Bulk Load Template is a single Excel file containing a single spreadsheet. Exported Bulk Load templates contain a single or multiple Provisioning/Data/Relation models like Sites, Devices or Customers. These are a part of the product and can be generated on demand for the corresponding models within the GUI.

Note: Field data is not exported when exporting the bulk loader for a template, and only the field headings (schema) are imported.

2.5. Bulk Loader Template Reference Set

A Bulk Loader Template Reference Set is a collection of multiple Bulk Load Template sheets for different models in a single Excel spreadsheet. The Bulk Loader Template Reference Set configures a basic system configuration, including NDL, Customer, Site, and dial plan elements. They provide typical loader templates, but partners may add or delete templates as necessary based on their needs. The Bulk Loader Template Reference Set provides examples of how to populate the various templates.

3 Administration Tools > Bulk Loader Menu

3.1. Bulk Load Overview

The bulk loader tools enable the quick and easy management of system data using pre-populated MS Excel formatted spreadsheets.

A spreadsheet template can be generated by the system for any of the resources in the system - either from the GUI or the API.

The data on the sheet includes column headers to indicate the hierarchy, action, search criteria and attribute names of the model to which the data applies. Rows include the data for model individual instances.

Note: To carry out a bulk load, the selected model should allow add operations in the Access Profile for the user.

Use a single sheet in the file to manage multiple templates by adding additional header rows and data under them. A file can include multiple sheets with a single or multiple templates on each.

When the file is loaded, it can either be processed immediately or scheduled for a date and time. A scheduled bulk load file is listed on the Schedule list view as a Single Execution schedule type and with resource type of data/BulkLoad. Items on the Schedule list are deleted once the scheduled item has been executed. This means that after a scheduled bulk load has been executed, you will no longer see it in the list of schedules.

A single parent transaction is created for the entire bulk load. Unless a sheet is set to execute rows in parallel, each row in the bulk load sheet results in a separate sub-transaction that is executed sequentially and synchronously. If a single sub-transaction fails, the bulk load transaction continues and does not roll back the preceding sub-transactions. In the case where a bulk load sub-transaction has other sub-transactions - for example a provisioning workflow with multiple steps - failure in any of the steps will cause a roll back of all the steps in the bulk load sub-transaction.

If a sheet is set to process rows in parallel, then by default, 14 rows are processed in parallel. Refer to the topic on the bulk load sheet layout for more details.

If a file is processed and further files are loaded, they are processed in parallel. Thus, bulk load transactions are executed in parallel, as with all transactions. Bulk load transactions are executed immediately.

Transactions, once started, cannot be canceled.

3.2. Data Export

Data can be exported in JSON format and as MS Excel spreadsheets.

The system JSON file format is used to Export and Import various operations on model instances. The operations available via JSON files are: Add, Modify, Delete. This Import and Export task is carried out from the GUI or API using the file Export and Import functionality.

The JSON file format for the different operations is available from the **Action** button on the GUI form of the specific model and choosing JSON as Export format from the drop-down list on the Export menu. The API provides a request URL and parameter for this task - refer to the API documentation. The export file format is a compressed JSON file. The import filename and format can be <filename>.JSON, <filename>.JSON.zip or <filename>.JSON.gz.

The Excel file format for data export of selected items can be carried out in the list or instance view of a model.

The commands are available from the **Action** button on the GUI form by choosing either Excel or Excel (formatted) from the drop-down list on the Export menu. The API provides a request URL and parameter for this task - refer to the API documentation. The export file format is a MS Excel XLSX file.

The Field Display Policy that applies to a menu item from which an Excel (formatted) export of data is carried out, is reflected in the Excel (formatted) exported sheet as follows:

- Titles of attributes
- Sequence of the attributes
- Group names
- Hidden fields, with the exception of mandatory fields.

3.3. Bulk Load Template Export

You can use the MS Excel format spreadsheet bulk load template of a model to easily create a template of a sheet from the user interface. See [Bulk Load Template Sheet Layout](#).

The VOSS-4-UC multi-domain core supports the ability to generate a MS Excel format spreadsheet bulk load template for any of the resources in the system directly from the user interface.

You can populate the template sheet with data and then load it using the Bulk Load administration tool.

Excel Bulk Load operations using spreadsheets support multiple (tabbed) worksheets that are loaded in tab sequence. Defined Configuration Templates on the system can be referenced in the sheets and applied during the Bulk Load operation.

