



EngageOne Server

Administration Guide

Version 4.4 Service Pack 11



Table of Contents

1 - EngageOne Server overview

EngageOne Server operating modes.....	5
User roles available in EngageOne Server.....	6
Definition of terms.....	7
Architectural overview.....	10

2 - Getting started

Communities.....	14
Running EngageOne Server Admin.....	16
Common operations.....	17

3 - Template management

About template management.....	21
Folder management.....	24
Effective dates and withdrawn dates.....	26
Spell Checker.....	28
Document Classes.....	28
Working with templates.....	28
Message Content management.....	39
Active Content management.....	41
Working with EngageOne Video.....	44
Working with EngageOne Communicate.....	45

4 - External File management

Working with external files.....	47
Working with images referenced by external key maps.....	48
Using the EngageOne Key Map Generator.....	49

Using key maps.....	50
Using lookup tables.....	55
Using RTF files.....	58

5 - Delivery management

About delivery management.....	61
Output variables.....	62
Resources.....	67
Devices.....	74
Recipients.....	78
Compatibility modes.....	84
Delivery channels.....	85
Delivery options.....	118

6 - Community Administration

Document class management.....	121
Retention policy rules.....	121
Retention rules run rate.....	122
Retention history.....	123
Spell checker management.....	123
Data map management.....	124
EngageOne Communicate Integration.....	127
Vault Databases.....	129
Diagnostics.....	129

7 - System administration

About system administration.....	131
The configuration framework.....	131
Diagnostics.....	132
Resolving composition issues.....	133
Repository configuration.....	133
Active Drive configuration.....	134

Account unlocking.....	136
------------------------	-----

8 - Event Monitor

Event Monitor.....	139
EngageOne Notifications.....	140
EngageOne events.....	141
Event Monitor tables.....	148
Event configuration.....	153
Logging events.....	164
Attachment conversion process.....	166

9 - Batch processing

Accumulated batch.....	168
Non-accumulated batch.....	175
NA batch setup and performance optimization..	200
Purging using batch.....	204
Log file location.....	209
Logging information (accumulated and non-accumulated batch).....	209
Modifying log file size and count.....	210
Batch error handling.....	212
Batch threshold settings.....	214
Setting up the batch programs on a separate machine.....	217
Running batch with integrated security database connections	219

10 - Managing your EngageOne environment

Managing disk space.....	221
System performance recommendations.....	225
Tuning the system for performance.....	234
System interactions and problem solving.....	244
Monitoring the Interactive Editor's WebSocket connections.....	260

11 - EngageOne Server system monitoring

Use case 1: Monitoring with Oracle's JConsole..	264
Use case 2: Monitoring with commercial monitoring tools - example: AppDynamics.....	267

1 - EngageOne Server overview

EngageOne Server is an enterprise solution for dynamic communications. Document templates and optional content objects are defined by the template designer in a core application called Designer, and managed in the content repository. These templates are accessed by front-office users through EngageOne Interactive or custom web-based applications to create and modify documents in a controlled manner and distribute the documents through various delivery channels.

Many of the same features can be enabled via custom web applications using web services.

In this section

EngageOne Server operating modes.....	5
User roles available in EngageOne Server.....	6
Definition of terms.....	7
Architectural overview.....	10



EngageOne Server operating modes

EngageOne Server allows you to generate relevant communications through a mix of batch and real time processes delivered through your customer's preferred delivery channel using one of the following modes of operation:

Interactive

Interactive allows for the creation, control and delivery of ad hoc customer correspondence, such as letters and offers, based on a standard Designer publication templates. In this scenario the designer creates a template which will be launched and interactively customized by a front-office user before delivering the document. Data can be supplied to your interactive communication from your existing business system on a push/pull basis and can be modified by the end-user via data prompting. These features are governed by the design of your template and associated data model.

Interactive is delivered as an "out of the box" application or as a collection of web services together with the editors to allow for data to be supplied to the interactive communication; this allows for flexible deployment. Loose integration and lower cost of ownership can be achieved with the standard application and the data push capability. More streamlined integration can be implemented using the web services and together with your organization's chosen editor type to tightly integrate the interactive features into your core business systems. The process of submitting documents for batch delivery in EngageOne Interactive is referred to as accumulated batch.

A note about editor types

Interactive can be configured to use multiple editor types, this is configured during the installation of EngageOne. For details about the various editor types, refer to the EngageOne Server Installation guides.

EngageOne Batch

EngageOne Batch allows you to vary template and delivery options for each customer record and includes the ability to use different template versions through the Effective date feature. EngageOne integrates with Vault and EngageOne Deliver and configuration is made using the Administration client. The Designer/Generate solution offers greater performance but is tied to a single template. Applications that are more appropriate for EngageOne high-speed batch are for example, account statement applications with high-volume requirements and small Service Level Agreements. This type of batch processing is known as non-accumulated batch. Where optimum non-accumulated batch processing speeds are required, use Express Batch processing option; for detailed information refer to [Express Batch](#) on page 192.

EngageOne On-Demand

EngageOne On-Demand allows you to deliver communications in real-time for business processes that require this. Some examples include account activities through your portal or self-service options. On-Demand uses the standard delivery option features of EngageOne which caters for all

mulch-recipient/channel output in forms other than real-time. For example, an archive copy may be queued up and processed later in a batch job. Where optimum On-Demand processing speeds are required, use the Express Immediate option; for detailed information refer to [Compatibility modes](#) on page 84.

User roles available in EngageOne Server

Administrative:

- Delivery Manager
- External File Manager
- Community Administrator
- Workflow Administrator
- Template Manager

Interactive:

- Document Manager
- Editor with Saving New Templates
- Editor

Project:

- Reviewer
- Project Manager

Vault:

- Viewer
- Viewer with Reprint

Definition of terms

Active-drive	The default suggestion of the shared path made during installation by the EngageOne Installer. See Shared Path below.
Community	A community represents a line of business, department, or section in your business environment.
Device	Indicates the output datastream. Represented with a metafile and a HIP file provided by Designer. Can be imported into the EngageOne repository. In the EngageOne database schema, a device entry will contain an internal device identifier, device name, metafile resource identifier and HIP file identifier.
Delivery channel	Delivery Channels are reusable and define the properties used to compose an output stream.
Channel details	Set of sub tables (e-mail, fax, print, archive, etc.) for a delivery channel object. Sub tables provide additional information regarding channel details.
Delivery options	A set of one or more delivery channels that can be directly associated with one or more EngageOne templates.
Express Batch	This option can improve NA batch performance; in this mode, NA batch skips additional configurations provided by Engage One Server during delivery channel definition, thus reducing I/O operations.
Default Batch	This delivery channel option allows for a full set of configuration options; it is used in NA and Accumulated batch document composition settings.
Express Immediate	This delivery channel option has limited configuration options to allow for increased performance for document composition in an EngageOne Server On-demand setting

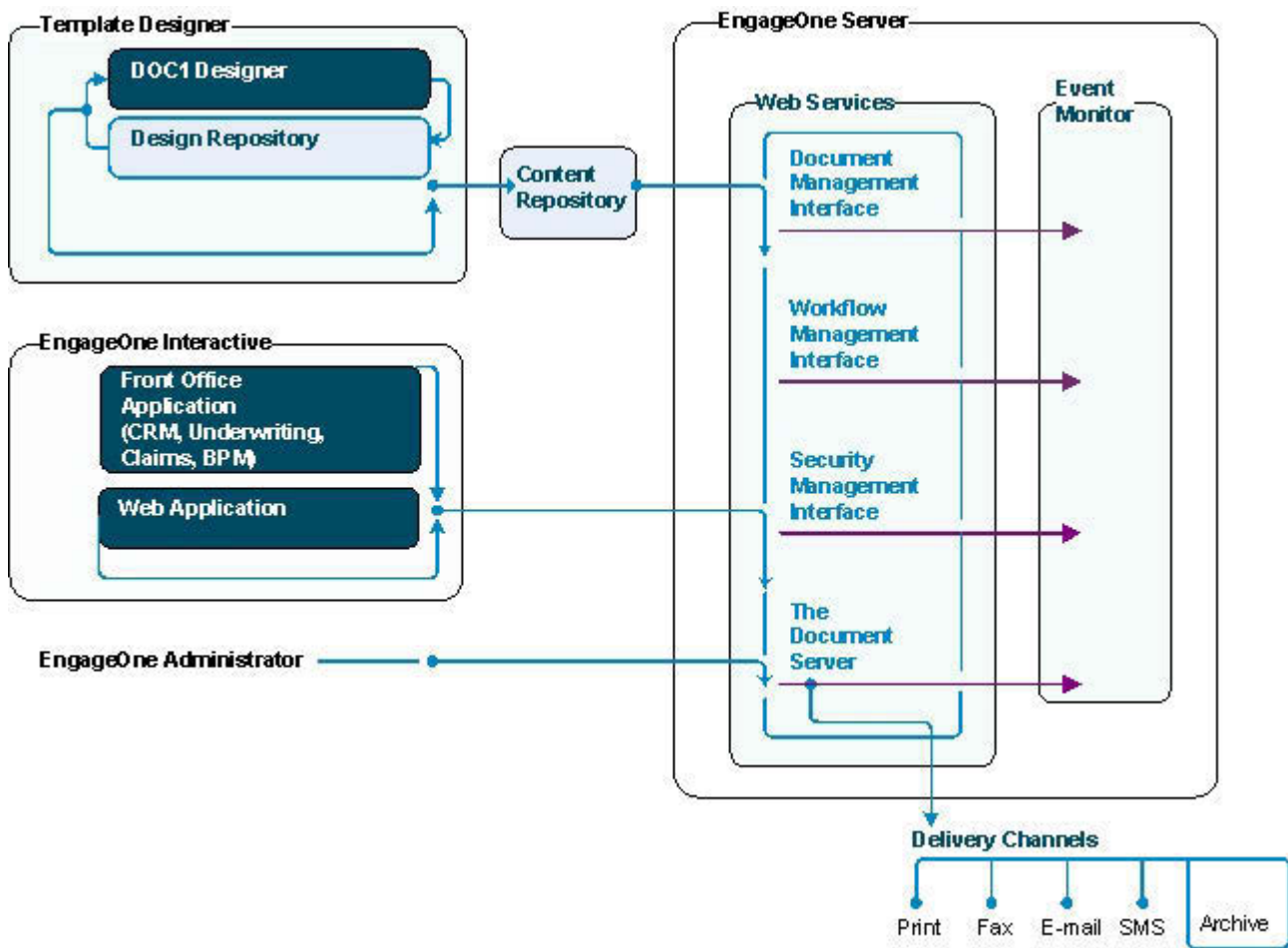
Default Immediate	This delivery channel option provides a full set of configuration settings for document composition in an EngageOne Server On-demand setting.
Manipulation detail	Sub table for a delivery channel object. Provides more information regarding manipulation details. Applies only for type=Print and mode=Batch.
Delivery option	One or more delivery channels. Can be directly associated with one or more EngageOne templates.
EngageOne server	The core system including the Administration user interface, Web services, and the document server.
HIP file (for EngageOne)	Created by a publishing task, a HIP file contains the information required for a publication in the production environment (except resources).
Key map	<p>A key map is a set of key value and image reference pairs intended for document composition.</p> <p>A key map will map keys for keyed objects to the image or Active Content, when using resources in the Work Center. If images are supplied externally, it maps keys to sample images for use when designing, with the images provided via an external control.</p>
Metadata	Metadata is associated with folders and templates. Used for template search and retrieval.
Output variable	Reserved keyword that is system or user defined in EngageOne as a variable name. Can be resolved during system run time.
Recipient	Name allocated to the person receiving a copy of the interactive document. This name must exactly match the name condition to set carbon copy options in Designer.
Resources	References used to identify font and images required to print or present the resulting output datastream on the intended output device.
Shared path	Common term for the active-drive. Contains the content repository, license file, and configuration settings.

Template

Package that contains the publication and the interactive sections of a document, ready for use by front-office users in EngageOne.

Architectural overview

The diagram below illustrates the main components of the EngageOne environment.



Template designer

A template consists of a traditional Designer publication design, along with interactive data definitions. Templates describe the front office user interactions required to complete a document from the template. The interactive data model is defined and exposed using the W3C XForm standard. The XForm can be used by the system integrator to push system data into the interactive document. For more detailed information about XForms, see the EngageOne Programmer's Reference Guide.

Template designer can allow the front-office user to edit or enter free form text, as required in an EngageOne document design, by selecting a section of text in a paragraph object and defining this as editable. Text not marked as editable will be protected, and text that is marked as editable can be manipulated by the front-office user.

In addition to the basic template, the template designer can create optional content. Property values are defined enabling them to be conditionally placed within an Active Content Group. When Active Content is marked as interactive, the front-office user is dynamically presented with choices of content to help complete their document.

Once the EngageOne template has been designed and data fields defined, it is named and published for EngageOne. The XForm files that define the document's data model are included in the zip file that makes up the template.

Template deployment is the process of importing these files into the EngageOne content repository, using the EngageOne Administration client. The same process is used when the template is imported.

Content repository

Content repository is the content management system that provides file storage and version control for EngageOne. It houses the templates used to create documents, the supporting files needed for document composition, and document instances waiting for delivery.

EngageOne server

There are three main tiers in EngageOne server.

- The Web Services layer exposes EngageOne Communication Services to a custom built front office application, or EngageOne Interactive, both delivered as part of the EngageOne standard install.
- The Business Layer provides:

Document management, including storage and retrieval of templates (along with template version control), and work queues (including storage and retrieval of document instances).

Delivery Services, which defines delivery channels and provides services to generate draft and final copy output.

- The Event Monitor serves two purposes. It monitors the health of the system and tracks all system activities. Tracking information can be used for reporting purposes. For example, how often a specified template is used, or how quickly a template is processed.

Delivery channels

Delivery channels supported by EngageOne include Print, Fax, e-Mail, SMS and archive. Output can be requested for real-time delivery on a local printer in either draft or final format, or queued for EngageOne batch processing for print devices such as AFP or Postscript printers.

EngageOne administration

The EngageOne administrator uses a Web application to set up and configure the EngageOne environment. The Web application is delivered as part of the standard EngageOne Server install.

The EngageOne Administration Web interface supports the management of folders, templates, images and delivery channels. For security purposes, role based access and user roles are defined in the System Administration section. System diagnostics are handled by the EngageOne administrator.

EngageOne Interactive

EngageOne Interactive is a stand-alone front office application delivered with the EngageOne product. It provides a browser-based interface for front-office users to create customized documents. The Web client can support a widely distributed user base, including remote users.

In many cases however, a custom designed application might be more appropriate for addressing your own business needs. Using the Web Services interface you can achieve seamless integration between EngageOne and your existing business systems, creating your own application that will replace the sample one provided.

Note: See the EngageOne Programmer's reference Guide for details on using Web Services to build your own Web application.

After logging in to EngageOne, front-office users will create a customer document based on a template chosen from the template repository (designed by the template designer). The Interactive Editor provides the interface for data prompting, optional content selection, and secured free form editing. See the EngageOne Programmer's Reference Guide for more details about the Interactive Editor.

Work can be created, saved, retrieved and reassigned in the work queue process. Once completed, the document can be delivered through various delivery channels for distribution.

For more information about using the standard Web application provided with the EngageOne system, see the EngageOne Interactive User's Guide.

2 - Getting started

EngageOne Server Admin requires that you provide a valid user-id and password combination to access its features.

In this section

Communities.....	14
Running EngageOne Server Admin.....	16
Common operations.....	17




Communities

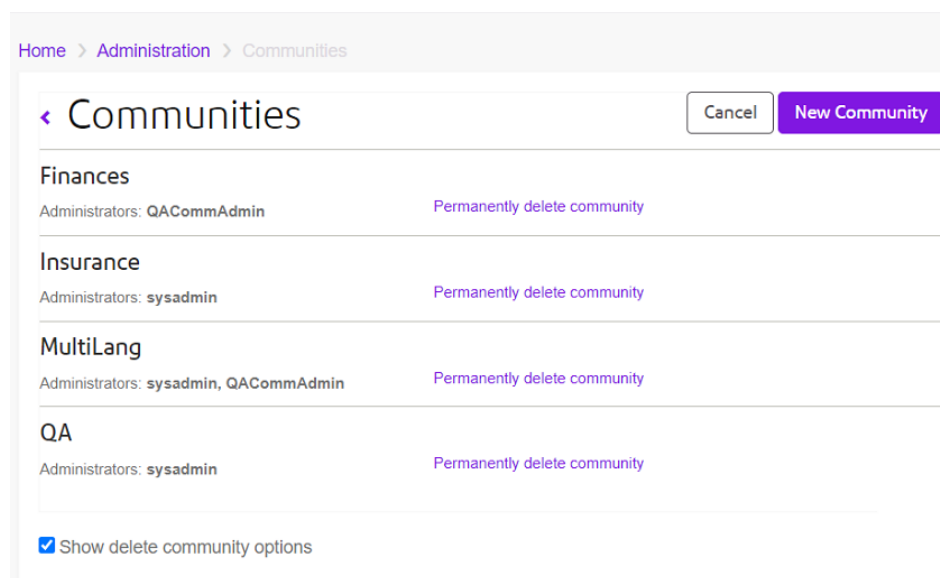
An EngageOne Server Community represents a line of business, department, or section in your business environment.

You need to create at least one community to work within before using EngageOne Server Admin.

Note: as a system administrator you can access Admin to perform system administration tasks, for example, to create Communities, perform diagnostics and repository configuration.

To access the Communities page

- Click the Admin  icon
 - From the System Administration section of the page, click Communities.



Working with Communities

To create a new Communities

- Click on **New Community** you are prompted for:
 - Name
 - Administrator(s)

Note: You can choose one or more users from your organization's LDAP directory service.

To edit an existing Community


- Click on an existing community's name, for example, ***Billing*** above. You can change the community's name and add/remove **Administrators**.

To delete a Community

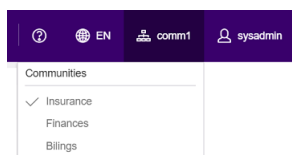
- Click **Show delete community options** to allow deletion of the selected community.

Running EngageOne Server Admin

To access EngageOne Server Admin:

- Click the Admin  icon
 - From the Community Administration section of the page, click **EngageOne Server Admin**

The EngageOne Server Admin page is displayed. You have access to, and can switch between communities in which you have Administrator access rights, refer to **Communities** on page 14 for further details.









Note: Select the required language from the list of supported languages; in the example above EN has been selected. The language you select dictates date and time formatting. If you wish to change this formatting and still retain the selected user interface language, this can be done directly from the browser in Chrome, alternatively, set the appropriate regional settings for your operating system.

Common operations

Common list view operations such as importing resources and page navigation, use the same user interaction. This section describes these interactions.

List view operations

New +	Create a new item.
Delete 	Delete a selected list view item.
Import 	Import a resource.
Export 	Export a resource.
Copy 	Copy a selected resource.
Update 	Update a selected resource.
Used By 	Identifies which resource uses the selected list view item.

Page navigation



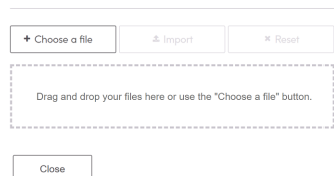
Page navigation controls allow you to:

- Select the number of items to be displayed using the **per page** drop-down.
- Navigate to a specific page by:
 - First, previous, page number, next and last page.

Importing resources

Templates

Import



- Click **Chose a file**
 - Use the Windows Open dialog or drag and drop one or more files into the indicated area as required.

Note: click **Reset** to clear the list of files to be imported.
- Click **Import** to start the Import process.

