LOAD PROCESS GUIDE

Campaign Manager 6.0

VERSION CONTROL

Version	Date	Author	Changes
1.2	24 October 2017	D Cooper	Process Attribution Rules Example updated.
1.1	19 June 2017	D Cooper	Contact / Response History Roll-Off updated.
1.0	28 April 2017	P Trevitt	Release

RELATED DOCUMENTS

The related documents are located in the Alterian product help.

Name
Campaign Manager 6.0 Installation Guide
Campaign Manager 6.0 Upgrade Guide
Campaign Manager 6.0 Architecture Guide
Campaign Manager 6.0 Backup Guide
Campaign Manager 6.0 Data Flow and Structure
Campaign Manager 6.0 Release Note

TABLE OF CONTENTS

1. INTRODUCTION

This document provides details on the required stages to implement the data load process for a Campaign Manager 6.0 system. It describes how to create an automated load process around the Campaign Manager application that can potentially receive incoming event data and execute scheduled campaigns. It details how the system can be paused and restarted, emphasizing the order that tasks are performed.

IMPORTANT NOTE: This document describes the process for a clean Campaign Manager 6.0 system and considers the best approach for an upgrade to build a new Campaign Manager script set.

A full example script is supplied at the end of the document as a template.

If upgrading from an Alterian Marketing Suite (AMS) installation, a thorough review of load scripts is required, along with a review of user practices in the application that might interact with the load process, such as data engineering.

For example, the Scriptcreator function in iLoader was built around AMS user based engineering, and its use with a Campaign Manager system is not advised, as there are many situations where it will not be able to reverse engineer user created data due to application based storage of some of the engineering syntax. In these instances, it is more efficient to use in-line engineering in Campaign Manager.

Scriptcreator can continue to be used for capturing and recreating links. For more information on using the data engineering tool in Campaign Manager, refer to the online help.

For more information on implementing data engineering in Campaign Manager, refer to the 'CM 6.0 Best Practice Guide – Data Engineering in Campaign Manager and Engine' document.

The Campaign Manager application has processes/services that must be managed to ensure the correct data flow occurs to avoid data corruption or data access problems. This document details how to manage them in your overall load process.

For specific details on the services, refer to the Campaign Manager 6.0 Architecture Guide.

An understanding of the data structure and data flow for campaign activity is vital to see the key relevance of the Load Process stages. For specific details on this subject, refer to the Campaign Manager 6.0 Data Flow and Structure guide.

The 'Required Load Process Stages' and 'Detail of Creating a Scripted Process' sections in this document look at each load process stage in turn, initially at a high level, then in

more detail and based on the interaction between processes. There may be some repetition of content where required.

1.1. ASSUMPTIONS

- Users have consulted the Campaign Manager 6.0 Upgrade document to ensure all upgrade considerations have been assessed.
- Administrators will add further error handling, alert email management etc., to their load scripts as required.
- Parameter values, network and local paths etc. used in these examples are simplified for ease of understanding, and cannot simply be cut and pasted into existing scripts without review and edit.
- Users of this document will be familiar with using iLoader.
- This best practice document is based on an overnight load process where a data load is started once you have ensured all users are off the system, as per current Engine load strategies. This will involve users understanding when to log off the system and potentially calling a shutdown prior to commencing the process.

2. REQUIRED LOAD PROCESS STAGES - OVERVIEW

NOTE: There are defined actions that must occur to ensure a robust load process. Failure to follow these guidelines could result in corruption or loss of data.

The following is a high-level description of each task in the correct order. Details of how to carry out these stages are detailed later in this document.

1. **Pause Campaigns and wait on completion of current processing.**

   A Campaign Manager campaign constantly polls for audience changes and incoming events, as well as reacting to those events and implementing tactics. Do not start the load process while this is taking place. Campaigns should be paused to stop any new activity starting once all current processing campaign activity has stopped.

2. **Ensure all of Campaign Manager's "State" processing is complete.**

   Some enhancements made in Campaign Manager 2016 R1, specifically the introduction of Campaign History Data Table Structure /Intra-Day reporting and the removal of Event data from SQL, mean that some Engine activity (for example, Event data loads to Engine and staging table archiving) may continue even when campaigns are paused. This stage ensures that these processes exit cleanly before the load process continues.