The field specific help of the product can be used to assist the user with populating the bulk loaders with the correct data. See [Bulk Load Template Sheet Layout](#).

3.4. Bulk Load Sheets

An exported bulk load template is a workbook containing a single sheet and serves as the basis for bulk loading. A workbook can also be created that contains more than one sheet as a tabbed workbook.

For tabbed workbooks, bulk load transactions are carried out from the leftmost sheet or tab to the rightmost. For example, if a site is to be added under a customer, the customer sheet tab should be to the left of the associated site.

The spreadsheet workbook is in Microsoft Excel .xlsx format. Any name can be provided for the workbook and the same filename can be loaded multiple times, although the best practice is to use different names.

To bulk load data, preliminary steps need to be carried out. Verify existing information on the sheet and determine required information in order to complete the required data and prepare the spreadsheet.

3.5. Bulk Load Template Sheet Layout

This topic describes a typical generated sheet when using the Export Bulk Load Template menu option.

Colors and styles are applied to the exported sheet:

- dark colors style for header rows
- yellow text for base group titles
- mandatory fields have red title text headers
- optional fields text headers are in white

Although an attribute that has nested attributes may be optional, if this attribute has mandatory nested attributes, then the containing attribute becomes mandatory. If a field is mandatory, it is shown on the sheet regardless of any Field Display Policy instruction to hide it.

The Field Display Policy that applies to a menu item from which a Bulk Load Template Export is carried out, is applied to the exported sheet as follows:

- Titles of attributes
- Sequence of the attributes
- Group names
- Hidden fields, with the exception of mandatory fields.

Note:

- Macros can be included in the loader to either be loaded as text or evaluated as part of the load. See documentation in this guide around *evaluate_macros* header for more details on macro behavior in the loaders.
- A single sheet of a file can be used to manage multiple templates by adding additional header rows and data under them. A workbook file can include multiple sheets with single or multiple templates on each.

entity: relation/HuntGroupRelation; hierarchy: sys; parallel: False; parallel_transaction_limit: ; template: ; meta_pref															
\$hierarchy	\$action	\$search_fields	\$device	\$template	\$ndl	\$pkid	pattern	name							
# Base							# Hunt Pilot Pattern								
Comment	Hierarchy	Nod	Action	Search	Field	Device	CT	Templa	Network	Device	List	Unique	Identifie	# Hunt Pilot Pattern	# Name

Refer to the example sheet snippet. A bulk load sheet contains the following information:

3.5.1. Sheet name (tab on spreadsheet workbook)

Any name can be provided on the tab or sheet. If the name is prefixed with a # on the tab, the sheet is ignored during loading.

3.5.2. Row 1 - Resource and instructions

The exported bulk loader template will have the resource as target entity (model) as well as the hierarchy shown on the top row of the sheet. Verify the entity in the first row of an exported sheet. The reference data in the first row is of the format shown below, with variable values indicated in {}:

```
entity: {entity name}; \
hierarchy: {hierarchy}; \
parallel: {True | False}; \
parallel_transaction_limit: {n}; \
template: {config_template}; \
meta_prefix: {c}; \
evaluate_macros: {True | False}
```

- *entity*: *{entity name}*: the name of the model, in the format *{modeltype}/{model name}*, for example *data/User*
- *{hierarchy}*: the hierarchy, in the format *{level1}.{level2}.{level3}*, where *{level1}* is the first system level. Verify the hierarchy at which the bulk load should take place.
- *parallel*: True or False. By default, the value is False and rows are processed sequentially. If multiple templates are entered on a single sheet, they should all *only have a single value*: True or False.

Sheet rows can be processed in parallel. The sheet should then not contain multiple, sequence dependent models. If there are a large number of rows for complex models on the sheet, the duration of a bulk load transaction is significantly reduced by parallel processing.

By default, 14 rows are processed in parallel, since bulk loads are low priority transactions that are limited to 50% of the maximum allowed parent transactions, which is by default set to be 30 per unified node. The default value supposes that one slot is used by the parent bulk load transaction itself.

The maximum allowed parent transaction limit can be modified from the Command Line Interface (CLI) using the command: **voss workers <number>**.

- *parallel_transaction_limit*: the maximum number of rows that can be processed in parallel by the bulk load at any given time. The minimum value that can be set is 1 and the maximum is 100.
- *template*: The Configuration Template *{config_template}* that is associated with the user's menu item for the *{entity_name}* from which the export was carried out. The exported sheet will show a row of values from the Configuration Template.