External files

The mechanism for importing external files is very similar to the template import, except that you:

- are prompted to provide a name for the external file
- must specify the save location
- can only import a single file at a time

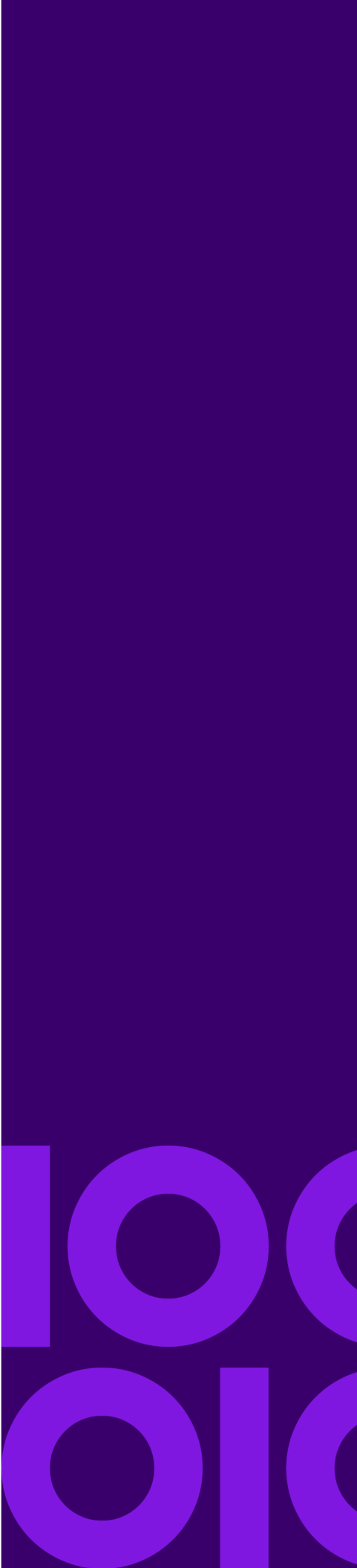
3 - Template management

Note: Before you can use a template in EngageOne Administration, you must define a default device for viewing and define at least one delivery option. Devices are created in Designer and imported into EngageOne. See [Delivery management](#) on page 60 for more information.

The search features allow you to find the resources quickly. The basic search provides for simple resource name searches with some filtering and sorting capability. The advanced search, on the other hand, allows you to perform searches based on the attributes of the resource. The sections that follow provide detailed information on the available search options.

In this section

About template management.....	21
Folder management.....	24
Effective dates and withdrawn dates.....	26
Spell Checker.....	28
Document Classes.....	28
Working with templates.....	28
Message Content management.....	39
Active Content management.....	41
Working with EngageOne Video.....	44
Working with EngageOne Communicate.....	45



About template management

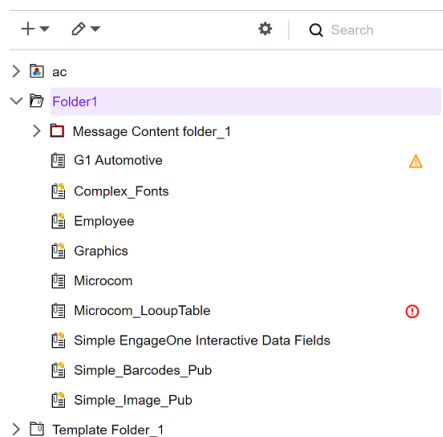
In template management, you can:

- Specify user access rights to folders, templates and Active Content
- Modify delivery options

There are two types of EngageOne templates: interactive and non-interactive. Interactive templates use the interactive data model and can be used with the Interactive Web application.

Non-interactive templates use the Designer data model and are used with the EngageOne batch processes, as well as the interactive data model.

For more information about non-interactive templates, see [Non-accumulated batch](#) on page 175



- Click + to create new folders, sub-folders and to add templates, Active Content, and Message Content to folders. You can also create folders and template at the root level from sub-folders using the appropriate right-click menu option.
- Click ✂ to cut, copy, paste and delete a folder or template or, right-click the item. Note that the **Export** option can be used for diagnostic purposes.
- Use the Search and Filter option to find resources in the treeview. Note that you can perform basic searches or more complex searches by clicking **Advanced Search**. Refer to [Common operations](#) on page 17 and [Using the advanced search feature](#) on page 23 for further details.
- Change the name of the template or Active Content, and specify document class on the **General** tab.
- Specify user access rights to folders or templates from the **Access Rights** tab.
- Specify one or more delivery options for templates from the **Delivery Options** tab.
- Apply retention rules to templates and Active Content from the **Retention Rules** tab.
- View template resources and identify any missing resources on the **Resources** tab.

- View version metadata, view a template version, and roll-back to a previous version on the **Versions** tab.
- Warning icons indicate:
 - Red exclamation mark - indicates the template is missing resources and cannot be processing any further until the missing resource(s) are available.
 - Yellow triangle - indicates that certain resources are missing, however, processing of the template can continue.
- Set a date to make the template available, and a withdrawn date for removal.

Do not delete or rename images, Active Content, or Message Content that have been used by EngageOne. Import new resources into Designer using new names. Ensure Key Maps are updated where images are revised.

You can manage different template versions by configuring effective and withdrawn dates, enabling the front-office user to select a given version from a template list according to the dates specified.

Retention policies allow you to manage your templates and their validity. Set up the retention rule name with its valid time frame in System Administration, and apply retention policies to folders or templates and Active Content. For more information, see [Retention policy rules](#) on page 121

If a custom template was created from a document in EngageOne Interactive, it would have been saved to the template folder by the front-office user. The template automatically displays in template management. If the custom template replaces the original, a new version number is automatically assigned.

Using the standard search

This type of search allows you to find resources based on the name of the resource and provides filtering and sorting options.

The screenshot shows the standard search interface. At the top, there is a navigation bar with a plus icon, a pencil icon, a gear icon, and a search bar labeled "Search". Below this is a filter dropdown menu showing "All Types". Underneath the filter are two dropdown menus: "Name" and "Ascending". At the bottom, there is a link for "Advanced Search" and a checkbox labeled "Show path".

- Click **Show path** to display the path for each resource returned in the search.
- Click **Reset** to clear the previous set of search results.
- Enter your search string in the **Search** entry area.
- Click in the filter drop-down to select the required filter to apply to the search.
- Specify sorting and ordering options for the results.

Using the advanced search feature

The advanced search allows for more comprehensive search options and presents results in a tabular form.

Advanced Search

Content Type

Field Type


Keyword Search

Date Type

From

To

Search result 4 of 4

Name	Description	Type	Effective Date	Withdrawn Date	Path	Version
 MultiLingRecipientTemplate	MultiLingRecipientTemplate	Interactive Template			/Template1/MultiLingRecipientTemplate	1.0
Publishable Active Content	Publishable Active Content	Active Content			/ac/Publishable Active Content	1.0
Simple EO-InterDataFields-PromptedforOnXform	Simple EO-InterDataFields-PromptedforOnXform	Interactive Template			/Simple EO-InterDataFields-PromptedforOnXform	1.0
Simple_Barcodes_Pub_withAC	Simple_Barcodes_Pub_withAC	Interactive Template			/Simple_Barcodes_Pub_withAC	1.0

You can choose from one or more of the following attributes:

- Select the required **Field Type** - Name, Description, Path, Version, Created By or Last Modified By
- Keyword Search - allows you to search on free-format text entered in this entry field.
- Select the required **Data Type** - Created Date, Last Modified Date, Effective Date or Withdrawn Date

Note: You can select a single **From** date or a date range using **From** and **To** date selections.

Other points to note:

- Templates with missing dependencies are indicated by a red exclamation mark icon in the leftmost column.
- Results are displayed in a tabular form and can be sorted on a column sorting control.

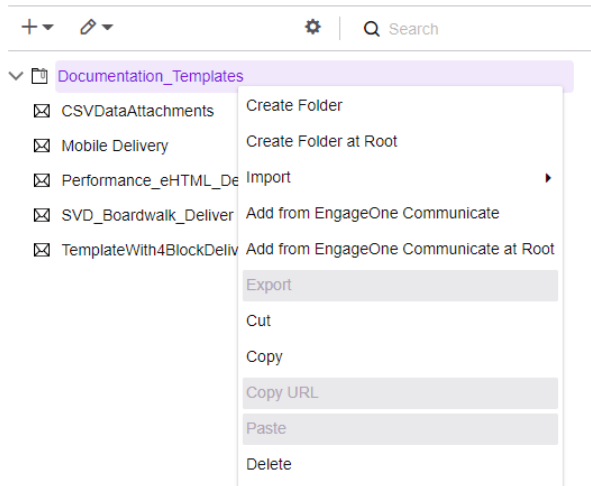
Attention:

It is possible to have multiple versions of the same template available in the system. In this scenario, the advanced search applies search criteria to the latest version of the template only; earlier versions are ignored.

Template operations

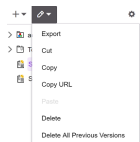
Context menu

A context menu is available when are working with resources in the tree-view area of the EngageOne Server Admin workspace. Click the right mouse button to activate the context menu.



The options available to you depend on the type of resource selected in the tree-view. For example, in the image above, because the Message Content folder is selected, you can perform appropriate actions from the context menu.

Further template options are available by clicking the  button:



Folder management

You can define an unlimited number of folders for templates, Active Content, and Message Content.

A folder name can contain up to 255 characters. Folders can be located at the root level or nested in other folders. Folder settings apply to the folder sub-levels, including templates, Active Content, and Message Content. Specifying settings at an individual level override higher nested settings. Templates and Active Content cannot be published independently together in one folder.

When the front-office user creates a new document, the document version is used. The same version is retained throughout the document life cycle even if the template is updated.

Templates are automatically given a unique version identifier. Template metadata can be used to retrieve templates by effective date.

Templates can be exported from the content repository, (for example, for diagnostic purposes), or migrated to another environment (for example, test or production). Template export does not export the related Key Map or Active Content. In order to use a template in other environments, you must export all related key maps and Active Content separately.

Note: A template cannot be deleted if it is referenced by another work item.

To create a new folder:

1. From the **Templates** tab, in the tree view click the template parent folder.

To create a parent folder, hold **CTRL** and click to deselect all folders. Then you can create a new folder at the root. You can also right-click and select **Create Folder at Root**.

2. Click **New** and select **Folder** from the menu.
3. The **Create Folder** dialog box is displayed.
4. Enter the folder name and select the folder type.

There are three types: Template, Active Content, and Message Content.

Create new folder

Destination \

Type name for a new folder

☒ Template ☐ Active Content ☐ Message Content

Document Classes

☐ Name

No data

Create Cancel

The document class defines the metadata, and is retrieved from the root folder. You can override the metadata after you have created the folder.

5. **Templates** tab folder options:

- **Access Rights:** Add and remove users to the **Selected** pane, using the left or right arrow buttons.

If you **Override** inheritance, the template is unchanged even if you update properties in the child folder level. If all access rights are removed, the template inherits its rights from the groups associated with the current community.

- **Delivery Options:** Add delivery options to the **Selected** pane, using the left or right arrow buttons.

If the delivery option is missing resources, the tick icon next to the delivery option name for the template is replaced by the exclamation icon. If you all delivery options are removed from a template, the template inherits the delivery options of the parent folder. For more information about missing resources, see [Resources](#). Note that templates within a folder inherit delivery options assigned to that folder only if the template supports the delivery option.

- **Retention Rules:** Apply to a template or folder from the **Retention Policy** list.
 - **Version** : Change a folder version.
 - Click **Set Value** to open the **Metadata Details** dialog for updating metadata values.
6. Missing resources can be resolved by exporting the resource list to Designer, and re-importing the template into EngageOne Administration. Click **Show missing resources only** to filter the list to show resources missing from the template or Active Content. For more information, see [Resources](#) on page 67 and the Designer User's Guide.
 7. To cut and paste a template:
 - a) In the tree view, locate the template you want to remove or move.
 - b) Click **Edit/Cut**.
 - c) Locate the template folder to move the template.
 - d) Click **Edit/Paste**.
 8. To rollback to a previous version:
 - a) Select a template, and click the **Version** tab.
 - b) Highlight the version you want to rollback to, and click the **Rollback** button. This makes a copy of the template version. The copy becomes the newest version, and the process is recorded in the **Comments** field. To delete a version of a template: If appropriate, you can highlight multiple versions, when you click **Delete** all highlighted versions are deleted.
 9. To delete a version of a template:
 - a) Select a template, and click the **Version** tab.
 - b) Highlight the version to delete, and click the **Delete** button.
 10. To delete a template, locate the template to delete, and select **Edit/Delete**.

Effective dates and withdrawn dates

Effective date and withdrawn date information is associated with each template version as metadata. When you import a template for the first time (version 1.0), the default value for effective date and

withdrawn date is blank. If no effective and withdrawn date are defined, new templates will inherit the properties of the parent folder.

When you import a subsequent version of a template (2.0 or greater), the template will inherit the metadata properties of the previous version. If the effective date and withdrawn date had been specified for the previous version, the default value for effective date in the new version will be the current date, and the default value for withdrawn date will be "2999-12-31". If the values are blank, the new version inherits the blank value. Use the **Version** tab to modify the effective date and withdrawn date.

During template search or document processing, template selection is based on finding the template whose effective date is *less than or equal to* the input effective date, and withdrawn date is *greater than* the entered effective date. The most recent version of a template is used if more than one template is found, or if the effective date is not specified.

The following table shows the effective date and withdrawn date for two template versions:

Version 1		Version 2		Input Effective Date	Template Version Returned
Template Effective Date	Template Withdrawn Date	Template Effective Date	Template Withdrawn Date		
2012-02-01	<Blank>	2012-03-01	2012-03-15	2012-02-01	1.0
2012-02-01	<Blank>	2012-03-01	2012-03-15	2012-02-07	1.0
2012-02-01	<Blank>	2012-03-01	2012-03-15	2012-03-01	2.0
2012-02-01	<Blank>	2012-03-01	2012-03-15	2012-03-07	2.0

Scenarios that illustrate the results for different effective dates and withdrawn dates:

- If you search for a template with an input effective date of 2012-02-01 or 2012-02-07, version 1.0 is returned.
- If you search for a template with an input effective date of 2012-03-01 or 2012-03-07, version 2.0 is returned.
- If the effective date is outside the effective date range, the system returns an error message that no template was found.
- If input effective date is not specified, the current date is used as the effective date:
 - If the current date is in the range for version 2.0, version 2.0 is returned.
 - If the current date is before or after the range for version 2.0, version 1.0 is returned.

Spell Checker

Configure how the spell checker works in EngageOne Interactive by selecting one of the options from the General tab.

- **Inherit from Parent:** Use the parent folder setting. This is the default for all templates and template folders. If all templates in a folder are set to **Inherit from Parent**, you can control the spell checker behavior for these templates by selecting the folder and changing the setting in the **General** tab.
- **Closable:** Allows EngageOne Interactive users to close the Spell Checker when prompted without completing the spell check on the work item.
- **Not Closable:** Prevents EngageOne Interactive users from closing the Spell Checker when prompted without completing the spell check on the work item.

For related information, see the EngageOne Interactive User's Guide.

Document Classes

Use document classes to define metadata fields, associate metadata fields to a template with values for the fields, and search for templates using metadata. EngageOne Interactive and web services support search using metadata fields.

For example, define a metadata field called "Department", then associate it with a template or template folder with a value of "Finance". You can associate the metadata field with other templates and folders with different values ("Insurance", "Legal", etc.). You can search for templates, Active Content, or Message Content that applies to the "Finance" department or other departments.

For more information about document classes, see [Document class management](#).

Working with templates

Export a template from the content repository (for example, for diagnostic purposes), or migrate to another environment such as test or production. When you export a template, you select the template to export and provide the path and filename. The related key map or Active Content are not exported, these must be exported separately.

Importing templates

Once an EngageOne template has been designed and data fields defined in Designer, it is named and published for EngageOne. The XForm files that define the document's data model are included in the template zip file. Template deployment is the process of importing these files into a content repository (Vault, or a third party content repository).

If a different version of a template is imported, any video configuration associated with the existing version of the template will be copied to the new version.

When creating a new template version from an old version, the video configuration associated with the old version is copied to the new version.

Refer to "*Video management*" in the Compose User's Guide for details on how template managers can configure templates to contain personalized video links.

Note: It is necessary to set appropriate "Access Rights" and "Delivery Options" at the parent folder level before attempting to rollback templates.

To import a template:

1. From the tree view click the parent folder to receive the template.
2. Click +, and select **Template** from the menu.
3. Refer to [Importing resources](#) on page 19 for further information.

Exporting templates

Export a template from the content repository, (for example, for diagnostic purposes), or migrate to another environment such as test or production. When you export a template, you select the template and provide the path and filename. When you export a template, you select the template to export and provide the path and filename. The related key map or Active Content are not exported, these must be exported separately.

To export a template:


1. From the treeview click select the template to be exported and click .

Note: Alternatively right-click the template in the treeview.


2. Click **Export**, select a location for download from the Explorer window.
3. Click **Save**.

Cut, copy and pasting templates


To cut/paste, copy/paste a template:

1. From the treeview select the template to remove or copy.
2. Click 

Note: Alternatively right-click the selected template.

3. Select **Copy** or **Cut**.
4. Locate the template folder to receive the template.
5. Click **Edit/Paste** available via the  icon or, by right-clicking the receiving template folder.

Deleting templates

To delete a template: from the treeview locate the template you want to delete, click /Delete.

Note: Alternatively, right-click the required template and select Delete from the context menu.

Note: If the template is referenced by another work item it cannot be deleted.

Note: Where a template is associated with an output device, the output device will not be removed if it has a delivery channel association.

Adding or modifying the document class

To add or modify the document class: from the **Templates** tab, click the **General** tab.

The screenshot shows the 'Document Classes' pane. At the top right is a '+ Add' button. Below it, the selected class is 'Invoice001', with a dropdown arrow to its right. Under the 'Metadata:' label, there is a table with two columns: 'Field' and 'Value'. The 'Field' column contains the text 'Field', and the 'Value' column contains a dropdown menu with the number '1' selected. To the right of the table is a trash bin icon. At the bottom left are 'Save' and 'Cancel' buttons.

Field	Value
Field	1

The **Document Classes** pane is where predefined document classes can be associated with the selected template. Use the **Add** button and the Bin icon as required.

Note: Use the Document Class tab in Community Administration to create document classes as required. Refer to [Document Classes](#) on page 28 and [Document class management](#) on page 121

Modifying user access

To modify user access to a template:

1. From the **Templates** tab, in the tree view click on the template you want to modify.
2. Click **Access Rights**. Click the **Override** button.
3. Add users to the Selected pane as required using the left or right Arrow buttons. Note: The user is only able to modify the access rights of templates and folders from the groups associated with the current community.
4. Click **OK**.

Note: If you override inheritance, the template will be unchanged even if you update properties at the child folder level.

Removing access rights

To remove all access rights associated with a template:

1. From the **Templates** tab, in the tree view click on the template you want to modify.
2. Click **Access Rights**.
3. Move all user groups in the **Selected** list to the **User** list. Note: The user is only able to modify the access rights of templates and folders from the groups associated with the current community.
4. Click OK.

Note: When you remove all access rights from a template, the template inherits access rights of the parent folder.

Changing delivery options

To add or modify delivery options associated with a template:

1. From the **Templates** tab, in the tree view click the template to modify.
2. Click **Delivery Options**.
3. Add delivery options to the **Selected** pane using the left or right Arrow buttons.

Note: If you override delivery options, delivery options will be unchanged even if you update properties at child folder level.

To remove all delivery options associated with a template:

1. From the **Templates** tab, in the tree view click the template to modify.
2. Click **Delivery Options**.
3. Move all delivery options in the **Selected** list to the **Options** list.
4. Click **OK**.

Note: When you remove all delivery options from a template, the template inherits the delivery options of the parent folder.

Applying retention policies

To apply a retention policy to a folder or template:

1. From the **Templates** tab, in the tree view click the template, or child folder to apply a retention policy.
2. Click **Retention Rules**.
3. Click the **Retention Policy** drop-down list, highlight the rule you want to apply. Refer to [Retention policy rules](#) on page 121 for details on creating Retention rules.