3. **Drop User Created database Objects and Campaign History Engineering.**

   Campaign Manager users can create in-line engineering or table column engineering as part of their daily activity. These can be dropped via a Campaign Manager document containing one or more Engine Cleanup tools with the relevant selection made. Run the document using an iLoader command.

NOTE: As the next two stages relate to Campaign History, all Data Engineering created on the Campaign History tables should also be dropped at this point. Clearing these down will optimize the performance of Campaign roll-off.

4. **Contact/Response History Roll-Off.**

   At any time, administrators can select a campaign's Contact and Response history data to be removed, along with deleting its Master Campaign table record. The actual removal of data occurs at this stage in the load process.

5. **Disable Datasource.**

   Flags against each Datasource in the system control access to that Datasource. This process disables access to the Datasource by any campaign processing services or Campaign Manager users.

6. **Engine Shutdown/StartUp.**

   An optional stage if file access is required to the Engine repository for reasons such as performing a backup.

7. **Engine Data Warehouse load.**

   At this stage, the Datasource is disabled, the campaigns are paused and all existing activity has completed. The system is ready to commence the Data Warehouse load.

IMPORTANT NOTES

a. As the Campaign Manager system is now fully shut down, it is an appropriate time to perform system backups. For details on this subject, refer to the Campaign Manager 6.0 Backup Guide.

b. Engine has some significant changes that require a full overview of existing Customer Data Warehouse load scripts. This is especially true if the scripts have been in existence for some time and have not made use of recent improvements to load processes. For more information, see CM 6.0 iLoader Best Practice - Strategies for Loading Data.

c. For information on minimizing downtime and utilizing initiatives such as A/B repository switching and using staging servers, see CM 6.0 Best Practice Guide - Minimizing downtime with AB Repository switching.

8. **Enable Datasource.**

   Enable the Datasource, as the next stages need to interact with the Campaign Manager services.

9. **Response Attribution.**

   This section of the load process runs the attribution rules created in a campaign to link certain events back to the relevant Contact record on the Contact History table. This process creates a new key column on the tables being attributed to.

NOTE: If a REFRESH or an UPDATE load strategy is performed on the table being attributed, the table's primary key and the new attribution key would need to be dumped and re-imported and applied after the refresh. This process is covered in the 'Attribution' section and is very important to ensure consistency of attribution across data loads.

10. **Document Execution.**

    Process any Campaign Manager documents to execute, prior to exposing the system to the users. This might include segment documents that contain Data Engineering or documents that create Grid or Charts for Dashboards or for export to Excel.

11. **Restart Campaigns.**

    Once the Datasource is enabled and the data load has completed, campaigns can now be started. Note: this only restarts campaigns that were paused by stage one. Any manually paused campaigns remain in a paused state.

3. DETAIL OF SCRIPTING PROCESS

This section provides more detail for each stage in the load process to give a better understanding on how to implement these processes.

3.1. PARAMETERS REQUIRED IN THIS SCRIPT

During the following script stages, several parameters are required. They are defined here, then referenced through the document.

- **URL**: A URL is used to access the Campaign Manager application. The URL required in the load processes is the application URL with /services appended.
- **Static Token**: A static token is required when connecting to the Campaign Manager API via iLoader. Create the token in the Static Token Management admin screen and associate it with the 'system'.
    - To create a static token:
        1. Log in to Campaign Manager.
        2. From the Settings section of the slide-out navigation, select Static Token Management.
        3. Click Add .
        4. Click the drop-down arrow on the Associated User field to select the user to associate the static token with.
        5. Click Save and note the static token value for later use.
- **DatasourceId**
    - The DataSourceId for each Datasource should be noted for later use.
        - To locate Datasource Id's:
            1. Log in to Campaign Manager.
            2. From the Settings section of the slide-out navigation select Datasources.
            3. For each Datasource, note the Config ID shown on the General tab.

NOTE: In the examples the following values are used:

URL = http://cm.br-pm-v08/services/

Token = 3af859fe-e749-4ebd-87ac-766cc5e638e7

DataSourceId = 102

In all cases, if more than one DataSourceId is used the commands need to be repeated.