When a sheet is created to bulk load, the Configuration Template should be available on the target system and it will only apply to rows on the sheet that has **add** specified in the # *Action* column.

Note: this header item is not used when Configuration Templates are loaded.

- *meta_prefix*: By default, the value is \$. The # character cannot be used, as it is used for comments. The character is prefixed to the # *Base* group of columns in Row 2.

The purpose of the prefix is to distinguish a special set of base columns from the entity attributes on bulk load sheets.

Note that the bulk load sheets will fail to load if the # character is used as prefix. An error message will be shown in the transaction log.

- *evaluate_macros*: By default, the value is False. When set to True, named macros can be added as values to be evaluated before the sheet is loaded. Otherwise, the value is a string.

The format of the macro is `{{ fn.bulkload_evaluate macro.NamedMacro }}`, where *NamedMacro* is the name of an existing *data/Macro* instance. The function prefix *fn.bulkload_evaluate* is required in the value for the macro to be evaluated.

- For examples, see [Bulk Load Sheet Macro Evaluation](#).

Note: `fn.bulkload_evaluate` is not available via the Macro Evaluator. For testing purpose using the Macro Evaluator, please use the `fn.evaluate` function prefix.

3.5.3. Base columns (grouped by # Base in Row 3)

The list below describes the column values with the default value of `meta_prefix`, in other words, column names by default prefixed by `$`.

The purpose of the columns is to provide more detailed instructions or overriding data for a row.

- **Comments:** Any row that contains a `#` character in column A is considered a comment row and will be ignored. Empty rows are also ignored. Column A - the first column - is also a `# Comment` column, so that any value entered in it is considered a comment. If all rows on a tab are commented, but the tab name itself is not commented, the tab sheet load will fail.
- **\$hierarchy:** A hierarchy column with the name `# Hierarchy Node` is also available so that individual rows of a sheet can be loaded to a specified hierarchy. If a hierarchy is specified in this column for the row, it takes precedence over the hierarchy in the first row. The format for the hierarchy in the row is the same as for the first row: the full hierarchy, with levels separated by dots.
- **\$action:** Any row that contains an action in the `# Action` column : **add**, **delete**, **modify**, **execute** or a custom action name, will have the action carried out. The action values in the column are case insensitive.

If no action is entered, the **add** action is carried out. The list below shows the functionality for the values entered in the row. Also refer to the Search Fields entry below.

- **add** or empty - the data in the attribute columns is added. Any values in the `# Search Fields` column are ignored.
- **delete** - the row matching the unique criteria in the `# Search Fields` column is deleted.
- **modify** - the row matching the unique criteria in the `# Search Fields` column is updated with values in the attribute columns. Refer to the Search Fields entry below.
- **execute** - if the action is available for the model, the row matching the unique criteria in the `# Search Fields` column is executed, using any values entered in attribute columns.
- custom action name - if the custom action is defined for the model, it is carried out for the row matching the unique criteria in the `# Search Fields` column.
- **\$search_fields:** The column applies to rows where the action is not **add** and consists of a colon-separated list of attribute names and values, for example, `fullname:'John Smith', username:jsmith`.

Note however that the `pkid` field takes precedence over search fields criteria.

- **delete** - the search fields and corresponding attribute values uniquely identify the model instance to delete.
- **modify** - the search fields and corresponding attribute values uniquely identify the model instance to modify, with the values to modify in the attribute columns.
- **execute** - the search fields and corresponding attribute values uniquely identify the model instance to execute.

Where the sheet is for a Relation model, only the left model attributes in the Relation can be in the Search Fields column. This is the standard search behavior for Relations.

- `$device`: The column is used when a sheet includes attribute columns that belong to a device model. This column then contains the comma-separated list of business keys of the device model, as well as its hierarchy. These values narrow the search for the device to which the data in the sheet applies. Examples of such sheets would contain device models or relations that have device model attributes in the left hand association of the relation.

The format of the values in this column is:

```
<business_key1>,<business_key2>,...,<business_keyn>,<device_hierarchy>.
```

For example, if a Unified CM instance in a model `data/CallManager` has `host` and `port` as business keys, the value would for example be: `10.120.2.175,8443,sys.Varidion.InGen.Tokyo`.