Note: if you override inheritance, the template is unaffected even if you update properties at child folder level.

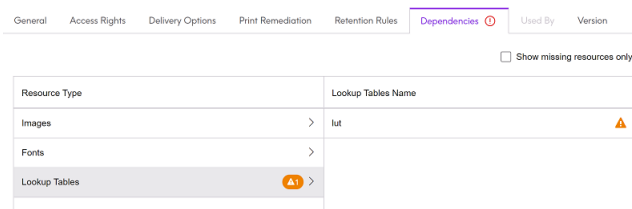
Viewing dependencies

To view dependencies:

1. From the **Templates** tab, in the tree view, click the template to view resources.
2. Click **Dependencies**.

Note: an orange exclamation mark is displayed on the Dependencies tab to indicate the template is missing resources.

3. The Resource Type section displays the resources types applicable to the template. An orange triangle icon indicates missing resources.



Resource Type	Lookup Tables Name
Images	lut
Fonts	
Lookup Tables	

4. Resolve missing resources by exporting the resource list to Designer and re-importing the template into EngageOne Administration. For more information, see [Exporting resources](#) on page 72 and the Designer User's Guide.
5. Click the **Show missing resources only** check box to filter the list to show resources missing from the selected template or Active Content.


Font and image resource properties

There may be times when you want to view the properties of a font or image. For example, if you have two images of the same name from a different repository, you can expand the image in the **Resources** tab to examine its properties.

Use **Export Resource List** to resolve resource problems. For more information, see [Managing resources](#).

1. From the **Templates** tab, in the tree view click the template.
2. Click the **Resources** tab.
3. Expand the **Fonts** or **Images** item.
4. Expand a font or image to show its properties.

For descriptions of each property, see "Font and Image Properties."

Resource Type	Image Name	Image Details
Images >	Details.bmp	Checksum: 2d16ca287d8e91151783c3633946c976
Fonts >	logo.bmp	Bit Depth: 0 Format: 36
Lookup Tables  >		Type: Windows Bitmap Height: 281 Resolution: 299 Width: 744

Property	Property Type	Description
Advanced Graphics	Font, Image	Yes or No. If Yes, this item supports Advanced Graphics features.
Bit Depth	Image	0, 1, 2, 8, 16, 24, 32. The color depth of an image.
Bold	Font	Yes or No. If Yes, the font is a Bold variation of the Typeface.

Property	Property Type	Description
Character Set	Font	<p>The character encoding system of the font. The values are:</p> <p>ANSI</p> <p>DEFAULT</p> <p>SYMBOL</p> <p>MAC</p> <p>SHIFTJIS</p> <p>HANGUL</p> <p>JOHAB</p> <p>GB2312</p> <p>CHINESEBIG5</p> <p>GREEK</p> <p>TURKISH</p> <p>VIETNAMESE</p> <p>HEBREW</p> <p>ARABIC</p> <p>BALTIC</p> <p>RUSSIAN</p> <p>THAI</p> <p>EASTEUROPE</p> <p>UNICODE</p>
Checksum	Image	A fixed-size datum computed from an arbitrary block of digital data used to uniquely identify two resources.
Font Type	Font	The font format. Examples are: Type 1 (PostScript), TrueType.
Format	Image	Format of the image. Examples are: JPG, PNG, and BMP.
Height	Image	The image height in pixels.
Italic	Font	Yes or No. If Yes, the font is an Italic variation of the typeface.
Resolution	Image	The image resolution in DPI (dots per inch).

Property	Property Type	Description
Size	Font	The point size of the typeface.
Typeface	Font	The visual representation, or type design, of the font. Also known as font family.
Weight	Font	A whole number. The thickness of the character outlines relative to their height.
Width	Image	The image width in pixels.

Viewing Used By details

You can view **Used By** details for the selected Active/Message content resource.

To view Used By information:

1. Select the required Message or Active content resource from the tree-view.
2. Click **Used By**.
3. Information on the resources which use the selected Active/Message content is displayed.

Changing effective and withdrawn dates

To change effective and withdrawn dates for a template:

1. From the **Templates** tab, in the tree view click the appropriate template.
2. Click the **Version** tab.
3. Highlight the version to change the effective and withdrawn dates.
4. Click the calendar controls, and select the dates required for template version control.
5. Optionally enter comments as required.

General
Access Rights
Delivery Options
Print Remediation
Retention Rules
Dependencies
Used By
Version

Delete All Previous Versions
Search...

<input checked="" type="checkbox"/>	Version	User	Date	Effective Date	Withdrawn Date
<input checked="" type="checkbox"/>	1.0	sysadmin	11/05/2020 11:22 AM	11/06/2020	11/07/2020

1

VERSION

Effective Date
11/06/2020

Withdrawn Date
11/07/2020

Comments

Save
Cancel

Message Content management

Content Author is used for creating, maintaining and reviewing messages that provide dynamic content for publications produced by Designer. This message content for EngageOne can be referenced (and shared) by document templates.

Working with message content

Content Author is used for creating, maintaining and reviewing messages that provide dynamic content for publications produced by Designer. This message content for EngageOne can be referenced (and shared) by document templates.

Message Content archives are created in Content Author, and do not come from the same repository as templates and Active Content. Message Content is managed separately from templates and Active Content. Message Content must be imported into a Message Content folder.

Message Content that you import into EngageOne Server will not appear in Designer, EngageOne Interactive, or any custom application that uses the editor.

Message Content that you import into EngageOne Server is tracked by Message Area IDs. Message Content is created and maintained in a framework set up by the Content Author administrator. This includes projects and folders to which messages are added, and message areas for which messages are targeted. Message areas represent reserved space in document designs for which messages are scheduled for inclusion. They can be thought of as rectangles that define the maximum size of one or more messages. For more information about Message Content, see the **Content Author User's Guide**.

You can cut and paste Message Content from one Message Content folder to another. You cannot copy Message Content. You can also delete a Message Content if that content is *not* referenced by an active content or template. Moved or deleted Message Content items are shown in the history log.

If you have any problems creating or generating documents that contain Message Content, navigate to **Templates**, select the template, and then click the **Resources** tab. Expand the Message Area to view any missing resources. If there are missing resources, see [Working with missing resources](#).

The following output devices are supported for Message Content:

Note: You must ensure that the device resources used in Message Content match the device resources used in templates. If the resources do not match, the Message Content will display missing resources.

- AFPDS 240, 300, 600 - bitmap (FOCA) and outline (FOCA Type1)

- HTML for e-mail (eHTML)
- IJPDS 240, 300, 600
- Line data
- Metacode
- PCL - bitmap and outline, black/white and color image
- PDF 72 & 300
- Postscript 72 & 300 - Type 1 & Type 42 (automatic)

These Designer resource features are not supported for Message Content:

- Font subsetting for Postscript
- Open Type Fonts (.otf fonts) using Compact Format compression
- Windows Image types using AFP Object Containers

Message Content options

1. To import Message Content:
 - a) Select the Message Content folder to store the Message Content.
 - b) Click + or right click in the treeview area, and select **Message Content**. The document class control is disabled for Message Content. You can export Message Content from the content repository for diagnostics purposes as well as migrate to a new content repository.
 - c) Refer to [Template operations](#) on page 24 and [Importing resources](#) on page 19 for further information.
2. To export Message Content:
 - a) Select the required Message Content from the tree-view.
 - b) Right click in the treeview area, and select **Export**.
 - c) The export process starts automatically, you are notified when complete.
 - d) Refer to [Template operations](#) on page 24 for further information.
3. To cut and paste Message Content:
 - a) In the tree view locate the Message Content to remove or move.
 - b) Right click in the treeview area, and select **Cut**.
 - c) Locate the Message Content folder that will receive the Message Content.
 - d) Right click, and select **Paste**.
 - e) Refer to [Template operations](#) on page 24 for further information.
4. To delete Message Content:
 - a) In the tree view locate the Message Content to delete.
 - b) Right click in the treeview area, and select **Delete**.
 - c) You are prompted to confirm deletion.
5. To rollback to a previous version:



- a) Select Message Content, and click the **Version** tab.
- b) Highlight the version to rollback to, and select the **Clock** icon.
This will copy that version, updating the template to the newest version. The process will be recorded in the **Comments** field.

To delete a version of Message Content:

Note: You can highlight more than one version. When you click Delete, all highlighted versions will be deleted.

6. To delete a version of Message Content:



- a) Select Message Content, and click the **Version** tab.
- b) Highlight the version you want to delete, and click the **Bin** icon.
Click **Delete All Previous Versions** to delete all versions prior to the selected version.

Active Content management



Public Active Content that is published independently must be imported into an Active Content folder. If Active Content was published independently in Designer it must be imported into an Active Content folder. Private Active Content (used by a single template design) is included when you publish resources for EngageOne. For more detailed information about Active Content and how it is used, see the "Designer User's Guide".

You can cut and paste Active Content from one Active Content folder to another folder. You cannot copy Active Content. You can also delete Active Content if it is *not* referenced by a work item or template. Active Content items that have been moved or deleted are shown in the history log. You can export all public Active Content from the content repository for diagnostic purposes as well as migrate to a new content repository. When you export public Active Content, you can select the Active Content to export (or all), and provide the fully qualified path and filename for the exported Active Content. The export process creates a valid Active Content zip file which includes creation of the Active Content metafile. You can work with document classes, access rights, delivery options, retention rules and versions associated with Active Content. The document class controls are disabled for Message Content. This is handled in the same way as for templates.



See [Working with templates](#) for details about these tabs: **General**, **Access Rights**, **Delivery Options**, **Retention Rules**, **Resources**, and **Version**.

Active Content options

1. To import Active Content:
 - a) Select the Active Content folder to store the Active Content.
 - b) Click + or right click in the treeview area, and select **Active Content**.
 - c) Refer to [Template operations](#) on page 24 and [Importing resources](#) on page 19 for further information.
2. To export Active Content:
 - a) Select the required Active Content from the tree-view.
 - b) Right click in the treeview area, and select **Export**.
 - c) The export process starts automatically, you are notified when complete.
 - d) Refer to [Template operations](#) on page 24 for further information.
3. To cut and paste Active Content:
 - a) In the tree view locate the Active Content to remove or move.
 - b) Right click in the treeview area, and select **Cut**.
 - c) Locate the Active Content folder that will receive the Active Content.
 - d) Right click, and select **Paste**.
 - e) Refer to [Template operations](#) on page 24 for further information.
4. Deleting Active Content
 - a) In the tree view locate the Active Content to delete.
 - b) Right click in the treeview area, and select **Delete**.
 - c) You are prompted to confirm deletion.
5. To rollback to a previous version:

  [Delete All Previous Versions](#)

 - a) Select the required Active Content from the tree-view, and click the **Version** tab.
 - b) Highlight the version to rollback to, and select the **Clock** icon.
This will copy that version, updating the template to the newest version. The process will be recorded in the **Comments** field.
6. To delete an Active Content version:

  [Delete All Previous Versions](#)

 - a) Select Active Content, and click the **Version** tab.
 - b) Highlight the version you want to delete, and click the **Bin** icon.
Click **Delete All Previous Versions** to delete all versions prior to the selected version.

Exporting Active Content

1. Select the Active Content to export.
2. Click **Edit**.
3. Click **Export**, and select a location for download from the Explorer window.
4. Click **Save**.

Cut and paste Active Content

1. From the **Templates** tab, in the tree view locate the Active Content to remove or move.
2. Click **Edit/Cut**.
3. Locate the Active Content folder to receive the new Active Content.
4. Click **Edit/Paste**.

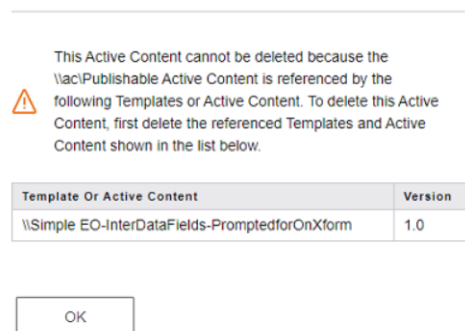
Deleting Active Content

To delete Active Content: locate the Active Content to delete, and select **Edit/Delete**.

If the Active Content is in use it cannot be deleted.

If the Active Content is referenced by a template or other Active Content, the following message box displays. "You cannot delete the selected Active Content until you delete the referenced items shown in the list." A similar message box displays if you delete a folder that contains Active Content referenced by a template or other Active Content.

Warning



Note: Where the Active Content is associated with an output device, the output device will not be removed if it has a delivery channel association.

Rolling back Active Content

1. Select Active Content, and click the **Version** tab.
2. Highlight the version to rollback, and click the **Rollback** button.

This will copy that version, updating the Active Content to the newest version. The process will be recorded in the **Comments** field.

To delete a version of Active Content:

Note: You can highlight more than one version. When you click Delete, all highlighted versions will be deleted.

Working with EngageOne Video

EngageOne can optionally be integrated with EngageOne Video so as to send communications that contain personalized video links.

To make use of this feature your connection to EngageOne Video must first be established.

Refer to the "*EngageOne Installation Guide*" for configuring the connection to EngageOne Video.

Then, EngageOne's use of specific video projects must be configured in EngageOne Video.

Provide the following information to software.support@precisely.com for each video project that your EngageOne installation is going to consume:

- environment - EngageOne Video environment. For example, prod, preprod, euprod.
- system - Name representing the EngageOne installation that will use the video project. For example, "EngageOne".
- category - (optional) Name that categorizes the video project. For example, "Promotional".
- validFrom - (optional) Start date and time when the video project will be available to EngageOne.
- validTo - (optional) End date and time when the video project will stop being available to EngageOne.
- urlPattern - Format of the URL that will be inserted into communications created by EngageOne.

The urlPattern must contain a token of the format {uid} indicating where the unique identifier that personalizes the video will be inserted.

For example, <https://preprod.rtcvid.net/billexplainer/embed-iframe.php?uid={uid}>.

- projectName - Name of the video project. For example, "BillExplainer".
- projectDescription - (optional) Description of the video project.

Once the video project usage configuration has been applied to EngageOne Video, template managers can select from a list of available video projects in the **Template Management** area of the Compose application. See "*Video management*" in the Compose User's Guide for details on how template managers can configure templates to contain personalized video links.

Video link tracking

If you are using data captured by EngageOne Video to report the video links that have been clicked, you should consider filtering out clicks made by EngageOne Interactive users when viewing or reviewing submitted Communications.

It is recommended that these clicks are removed from EngageOne Video statistics data by registering internal IP addresses in the `test_ips` project setting.

For more information, see "Project Settings" in EngageOne Video's Director application.

Working with EngageOne Communicate

EngageOne Server can optionally be integrated with EngageOne Communicate to compose responsive HTML communications via EngageOne Communicate templates using EngageOne Server high speed batch capabilities. EngageOne Deliver will typically be used for delivery.

For detailed information on EngageOne Compose/EngageOne Communicate configuration and integrated features, refer to: <https://support.precisely.com/products/engageone-compose/>.

To use this feature you must:

- have an EngageOne Communicate account in place. Contact your support team for details on account registration.
- create a connection to allow integration between EngageOne Server and EngageOne Communicate, refer to **Retrieve Communicate account credentials** on page 127 for details.

Use the **Add from EngageOne Communicate** and **Use the Add from EngageOne Communicate at Root** import option to make EngageOne Communicate templates available in EngageOne Server. Refer to **Template operations** on page 24 for further information.

4 - External File management

In this section

Working with external files.....	47
Working with images referenced by external key maps.....	48
Using the EngageOne Key Map Generator.....	49
Using key maps.....	50
Using lookup tables.....	55
Using RTF files.....	58



Working with external files

Images in the EngageOne environment can be handled in two ways.

- Images used directly in templates and managed inside Designer. These images need to be imported into the Designer environment, and are introduced into the EngageOne server via templates and HIP files. Once inside the EngageOne Server these images cannot be added, removed or updated.

Internally referenced images are prepared in **Publish Resources for EngageOne**. This produces the zip file ready to be imported into EngageOneAdministration.

- Images referenced in templates by external key maps.

Note that external keyed images are not supported with an eHTML device.

Key maps are used to dynamically map Designer keys to image resources, so that images may be updated without updating every associated template. During **Publish for EngageOne** the external key map reference is retrieved automatically. You can also use the wildcard feature to specify a location where all image files are retrieved.

Note: Do not delete or rename images or Active Content previously used by EngageOne. Import new resources into Designer using new names. Ensure key maps are updated where images are revised.

Lookup Tables are used to perform value substitution during document composition using a file containing key name and value pairs. Import lookup table files and index these tables on the EngageOne Server. This allows the files to be referenced and included in a Generate process for a template.

RTF files include external content from RTF documents during document composition. The entry point for that content is marked in the template by defining special variables. The variable is resolved as a reference to an external file. Import RTF files so these files can be referenced and included in a Generate process for a template.

Working with images referenced by external key maps

Note: If you want to manage external images and key maps in EngageOne, you will need to install the EngageOne Key Map Generator, a standalone Windows application provided with your EngageOne package. See the EngageOne Installation Guide for installation instructions.

The EngageOne Key Map Generator allows you to select a target print stream and allocate image formats to convert source images into target image formats. Note that the attributes for the target print stream selected in the tool should match the attributes for the output device selected in Designer. See the following example for a custom AFP print stream.

The target image metrics are read, and a Generate equivalent key map XML file is created. The Generate key map is zipped with image files that are ready to be imported through external file management to a valid location on the EngageOne server.

See [Using the EngageOne Key Map Generator](#) on page 49 for instructions on using this tool.

1. If you want to use the wildcard key map feature, edit the key map archive you created above using the instructions in [Using the wildcard key map feature](#) on page 53.
2. Use External File Management in EngageOne Administration to import the zipped file from the Key Map Generator. Any updates to key maps will add new key entries to the existing key map file, and copy the additional image files to the target location on the server. A template referencing a key map is set to **incomplete** if the key map does not exist, and **complete** if the referenced key maps and Active Content exist. A template containing references to key maps, or missing Active Content, cannot be used for document composition and is presented with the appropriate warning icon in the **Delivery Management/Resources** view.

Note that you must check the **External File Management** check box in **System Administration/ User Roles** to display the Images tab.

When a key map is created or imported, the system checks for templates that reference a key map of the same name, and updates those templates so that their status is no longer **incomplete**.

See [Using key maps](#) on page 50 for further details.

Using the EngageOne Key Map Generator

Use the Key Map Generator to create a key map for each external key map referenced by your templates. This will create images in all the formats you need, plus the key map file, and zip them use in the EngageOne environment. You then use the external file management function in EngageOne Administration to import the new key map.

Note: You must create image formats using the key map tool for all devices you have configured in Administration. If you create image formats for some of the devices you use, this may cause unexpected results when you try to publish to these formats.

Points to note when using the tool:

- The output directory must have write access permission.
- Maximum supported custom image size used for conversion is 23.4 inches x 33.1 inches (paper size A1).
- Supported input image types are: PSG, PSEG, PNG, FS10, F10, F45, FS45, GIF, GIFF, JPEG, JPG, BMP, EPS, TIFF and TIFF. Supported output image types are: PNG, PSEG, F45, JPG, BMP, EPS, PCL, GIF, and TIFF. The PNG image format will automatically be selected for every run.
- Key names in the key map are used to find the images. Key names are case sensitive. If the exact name is not used, the keyed image will not display. A warning will be entered in the log file, but there will *not* be a composition failure.
- DBCS file names are not supported.