TIP: The values can be set as variables at the start of a script allowing variables to be used in command. Example scripts in this document use the following variables and values:

```
Set urlvar = {http://cm.br-pm-v08/services/}

Set tokenvar = {3af859fe-e749-4ebd-87ac-766c-
c5e638e7}

Set datasourceidvar = {102}

;If this system is going to continue to load the staging Contact and Response tables overnight, therefore not using Intra-Day reporting, a variable for Campaign History Datase should be set to the value as defined in the Campaign Key section of the Datasource Admin.

Set CMHistoryDB = {Campaigns_History}
```

3.2. SCRIPT STAGES

The previous section gave an overview of the stages in the load process. This section provides more detail for each stage, to allow a better understanding of how to implement these processes.

3.2.1 PAUSE CAMPAIGNS AND WAIT FOR COMPLETION OF CURRENT PROCESSING

For this part of the script, iLoader is required to call two Campaign Manager API functions using the CALLCMAPI command:

- PauseDatasourceCampaigns: places all active campaigns into a paused state.
- AreDatasourceCampaignsPaused: checks current activity and returns a list of campaigns still actively executing.

The following iLoader script segment calls a pause on campaigns in the specified Datasource, and then creates a loop repeatedly checking the list of campaigns returned by the AreDatasourceCampaignsPaused call. Once this list is empty, that is "[]", the process is allowed to continue.

```
;Pause Campaigns and wait on completion of current processing

CALLCMAPI PauseDatasourceCampaigns, %urlvar%,"Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" ,ResultV

REPORT {Pausing Datasource Campaigns}
```

```
SET ResultV=""
Loop1:
IFVAL %ResultV% = "[]"
GOTO Loop1End
ENDIF
;Loop check to confirm all campaign activity is finished
CALLCMAPI AreData-
sourceCam-
paignsPaused,%urlvar%,"DatasourceId":"%datasourceidvar%"
|"Token":"%tokenvar%", ResultV
REPORT {Campaigns Running: %ResultV%}
GOTO Loop1
Loop1End:
REPORT {All Campaign activity complete}
REPORT {}
REPORT {Finished Pause}
REPORT {}
```

NOTE: %ResultV%: The above commands take this and set it as an iLoader variable. It is simply a name representing a list of active Campaign IDs.

3.2.2 ENSURE ALL OF CAMPAIGN MANAGER'S "STATE" PROCESSING IS COMPLETE

As part of enhancements made in Campaign Manager 2016 R1, specifically the introduction of Campaign History Data Table Structure /Intra-Day reporting and the removal of Event data from SQL, some Engine activity (for example Event data loads to Engine and staging table archiving) may continue even when campaigns are paused. This stage ensures that these processes exit cleanly before the load process continues.

```
;Ensure all of Campaign Managers "STATE" processing is complete
REPORT {Mark Datasource as Pending Disabled}
CALLCMAPI MarkDatasourceAsPendingDisable, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Pending DisableResult %ResultV%}
SET ResultV=""
Loop2:
```

```
IFVAL %ResultV% = "[]"
GOTO Loop2End
ENDIF
REPORT {}
REPORT {Checking if Datasource jobs completed}
REPORT {}
CALLCMAPI AreDatasourceJobsComplete, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Jobs Running: %ResultV%}
GOTO Loop2
Loop2End:
REPORT {All Datasource jobs complete}
REPORT {Finished job termination}
REPORT {**}
```

3.2.3 DROP USER CREATED DATABASE OBJECTS

Campaign Manager users can create in-line engineering or table column engineering as part of their daily activity. As the next two stages relate to Campaign History, all Data Engineering created on the Campaign History tables should also be dropped at this point. Clearing these down will optimize the performance of Campaign roll-off.

Drop these via a Campaign Manager document containing one or more Engine Cleanup tools with the relevant selection made. Execute Campaign Manager Documents via an iLoader command.

iLoader created History based engineering can be dropped by iLoader.

For Data Engineering options, refer to the Best Practice Guide - Data Engineering in Campaign Manager and Engine.