- `$template` (Configuration Template): If a row that contains a Configuration Template name that applies to the model, this template is applied to the row when it is loaded. Upon bulk loading, values in this column will override any value for `template` in the sheet header.
- `$ndl` (Network Device List): The column is used when a sheet includes attribute columns that belong to a device model. This column then contains the name of the Network Device List that includes the required device in the list of devices. If the Device column is also filled in, then the value in the Network Device List column overrides it.
- `$pkid` (Unique Identifier): On modify, delete, execute, and custom action operations, this `pkid` is used to identify the resource represented in the row data. The `pkid` field takes precedence over the search fields criteria when locating a resource. The `pkid` is unique to the resource on the particular database and cannot be relied upon when attempting to manipulate an identical resource on a different database.

3.5.4. Row 2 - Column names

- base column names (prefixed by the `meta_prefix` character and listed above)
- attribute names. Entity attribute names show as column header data in the spreadsheet.

Columns can be in any order in a row. Nested object attribute names follow a dot notation.

Array objects will be sorted, so that attributes with names such as `filter_fields.<number>.xx` will be in sequence: `filter_fields.0.xx`, `filter_fields.2.xx`, and so on - before further ordering (represented by `.xx` here) is applied.

- If a column header starts with a #, the column will not be loaded.
- If a column header is blank, this indicates the end of the sheet header. Subsequent columns will not be loaded.

entity: relation/HuntGroupRelation; hierarchy: sys; parallel: False; parallel_transaction_limit: ; template: ; meta_prefix: #																			
Hierarchy	Action	Search_fields	Sdevice	Stemplate	Sndl	Spkid	pattern	name											
# Base							# Hunt Pilot Pattern												
# Comment	Hierarchy	Nod#	Action	Search	Fiel	# Device	FT	Templa	Network	Device	L#	Unique	Identifie	#	Hunt	Pilot	Pattern	#	Name
	sys.Provider1Customer52																33121		
#	svs.Provider1Customer52																33122		

3.5.5. Row 3 - Group or description

The row provides a description of a column or columns (as for example `# Base` for the sheet base columns), or else the group name of attributes that are grouped on the GUI as tabs on the detail or input GUI form.

A group is specified in the row by merging the group name across all the columns of the group. For attributes that are required and are not grouped in the GUI (or may be hidden in the GUI), the group name: Not Grouped Fields is given on the sheet.

“Default” values of attributes in this group need to be removed from an exported sheet before the sheet is used to bulk load rows.

3.5.6. Row 4 - Title

Title of:

- the reference for base column names (hierarchy, action and so on)
- the column attribute as on the GUI. This title may be modified by a Field Display Policy.

3.5.7. Data rows

The exported template contains no data.

Important: As a part of bulk loader sheet design, attention should be paid to the API payload posted to the system. The data entered in the loader sheet columns should correspond with the API payload.

GUI drop down lists may contain user-friendly titles, while the actual value sent to the API may differ.

3.6. Bulk Load Limitations

The VOSS-4-UC automation templates include a rich set of features that incorporate the use of Configuration Templates, customizable Field Display Policies and GUI rules that are used to enhance the user experience of the user interface.

Generated bulk load templates for certain resources that make use of such advanced features are limited in their ability to produce the same provisioning results that can be obtained when using the user interface and as such require additional consideration.

An overview of these limitations is provided below, whilst the implications of these limitations, how they may apply to specific resources in the system as well as how best to use the generated loaders, are documented in the Bulkload Reference Guides under resource specific sheet notes.

- Certain fields are used to link together different resources (for example, the name of the Cisco Unified CM remote destination is linked to the user ID of the user name).

These fields may not be exposed in the user interface or may, for some resources, be exposed as read-only in the user interface. Such fields are currently exposed as mandatory fields in the generated bulk load templates. The fields and the specific conventions that are used in the template to link the fields together are highlighted in notes specific to the resource. For example, the value for remote destination name should be specified as `RDP-<username>`.

- Certain fields are derived from other data in the system. The notes specific to the resource highlights where to get the possible values for such fields. Examples of this are key-value type fields of a phone's vendor configuration settings.

- GUI Rules defined in the user interface that are not replicated in the backend workflow need to be considered in the loader to get the same provisioning results as the GUI. For example, the GUI rules may:
 - Set a default value for a visible field (fixed value or derived from other data in the system). This column and corresponding value will have to be included in the loader for this to be provisioned.
 - Set a value for a hidden field - This column and corresponding value will have to be included in the loader for this to be provisioned. Note that this means that fields may be included in the loader that would not be visible in the user interface.
 - Make a field visible depending on some condition such as the value of another field (for example, a check box being selected). The column(s) need to be included in the loader and populated under the appropriate conditions.