To generate external keyed images for EngageOne: locate the .exe file for your language in C:\Program Files (x86)\EngageOne\EngageOne Compose\EngageOne Key Map Generator

- **Target Print Stream** – select the target print stream.
- **Use Default Image Formats** – if you leave this box checked, the appropriate image formats are automatically selected. If you uncheck this box you can select as required.
- **Input Directory** – select to convert all images contained in the directory.
- **Include Subfolders** – select to convert all images in the sub-folders of the directory.
- **Input File** – select to convert one image only.
- **Key Map Filename** – enter the name of the output XML file to be created.
- **Use image Defaults** – if you do not wish to use the image defaults, uncheck this box and enter the preferred image settings.

Using key maps

When a template or Active Content containing a key map is imported and the key map does not exist in the system for the current community, an empty key map will be created with the key map name. In this case, you will need to use the update key map option to import the keys to the key map, to define the target directory.

Note that an empty key map will count as non-existing when the check is made if a template or Active Content is complete.

External keyed images are supported for on demand, batch processing, and the Interactive Editor with either wildcard keys or non-wildcard keys. However, wildcard keyed images will not be displayed in Interactive Editor.

Key Maps

<input type="checkbox"/>	Name	Image Path
<input type="checkbox"/>	ExternalKeyMapBmp	c:\
<input checked="" type="checkbox"/>	ExternalKeyMapJpg	c:\

10 per page

1

Key Map list – displays a list of defined key maps for the current community. Refer to the image that follows and [List view operations](#) on page 17.

- **Name** – the key map name cannot be edited.
- **Image Path** – the path on your server where images are uploaded.

Note: these fields are for reference and cannot be edited.

KEY MAP

ExternalKeyMap.jpg

Name *(required)*

ExternalKeyMap.jpg

Target Directory *(required)*

c:\

Key Map Images

Key Name	Image Name
croxley	croxley
Samaritn	Samaritn

Save Cancel

Key Map details pane - displays information on the selected key map in the Key Map list. Refer to the image that follows.

- **Name** - the name can be edited. If the key map is referenced by a template, or Active Content, the link may be broken and the template or Active Content will become incomplete.
- **Target Directory** - if you change the target directory details, all images defined in this key map will be moved to the new target directory.
- **Key Name** - the name of the key value which dictates the image is to be selected for use in the template/active content.
- **Image Name** - the name of the image that is selected according to its **Key Name**.

Importing

To import a key map:

1. Click the + icon
Refer to [Importing resources](#) on page 19
2. Enter a unique key map name in the community. If you want to link the key map to a template, the key map name has to match the key map defined in the template master file.
3. Enter the target directory to store the image files. This does not have to be in the content repository, as long as it is accessible to the server. In a clustered environment, the directory should exist and be accessible to all server nodes, and should be a UNC path. For example, \\161.228.45.143\keymap. Make sure that the target directory exists when importing a key map.

If the target directory does not exist, an error message displays. Check the directory path and make sure it is correct.
4. Click **Import** to run the import process.

Updating

If the target directory is defined, it will be read only. The update will add additional key map keys and images to the selected key map.

To update a key map:

1. From the **External Files** tab the Key map tab is active by default
2. After highlighting the required key map in the list view, click the ↗ icon.
3. The **Update Key Map** dialog is displayed. You cannot change the key map name here.
4. Click **Choose a file** and select the required key map zip file. Alternatively you can drag and drop the key map zip file to the indicated area.
5. If the **Target Directory** is not defined you can enter where to store the image file.
6. Click **Import** to upload the selected zip file.

Using the wildcard key map feature

The wildcard feature provides all required images from a location, and passes in the image name to be pulled dynamically through the data using EngageOne Interactive or a custom application.

Note: Key words are case sensitive. Make sure you do not modify key words or delete attributes while editing the key map XML file.

To use wildcard feature in a key map archive:

1. Create the key map archive. See the instructions on [Using the EngageOne Key Map Generator](#) on page 49.

Note that you do not need all images in the input directory when creating the key map archive, but you must have at least one.

2. After creating the key map archive, open the archive and edit the key map XML file using an XML editor.
3. Locate the first `<KeyEntry>` element, and note the values for the attributes highlighted in this example:

```
<KeyEntry Key="Samaritn">
  <Image Width="354" Height="325" Resolution="72" >
    <EOImageDeviceInfo Device="DOC1InteractiveEditor"
      DeviceType="DOC1INTERACTIVEEDITOR" Resolution="72" ImageType="PNG"
      FileName="Samaritn.PNG" />
    <EOImageDeviceInfo Device="PDFNC" DeviceType="PDF" Resolution="72"
      ImageType="BMP" FileName="Samaritn.BMP" />
  </Image>
</KeyEntry>
```

4. Modify the first `<KeyEntry>` element:

```
KeyEntry Wildcard="%"
  FileName="%.PNG"
  FileName="%.BMP"
```

5. Add the `ResourceName` attribute to the `<EOImageDeviceInfo>` element for each device:

```
ResourceName="%"
```

6. Each device has an `<EOImageDeviceInfo>` element in the `<Image>` element. For each `<EOImageDeviceInfo>` element, the `FileName` and `ResourceName` attribute, as shown in steps 4ii, 4iii, and 5i.

If you are targeting a PDF device and using JPEGs, you must also add the `ColorSpace` attribute to the PDF `<EOImageDeviceInfo>` element. Without this keyword in the key map file, the PDF will contain the incorrect value `/ColorSpace /DeviceRGB` in the PDF output file. Value 2 indicates `DeviceRGB`, and 3 indicates `DeviceGray`. For example, `<EOImageDeviceInfo Device="Default PDF" DeviceType="PDF" ColorSpace="2" Embed="true" Disposal="Retain" ImageType="JPEG" ResourceName="" FileName="%.JPG" PDFPixelWidth="364" PDFPixelHeight="480" />`

7. Search for all remaining `<KeyEntry>` elements, and delete them.

The resulting key map XML file should be similar to this example:

```
<KeyEntry Wildcard="">

<Image Width="354" Height="325" Resolution="72" >

<EOImageDeviceInfo Device="DOC1InteractiveEditor"
DeviceType="DOC1INTERACTIVEEDITOR" Resolution="72" ImageType="PNG"
FileName="%.PNG" ResourceName=""/>

<EOImageDeviceInfo Device="PDFNC" DeviceType="PDF" ColorSpace="2"
Resolution="72" ImageType="BMP" FileName="%.BMP" ResourceName=""/>

</Image>

</KeyEntry>
```

8. Save the key map XML file.
9. Update the key map archive with the modified key map XML file.
10. Import the key map archive in EngageOne Administration. For more information, see [Using key maps](#) on page 50.

Using lookup tables

Lookup tables perform value substitution during document composition by using a file containing key name and value pairs. When a template or Active Content containing a lookup table is imported, and the lookup table does not exist in the system for the current community, an empty lookup table will be created with the lookup table name. In this case, use the **Update** lookup table option to import the table to the map, to define the target directory.

Note: An empty lookup table will count as non-existing if a template or Active Content is complete.

Lookup tables are supported for non-interactive templates used in on demand and batch processing. Lookup tables are not supported in the Interactive Editor (Designer 6.0 does not support lookup tables with interactive templates).

Lookup tables can be set up in EngageOne Administration, then managed on a regular schedule. You can manage lookup tables using an automated production process that defines the lookup table location, then manages table updates in your process.

Lookup Table list – displays list of defined lookup tables for the current community.

- **Name** – the name of the lookup table.
- **File Path** – the path on your server where the lookup table is uploaded.

Note: these fields are for reference and cannot be edited.

	Name	File Path
<input type="checkbox"/>	lut	c:\lut.txt
<input type="checkbox"/>	Lookup	c:\Non Interactive Lookup Table.lut

Lookup Table pane – displays information on the selected lookup table in the lookup table list. Refer to the image that follows.

- **Name** – the name can be edited. If the lookup table is referenced by a template, or Active Content, the link may be broken and the template or Active Content will become incomplete.
- **File Path** – if the target directory is changed, the lookup table will be moved to the new target directory.

LOOKUP TABLE

Lookup

Name *(required)*

Lookup

File Path *(required)*

c:\Non Interactive Lookup Table.lut

Save Cancel

Importing

To import a lookup table:

1. From the **External Files** tab, click **Lookup Tables**.
2. Click +. Refer to [Importing resources](#) on page 19

Enter a unique lookup table name in the community. If you want to link the lookup table to a template, the lookup table name has to match the lookup table name defined in the template master file.

3. Click **Choose a file** and browse for the lookup table zip file you want to import, click **Open**. Alternatively, you can drag your key map lookup table to the indicated area.
4. In **Target Path**, enter the path string, including the file name, where the lookup table file will be stored. This does not have to be in the content repository, as long as it is accessible to the server. In a clustered environment it should exist and be accessible to all the server nodes and it should be an UNC path, for example: \\161.228.45.143\lookuptables\example.lut.


If the target directory does not exist, an error message displays. Check the path to the directory and make sure that it is correct.

5. Click **Import** to run the import process.

Updating

To update an existing lookup table:

Note that if the target directory is defined already, it will be read only. The update will replace the existing file.

1. From the **Lookup Tables** tab, highlight the existing lookup table to which you want to update the associated file.
2. Click the  icon.

The **Update Lookup Table** page is displayed. You cannot change the lookup table name here.

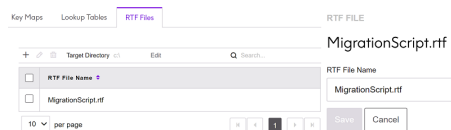
3. Click **Choose a file** and select the required lookup table file. There is no restriction on the file type extension used. Alternatively, you can drag your lookup table file to the indicated area.
4. Enter the **Target Path** where the lookup table is stored.
5. Click **Import** to upload the selected zip file.

Using RTF files

RTF Files are used to include external content from RTF documents during composition of a document. When a template, or Active Content that contains an RTF file is imported and the RTF file does not exist in the system for the current community, an empty RTF file will be created with the given RTF file name. In this case you will need to use the **Update** option to import the RTF file to the map to define the target directory.

RTF Files list – displays list of defined RTF files for the current community. Refer to the image below.

- **Name** – the name of uploaded RTF files.
- **Target Directory** – the path on your server where the RTF files are uploaded.



RTF Files details pane - displays information on the selected RTF file in the RTF File list. Refer to the image below.

- **Name** - the name can be edited. If the RTF file is referenced by a template, or Active Content, the link may be broken and the template or Active Content will become incomplete.

Importing

To import a RTF file:

1. From the **External Files** tab, click **RTF Files**.
2. Click +. Refer to [Importing resources](#) on page 19

Note: The first time you import an RTF file in a community you will be prompted for the **Target Directory** where all RTF files for this community will be stored. You can subsequently change the location by entering a new path and clicking **Edit**.

If the target directory does not exist, an error message displays. Check the path to the directory and make sure that it is correct.

Select the **Inherit name from file** check box to use local filename for the uploaded filename. Otherwise, enter a unique RTF file name within the community. If you want to link the RTF file to a template, the RTF file name has to match the RTF file name defined in the template master file.

3. Browse for the RTF file you want to import and click **Open**.

Updating

Note that if the target directory is defined already, it will be read only. The update will add the additional RTF file keys and RTF files to the existing map.

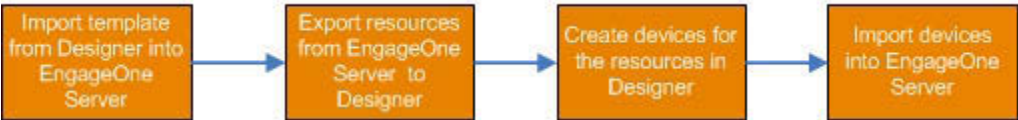
To update an existing RTF file:

1. From the **RTF Files** tab, highlight the existing RTF file to which you want to update the associated file.
2. The **Import RTF Files** dialog is displayed. You cannot change the RTF file name here.
3. Click **Choose a file** and select the required RTF file. RTF files can be of type .txt or .rtf. Alternatively, you can drag your RTF file to the indicated area.

5 - Delivery management

Devices are created in Designer and must be exported from Designer for importing into EngageOne.

The basic process flow is illustrated in the following diagram.



In this section

About delivery management.....	61
Output variables.....	62
Resources.....	67
Devices.....	74
Recipients.....	78
Compatibility modes.....	84
Delivery channels.....	85
Delivery options.....	118

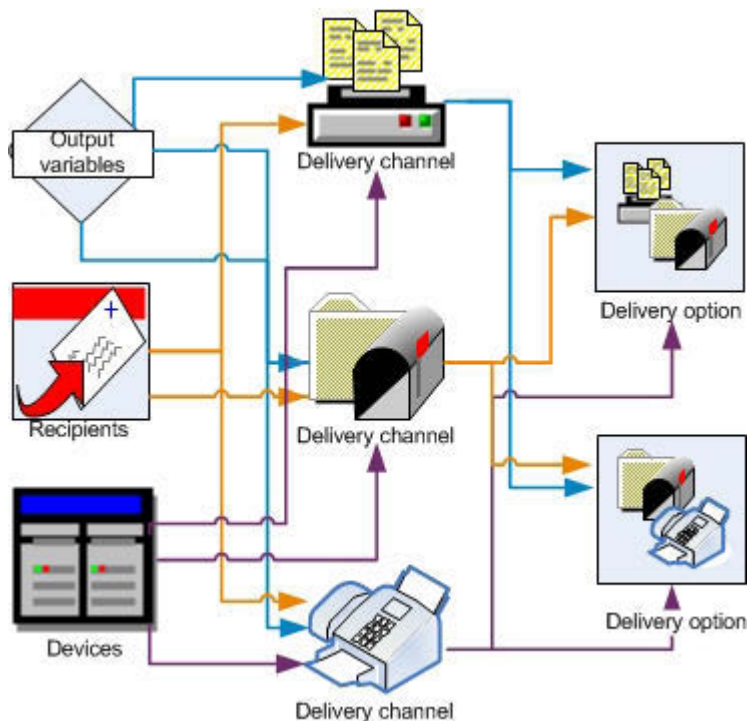
About delivery management

Once the devices have been received you can create delivery channels and delivery options. Delivery options define how a device should be used, for instance, they provide a method to group one or more delivery channels into a single delivery choice that can be selected by the front-office user. Supported delivery channels include; local print, batch print, archive, fax, EngageOne Deliver, and e-mail. This provides automatic carbon copy processing using recipients, meaning that the front-office user need only select one delivery choice and carbon copies can automatically be created and delivered.

Delivery management also requires output variables, which are used to pass data from the input data stream to the output infrastructure.

Note: EngageOne Deliver is the new name for EngageOne Digital Delivery/e-Messaging.

The diagram below shows the delivery channel architecture.



Output variables

Output variables are reserved keywords provided either by the system, or user defined and are resolved at run time.

Output variables are used in:

- print file paths and names
- report file paths and names
- archive file paths and names
- inclusion conditions
- sorting
- report file contents
- index file contents

Be aware that if you duplicate a variable definition within the same template, when batch processes the channel the last defined variable XPath value will take precedence.

Note that the XPath field is editable. However, do not attempt to enter the XPath value for a non-interactive template. Instead, use Browse to select the XPath value. For more information, see [Using non-interactive templates](#) on page 184.

Structured XML journal variables are automatically imported along with your template into EngageOne for use with batch reporting. These are listed as Reserved/Journal and consist of any variables defined in a Structured XML Journal object in Designer.

You can select the default system variables or use the displayed user defined variables instead.

It is important to note that documents created from EngageOne Communicate imported templates are generated directly in EngageOne Communicate. If you use these templates with JSON formatted data, you must define a separate set of variables because JSON data is automatically converted to our internal data format before output variables are calculated. Refer to [User defined variables](#) for further information on defining variables.

If you intend to use a template imported from EngageOne Communicate with both JSON data and data in other supported formats, e.g., XML, together with Output Variables when generating a document, e.g., at the channel level or BATCH configuration level, you should:

- duplicate the elements using an Output Variable and in one copy use only variables for JSON data and in the other copy use only variables for the other data formats.
- duplicate elements that use the above mechanisms, such as Delivery Option, and split them into those that use only variables for JSON data in one copy and only variables for other data formats in the other.
- use only elements with Output Variables definitions that match the format of the data used (JSON, XML, etc.) to generate documents.

System variables

System variables are provided with the EngageOne installation. They predefine placeholders in the template instance file and the variable name and data type are displayed in the list as **Reserved**. Footnoted system variables cannot be used as inclusion conditions in immediate delivery channels.

Note that you cannot change or delete a system reserved variable.

System Variable	Description	Delivery Channel
COMPOSITION_DATE*	Date the document was composed.	Batch/Archive/Email/FAX
DOCUMENT_OFFSET	Binary offset of the start of the document in the output stream.	Report File (structured journal)
DOCUMENT_PAGE_NUMBER	The current page number within the document being processed	Report File (structured journal)
FAX_NUMBER	The fax number of the recipient.	FAX
INCREMENT*	Sequence number variable to be used in reports, index files, and to ensure unique file naming when partitioning.	Batch/Archive
INTERACTIVE_DOCUMENT_SELECTOR	The document selector value triggers the inclusion and exclusion of one or more documents within a template for the given composition. This system variable can only be used with the interactive data model. For more information, see Input files on page 177.	EngageOne Deliver
INTERACTIVE_RECIPIENT	The value provided at composition time to help trigger logic in the template that varies the content based on the recipient value. This system variable can only be used with the interactive data model. For more information, see Input files on page 177.	Batch/Archive/Email/Print/FAX
JOB_PAGE_NUMBER	The current page number of the entire production job, starting from 1	Report File (structured journal)

System Variable	Description	Delivery Channel
MAIL_BODY	The text content of the e-mail.	Email
MAIL_FROM	The e-mail address of the sender.	Email
MAIL_SUBJECT	What the e-mail refers to.	Email
MAIL_TO	The e-mail address of the recipient.	Email
PAGE_OFFSET	Binary offset of the start of the page in the output stream.	Report File (structured journal)
PRINT_FILENAME*	Used to write the name of the generated print file to an archive index file. This may be required to support archive products such as FileNet Panagon or P8.	Batch/Archive/Email/FAX
SYSTEM_DATE	Date the batch job was run.	Batch/Archive/Email/FAX
SYSTEM_TIME	Retrieves the system time whenever it is needed, for example, job start time and job end time.	Batch/Archive/Email/FAX
TEMPLATE_INDEX	The sequence number of a publication within its final destination stream.	Report File (structured journal)
TEMPLATE_OFFSET	Binary offset of the start of the template in the output stream.	Report File (structured journal)
TEMPLATE_PAGE_NUMBER	The current template page number within the document being processed	Report File (structured journal)
TOTAL_PAGES*	Total number of pages counting front and back.	Batch/Archive/Email/FAX
TOTAL_PAGES_RECTO*	Total number of pages counting front only (sheet count).	Batch/Archive/Email/FAX
* These system variables are not a valid inclusion condition for immediate delivery channels.		

User defined variables

Create user variables that map to the data model of a template regardless of the data format that the template implements so that these variables can be used by the various processes that utilize variables on the EngageOne server. User-defined variables are any other variables that you need to create your output infrastructure. You must specify the generic XPath, which is the data location of the input file. Input files can be XML, keyed record, or delimited.

Note: User-defined output variables containing special characters (for example - / * ?? !) are omitted from reports.

The syntax is XPath-based regardless of the format of the input file. Interactive templates always follow the interactive data model path:

```
InteractiveDataModel/Publication/DataNode
```

The syntax for selecting data elements for keyed record and delimited data models will look something like this:

```
CUSTOMER01/ACCOUNT_NUMBER
```

If you want to override this generic path you must enter the template-specific XPath by selecting the template and entering the variable XPath name location. If you select an XPath for a specific template, the template selection is bypassed and the data elements are automatically provided. Be aware that if you duplicate a variable definition within the same template, when batch processes the channel the last defined variable XPath value will take precedence.

To add or modify a user defined output variable: from the **Delivery Management** tab, click **Output Variables**.

The screenshot displays the 'Output Variables' management page. On the left, a table lists existing variables. On the right, a form allows editing a variable named 'MyString'.