The RunDocument function has the following format:

```
REPORT {Processing Segment Document to drop User Engineering}
CALLCMAPI RunDocument, %urlvar%,”Token”:”%-
tokenvar%”|”IgnoreLock”:”0”|”Id”:”1234”|”SaveResults”:”1”
REPORT { Segment Document processing complete}
```

Additional parameters are defined as follows:

- “IgnoreLock”:”<value>” - The IgnoreLock setting relates checks to see if the document is locked that is potentially being accessed. This is a 0 or 1 setting where = 1

will still process the document even if it is locked.

- "Id":"<value>" - This is the internal document ID. To assist with this, the document ID has been added to the document preview in the Document Explorer in CM.
- "SaveResults":"<value>" - This is equivalent to the Save Results check box on the scheduled document dialogue in Campaign Manager where, for documents that contain report Grids or Charts, the user would want to save those results to the document. This is likely to be set to '1' for documents with Reporting and '0' for Data Engineering documents.

NOTE: PIPE separators between the name.<value> parameters.

3.2.4 CONTACT/RESPONSE HISTORY ROLL-OFF

At any time, administrators can select a campaign's Contact and Response history data for removal, along with deleting its Master Campaign table record. The actual removal of data will occur at this stage in the load process. The selection of campaigns for removal is carried out via Campaign History Roll-off in the Settings section of the slide-out navigation pane, by the following two options:

1. Purge: Fully removed with no back up.
2. Roll-Off: Removed from History and rolled off to a text file.

The following command from iLoader interrogates the list of campaigns and triggers a delete on data related to the selected campaigns.

```
REPORT {Rolloff Campaigns - Start}
Rolloffcampaigns %urlvar%, %tokenvar%
REPORT {Rolloff Campaigns - Finish}
```

3.2.5 DISABLE DATASOURCE

For this part of the script, a function for the CALLCMAPI command is required.

```
REPORT {Disabling Datsource}
CALLCMAPI DisableDatasource, %urlvar% , "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Datasource Info: %ResultV%}
```

The result retuned will always be True by way of confirming the call has been made.

3.2.6 ENGINE SHUTDOWN/STARTUP

This process is important for several reasons, firstly to ensure that the services do not continue to poll the Datasource for updates during a load process, but also it is useful

at the start of a load to clear Engine buffers, connections and memory allocation to begin with a clear repository.

The suggested iLoader commands to perform these tasks use the Engine Nucleus Dispatch call to request an Engine Shutdown.

If the parameter is set to '0', this command waits 10 seconds then implements an Engine Shutdown call, allowing Engine to drop its session and objects. The example below shows 20 seconds to give Engine a little longer.

The dispatch call does not perform an 'End Process' of the Engine process and with the Campaign Manager system. An End Process on the Engine process will always have the potential to cause database corruption and must be avoided.

After the DISPATCH call, use iLoader commands to repeatedly perform a check to see if Engine has closed. The following example will loop 20 times and then stop looping. At this point further error situation handling should be put in place as required.

```
REPORT {***Starting DISPATCH Shutdown process with 20 second
wait***}
DISPATCH 25000, 20
SET Number_of_loops=1
tbLoop:
REPORT {Number_of_loops is now = %Number_of_loops%}
IF VAL %Number_of_loops% > 20
GOTO LoopEnd
ENDIF
IF_NOT_SERVER_STOPPED
REPORT {This Server is NOT Stopped}
WAIT_FOR_SERVER_STOP 20
REPORT {Waited 20 secs and Server is still NOT Stopped}
ADD Number_of_loops,1
GOTO tbLoop
ELSE
REPORT {This Server IS Stopped}
ENDIF
GOTO LoopStopped
LoopEnd:
REPORT {LoopEnd, Number_of_loops > 20}
LoopStopped:
```

```
REPORT {LoopStopped, End of looping}
```

3.2.7 ENGINE DATA LOAD

At this stage, the Datasource is disabled, the campaigns are paused and all existing activity has completed.

As the Campaign Manager system is now fully shut down, this is an ideal time to perform system backups. For more information, see the **Campaign Manager 6.0 Backup Guide**.

Engine has some significant changes that require a full overview of existing Data Warehouse load scripts. This is especially true if the scripts have been in existence for some time and do not make use of recent improvements to load processes.

For more information, see the **CM 6.0 iLoader Best Practice - Strategies for Loading Data Guide**.