A GUI Rule may for example disable input fields based on the state of a check box. On the sheet, the ticked check box is represented as TRUE in the column. The columns associated with the disabled fields should not be filled.

	Y	Z	AA	AB	
1					
2	forwardHuntNoAnswer_destination	forwardHuntNoAnswer_usePersonalPreferences	forwardHuntNoAnswer_callingSearchSpaceName	forwardHuntBusy_destination	forwardH
3					
4	# Destination	# Use Personal Preferences	# Calling Search Space Name	# Destination	# Use
5		TRUE			

Red boxes labeled "Disabled" are placed over columns AA and AB, with red lines pointing to the TRUE value in column Z.

To overcome the complexities introduced by the above limitations, a set of sample bulk load sheets have been generated that enable users to get started quickly and to leverage the best practices developed by VOSS.

3.7. Sample Bulk Loaders

Sample loaders enable a quick start by providing working examples of the most frequently used loaders. These can be customized according to user requirements and data.

Furthermore, sample bulk load sheets incorporate best practices for using bulk loaders; ensuring rapid customer and subscriber on-boarding.

Note that the sample loaders are built according to the default Field Display Policies and Configuration Templates that are shipped with the product. Since these are configurable, the use of non-default Field Display Policies or Configuration Templates may result in a change of the sample loaders. For example, if an additional field is exposed by the Field Display Policy, it needs to be added if it is to be managed in the loader.

The latest sample bulk loaders can be obtained from your account team.

3.8. Export Data Sheet Layout

This topic describes an exported sheet, either formatted or not formatted.

For both exported sheet formats, the header and column layout shows correspondences with the Bulk Load Template sheet.

The following items apply to sheets containing data exports:

- The `# Comment` column shows the text “Exported data” in green for each row of data.
- The `# Hierarchy Node` column shows the source hierarchy of the exported row. The `hierarchy:` value in the sheet header shows the hierarchy from which the data export was run.
- If device instances were exported, the `# Device` columns shows the business key of the device (for example, comma-separated: host, port, hierarchy).
- If a Configuration Template was applied during the export - for example if it applied to the GUI form - `# CFT Template` column will show this name for each row, as well as the `template` value in the sheet header. If a sheet is used for loading, the row value overrides the sheet header value.
- The `# Unique Identifier` contains the `pkid` that is used to identify the exported resource represented in the row data. On modify, delete, execute, and custom action operations, this `pkid` is used to identify the resource instance on the database represented in the row data.

The `pkid` field:

- takes precedence over the search fields criteria when locating a resource
 - is unique to the resource on the particular database and cannot be relied upon when attempting to manipulate an identical resource on a different database
- For a formatted Excel export, the columns in the sheet correspond with an exported Bulk Load Template sheet.
 - For a non-formatted Excel exported sheet, the columns correspond with the properties of an exported JSON file. For example, only properties where strings are not empty and boolean values are set, are exported.
 - A formatted, exported sheet of data can be used just as a Bulk Load Template sheet to bulk load data. For other Actions, the `# Search Fields` column needs to be completed. Refer to [Bulk Load Template Sheet Layout](#).
 - “Default” values of attributes in any Not Grouped Fields group need to be removed from a sheet before it is used to bulk load rows.
 - The rows and data in the columns of an exported sheet are bound by the limitations of the MS Excel format. For example, model data with property values longer than 32,767 characters (maximum length of MS Excel cell contents) will be truncated in the exported sheet.

3.9. Template Location and Template Reference Set

You can download bulk loading templates from the VOSS-4-UC GUI using the **Actions** menu on the upper right-hand side of the screen. You can select **Export Bulk Load Template** to open the file in Excel or to save the file to your desktop or removable storage device.

The latest BulkLoading Template Reference Set is available from your account team.

3.10. Bulk Loading a File

Completed Bulk Load XLS sheets can be loaded immediately. Verify the following:

- Ensure that:

- The file has a valid file extension `.xlsx`. An error message will display on the user interface to indicate that the file does not have a valid file extension.
 - Any Configuration Templates that are referenced, are available.
 - Any comments that your spreadsheet application allows you to add (for example showing as a marker in the cell with a pop-up) have been removed. Otherwise, an error message is shown: “An error occurred while opening the workbook. For possible resolution, please remove all comments from the worksheets and try again. If the problem persists, contact your administrator for support.”
- To send empty values for a files, type a space in the cell of the value column on the sheet.
 - Spreadsheet formulas (such as `'=7+2'`) in data will not be evaluated.
1. Browse to the hierarchy level at which you want to perform the bulk load.
 2. Choose **Administration Tools > Bulk Load**. The **Bulk Load**** form is displayed.
 3. Click **Browse** to open the file upload dialog.
 4. Click **Bulk Load File** on the button bar to carry out the bulk load.
 5. Choose **Administration Tools > Transaction** to inspect the bulk load in the transaction log if necessary. The **Execute Bulk Load** sub-transaction list shows the transaction for each row of the sheet.
 6. Alternatively, to schedule the bulk load, clear the **Execute Immediately** check box and add scheduled date and time values in the mandatory **Execution Date**, **Execution Time** and **Execution Timezone** fields. A scheduled bulk load is shown on the list view of the Schedule and has the name and upload load date of the sheet.

3.11. Bulk Load Sheet Macro Evaluation

Bulk load sheets can be configured to allow for macro evaluation.

The first row of a bulk load sheet has a variable to enable or disable macro evaluation (see [Bulk Load Template Sheet Layout](#)):

```
evaluate_macros: {True | False}
```

- If the variable is set to `True`, macros in a sheet will be evaluated upon loading. In this case, it is important that:
 1. The macro be prefixed with `fn.bulkload_evaluate`.
 2. A named macro must be used, in other words, a `data/Macros` instance.
 3. If the named macro that is used evaluates to a boolean or integer value, it will be evaluated and the sheet will be processed with that value.

For example:

1. If the sheet is used to update a site at its hierarchy, and the `evaluate_macros`: value is set to `True` in the first row, then:
 - The macro `{{ fn.bulkload_evaluate macro.SITENAME }}` will be evaluated to the site name when the sheet is loaded, but inserted as plain text.
 - The macro `{{ fn.evaluate macro.SITENAME }}` will be *not* evaluated to the site name when the sheet is loaded, but inserted as plain text.
 - The macro `{{ macro.SITENAME }}` will be *not* evaluated to the site name when the sheet is loaded, but inserted as plain text.

- The macro `{{ input.sitename }}` will be *not* evaluated to the site name when the sheet is loaded, but inserted as plain text.
 - Rows containing entries with a *combination* of the type `{{fn.bulkload_evaluate <named macro> }}` and other types macros will *only evaluate* the former type and load others as plain text.
2. If the sheet is used to update a site at its hierarchy, and the `evaluate_macros`: value is set to `False` in the first row, then *all* macros entered will be inserted as plain text.

Note: If the named macro needs to be tested with the macro evaluator, the format is `{{ fn.evaluate macro.SITENAME }}`.

- See the topic on macros in VOSS-4-UC documentation for named macro examples.
- For further details, also refer to the Advanced Configuration Guide and Named Macro Reference.

3.12. Carry out a Bulk Load Template Export

The button to export a Bulk Load Template of a model is available on any of the following forms: Model design form, Model instance, Model list view.

1. Choose the Hierarchy where the model is available.
2. Choose the required form and click **Action > Export Bulk Load Template** on the button bar to export the bulk load template.

A MS Excel sheet is created that contains the bulk load template for the selected model. The sheet is available in the download directory of the browser application.

Use the bulk load template sheet to enter data.

Use the Bulk Load administration tool to upload the bulk load template sheet.

3.13. Bulk Load Transactions

The transaction log is available on the user interface after a bulk load transaction has been run. Refer to the topics on transactions and viewing transactions in the Core Feature Guide.

Choose **Administration Tools > Transaction**. Bulk load transactions show in the log:

- In the list view, the bulk load is shown in the Action column of the log. If the bulk load was scheduled, this is shown as a schedule with the detail column indicating it to be a bulk load. The Action column will show “Execute Bulk Load” or “Execute Schedule” respectively.
- The submitted, start and stop time for the entire bulk load transaction is also shown.
- The Detail section will hold the name of the file that is bulk loaded as well as the workbook sheet number and the number of successful rows out of the total, for example:

```
[ 8/9 ] succeeded from [ 1 ] sheet in data_Users_bulkloadsheet.xlsx.
```

Checks are made to validate the user’s access profile, the provided hierarchy information and data constraints for the bulk load transaction when updating the target models. The parent bulk load transaction will show the error message if this validation fails and no rows will be loaded.

Where rows are loaded, each row in the bulk load sheet appears as a sub-transaction within the bulk load transaction. The Message box shows the number of successful and failed rows loaded.