Variable Name	Data Type	Variable Type	Variable XPath
DOCUMENT_OFFSET	Number	Reserved	
DOCUMENT_PAGE_NUMBER	Integer	Reserved	
FAX_NUMBER	String	Reserved	//DeliveryInformation/FaxNumber
INCREMENT	String	Reserved	
JOB_PAGE_NUMBER	Integer	Reserved	
MAIL_BODY	String	Reserved	//DeliveryInformation/EmailBody
MAIL_FROM	String	Reserved	//DeliveryInformation/EmailFromAddress
MAIL_SUBJECT	String	Reserved	//DeliveryInformation/EmailSubject
MAIL_TO	String	Reserved	//DeliveryInformation/EmailToAddress
MyString	String	User	//InteractiveDataModel/Publication/String

Variable Details for 'MyString':

- Name:** MyString
- Generic XPath:** //InteractiveDataModel/Publication/String
- Data Type:** String
- Variable Type:** User
- Template Specific:**

Template	Variable XPath
Simple EO-InterDataFields-PromptedforOnXform	//InteractiveDataModel/Publication/Integer-Field

- To delete, select the required output variable in the list view and click on the Bin icon.
- Click + to open the right pane for entering new variable details.

- Click on an existing variable to show its current attributes in the right pane for modifying.
- When you click on **Generic XPath**, you are prompted to select a template first, then to select the field from the data dictionary. This will automatically return the proper XPath expression. Select the variable XPath location from the list for the template selected.
- If you want to override the generic XPath, click on the + icon to open the Template list. Note that only XPaths for elements with simple type definitions are shown. XPath with complex types and anyType are not visible.
- Click on the Bin icon in the **Template Specific** pane to delete a highlighted, template specific variable XPath.

Considerations for templates imported from EngageOne Communicate

If the selected template was imported from EngageOne Communicate, then the XPath data dictionary will contain both:

- XPath expressions for variables associated with EngageOne Communicate data in JSON format referenced by the template,
- and XPath expressions for common data formats such as XML, which can be used for any template imported directly from Designer.

To distinguish between them, it is sufficient to compare their syntax. XPath for ordinary data formats addresses elements by their names, while XPath expressions associated with EngageOne Communicate JSON data address individual elements based on the content of their attributes.

For example:

- non-EngageOne Communicate data XPath expression sample:

```
/Customer/FullName
```

- EngageOne Communicate JSON data-related XPath expression sample:

```
/InteractiveDataModel/Publication/field[@name='Customer']/field[@name='FullName']
```

If you want to create an output variable for a template imported from EngageOne Communicate that reads its value from a JSON data source, then note that the list of suggested XPath expressions in **User defined variables** on page 65 only contains entries for the data referenced by the imported template. If you want to address any other fields in the JSON data source with an output variable, you will need to manually create an XPath expression using the same naming convention as the example above, where the JSON data is prefixed with `/InteractiveDataModel/Publication` and the individual JSON data fields are addressed by the "name" attribute value, e.g.

```
field[@name='JSON_field_name'].
```

Resources

Resources are the fonts and images of an EngageOne template that are associated with a device. These resources are referenced by templates and contained within devices, which are then used together to generate your documents. Font and image resources are published from Designer with the templates and imported into EngageOne Server when you import those templates.

EngageOne templates published in Designer do not contain any external images, lookup tables, and RTF files (that are managed inside of Designer) for target output datastreams. External resources are imported and managed using the **External Files** tab. This architecture allows EngageOne to re-use previously deployed resources wherever possible.

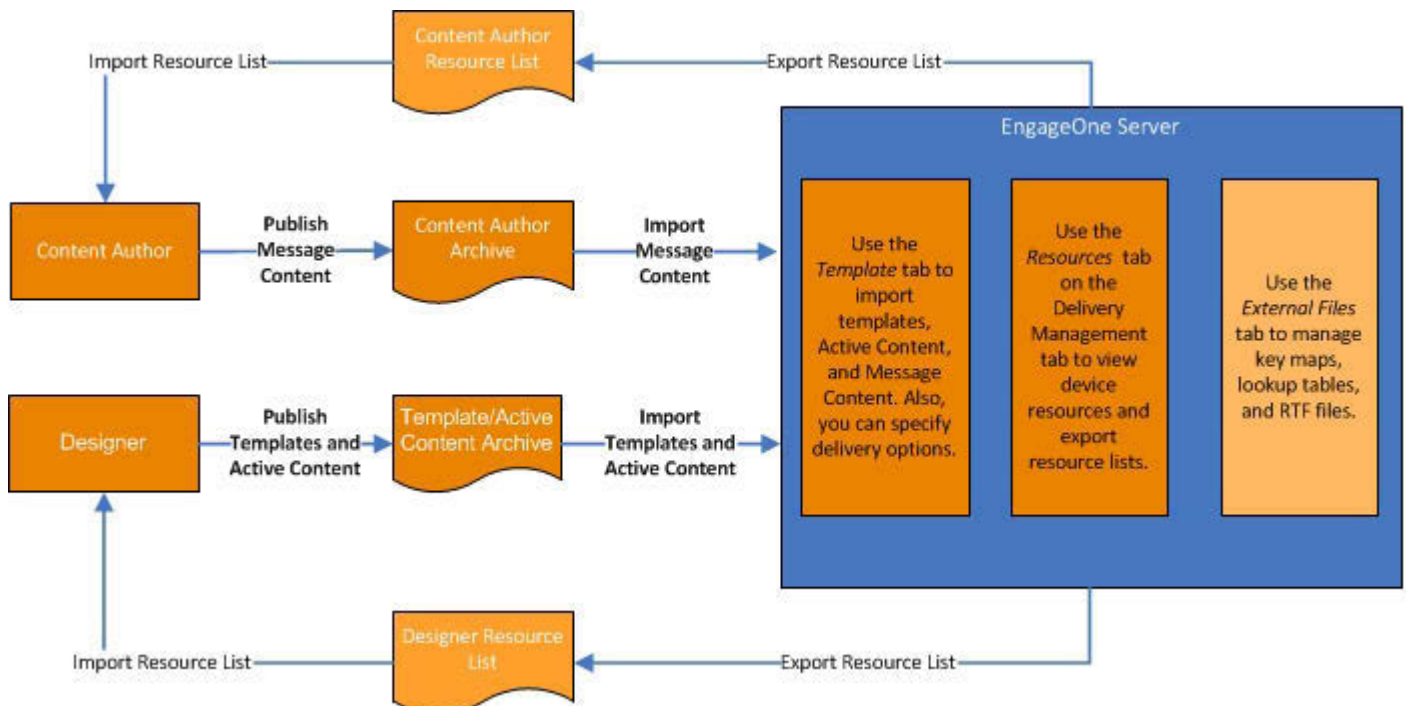
Note that referenced images, lookup tables, and RTF files in a template are handled differently. For more information, see [External File management](#) on page 46.

Topics in this section include:

- Managing resources
- Publishing new device resources
- Viewing resource properties
- Working with missing resources
- Exporting resources

Managing resources

Resource management requires coordination of the fonts and images used by templates, devices, Active Content, and Message Content. Templates, devices, and Active Content are defined in Designer. Message Content is defined in Content Author. This section explains how to manage these resources effectively. The following diagram shows the resource workflow among Designer, Content Author, and EngageOne.



When the template designer publishes a template, Active Content, or Message Content the device resources are automatically published with the templates using the device previously set up by the output device manager. Subsequent requests to publish a template only convert newly used resources that have not been converted from a prior publishing request. For more information about publishing resources from Designer, see [Devices](#) on page 74 and the Designer User's Guide.

Publishing new device resources

Occasionally, new devices are created and a template is published with the new device. When this occurs, you may encounter missing resources. If there are any missing resources, export the resource list from EngageOne Server and republish the device resources from Designer.

Note that if you upgraded from a previous version, make sure you publish a template or Active Content and import it into EngageOne Server before exporting a resource list back to Designer. This is particularly important if you have multiple repositories.

If a resource exists in multiple repositories, EngageOne Server updates the existing resource with any new information about the resource, including any additional repositories to which it is linked. When a resource is missing in multiple repositories, export the resource list for each repository and republish the device resources from Designer.

To avoid any missing resource problems, follow these steps:

1. Create the new device in Designer.
2. Publish the template from Designer and import it into EngageOne Server.
3. Use the **Export Resource List** option to export the resource list to a local file, which is then imported to Designer to resolve any missing resources. Note that exporting all of the resources and not just those of a selected template ensures that all devices created in Designer contain all resources for all templates in EngageOne. The **Resources** tab indicates if any resources are missing in a device as shown below. For more information, see [Working with missing resources](#).
4. Use the **Publish Resources For EngageOne** option in Designer, which allows the input file to be browsed and read and then prompts for the target output device definitions to publish to. A single ZIP file is created that contains all of the resources for the selected devices. This zip file is then imported into EngageOne Administration to create or update all devices installed on the server. The three publishing options are:
 - **Publish all resources**
 - **Publish missing resources**
 - **Publish for selected output device**
5. Import the device into EngageOne Server. For more information, see [Devices](#) on page 74.

Viewing resource properties

There may be times when you want to view the properties of a font or image. For example, if you have two images of the same name from a different (or even the same) repository, you can expand the image in the **Resources** tab to examine its properties.

Use **Export Resource List** to resolve any resource problems. For more information, see [Exporting resources](#) on page 72.

To view font and image resource properties:

1. From the **Delivery Management** tab, click the **Resources** tab.
2. Expand the **Fonts**, **Images** or **Advanced Features** item.

Export Resource List

Create Resource Report

Resource Type

Advanced Features

▲2 >

Fonts

▲75 >

Images

▲6 >

Image Name (6)	Image Details
bcbs.GIF	Checksum: 81d96ba33fc21ac4c960680161803a0f
bcgo4cd-cmyk-600.tif	Bit Depth: 8
ColinGnew.jpg	Format: 34
DogsJpg.jpg	Type: GIF (8 Bit RGB)
	Height: 60
	Resolution: 96
	Width: 120

Devices

Used By

⚠ Incompatible device

The following devices are incompatible with the resource.

- AFPDS 240 Bitmap Color (AFP240Bitmap)
- AFPDS 240 Bitmap Color (BarcodesAFP240bitmap)
- AFPDS 240 Outline Color (SEOINT-AFP)
- AFPDS 300 Outline Color (BarcodesAFP300OutlineColour)
- AFPDS 300 Outline Color (AFP300OutlineColour)
- eHTML (eh (ww))

Note: the image above illustrates the standard resources view.

3. Select **Advanced View** to switch between Advanced and standard resource views. The **Advanced View**, is tabular in format and presents information for the selected resource type, where each column in the table represent an output device. An orange triangular exclamation mark indicates that for Fonts/Images the resource are missing, and for Advanced Feature items that the feature is not supported.

Note: that the Advanced View option may significantly impact your browser performance and cause it to be unstable when using Internet Explorer 11. We recommend the latest version of Google Chrome for this option.

Selected type	Selected resources	Device 1	Device 2	Device 3	Device 4
Advanced Features	This Device does not support Unicode. Resource: Unicode Device: AFPDS 240 Bitmap Color (AFP240Bitmap)				
Resources					
Unicode		Missing Resource	✓		
Advanced Graphics		✓	✓		

For descriptions of each property, see [Template management](#) on page 20.

Working with missing resources

If any of the following are missing in a device, the Missing Resources icon displays in the Resources tab for any missing fonts and images. Additionally, the Templates tab displays these missing resources:

- Key map
- Active Content
- Message Content
- Look-up tables

Missing resources and EngageOne Interactive

As long as the Active Content, Message Content, and key maps are not missing, the template is considered complete and can be used by EngageOne Interactive and NA batch. However, delivery options that contain any device that are missing a device resource (such as a font or image) will not be usable for the template, so that delivery option will not be listed in the **Delivery Options** dialog box.

Missing resources do not necessarily mean that the template is unavailable to EngageOne Interactive. For example, you can have a template indicated on the **Resources** tab as having missing fonts, images and Active Content, but if the only thing missing for this Active Content is a device resource (that is, it has all of the referenced Active Content), then it is considered complete and will be available

to EngageOne Interactive. The document can be edited, but cannot be submitted for delivery until all of the resources are available.

Missing resources and Message Content

Make sure that the device resources used in your Message Content match the device resources used in your templates. If they do not, the Message Content will display missing resources.

Exporting resources

Devices are listed on the **Resources** tab by the device identifier with the device name in parentheses, for example **AFPDS 240 (afp240)**. Columns headings are sortable. The **Resources** tab shows the fonts and images used by each device that is defined in your system. If a resource required by the device is missing, it is indicated by a red Missing Resources icon.

The **Resources** tab also shows the advanced features of each device. Advanced features include support for Unicode and advanced graphics capabilities related to rotation and charts. For more information about these advanced features, see the Designer User's Guide.

To export resources to Designer:

1. From **Delivery Management**, click **Resources**.
2. The **Resource View** presents the EngageOne device resources in a tree view. You can navigate to the individual EngageOne resources, such as fonts and images, by clicking on the required item.
 - An orange triangular exclamation mark indicates that for Fonts/Images the resource is missing and for Advanced Feature items that the feature is not supported.. For details, see [Working with missing resources](#) on page 71.
 - Click **Show missing resources only** to filter the display to only show resources that are missing from one or more devices
3. To generate a resource list or resolve missing resources, click **Export Resources**.
4. The **Export Resource List** dialog box displays a list of repositories for which you can export a resource list

Export Resource List

Select a repository to export the resource list into.

Repository Alias	Database
 EO_SanityDB3	NCSRV2012R2U102\SQLSER...

☐ Show repositories with missing resources only



- Click **Show repositories with missing resources only** to filter the display to only show repositories that are missing from one or more devices.

5. Click **Export**.
6. Select the location for download in the Explorer window and name the file as required.
7. Click **Save**.

Resource reporting

Use resource reporting to create a file containing a list of all resources defined on your system.

1. From **Delivery Management**, click **Resources**.
2. Click **Resources Report**.
3. Select the location for download in the Explorer window and name the delimited file as required.
4. Click **Save**.

Devices

When you publish a template, all of the resources needed for that template (such as fonts and images) are included in the resultant ZIP archive. When you publish publishable Active Content, all of the resources that are needed for that active content such as fonts and images are included. Externally referenced resources such as external keyed images, lookup tables, and RTF files are not managed by this process and must be managed by an external process of your own design.

Note that keyed images are handled differently in EngageOne. For details see **“External Files management and EngageOne”** in the Designer User’s Guide.

When the completed template is imported into EngageOne, resources for that template are gathered by querying all the templates for the resources in the list. Using the **Delivery Management/Resources/Export Resource List** option in EngageOne Administration you can export template resources to a local file in XML format, which is passed manually to Designer.

The **Publish Resources for EngageOne** option in Designer allows the input file to be browsed and read and then prompts for the target output device definitions to publish to. A single ZIP file, which contains a ZIP archive for each selected device, is created that contains all the resources for the selected devices. This ZIP file is then imported into EngageOne Administration to create or update all devices installed on the server.

If publishing cannot complete because the requested file contains missing resources you must resolve them and then rerun the publishing process. Possible reasons could be that:

- Resources referenced by the templates were published from another Designer repository
- Resources are missing due to a database restore from an older backup
- Resources may exist, but on a different strand

Additionally, a check is done to ensure that the Designer repository used to publish the resources matches that of the template resources.

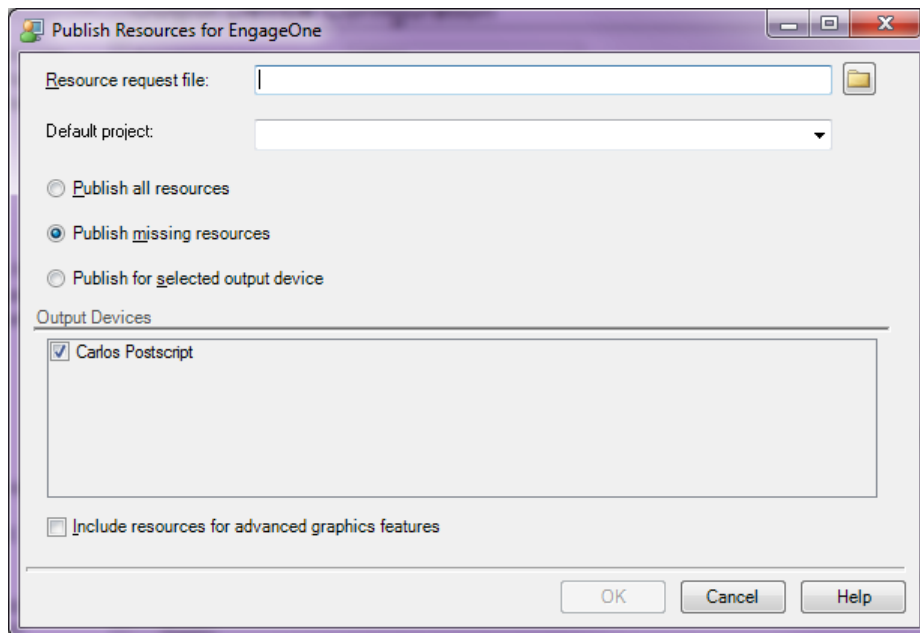
The basic process flow is illustrated in the following diagram.



However, because resources can be published along with the template, resources for a new device can be published without having to publish every template.

For example, if you have an EngageOne environment with an AFP device and a PDF device, you can introduce a PostScript device by exporting the resource request file from EngageOne Server and use **Publish resource for EngageOne** and select a PostScript device.

To publish resources for EngageOne: select the **Publish Resources for EngageOne** menu option. This displays the dialog box for selecting the output devices required.



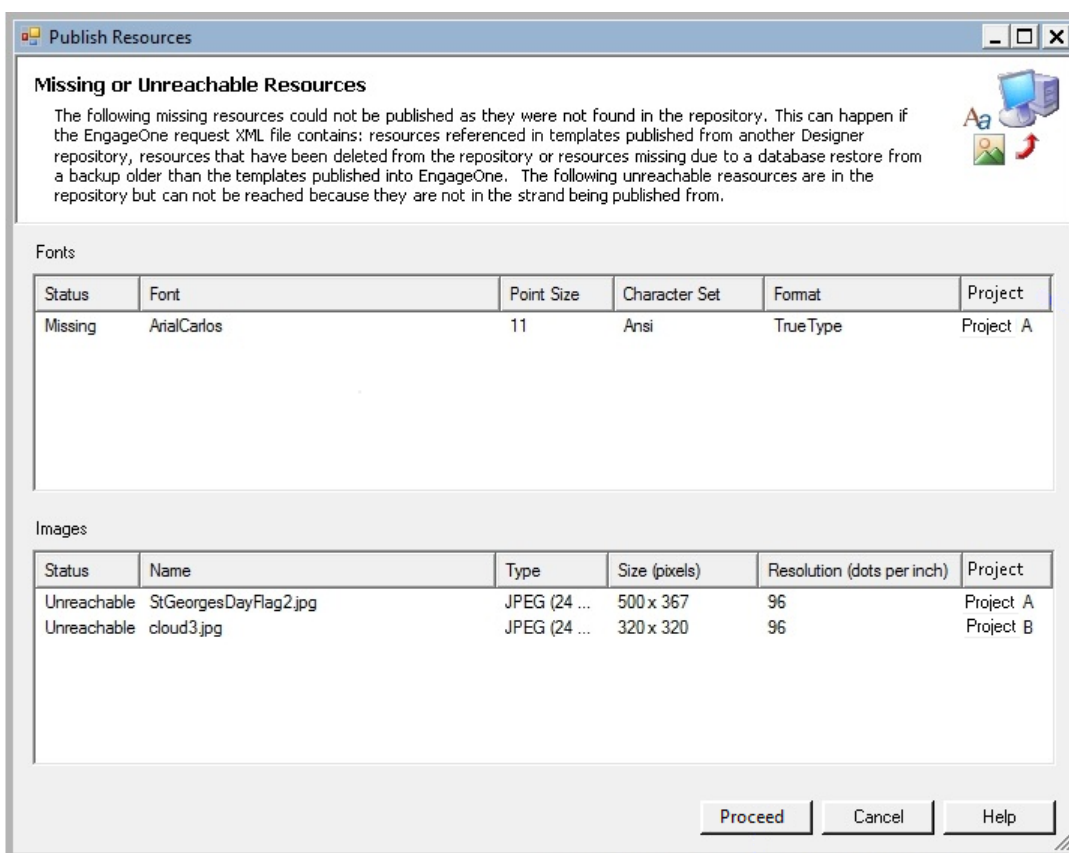
- Enter the name of the request XML file produced by the EngageOne application, or you can browse for it.
- Select one of the publishing options
- Select the default project to be searched for resources that can't otherwise be found.
- Select one or more EngageOne enabled output devices as required from the project specified above.
- If advanced graphics – such as rotated text boxes – are used then you must select the **Include resources for advanced graphics features** option.