The subject of minimizing downtime using A/B repository switching is covered in the **CM 6.0 Best Practice Guide - Minimizing downtime with AB Repository switching**.

Although the load of the Customer Data Warehouse is more in the domain of specific systems, the campaign data workflow will have created Engine links between tables that contain a campaign key (s), for example [Demo].[Customer] and the system created Campaign Manager State tables. It is important to ensure these links are recreated.

Prior to any Customer Date Warehouse loads, the existing links must be captured using iLoader standard commands, and then for completeness they should be dropped.

The following script example will create a link script called StateLinks.txt, containing all links from the [_CampaignTables] database and then drop them. This will include links to other repositories if an A/B repository switch process has been employed.

```
SC_STORELINKSFORDATABASE StateLinks.txt , [_CampaignTables]
```

Drop all links from [_CampaignTables]

At this point, the Customer Date Warehouse load can take place or the repository switch can occur.

At the end of that process, the following command is used to recreate the links.

```
Process StateLinks.txt
```

3.2.8 ENABLE DATASOURCE

For this part of the script, one function for the CALLCMAPI command is required.

```
REPORT {Enabling Datasource}
CALLCMAPI EnableDatasource, %urlvar%, "Data-
```

```
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Datasource Info: %ResultV%}
```

3.2.9 PROCESS ATTRIBUTION RULES

Attribution rules can be set up against a creative in a campaign. These rules will run against a specified table and find the responses in that table identified by the rule related to the outbound communication for that creative in the campaign. This process then creates a new key column of type FIELD and format TEXT on that table and link that key back to the HistoryId on the Contact History table to facilitate campaign response reporting.

The naming convention for the new key column is:

_HistoryDatabasename_CampaignKey_HistoryId

For example:

_CM_History_Demo_Customer_Cust ID_HistoryId

The following command runs attribution for any attribution tools across all campaigns (including multiple campaign keys) within Project 1 (i.e. Demo database). In this example the parameters would need to be changed for the desired environment.

This command requires URL, Token and DatasourceId parameters.

NOTE: The URL for this command has additional parameters. In this case, the standard application URL is amended with /Services/EMService.svc/cert for Http and /Services/EMService.svc/httpscert for Https

For example:

http://cm.br.qa.com/services/EMservice.svc/cert

http://cm.br.qa.com/services/EMservice.svc/httpscert

Example:

```
REPORT {Starting campaign attribution}
ATTRIBUTERESPONSES %urlvar%/services/EMservice.svc/cert, %token-
var%,%datasourceidvar%
REPORT
{Attribution processing complete}
```

For each attribution table, the process attributes any Order records on the table that meet the attribution query, and stores the internal URN row number of the last record it attributed. It then continues from that point in the next day's attribution process. Presuming that the table being attributed has an append load strategy, such as appending

new order transactions to an existing order table, any campaign can only attribute against new orders.

If data in the attribution table is deleted/rolled off, or becomes corrupt, you may need to reset the attribution urn to force the system to re-attribute the newly loaded records. To achieve this, two commands exist in iLoader as follows:

a. RESETALLATTRIBUTIONURN url, token, datasourceids

```
RESETTABLEATTRIBUTIONURN %urlvar%/EMservice.svc/cert,
%tokenvar%, [Project1].[Demo].[Customer]
```

RESETALLATTRIBUTIONURN %urlvar%/EMservice.svc/cert, %tokenvar%, %data-sourceidvar%

Resets the attribution URNs to that of the last row in the tables, so that the next attribution pass will begin processing from there. Do this for all response tables that are referenced by attribution nodes for the specified list of Datasource ids.

Example Usage:

1. Response table entries rolled off – Reset start urn back to the resulting last row in order that any subsequently loaded records are not missed.
2. Reattribute reloaded data – calling this when the table is empty (before the reload) will reset the start attribution urn to -1 causing it to reconsider the whole table again.

b. RESETTABLEATTRIBUTIONURN url, token, fullyqualifiedtablename

Resets the attribution start URN for a single response table. Table name should include datasource, for example [Project1].[Demo].[Customer]

```
RESETTABLEATTRIBUTIONURN %urlvar%/EMservice.svc/cert,
%tokenvar%, [Project1].[Demo].[Customer]
```

NOTE: The attribution process is aimed at tables that have an append load strategy, as this is considered the norm for tables being attributed such as transaction tables. This is the method that the attribution process is tested.