For each loaded sheet, bulk load transactions are run in series for each row. Multiple bulk load sheets can be loaded and these transactions will load in parallel.

Sheet rows can be processed in parallel. The sheet should then not contain multiple, sequence dependent models. Refer to [Bulk Load Template Sheet Layout](#).

For each row of the bulk load sheet carrying out the default add action, a Create action is shown on the list of transactions. Sheet rows that led to a successful Create action have a Success status, while rows that failed show a Fail status. If a row fails, the load process continues. For failed actions, the transaction can be selected to show the error message.

If one or more rows of the sheet failed to load, the Bulk Load Sub Transaction shows a Success status, while the Log list will show "error" for failed rows.

On the list of sub transactions, choose the transaction **Link** hyper-link to inspect the details of each sub transaction. For example, the submitted, start, and stop time for the bulk load sub transaction corresponding with a row on the bulk load sheet is shown. In the case of a failed sub transaction, further information about the failure - such as the error message and row data - is shown in the sub transaction Link.

A canceled bulk load transaction means the Processing worksheet sub transaction, as well as all sub transactions within the worksheet transaction in a Processing or Queued state, will fail.

For parallel transactions, multiple resource transactions may be in a Queued or Processing state. By default, 14 rows are processed in parallel. Refer to [Bulk Load Template Sheet Layout](#) for more details. If a worksheet transaction fails as a result of bulk load transaction cancellation, subsequent worksheet tabs in the bulk load workbook will not be processed by the bulk loader.

3.14. Bulk Export of Model Data

1. Choose the hierarchy to which the model belongs.
2. Choose the items to be exported from the List view and click the **Action > Export**.
3. From the **Export format** drop-down, choose the required export format and click **OK** to export the selected items. The following file formats correspond with the selected drop-down item in the list:

- JSON - an export containing data in JSON format as in the system database. Item properties such as strings that are empty or Boolean values that are not set, are not included. The filename format is:

```
export_YYYY-MM-dd_HH_MM_SS.nnnnn.json.zip
```

- Excel - an export containing data and Excel columns for all fields as shown in the JSON export format. The filename format is:

```
<type>_<name>_exportedsheet_CCYY-MM-DD_HH-MM-SS.xlsx.
```

An example would be:

```
data_User_exportedsheet_2015_01_01_12-25-22.xlsx.
```

- Excel(formatted) - an export containing data and Excel columns as arranged by any Field Display Policies that apply. The columns correspond with those of a Bulk Load Template export sheet. This sheet can therefore be used to modify and update data if required. The filename format is:

```
<type>_<name>_exportedsheet_formatted_CCYY-MM-DD_HH-MM-SS.xlsx.
```

An example would be:

`data_User_exportedsheet_formatted_2015_01_01_12-25-22.xlsx`.

4. If required, the export JSON file can be decompressed, and the JSON file (.json) can be opened in a text editor. The XLSX file can be opened in MS Excel.

Note: The bulk export of Device Model data will export locally *cached* data, not data on the device itself.

4 APIs

4.1. Bulk Load API

Two API calls are required.

Task	Call	URL	Parameters	Response
Submit file	POST	<code>/api/ uploadfiles/</code> This URL will be moved to <code>tool/UploadFile</code> in future.	<code>hierarchy=[hierarchy]</code> <code>Content-Type: multipart/form-data</code> <code>name='uploadedfile'</code> <code>filename=<filename></code> the file to upload	<code>{"uploadedfiles": [{"id": "<file_id>",&br/>"name": "<filename>"}]}</code>

The response is HTTP 202

Task	Call	URL	Parameters	Payload
Bulk Load	POST	<code>/api/tool/ BulkLoad/</code>	<code>method= bulkload_spreadsheet</code> <code>hierarchy=[hierarchy]</code>	Examples: <code>{'bulkload_file': '<filename>',' 'execute_immediately': true}</code> or: <code>{'bulkload_file': '<filename>',' 'execute_immediately': false 'execute_date': '2013-06-20', 'execute_time': '12:00:00', 'execute_timezone': '0'}</code>

The following `curl` commands illustrate the two steps:

Step 1

```
curl -H 'Authorization: Basic <auth_key>'
-F uploadedfile=@"<file>.xlsx"
'http://<hostname>/api/uploadfiles/'
```

Step 2

```
curl -H 'Authorization: Basic <auth_key>'
-H 'Content-Type: application/json'
-H 'accept: application/json'
--data-binary '{"bulkload_file": "DEMO.xlsx", "execute_immediately": true}'
'http://<host>/api/tool/BulkLoad/?hierarchy=[hierarchy]&
method=bulkload_spreadsheet&
nowait=true&
format=json'
```

The response to this call is for example as in the following table.