The three publishing options are:

- **Publish all resources:** publish all of the resources requested in the EngageOne resource list for all of the output devices in the EngageOne resource list.
- **Publish missing resources:** publish only the resources that EngageOne resource list states it does not have for certain output devices.
- **Publish for selected output device:** publish all of the resources requested in the EngageOne resource list for the selected output device. This option is useful when a new output device needs to be added to EngageOne Server. The device is created in Designer and then the resources are sent back to EngageOne Server, which then adds the new device.

Click **OK**. Using the Explorer dialog box provided, name and save the file to the location specified by your EngageOne Administrator. A file package is created ready for importing into the EngageOne Server. For more information about delivery management see the EngageOne Administration Guide.

An example Publish Resources dialog:



If a resource cannot be found, you must resolve them and then rerun the publishing process. The **Missing Resources** dialog box lists all the fonts and images that cannot be found in the Designer repository but are present in the resource request file.

- The project column indicates where the image was originally published from.
- You can proceed with those resources that can be published. If a resource cannot be found, you must resolve them and then rerun the publishing process.

Managing devices in EngageOne

Devices are listed by the device identifier with the device name in parentheses, for example **AFPDS 240 Outline B/W (afp240)**. Columns headings are sortable. The **Import** device option in EngageOne Administration allows you to browse for a ZIP file to import the devices and related resources into EngageOne.

During the import process, EngageOne unzips the file and interrogates the metafile descriptor included in the ZIP file to determine the characteristics of the device.

These include:

- the unique name given to a device with particular configuration values
- the basic type of the device (AFP, PDF, etc.)
- the resolution of the device
- a list of all the fonts and images it supports (resource information).

If a new device is imported, any existing key maps as defined in **Images**, are updated to support the new device. If the device with the same name already exists in the system, the existing device will be updated with the new device file.

Be aware that devices must be updated using the **Import** option when new resources are used within a template. Failure to do so will result in preview and publish errors.

The **Export** devices option is used if you need to send device information to EngageOne Support for diagnostic purposes.

To manage devices: from the **Delivery Management** tab, click **Devices**.

The screenshot shows the EngageOne Server Admin interface. The top navigation bar includes 'EngageOne Server Admin', 'Templates', 'External Files', 'Delivery Management' (selected), 'Community Administration', and 'System Administration'. The left sidebar shows 'Output Variables', 'Resources', 'Devices' (selected), 'Recipients', 'Delivery Channels', and 'Delivery Options'. The main content area displays a table of devices:

Name	Type	Resolution	Preview
AFPDS 240 Bitmap Color (AFP240Bitmap)	AFPDS	240	<input type="radio"/>
AFPDS 240 Outline Color (SEOINT-AFP)	AFPDS	240	<input checked="" type="radio"/>
AFPDS 300 Outline Color (AFP300OutlineColour)	AFPDS	300	<input type="radio"/>
eHTML (eh (ww))	eHTML	96	<input type="radio"/>
eHTML (Leg_eHTML_All_Comp)	eHTML	96	<input type="radio"/>
eHTML (Fluid_eHTML_Split)	eHTML	96	<input type="radio"/>
eHTML (Leg_eHTML_Split)	eHTML	96	<input type="radio"/>
eHTML (SEOINT-eHTML)	eHTML	96	<input type="radio"/>
eHTML (Fluid_eHTML_All_Comp)	eHTML	96	<input type="radio"/>
eHTML (ehtmlLegacy)	eHTML	96	<input type="radio"/>

Below the table is a pagination control showing '10 per page' and page numbers 1, 2, 3. To the right, the 'Resources' pane is active, showing a list of resources:

Name	Type
Arial 11 pt	FONT
Arial Bold 11 pt	FONT
Arial Italic 11 pt	FONT
Arial Narrow 11 pt	FONT
Arial Narrow 12 pt	FONT
Arial Narrow Bold 11 pt	FONT
Baskerville Old Face 12 pt	FONT
Baskerville Old Face Bold 12 pt	FONT
Baskerville Old Face Italic 12 pt	FONT
bcgo4cd-cmyk-600.tif	IMAGE

The 'Used By' tab is also visible but not selected.

- The Import button allows you to browse and select the zip file for importing.
- The Export button allows you to select a location for downloading the selected device as a ZIP file containing the HIP and metafile files.
- The Resources pane gives information (based on the metafile) for a selected device.
- The Used By tab gives information (based on the metafile) for a selected device.
- Select a PDF device to use for the preview option in EngageOne Interactive.

About the Preview option

The Preview option allows you to preview a template using the selected output device in the EngageOne Interactive application. Note: this functionality is only available when the Interactive application is configured for cross-browser operation. You also need to ensure you have a compatible viewer installed in your environment to view the communication.

Clicking on the Preview option allows you to view **Resources** and **Used By** details.

Recipients

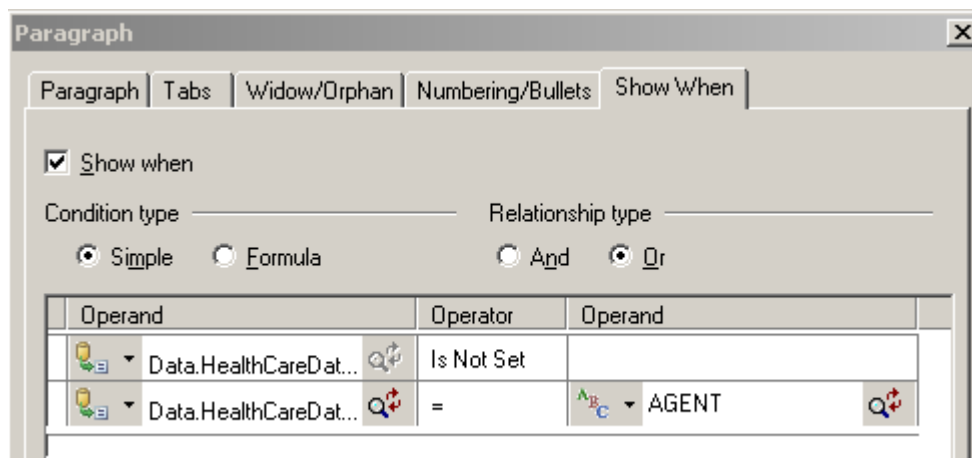
Recipients are used for carbon copy processing. Before you can use recipients to set up a delivery infrastructure you must define them. Recipient values must exactly match the values used within your document logic in the template design in Designer, for example, Agent and AGENT are not a match.

Recipient conditions in the template are used to produce non-unique content. For example, if you do not want the user AGENT to receive the content designated to INSURED then you must check a condition against the recipient field for the content in your template design: if the recipient field **Is Not Set** (front-office user needs to see everything), or is equal to AGENT.

You must then set the **Inclusion Condition** in the delivery channel to match.

Note that it is recommended that you use uppercase letters for recipient identifiers.

To set a recipient condition in Designer: from the **Format** menu option, click **Paragraph**. On the **Show When** tab of the **Paragraph** dialog box check the **Show When** option and enter the condition as required.



Select the **Show when** option to make the paragraph content conditional.

- Select the **Simple** condition together with the **Or** relationship type.
- Click to select the data field **Delivery Information/ Recipient** from the data dictionary.
- Select the comparison **Is Not Set** (means the front-office user will see everything in the document).
- Set the **Operator** equal to (=) and enter the name of the recipient in the **Operand** field.

For more details about conditional paragraphs and creating conditional expressions see the Designer User's Guide.

In order for the recipient processing to work, the recipients must be defined in EngageOne Administration after you define the recipient condition in Designer. The **Recipient** tab **XPath** field corresponds to the recipient data field defined in the Designer document logic, as shown in the next task.

To add a new or modify an existing recipient: from the **Delivery Management** tab, click **Recipients**.

Note that the XPath field is editable. However, do not attempt to enter the XPath value for a non-interactive template. Instead, use Browse to select the XPath value.

For more information, [Using non-interactive templates](#) on page 184

- Click + to activate the right pane for entering the new recipient.
- Enter or edit the name of the recipient.
- Click on an existing recipient to show its current name in the right pane for modifying.
- Enter an XPath or right-click to select an output variable.

Click **Browse** to select an XPath from a selected template.

How to create carbon copies

The recipient identifier can be used to create non-unique documents. For example, based upon conditional testing in the template on **Recipient** you can deliver:

- the agent address page for the agent copy only, and
- the payer address page for the payer copy only.

Sometimes a business requires user control over who receives a carbon copy. This can be achieved through the use of template logic and dummy delivery channels by:

- Defining the data fields with required properties in Designer – see using the Interactive Data Editor section in the Designer User's Guide.
- Inserting a CC block that will prompt the user for each copy they might want (if appropriate). This is done in the template in Designer based upon user selection – see the Designer User's Guide.
- Creating a data variable for each channel including the prompts set up in the previous step. See **Output variables** on page 62.
- Defining delivery channels with inclusion conditions using the data variables used above. Select the recipient based upon which copy you want to duplicate, or select no recipient.

Following is an example of how you might create user selectable carbon copies.

1: define data fields in the interactive data editor

1. From Designer, open a sample Interactive Data definition.
2. Add the required data fields and corresponding properties with prompts to your data model, for example:

Field	Properties
BrokerCopy	Type: String Prompt: Do you need a broker copy? Default: N: Y, N
PhysicianCopy	Type: String Prompt: Do you need a physician copy? Default: N Mandatory Choices: Y, N

3. Save and issue.
4. Update the dictionary and data map.

2: update your document to prompt for data

1. Open your publication in Designer.

2. Insert text after the signature block, for example:
Include only if BrokerCopy = Y or PhysicianCopy = Y
CC:
If BrokerCopy = Y Broker Copy
If PhysicianCopy = Y Physician Copy
3. Save your publication.
4. Test the change using **Preview for EngageOne** and validate that you are prompted for these new data fields, and that the selected data is inserted into the document.
5. Run the **Publish** option for **Interactive**.

3: define new output variables

1. Using EngageOne Administration, select **Delivery Management/Output Variables**.
2. Add new output variables with the following properties:

Variable Name	Type	Generic XPath
BROKER_COPY	String	MyInteractivData/Publication/BrokerCopy
PHYSICIAN_COPY	String	MyInteractivData/Publication/PhysicianCopy

4: add new delivery channels for your additional copies

1. Using EngageOne Administration, select Delivery Management/Delivery Channels and enter the following.

For the Broker copy channel:

Property	Value
Name	Broker Batch
Type	Print
Mode	Batch
Device	AFP240
Recipient	
File Path	C:\EngageOne\Output\Print Files\
File Name	BROKER_\${SYSTEM_DATE}.afp
Partitioning	0
Inclusion Condition	\${BROKER_COPY} = 'Y'

For the Physician copy channel:

Property	Value
Name	Physician Batch
Type	Print
Mode	Batch
Device	AFP240

Recipient	
File Path	C:\EngageOne\Output\Print Files\
File Name	PHYSICIAN_\${SYSTEM_DATE}.afp
Partitioning	0
Inclusion Condition	\${PHYSICIAN_COPY} = 'Y'

2. Save.

5: update delivery options create test documents and validate using batch

1. Using EngageOne Administration, select **Delivery Management/Delivery Options**. See **Delivery options** on page 118 for more details about working with delivery options.
2. Add the **Batch Archive** delivery channel to your **Batch Print** delivery channel.
3. Save.

Compatibility modes

Compatibility modes determine the processing options used to compose your final communication. When defining your delivery channel, you can associate the following compatibility modes:

- **Express Batch** - can improve NA batch performance; in this mode, NA batch skips additional configurations provided by Engage One Server during delivery channel definition, thus reducing I/O operations.
- **Default Batch** - allows for the full set of configuration options allowed during delivery channel definition.
- **Express Immediate** - is used in an On-Demand setting to deliver communications in real-time. This compatibility mode has limited configuration options to allow for increased performance.
- **Default Immediate** - allows for the full set of configuration options allowed during delivery channel definition.

It is important to be aware that although your delivery channel may be assigned to an 'Express' type compatibility mode; you must also take into account:

- The compatibility modes of the delivery channels that form the delivery option associated to your template.
- Whether the template has the **Express Batch** option set in EngageOne Designer.

Express Batch composition is in force when:

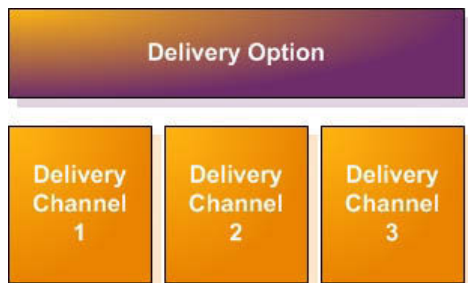
- The imported template is **Express Batch** enabled, refer to [Express Batch](#) on page 192 and the Designer User's Guide for further information.
- An **Express Batch** compatible delivery option is associated with the template. Refer to [Delivery options](#) on page 118 for further information.

Express Immediate composition is in force when:

- **Express Batch** is disabled for the imported template, refer to the Designer User's Guide for further information.
- An **Express Immediate** compatible delivery option is associated with the template. Refer to [Delivery options](#) on page 118 for further information.

Delivery channels

Delivery Channels are reusable and define how an output stream is constructed. One or more delivery channels can be allocated to a delivery option. This means the front-office user does not have to remember all the copies required when a document is submitted for delivery.



Supported delivery channels include:

- Local Print – intended to support delivery of a final, composed document back to the originator for printing on a local network printer. The action of printing the document locally is based upon the capabilities of the user's desktop.
- Batch Print – allows requests to be queued for later batch processing.

Note that batch delivery channels can also be used for non-accumulated batch. See [Non-accumulated batch](#).

- Archive – used to generate the appropriate print files (could be PDF) and indices to support writing the output to an archive product.
- EngageOne Deliver– used to integrate with EngageOne Deliver. EngageOne Deliver allows organizations to leverage EngageOne technology to embrace interactive e-mail and short message service (SMS) communications in their contact centers. That is, it allows end-customers to receive any communication they currently receive on paper via e-mail too, without the need for any redesign of content.
- E-mail – supports e-mailing a document as a PDF attachment to a single recipient and uses Java Mail API for delivery. Delivery management is provided by your e-mail provider. Also supports eHTML as a device, so that you can provide the e-mail body or an attachment from the template. For more information about eHTML, see [eHTML support](#). E-mail delivery channels cannot be used with batch processes, they must be set to immediate delivery.
- Fax – supports faxing of a PDF document. The default FAX implementation is a simple SMTP channel. If you have another fax provider see [Implementing a custom fax provider](#).

Do not use SPLIT as the Document Processing parameter when you create templates in Designer. Using SPLIT may cause the batch process to fail. Use the default COMPOUND parameter as the Document Processing setting for all templates used in EngageOne.

To add a new or modify an existing delivery channel: from the **Delivery Management** tab, click **Delivery Channels**.

DELIVERY CHANNEL

First delivery channel

Name

Recipient

Compatibility Mode
☒ Express Batch ☐ Default Batch ☐ Express Immediate ☐ Default Immediate

Device

Type



Print Inclusion Condition Sort Report Files

File path (required)

File name (required)

File partition number

Save Cancel

- Click + to activate the right pane for entering a new delivery channel.
- Click  to create a duplicate of the selected channel.
- Click  to create a duplicate of the selected channel.
- Click on an existing channel to display and modify the current settings.

Delivery channel properties

Name

Delivery channel properties

Type	<p>Select one of the following from the drop-down list: Print, E-mail, Fax and Archive. For a local Print select the Immediate mode. To send a Print file for batch processing select the Batch mode. The fully qualified path and name of the print file to be generated and print file names can contain variables such as counters, dates, literal values or values from the XForm data. In Batch mode there are other requirements to be set – see EngageOne batch for details.</p>
	<p>E-mail properties include; From Address, To Address, Subject, and Message Body. All properties can be defined using a combination of literal values and XPath values. All mail channel properties can also contain output variable inputs, literal value inputs, or values from a document instance XML identified by its XPath.</p>
	<p>Fax properties include Fax Number, Sender Name, Receiver Name, and Message description. All properties can be defined using a combination of literal values and XPath values. A fax document template should be created with a fax cover sheet. Use Recipient processing to control when the fax cover sheet is included in the package of properties of the fax delivery node. All fax channel properties can also contain output variable inputs, literal value inputs, or values from a document instance XML identified by its XPath.</p>
	<p>Archive is used to generate the appropriate print files and indices to support writing the output to an archive product. For archive there are other requirements to be set. For more details see Archive definitions.</p>
Compatibility Mode	<p>Your delivery channel can be assigned one of the following modes:</p> <ul style="list-style-type: none"> • Express Batch • Default Batch • Express Immediate • Default Immediate <p>Note that a Batch print channel allows requests to be queued for later batch processing. See Batch processing for more details. Immediate processing is when a document is generated immediately as a result of a self service/portal type action which invoke calls to EngageOne Web services.</p> <p>It is important to note that the Express Batch and Express Immediate compatibility modes will generally provide improved performance but have limited processing and Device/Type options compared to their Default versions.</p>

Delivery channel properties

Device	Each channel is tied at least to one device – see Devices .
Recipient	For carbon copy processing enter a recipient from the drop-down list. For details see Recipients .

EngageOne batch

If a channel is defined with a delivery mode of **Batch**, you create the properties on the channel and run the batch process against that channel. The batch process takes the channel name as input and exports all documents that have been composed for the given batch channel/batch job, concatenates the output into one or more files, sorts the output, partitions the output files, creates report files/audit files and prints them. The channel definition is maintained by the EngageOne server. For more information about batch see [Batch processing](#) on page 167.

The properties of a print batch channel include:

- Print file path and name.
- Inclusion Conditions – defines when a document is to be included in a batch stream, using output variables. For example, include testing on sheet count or testing on a variable that indicates if the document has a check or not. For a list of valid system variables that you can use in inclusion conditions, see [System variables](#).
- Sort – defines the ordering of the documents in the print file and any associated report or index files.
- Report files – defines the type of report required. System or user-defined variables are used to record activity when the job processes. These can help with things such as post-processing requirements, audit trail and archiving functions.

Configuring batch print properties

To configure batch print properties:

1. Click **Delivery Channels/Batch Mode/Print** to open the print file path and name dialog.
2. Enter a path name that is appropriate for your server setup. The path can contain variables and literal values. – Or – right-click on the box for a list of output variables. Click on the line required and the correct variable syntax will be entered for you.

3. The **File Name** can consist of a combination of literals and output variables and should include the `INCREMENT` system variable if partitioning is used.
4. If you used the `INCREMENT` system variable in the **File Name**, enter the **File Partition Number**. File partitioning is typically used in support of finishing equipment and sets the page count to include in a given print stream (without breaking a document). For example, if you want each

output record of the output document to be its own file, then set **File Partition Number** to 1. If you want the output document partitioned so that there are 100 output records in each file, set **File Partition Number** to 100.