If there is a requirement to attribute against a table that is loaded by an UPADTE strategy, some special considerations are required that are documented here but should be considered as varying from the norm and is not a method that Alterian currently test.

The RESETALLATTRIBUTIONURN resets the last attributed URL value to the last rows in the table to cover a data roll-off, but in an UPDATE stagey, the order of records may change so there would be a need to set the last attributed URL to the start of the table to allow re-attribution of the entire table. This can be done with two important notes:

1. Re-attributing the entire table will not re- attribute any rows with a populated attribution key.

2. Re-attributing the entire table adds extra processing which impacts on load times.

At this point there is no product command to reset the last attributed URL to zero. This is done using a BATCH command to interact directly with the SQL back end.

The following example shows %value% variables for the SQL Server and SQL DB values. It also has a hardcoded path for the sqlcmd.exe that may need to be edited.

```
REPORT {*** Resetting Attribution URN to -1 ***}

BATCH {cmd.exe /C ""C:\Program Files\Microsoft SQL Server-
\100\Tools\Binn\sqlcmd.exe" -S %SQLSERVER% -d %SQLDB% -Q
"UPDATE [Alchemy_TWS].[DS].[LastCampaignAttributionURN] SET
LastURN=-1""}
```

3.2.10 DOCUMENT EXECUTION

Campaign Manager documents can be executed via an iLoader command that allows segment documents with Data Engineering or Report creation to be included in this load process.

The RunDocument function has the following format:

```
REPORT {Processing Segment Documents}

CALLCMAPI RunDocument, %urlvar%,"Token":"%-
tokenvar%"|"IgnoreLock":"0"|"Id":"1234"|"SaveResults":"1"

REPORT { Segment Document processing complete}
```

Additional parameters are defined as follows:

- "IgnoreLock":"<value>": The IgnoreLock setting relates to a check to see if the document is locked i.e. potentially being accessed. This is a 0 or 1 setting where = 1 will still process the document even if it is locked.
- "Id":"<value>" : The internal document id. To assist with this, the document ID has been added to the document preview within the Document Explorer in Campaign Manager.
- "SaveResults":"<value>" : This is equivalent to the Save Results checkbox on the scheduled document dialogue in Campaign Manager where, for documents that contain report Grids or Charts, the user would want to save those results to the document. This is likely to be set to '1' for documents with Reporting and '0' for Data Engineering documents.

NOTE: PIPE separators between the name.<value> parameters.

3.2.11 RESTART CAMPAIGNS

Perform the campaign restart in iLoader using the ResumeDatasourceCampaigns function for the CALLCMAPI command. This will resume all campaigns that were paused by the API.

```
REPORT {Re-starting campaign activity}
CALLCMAPI ResumeDatasourceCampaigns, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%",ResultV
REPORT {Re-starting campaign activity}
```

The parameters used here are as the same as previously defined.