Response
<pre>{ "href": "/api/tool/Transaction/0b340a6f-b658-48bb-ac8c-7562adc5572d", "success": true, "transaction_id": "0b340a6f-b658-48bb-ac8c-7562adc5572d" }</pre>

- If the Bulk Load is to be scheduled, the payload of the second task includes schedule details:
 - *execute_immediately* is set to false
 - *execute_date* is added in the format YYYY-MM-DD
 - *execute_time* is added in the format HH:MM:SS
 - *execute_timezone* is added in the format of a numeric value in minutes relative to UTC. For example, UTC is 0, UTC+2:00 is 120, UTC-1:00 is -60, and so on.
- An entry is also generated in the schedule; that is, an instance is added to the data/Schedule module.
- If the second task payload has *execute_immediately: true*, a POST is generated to `/api/data/Bulkload/`. The payload includes the uploaded filename and a generated name and time stamp as well as a description, for example:

```
{ 'filename': '<file>.xlsx', 'description': 'Generated by Bulk Loader
Administration Tools', 'name': 'AnyUser.xlsx -- 2013-05-21
16:47:11.801664 (UTC)' }
```

To inspect the detailed progress and status of the transaction, use the API call from the response above:

```
GET /api/tool/Transaction/[pkid]
```

with parameters:

- `hierarchy=[hierarchy]`
- `format=json`

The response to this GET call is a JSON object that provides details of the transaction, as for example in the truncated snippet:

```

...
  "href": "/api/tool/Transaction/[pkid]
  "log_id": "53a8053ea616540708141f44",
  "message": "data_Countries_bulkloadsheets.xlsx is a valid
  "severity": "info",
  "time": "2014-06-23T10:45:18.029000",
  "transaction_id": "[pkid]"
}
],
"pkid": "[pkid]",
"resource": {},
"rolled_back": "No",
"started_time": "2014-06-23T10:45:17.813000",
"status": "Success",
"sub_transactions": [
  {
    "action": "Execute Resource",
    "detail": "Execute : data_Countries_bulkloadsheets.xlsx -- ...
    "status": "Success",
    "submitted_time": "2014-06-23T10:45:19.567000",
    "transaction": "/api/tool/Transaction/[pkid1]    ...
  },
  {
    "action": "Create Schedule",
    "detail": "Name:data_Countries_bulkloadsheets.xlsx -- 2014- ...
    "status": "Success",
    "submitted_time": "2014-06-23T10:45:18.912000",
    "transaction": "/api/tool/Transaction/[pkid2]    ...
  },
  {
    "action": "Create Bulk Load",
    "detail": "Name:data_Countries_bulkloadsheets.xlsx -- 2014-06 ...
    "status": "Success",
    "submitted_time": "2014-06-23T10:45:18.419000",
    "transaction": "/api/tool/Transaction/[pkid3]    ...
  }
],
"submitted_time": "2014-06-23T10:45:17.794000",

```

On the GUI, the same transaction displays as in the Transaction log image.

Transaction	
Submitted Time	Jun 23, 2014 12:45:17 SAST
Started Time	Jun 23, 2014 12:45:17 SAST
Completed Time	Jun 23, 2014 12:45:19 SAST
Duration	1.805 sec

Sub Transactions				
Action	Status	Transaction	Submitted Time	
Execute Resource	Success	Link	Jun 23, 2014 12:45:19 SAST	Execute : data_C
Create Schedule	Success	Link	Jun 23, 2014 12:45:18 SAST	Name:data_Cou
Create Bulk Load	Success	Link	Jun 23, 2014 12:45:18 SAST	Name:data_Cou

1 - 3 of 3

Log		
Time	Severity	Message
Jun 23, 2014 12:45:18 SAST	info	data_Countries_bulkloadsheet.xlsx is a valid utf-8 e

For long transactions, to retrieve a summary of the status of the transaction, the transaction can be polled, using `poll` in the URL, using the same parameters:

```
GET /api/tool/Transaction/poll/?transactions=[pkid]
```

In this case, there is a shortened response, for example:

```
{ "[pkid]":
  { "status": "Processing",
    "href": "/api/tool/Transaction/0b340a6f-b658-48bb-ac8c-7562adc5572d",
    "description": null}
}
```