Configuring batch inclusion conditions

To configure batch inclusion conditions:

1. Click **Delivery Channels/Batch Mode/Inclusion Condition** to open the **Inclusion Condition** field.
2. Enter an inclusion condition. You can use any output variable to determine if a document is included in the stream

DELIVERY CHANNEL

PDF300_Barcodes_Batch_Lnx

Name Recipient

Compatibility Mode
☐ Express Batch ☒ Default Batch ☐ Express Immediate ☐ Default Immediate

Device Type

Print **Inclusion Condition** Sort Report Files

Inclusion Condition

Note that you can right-click on the box for a list of output variables supplied by the system. Click on the line required and the correct variable syntax will be entered for you.

When entering your own inclusion conditions the following syntax rules must be applied:

Simple expression:

`{X} <> 10`

Complex expression:

`{X} 10 and {X} < 20`

Grouping:

`{X} 10 and not ({X} = 20 or {X} = 30)`

Note that in the above examples, X is the output variable.

Supported relational operators:	<, >, <=, >=, =, <>
Supported logical operators:	and, or, not
String literals:	enclose in quotes
Numeric values:	should NOT be enclosed in quotes
Date values:	use YYYY-MM-DD format and enclose in quotes
Time values:	use HH:MM:SS format and enclose in quotes, for example: \${Time} = "13:59:00"
Note that in the above examples quotes are used to avoid conflict with the operators (and, or, not, etc.) and dates with numbers. Quotes can be double or single.	

Configuring batch sort specifications

To configure batch sort specifications: click **Delivery Channels/Sort**.

DELIVERY CHANNEL

eHTML

Name: eHTML Recipient: RCP_1

Compatibility Mode:
☐ Express Batch ☒ Default Batch ☐ Express Immediate ☐ Default Immediate

Device: PDF 300 (Default PDF) Type: Print

Print Inclusion Condition **Sort** Report Files

+

Sort Field Name	Order By
\$(TheName)	Ascending

- Sort fields are included with field braces `${MYFIELD}`.
- Sort by any number of output variables in ascending and descending order.
- Sort by any type of output variables: Date, Integer, Number or String.
- When using output variable of Date type, the provided values have to be in 'YYYY-MM-DD' format to be properly sorted by.
- It is important to note when using output variables to sort, the value passed to an output variable will always be converted to the datatype defined for the output variable; this value will then be used in the sort.

Configuring batch report file specifications

To configure batch report file specifications:

1. Click **Delivery Channels/Batch Mode/Report Files**. You can define three types of report files:
 - Delimited – specify the delimiter then provide a list of output variables and literal values to include in your report.
 - Fixed width – requires, the field or literal value.
 - XML reports – one of its many uses is to create a Document Interchange Journal (DIJ) formatted file that records information about the documents within a Designer-generated output datastream.

When you specify a new target output file, or click on an existing one for editing, this opens further options for recording activity at template, document, or page level. Reports are normally generated within a publication, but you can also generate them at the start and/or end of the job.

It is important to note that for an XML report, you specify the root node, which is the wrapper tag, then you specify the node for the report level by clicking on the appropriate tab; **Template, Document, Page, Job Start, Job End**. Then you provide name and value pairs, the name being the element name and the value an output variable or a literal value.

2. Click the Plus button to add a line for entering the path. You can right-click the **Value** field for a list of output variables.
3. Click on the line required and the correct variable syntax will be entered for you. The list will include structured XML page offset and other key variables as well as any user defined variable elements that were configured in Designer. For more information about using structured journals please refer to the Designer User's Guide.

DELIVERY CHANNEL

eHTML

Name: Recipient:

Compatibility Mode: ☐ Express Batch ☒ Default Batch ☐ Express Immediate ☐ Default Immediate

Device: Type:

Print Inclusion Condition Sort **Report Files**

File Path	File Name	File Format
C:\temp\SYSTEM_DATE	report_SINCREMENT.csv	Delimited

Save Cancel

- Specify the report file path as appropriate.
- Use a combination of literal values and output variables as appropriate.
- You can re-size the columns as required.
- Select as required from the drop-down list; **Delimited**, **Fixed Width**, **XML**.

Using batch delivery options

Following is an example of how you might enhance batch delivery channels for a sample template to include sort criteria, inclusion conditions and report files.

1: add data variables for use in your delivery channels

1. From Designer, open an interactive data definition for your sample template.
2. Add the following data fields and corresponding properties to your data model:

Field	Properties
Color	Type: String Prompt: Select a color Mandatory Choices: Red, Blue, Green
Check	Type: String Prompt: Include a check? Mandatory Choices: Yes, No
Attachments	Type: String Prompt: Do you want any attachments? Mandatory Choices: Yes, No

3. Save and issue the document.
4. Update the dictionary and data map.

2: update your document to prompt for data

Data prompts are not available unless they are used in the document. To see if output processing is successful, add some text to each document footer.

1. Open your template in Designer.
2. Insert the following text into the footer of your document: `Selected color: [Color]`
`Selected check: [Check]` `Selected attachments: [Attachments]`
3. Save your template.
4. Test the change using **Preview for EngageOne** and validate that you are prompted for these three new data fields and the selected data is inserted into the document.
5. Select the **Publish** option for **Interactive**.

3: define new output variables

Next you define the output variables and map them to the new data fields.

1. Using EngageOne Administration, select **Delivery Management/Output Variables**.
2. Add three new output variables with the following properties.

Variable Name	Type	Generic XPath
COLOR	String	MyInteractivData/Publication/Color
CHECK	String	MyInteractivData/Publication/Check
ATTACHMENTS	String	MyInteractivData/Publication/Attachments

4: create your delivery channels

1. Using EngageOne Administration, select **Delivery Management/Delivery Channels**. See [Delivery channels](#) on page 85 for more information about working with delivery channels.
2. Create a delivery channel with the following properties:

Property	Value
Name	Insured Batch
Type	Print
Mode	Batch
Device	AFP240
Recipient	INSURED
File Path	C:\EngageOne\Output\Print Files\
File Name	INSURED_\${SYSTEM_DATE}.afp
Partitioning	0

3. Add a sort condition as `$(COLOR)ASC`.
4. Create a report file with the following properties:

Field	Value
File Path	C:\EngageOne\Output\Report Files\
Type	INSURED_\${SYSTEM_DATE}.csv
Format	Delimited
Delimiter	,
Fields	\${SEQ_NO} \${COLOR} \${CHECK} \${ATTACHMENTS} \${TOTAL_PAGES} \${TOTAL_PAGES_RECTO}

5. Create a delivery channel with the following properties:

Property	Value
Name	Agent Batch
Type	Print
Mode	Batch
Device	AFP240
Recipient	INSURED
File Path	C:\EngageOne\Output\Print Files\
File Name	AGENT_\${SYSTEM_DATE}.afp
Partitioning	0

6. Add a sort condition as \${COLOR} ASC .

7. Create a report file with the following properties and save:

Field	Value
File Path	C:\EngageOne\Output\Report Files\
Type	AGENT_\${SYSTEM_DATE}.csv
Format	Delimited
Delimiter	,
Fields	\${COLOR} \${CHECK} \${ATTACHMENTS} \${TOTAL_PAGES} \${TOTAL_PAGES_RECTO}

5: create your delivery option

Once you have your delivery channels defined, they must be associated with a delivery option in order to be used.

1. Using EngageOne Administration, select **Delivery Management/Delivery Options**. See [Delivery options](#) on page 118 for more details about working with delivery options.
2. Create a new delivery option with the following properties and save:

Property	Value
Name	Batch Print
Included Channels	Insured Batch

6: import your template and add your new channel

1. Select **Template Management**.
2. Import your updated template into the same folder as your original version if used.
3. Modify the available delivery options to include your new delivery option **Batch Print**.
4. Click **OK** to save.

7: create a document instance

Use EngageOne Interactive to create a document instance.

1. Log in to EngageOne Interactive.
2. Create six documents with the following properties:

Color	Check	Attachments
Red	Y	Y
Red	N	N
Green	Y	Y
Green	N	N
Blue	Y	Y
Blue	N	N

3. Submit each document, selecting **Batch Print** for the delivery option.
4. Approve each document.

8: run accumulated batch

Now run the batch job to process the documents and validate the output.

1. Open a command prompt using `Start/run/cmd`
2. Change directories to `<Batch bundle installation folder>/bin`
3. Execute the following command: `run-channel [domain] 'Agent Batch'`
4. If the batch was successful you will receive a message telling you the number of records read and the number of records processed.
5. Validate you have output by locating your print file in `c:\EngageOne\Output\PrintFiles`
6. Execute the following command: `run-channel [domain] 'Insured Batch'`
7. If the batch was successful you will receive a message telling you the number of records read and the number of records processed.
8. Validate you have output by locating your print file in `c:\EngageOne\Output\PrintFiles`
9. Validate the contents of each print file by double-clicking on the files. This will launch the AFP viewer for you.

9: add inclusion conditions

Now add some inclusion conditions to limit the output written to your print files.

1. Using EngageOne administration select **Delivery Management/Delivery Channels**.
2. Open the delivery channel **Agent Batch**.
3. Add an inclusion condition `${COLOR} = 'Red' and ${CHECK} = 'Yes'`
4. Save.
5. Open the delivery channel **Insured Batch**.
6. Add an inclusion condition `${COLOR} = 'Red' and ${CHECK} = 'Yes'`
7. Save.
8. Recreate the same 6 documents created in [7: create a document instance](#) on page 98.
9. Re-run your batch jobs as defined in [8: run accumulated batch](#) on page 98.
10. Validate that your output now only contains records where the color is Red and check is Yes.

Creating output for Archive

Following is an example of how you might create output for archive.

Step 1: create the archive channel

1. Using EngageOne Administration, select **Delivery Management/Delivery Channels**.
2. Add a new delivery channel, for example, with the following properties and click **OK**.

Field	Value
Name	Batch Archive
Type	Archive
Format	Vault
Print File Format	AFP
File Path	C:\EngageOne\Output\Report Files\
File Name	AGENT_\${SYSTEM_DATE}.csv
Format	Delimited

Delimiter	,
Fields	\$(COLOR) \$(CHECK) \$(ATTACHMENTS) \$(TOTAL PAGES) \$(TOTAL PAGES RECTO)

Step 2: update the delivery option

1. Click **Delivery Management/Delivery Options**.
2. Add the **Batch Archive** delivery channel to your **Batch Print** delivery channel.
3. Click **OK** to save.

Step 3: create some test documents

1. Using EngageOne Interactive, create documents with the following properties.

Color	Check	Attachments
Red	Y	Y
Red	Y	Y
Red	N	N

2. Submit each document, selecting Batch Print for your delivery option.
3. Approve each document.

Step 4: run batch

Run batch for the delivery channels associated with the delivery option.

Archive definitions

Archive is used to generate the appropriate print files (could be PDF) and indices to support writing the output to an archive product. EngageOne supports creation of index files for the Vault environment or if you use **Generic**, for any other vendor's archive system. Indices are used by the archive system to index the included documents for user search and retrieval. Note that an archive system could support print files, or split PDF files.

If you want to ingest EngageOne interactive documents into Vault.

To configure archive index specifications:

1. Click **Delivery Channels**.
2. Click **New**.
3. Select **Type/Archive**.
4. Select **Index**.
5. Enter the archive file path name using variables as appropriate
6. Select **Vault** or **Generic** as a **File Type**. Views will change depending on your selection – see Archive index settings below.

The screenshot shows the 'First Archive' configuration page. It has a 'Name' field, a 'Product' dropdown, and a 'Description' field. Below these are checkboxes for 'Document Type' (Interactive, Interactive, Interactive, Interactive). There are also fields for 'File Path' and 'File Type'. At the bottom, there is a table with two columns: 'Element Name' and 'Value'. The table contains two rows: 'ACCTNO' with a value of '20000' and 'DATE' with a value of '0000'. There are also buttons for 'Save' and 'Cancel'.

Archive index file settings

Specific Vault **File Type** options

Job Name

The name you want to identify the job with.

Document type

The name you want to identify the document type with.

Archive index file settings

Element Name / Value

Provide a name and value pairs, the name being the **Element Name** and the **Value** an output variable or a literal value.

You are able to map data to the standard Vault indexes like, account number, date, customer name and address lines. To support custom indexing, you can add any number of element names such as **DocID** and **Iguid** with their associated values.

Specific Generic **File Type**

When you specify a **Generic** file type, this opens further options for recording activity at template, document, or page level as well as at the start and/or end of the job.

Note that for an XML report, you specify the root node, which is the wrapper tag, then you specify the node for the report level by clicking the appropriate tab; **Template, Document, Page, Job Start, Job End**. Then you provide name and value pairs, the name being the element name and the value an output variable or a literal value.

XML **Format** options

Root Node

Specify the root node, which is the wrapper tag.

Node

Select the level you want to report on by clicking the required tab; **Template, Document, Page, Job Start, Job End** and then specify the appropriate node as per your selection.

Each node is optional, however, if you define a **Document** or **Page** node and **Template** node is empty, you must still define an empty **Template** node in order for the report output to function correctly.

Element Name

Specify the element name.

Value

An output variable or a literal value for the report level selected.

Delimited **Format** options

Delimiter

Specify the delimiter.

Column

Select the level you want to report on by clicking the required tab; **Template, Document, Page, Job Start, Job End** and then define the position of the item in the archive file, for the report level selected.

Archive index file settings

Value	An output variable or a literal value for the report level selected.
Fixed Width Format options	
Column	<p>Select the level you want to report on by clicking the required tab; Template, Document, Page, Job Start, Job End and then define the position of the item in the archive file, for the report level selected.</p> <p>Length – the column length</p>
Value	An output variable or a literal value for the report level selected.

To configure archive print specifications:

1. Click **Delivery Channels**.
2. Select **Type/Archive**.
3. Select **Print**.
4. Enter a path name that is appropriate for your server setup. The path can contain variables and literal values. – Or – right-click on the box for a list of output variables. Click on the line required and the correct variable syntax will be entered for you.

You may use output variables in a path name but typically it would be combination of literal values and output variables.

Note that you may use output variables in a path name but typically it would be combination of literal values and output variables.

5. The **File Name** can consist of a combination of literals and output variables and should include the `INCREMENT` system variable if partitioning is used.
6. Enter the **File Partition Number** if file partitioning should be used. File partitioning is typically used in support of finishing equipment and sets the page count to include in a given print stream (without breaking a document). For example, if you want each output record of the output document to be its own file, then set **File Partition Number** to 1. If you want the output document partitioned so that there are 100 output records in each file, set **File Partition Number** to 100.

Batch runs to Vault

The steps that follow must be completed prior to ingesting documents into Vault:

Note that this is a basic guideline only. For more detailed instructions refer to the appropriate Vault Guide.

To configure the environment:

1. Ensure that the Vault configuration has been modified to ingest the following files:

- **Profiles.ini**
- **Database.ini** (if being used)

For detailed information about modifying these files please refer to the Vault User's Guide.

2. Ensure that the EngageOne output device and delivery channel are configured correctly.

- You must have a delivery channel configured for **Archive**.
- **Archive** must be set to **Batch** mode.

For detailed information on configuring output devices and delivery channels refer to [Archive definitions](#) on page 101.

3. Ensure that the transfer mechanism (FTP or other script) is configured.

- The files must be routed to the **hot folder** in Vault Windows default path:
Vault\Server\download.

To run batch for Vault:

4. Run accumulated batch for the Archive delivery channel. This creates an accumulated print stream file, a resource file and an accumulated index file.
5. Transfer the files created in step one from the accumulated batch process to the Vault **hot folder**. You can view documents as they complete ingestion.

AFP resources

Note that when working with AFP files you must follow these steps before the files are transferred to Vault to ensure the availability of resources:

6. Open for Edit the `profiles.ini` file for the Vault Application.
7. Add the line `ExtractResources=distrib\Default` to the profile section used for EngageOne.
8. Users with potentially conflicting resource sets may extract to separate directories specifying:

```
ExtractResources=distrib\resourceset1 ResourceSet=resourceset1
ExtractResources=distrib\resourceset2 ResourceSet=resourceset2 . . .
ExtractResources=distrib\resourcesetn ResourceSet=resourcesetn
```

EngageOne Deliver configuration

EngageOne integrates with EngageOne Deliver allows organizations to leverage EngageOne technology to enable interactive e-mail and short message service (SMS) communications in their contact centers. That is, it allows end-customers to receive any communication they currently receive on paper via e-mail too, without the need for any redesign of content. EngageOne Deliver is a separately licensed product and additional EngageOne configuration is required to enable the product integration.

When the EngageOne Deliver type is selected from the **Type** list, Delivery Channels displays a six-tab data entry section to collect the appropriate values for this delivery channel. There are two supported EngageOne Deliver types: e-mail and SMS. If the selected outbound profile is of type **Email**, then the **Email Body** and **Attachments** tabs are enabled along with the **Outbound Profile** and **General** tabs. If the selected outbound profile is of type SMS, only the **Outbound Profile** and **General** tabs display.

Devices are selected when you configure the e-mail body and attachments. For e-mail body, only eHTML devices display in the device list.

The screenshot displays the EngageOne Deliver configuration interface. On the left, a table lists delivery channels. The 'Deliver_Batch_Channel' is selected. The right pane shows the configuration for this channel. The 'Outbound Profile' tab is active, showing settings for Profiles, Content Location, Description, Message Type, Digitally Signed, Archive, Archive Format, Download Folder, Vault URL, Insert Message, Beacon, and Content Verification.

Name	Type	Compatibility Mode	Device
<input checked="" type="checkbox"/> Deliver_Batch_Channel	Print	Express Immediate	PDF 300 (Default PDF)

10 per page

DELIVERY CHANNEL
Deliver_Batch_Channel

Name: Deliver_Batch_Channel Recipient: [Dropdown]

Compatibility Mode:
☐ Express Batch ☒ Default Batch ☐ Express Immediate ☐ Default Immediate

Type:
 EngageOne Deliver

Outbound Profile | General | Email Body | Attachments | Inclusion Condition | Report Files

Profiles: Immediate

Content Location: /mnt/active-drive/_emsg/emsg_two_shared/vendors/EngageOne/OutProfiles/Immediate

Description: For Immediate Delivery.

Message Type: HTML Email

Digitally Signed: Off

Archive: On

Archive Format: Original Parts only

Download Folder: C:\VaultDownloads

Vault URL:

Insert Message: Off

Beacon:

Content Verification: Off

Save Cancel

- Click **New** to activate the right pane for entering a new EngageOne Deliver delivery channel.
- Enter the new EngageOne Deliver delivery channel **Name**, or you can change an existing e-mail channel name here.
- The list of profiles changes depending on which mode is selected. See below for important **Poll for DIJ** settings.
- Select **Type/ EngageOne Deliver** from the drop-down list.

- Use the **Outbound Profile** tab to select one of the outbound profiles defined in the EngageOne Deliver service.
- Click on an existing EngageOne Deliver channel to display and modify the current settings.
- Use the **General** tab to enter e-mail address and subject information or SMS phone number and message text.
- Use the **Report Files** tab to define reports to create for the stream.
- Use the **Email Body** tab to define if the email body is provided, and if so whether it originates from a template or static content. This tab only displays when an e-mail outbound profile is selected.
- Use the **Attachments** tab to select and import e-mail attachments. This tab only displays when an e-mail outbound profile is selected. Use this tab to determine if a document is included in the stream.

EngageOne Deliver configuration overview

This section explains the high-level tasks to perform to successfully configure EngageOne Deliver requests in EngageOne.