4. FULL SCRIPT EXAMPLE

```
;DOCUMENT SECTION 3.1

;The following "Set" commands allow values to be set for vari-
ablesonce to avoid repeated data and for better management.

Set urlvar = {http://cm.br-pm-v08/services}

Set tokenvar = {3af859fe-e749-4ebd-87ac-766cc5e638e7}

Set datasourceidvar = {102}


;If this system is going to continue to load the staging
Contact and Response tables overnight, therefore not using
Intra-Day reporting, a variable for Campaign History Datase
should be set to the value as desined in the Campaign Key sec-
tion of the Datasource Admin.

;Set CMHistoryDB = {Campaigns_History}


;DOCUMENT SECTION 3.2.1

;Pause Campaigns and wait on completion of current processing

CALLCMAPI PauseDatasourceCampaigns, %urlvar%,"Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" ,ResultV

REPORT {Pausing Datasource Campaigns}

SET ResultV=""

Loop1:

IFVAL %ResultV% = "[]"

GOTO Loop1End

ENDIF


;Loop check to confirm all campaign activity is finished

CALLCMAPI AreData-
sourceCam-
paignsPaused,%urlvar%,"DatasourceId":"%datasourceidvar%"
|"Token":"%tokenvar%", ResultV

REPORT {Campaigns Running: %ResultV%}

GOTO Loop1
```

```
Loop1End:
REPORT {All Campaign activity complete}
REPORT {}
REPORT {Finished Pause}
REPORT {}

;DOCUMENT SECTION 3.2.2
;Ensure all of Campaign Managers "STATE" processing is complete
REPORT {Mark Datasource as Pending Disabled}
CALLCMAPI MarkDatasourceAsPendingDisable, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Pending DisableResult %ResultV%}
SET ResultV=""
Loop2:
IFVAL %ResultV% = "[]"
GOTO Loop2End
ENDIF
REPORT {}
REPORT {Checking if Datasource jobs completed}
REPORT {}

CALLCMAPI AreDatasourceJobsComplete, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Jobs Running: %ResultV%}
GOTO Loop2
Loop2End:
REPORT {All Datasource jobs complete}

REPORT {Finished job termination}
REPORT {**}

;DOCUMENT SECTION 3.2.3
```

```
;This section will remain commented out as the system for this sample script is using Intra_day updating of the Campaign History Tables
;REPORT {Starting Contact and Response archive process}
;ARCHIVETABLES _TacticOutputTables , %CMHistoryDB%
;REPORT {Contact and Response archive processing complete}

;DOCUMENT SECTION 3.2.4
REPORT {Processing Segment Document to drop User Engineering}
CALLCMAPI RunDocument, %urlvar%,"Token":"%-
tokenvar%"|"IgnoreLock":"0"|"Id":"1234"|"SaveResults":"1"
REPORT {Segment Document processing complete}

;DOCUMENT SECTION 3.2.5
REPORT {Rolloff Campaigns - Start}
Rolloffcampaigns %urlvar%, %tokenvar%
REPORT {Rolloff Campaigns - Finish}

;DOCUMENT SECTION 3.2.6
REPORT {Disabling Datsource}
CALLCMAPI DisableDatasource, %urlvar% , "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Datasource Info: %ResultV%}

;DOCUMENT SECTION 3.2.7
REPORT {***Starting DISPATCH Shutdown process with 20 second
wait***}
DISPATCH 25000, 20
SET Number_of_loops=1
tbLoop:
REPORT {Number_of_loops is now = %Number_of_loops%}
IF VAL %Number_of_loops% > 20
GOTO LoopEnd
```

```
ENDIF

IF_NOT_SERVER_STOPPED
REPORT {This Server is NOT Stopped}
WAIT_FOR_SERVER_STOP 20
REPORT {Waited 20 secs and Server is still NOT Stopped}
ADD Number_of_loops,1
GOTO tbLoop

ELSE
REPORT {This Server IS Stopped}
ENDIF

GOTO LoopStopped
LoopEnd:
REPORT {LoopEnd, Number_of_loops > 20}
LoopStopped:
REPORT {LoopStopped, End of looping}

;DOCUMENT SECTION 3.2.8
SC_STORELINKSFORDATABASE StateLinks.txt , [_CampaignTables]
Drop all links from [_CampaignTables]

;*** CUSTOMER DATA MART LOAD***
Process StateLinks.txt

;DOCUMENT SECTION 3.2.9
REPORT {Enabling Datasource}
CALLCMAPI EnableDatasource, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%" , ResultV
REPORT {Datasource Info: %ResultV%}
```

```
;DOCUMENT SECTION 3.2.10

REPORT {Starting campaign attribution}

ATTRIBUTERESPONSES %urlvar%/EMservice.svc/cert, %tokenvar%,
%datasourceidvar%

REPORT {Attribution processing complete}


;DOCUMENT SECTION 3.2.11

REPORT {Processing Segment Documents}

CALLCMAPI RunDocument, %urlvar%,"Token":"%-
tokenvar%"|"IgnoreLock":"0"|"Id":"1234"|"SaveResults":"1"


REPORT { Segment Document processing complete}


;DOCUMENT SECTION 3.2.12

REPORT {Re-starting campaign activity}

CALLCMAPI ResumeDatasourceCampaigns, %urlvar%, "Data-
sourceId":"%datasourceidvar%"|"Token":"%tokenvar%",ResultV

REPORT {Re-starting campaign activity}
```