1. Configure the startup settings using the information received from the EngageOne Deliver administrator. For more information, see the **EngageOne Installation Guide**.
2. Configure the Outbound Profiles in EngageOne Deliver. For more information, see the Outbound Profiles section in the **EngageOne Deliver User's Guide**.
3. Identify document selector values used in the template using the information received from the template designer.
4. In Delivery Channels, configure an EngageOne Deliver type delivery channel.
5. For Interactive Data created in a previous version of Designer, a new, required **DocumentSelector** field is automatically added to the **Delivery Options** node. You must save the Interactive Data before closing to update it correctly. Also make sure you update your Data Map and Data Dictionary accordingly. See the **Designer User Guide** for more information.
6. To run EngageOne Server and EngageOne Deliver in batch mode, configure EngageOne Server to process the EngageOne Deliver delivery channel in batch mode as described in this guide. After batch runs, the data will be present in the applicable vendor folder on the EngageOne Deliver server.

To complete the batch processing, run `e-Messaging.bat <HTML_Batch_Folder>` on the EngageOne Deliver server. For example if your HTML batch folder is named `HTML_Email_Batch`, and EngageOne Deliver is installed at `C:\Program Files (x86)\EngageOne\EngageOne Compose\EO_Deliver`, you would run:

```
C:\Program Files (x86)\EngageOne\EngageOne
Compose\EO_Deliver\e-Messaging.bat HTML_Email_Batch
```

Creating a delivery channel for EngageOne Deliver e-mail outbound profiles

There are six items to configure for an EngageOne Deliver e-mail outbound profile:

- Outbound profile selection
- General settings for the from address, to address, and subject line of the e-mail
- Email Body settings for the source of the e-mail body

- Attachment settings for the source of any attachments to the e-mail
- Inclusion condition to determine if a document is included in the stream
- Report Files settings to define reports to create for the stream

To configure an EngageOne Deliver delivery channel for an e-mail outbound profile:

1. From the **Delivery Management** tab, click **Delivery Channels**.
2. From the **Delivery Channels** tab, click **New**.
3. In the **Type** list, select **EngageOne Deliver**.

4. In the **Profiles** list, select an e-mail profile.
5. Click the **General** tab.

Use the **General** tab to enter the **From Address**, **To Address** and **Subject** information. These fields can be literal text, variables, or both. Validation is performed on the **From Address** and **To Address** fields to ensure valid e-mail address format.

Right-click in any field to insert an output variable. For more information about using variables, see [Output variables](#) on page 62.

6. Select the **Use Vault to archive messages** check box to set up the Vault index fields for archiving email messages. Enter a value for each element name that you want to use as an index field. The ACCTNO and DATE fields are required. Right-click in any **Value** field to insert an output variable. Click the + icon to add a new element. To delete an element, select it and then click the – icon.

For more information about EngageOne Deliver email archiving, see the **EngageOne Deliver User's Guide**.

EngageOne Deliver configuration form, General tab. The form includes fields for From Address, To Address, Subject, and a table for Custom Headers. The 'Template will provide an attachment' checkbox is checked.

Custom Header	Value
ACCTNO	5
DATE	23/06/2019
Name	Value

7. Click the **Email Body** tab.

The **Static content provides Email Body** is the default value and is required for the **Template will provide an attachment** option to be selected in the Attachments tab. Note that when the **Template contains Email Body** is selected, the **Template will provide an attachment** is optional.

EngageOne Deliver configuration form, Email Body tab. The 'Template contains Email Body' radio button is selected.

8. Click the **Attachments** tab.

The **Attachments** tab provides two tabs for defining attachments that will be included for this delivery channel. Attachments can be provided from the template and from static content.

Templates can provide only one attachment. Static content can provide more than one attachment. The maximum number of attachments that can be defined for a delivery channel is dictated by the outbound profile selected for the delivery channel. For more information on outbound profile limitations, see the EngageOne Deliver User's Guide.

The **Template will provide an attachment** check box indicates that an attachment should be generated from the template. This check box is cleared by default. Leave this check box cleared if attachments will only be provided by the static content.

EngageOne Deliver configuration form, Attachments tab. The 'Template Provides Attachment' tab is selected, and the 'Template will provide an attachment' checkbox is cleared.

9. To provide the attachment from the template, click the **Template Provides Attachment** tab and select the **Template with provide an attachment** check box.

DELIVERY CHANNEL

eM001

Name: eM001 Recipient: [dropdown]

Compatibility Mode: ☐ Express Batch ☐ Default Batch ☐ Express Immediate ☒ Default Immediate

Type: EngageOne Deliver [dropdown]

Outbound Profile: General Email Body **Attachments** Inclusion Condition Report Files

Template Provides Attachment Static Content Attachments

☒ Template will provide an attachment

Attachment Name: [text field]

Document Selector Value: [text field]

Document Selector XPath: [text field] **Browse**

Device: PDF 300 (Default PDF) [dropdown]

COMPOSITION_DATE
COMPOSITION_TIME
DOCUMENT_OFFSET
DOCUMENT_PAGE_NUMBER
FAX_NUMBER
INCREMENT
JOB_PAGE_NUMBER
MAIL_BODY
MAIL_FROM
MAIL_SUBJECT
MAIL_TO
MyString
PAGE_OFFSET
PRINT_FILENAME
SYSTEM_DATE
SYSTEM_TIME

RECENT

None [text field]
None [text field] **Browse**

Cancel

- Right-click in the **Document Selector Value** or **Document Selector XPath** fields to insert an output variable.
- Click **Browse** to use the XPath dialog box to define the path.
- Enter the **Attachment Name**. The attachment name is the value that will be used for the attachment in the output. The attachment name can be a combination of text and output variables. If this field is empty, template name will be used as the attachment name.
- Enter the **Document Selector Value**.
- Enter the **Document Selector XPath** or click **Browse** to select the XPath from a list of values.

Note that the XPath field is editable. However, do not attempt to enter the XPath value for a non-interactive template. Instead, use Browse to select the XPath value. For more information, see [Using non-interactive templates](#) on page 184.

- In the **Device** list, select the device you want to use for this delivery channel. All devices defined in this community are listed.

10. To provide the attachment from static content, click the **Static Content Attachments** tab.

Static content can be one or more files that you select or import from a document store or one or more runtime expressions that get resolved at runtime to select the attachments from a document store.

- Click **+** and select one of the following:
 - **Select** to display **Static Content Selection** dialog box. Use this dialog box to select one or more files stored on the server or import additional static content. For more information, see [Selecting static content](#).
 - Click **Define** to display the **Runtime Expression Definition** dialog box.

Static File Name Expression

Use **Static File Name Expression** dialog box to define the attachment as a combination of one or more variables and text strings. The location can be either imported static content or a file system path. File system paths must be absolute paths and can be any valid file path that the server can access, including UNC paths.

Right-click in the **Expression Value** text box to display a list of output variables. For more information about variables, see [Output variables](#) on page 62. Click **OK** to return to the **Delivery Channels** tab.

- Select one or more items from the list and click to delete items from the delivery channel. Removing items from the list does not delete static content from the server.

11. Click **OK** to change the delivery channel settings.

Selecting static content

Use the **Static Content Selection** dialog box to import static content, organize static content in folders, update static content and delete static content. This dialog box is accessed from the **Email Body** tab and **Attachments** tab of an EngageOne Deliver delivery channel.

To create a folder: From the Static Content Select dialog box, click **New Folder**.

To import static content: From the Static Content Select dialog box, click **Import**. Then, click **Browse**, locate the content you want to import, and click **Create**.

To update static content: From the Static Content Select dialog box, select a static content item from the list and click **Update**. The name of the item displays in the **Name** field of the **Import Content**

dialog box. Then, click **Browse**, locate the content you want to import, and click **Create**. The contents of the new static content will replace the existing contents.

To delete static content: From the Static Content Select dialog box, select a static content item from the list and click **Delete**. Click **OK** to confirm. The static content item is deleted from the list. Deleting static content from the list also deletes the static content from the server.

To define static content intended for an email body: When providing a ZIP archive that will be used as the content for an email body on the EngageOne Deliver channel, the ZIP archive and contents must adhere to the following rules:

1. The HTML file that encapsulates the email body content must be in the root folder of the archive and have an `.html` extension.
2. Any images provided in that archive must be placed into an "image" sub-folder within the archive.
3. Any image references within the HTML should be fully qualified references to image resources provided by your webpresence. For example,
`src="http://my-company.com/xyz/Images/my-image.png"`

This ensures proper control of the referenced image asset.

4. Any image references within the EngageOne Deliver that are not external references, but are intended to reference images provided within the archive, must reference only the image name and not include any relative sub-folder paths, for example `src="my-image.png"`.

Creating a delivery channel for EngageOne Deliver SMS outbound profiles

There are four items to configure for an EngageOne Deliver SMS outbound profile:

- Outbound profile selection
- General settings for the phone number and SMS message
- Inclusion condition to determine if a document is included in the stream
- Report Files settings to define reports to create for the stream

To configure an EngageOne Deliver delivery channel for an SMS outbound profile: From the **Delivery Management** tab, click **Delivery Channels**.

1. From the **Delivery Channels** tab, click **New**.
2. In the **Type** list, select **EngageOne Deliver**.
 - Use the **Outbound Profile** tab to select one of the outbound profiles defined in the EngageOne Deliver service.
 - Use the **Report Files** tab to define reports to create for the stream.
 - Use the **General** tab to enter e-mail address and subject information or SMS phone number and message text.
 - Use the **Inclusion Condition** tab to determine if a document is included in the stream.
3. In the Profiles list, select an SMS profile.
4. Click the **General** tab.

Enter the **Phone Number** and **SMS Message** information. These fields can be literal text, variables, or both.

Right-click in any field to insert an output variable. For more information about using variables, see [Output variables](#) on page 62 of the EngageOne Administration Guide.

5. Select a linedata device from the **Device** list.
6. Select the **Use Vault to archive messages** check box to set up the Vault index fields for archiving SMS messages. Enter a value for each element name that you want to use as an index field. The ACCTNO and DATE fields are required. Right-click in any **Value** field to insert an output variable. Click the + icon to add a new element. To delete an element, select it and then click the – icon.

For more information about EngageOne Deliver email archiving, see the EngageOne Deliver User's Guide.

E-mail and fax

The e-mail and fax channel configurations support input fields for data such as sender address, receiver address, subject, etc. E-mail delivery channels cannot be used with batch processes, they must be set to immediate delivery.

Fax delivery facilitated by the EngageOne server provided with the EngageOne installation does not deliver a real fax, instead it uses SMTP internally. To be able to plug in your own fax system to the EngageOne system please see [Implementing a custom fax provider](#) on page 113. For more information about writing delivery information, see the **Additional features** chapter in the EngageOne Programmer's Reference Guide.

Note that when XPath's are required within a dialog you can right-click on the field for a list of output variables supplied by the system. Click on the line required and the correct variable syntax will be entered for you.

To configure an e-mail delivery channel: from the **Delivery Management** tab, click **Delivery Channels**.

- Click **New** to activate the right pane for entering a new e-mail delivery channel.
- Select **Type/ Email** from the drop-down list, then enter the new e-mail delivery channel **Name**, or you can change an existing e-mail channel name here.
- If you want to override the default XPath's for the e-mail details shown, click **Browse** to search for a specific **Template/XPath**.
- Click on an existing e-mail channel to show its current properties in the right pane for modifying.
- Select **Template contains Email Body** or **Template Provides Attachment**.

To configure a fax delivery channel: from the **Delivery Management** tab, click **Delivery Options**.

Note that the XPath's field is editable. However, do not attempt to enter the XPath's value for a non-interactive template. Instead, use Browse to select the XPath value. For more information, see [Using non-interactive templates](#) on page 184.

- Click **New** to activate the right pane for entering a new fax delivery channel.

- Select **Type/ Fax** from the drop-down list, then enter the new fax delivery channel **Name**, or you can change an existing fax channel name here.
- Click on an existing fax channel to display and modify the current settings.
- To override the default XPath for the **Fax Number** shown, click **Browse** to search for a specific **Template/XPaths**. See [Implementing a custom fax provider](#) on page 113 below.
- The default fax provider address is supplied. If you have your own fax solution see [Implementing a custom fax provider](#) on page 113 below.

Implementing a custom fax provider

The EngageOne server supports the ability to interface with any custom fax solution. If you have a fax provider that you wish to interface with, you can use a Java programming interface to supply the communication between the EngageOne server and your fax provider.

EngageOne comes with a sample custom provider implementation which e-mails attachments via SMTP. This sample merely serves as an example that a customer could use to implement their custom integration. The source code for the sample can be found in the **jar** located at

`ENGAGEONE_HOME/samples/fax-delivery-channel/target/fax-sample.jar`

The following steps outline how to implement and integrate your own custom solution.

To implement a custom fax provider:

1. Implement the interface:

```
com.pb.engageone.server.tds.core.delivery.fax.FaxFactory.
```

The client library that contains the interface, which should be used for your development is located under

`ENGAGEONE_HOME/samples/fax-delivery-channel/target/fax-sample.jar`.

This interface has one method called `getFax()`, which is responsible for creating a new `com.pb.engageone.server.tds.core.delivery.fax.Fax` instance.

2. Implement the `com.pb.engageone.server.tds.core.delivery.fax.Fax` interface.

This class is the actual fax provider. There is a single method in this class, which is called when a document must be delivered via fax:

```
void send( String faxNumber, String from, String to, String description,
String documentName, InputStream documentStream, RequestChannel
requestChannel) throws FaxException;
```

3. Package your classes into a jar and place that new jar on the server in a location to which the EngageOne application has access. For example, you could place it under:

`ENGAGEONE_HOME/composition/conf/fax`

4. In order for the EngageOne server to load your custom fax factory, you must update the EngageOne configuration file located at:

`ENGAGEONE_HOME/composition/conf/config-settings/config-settings.xml.`

The following shows an example of the `com.pb.engageone.server.fax` namespace:

```
<namespace name="com.pb.engageone.server.fax">
  <setting>
    <key>fax.factory</key>
    <value>com.pb.engageone.sample.fax.FaxFactoryImpl</value>
  </setting>
  <setting>
    <key>fax.factory.classpath</key>
    <value>fax-sample.jar</value>
  </setting>
  <setting>
    <key>fax.factory.classpath.basedir</key>
    <value>ENGAGEONE_HOME/composition/conf/fax</value>
  </setting>
</namespace>
```

In the above example, the name of the custom fax factory class is:
`com.mycompany.engageone.fax.FaxFactoryImpl.`

The jar containing the fax factory and fax implementation classes is:

`fax-sample.jar`

and the location of the `fax-sample.jar` is under:

`ENGAGEONE_HOME/composition/conf/fax.`

Sources for sample fax channel are available in build .zip file in the:

`samples/fax-delivery-channel` directory.

Note: if you require third-party dependent jars for your fax solution, you can include them in the `fax.factory.classpath` setting above, each separated by semi-colons (Windows) or colons (Unix). Alternatively, you can add them to the `META-INF/MANIFEST.MF/Class-Path:` setting in `fax-sample.jar`.

eHTML support

EngageOne supports the delivery of documents composed with the eHTML device. This device is supported in the following delivery channels:

- Archive delivery in batch mode.
- Email delivery in immediate mode. Additionally, you can:
 - Specify that the composed HTML will be sent as the body of email messages so that these messages can be produced and sent with HTML rich text format.
 - Specify that the composed HTML will be sent as an attachment so that email bodies can follow the original text body format provided by EngageOne.
 - Create a custom email implementation that delivers EngageOne-composed HTML rich text.
- FAX delivery in immediate mode.
- Print delivery for immediate and batch modes.

To define a delivery channel to use an eHTML device, enable the device for EngageOne in Designer. Thereafter, when you import a new template or import the device, it is available in the **Delivery Channels** tab.

Here is an example of the two email options:

Template contains the email body

DELIVERY CHANNEL

First delivery channel

Name

First delivery channel

Recipient

Compatibility Mode

☒ Express Batch ☐ Default Batch ☐ Express Immediate ☐ Default Immediate

Device

PCL TrueType 300 Color (PCL0ColorTrueType)

Type

Print

Print

Inclusion Condition

Sort

Report Files

File path (required)

\\test\doc

File name (required)

testdoc

File partition number

0

Save

Cancel

Template provides the attachment

Name	Type	Compatibility Mode	Device
PS72_Barcodes_immed	Print	Default Immediate	PDF 300 (Default PDF)

10 per page

DELIVERY CHANNEL

eHTML

Name: eHTML Recipient: [dropdown]

Compatibility Mode: ☐ Express Batch ☐ Default Batch ☐ Express Immediate ☒ Default Immediate

Device: PDF 300 (BarcodesPDF) Type: Email

Email Inclusion Condition Report Files

From Address (required): /InteractiveDataModel/Publication/DeliveryInformation/EmailFromAddress [Browse]

To Address (required): /InteractiveDataModel/Publication/DeliveryInformation/EmailToAddress [Browse]

Subject: /InteractiveDataModel/Publication/DeliveryInformation/EmailSubject

Message Body: /InteractiveDataModel/Publication/DeliveryInformation/EmailBody

[Save] [Cancel]

If the template contains an image, the associated eHTML driver returns the output file in ZIP format. This is a default behavior, and can be changed by modifying the `ehhtml.output.zip.image.files` parameter in the `config-settings.xml` file.

The default setting for this section is presented in the example below:

```
<setting>
<key>ehhtml.output.zip.image.files</key>
<value>true</value>
</setting>
```

If an HTML only response is required, this setting needs to be set to false. If necessary, the setting has to be modified independently for both immediate and batch delivery channels.

For immediate delivery channels, it is necessary to modify the setting in the `config-settings.xml` file present in:

```
<EngageOneInstallPath>\composition\conf\config-settings
```

It is recommended that the composition service be restarted if this setting is modified.

For batch delivery channels, the parameter should be modified in the `config-settings.xml` file present in:

```
<EngageOneInstallPath>\batch\conf\config-settings
```

Note: it is important to note that changes to `ehhtml.output.zip.image.files` have no effect on the EngageOne Deliver channel, the output is always zipped as this is only format acceptable by EngageOne Deliver.

Delivery options

A delivery option is a named group of one or more delivery channels. You can configure a delivery option that will be made available to front-office users as one, user-selectable delivery choice. When this choice is selected, the delivery channel(s) linked to it will also be distributed.

You can use any name for a delivery option, like Batch Print, Local Print, Insured Print, Agent Print, Archive, etc.

Delivery channels must first be defined – see [Delivery channels](#) on page 85 . You can allocate multiple delivery channels to a delivery option, thus providing automatic carbon copy processing.

You can export a delivery channel for diagnostic purposes or use it to migrate to another environment such as test or production and import it back again if required. When you export a delivery option, only the variables used within that option are exported.

Note that if you import a delivery option and there are any templates missing that reference it, you will be notified with details of which template is missing and its relative XPath – see the following example.

The missing template(s) will be indicated by a blank row within the **Template Specific – Variable XPath** list in the **Output Variables** panel. Moving the mouse over those items will show tooltips with details of the missing item(s). When you import the missing template the blank row(s) will be filled and automatically mapped.

To manage delivery options: from the **Delivery Management** tab, click **Delivery Options**.

The screenshot displays the 'Delivery Options' configuration page. On the left, a table lists various delivery options, with 'Combo_Barcodes_Lnx' selected. The right pane shows the configuration for this selected option, including a name field and two tables: 'Selectable Delivery Channels' and 'Selected Delivery Channels'.

Delivery Options	Compatibility Mode
<input type="checkbox"/> Archive_and_Immed_PDF300_Lnx	Default
<input type="checkbox"/> Attachments_Combo_Barcodes_Lnx	Default
<input type="checkbox"/> Attachments_Combo_Barcodes_Win	Default
<input checked="" type="checkbox"/> Combo_Barcodes_Lnx	Default
<input type="checkbox"/> Combo_Barcodes_PDF_PS_Only	Default
<input type="checkbox"/> Combo_Barcodes_Win	Default
<input type="checkbox"/> Immed_eMessaging_Lnx	Default Immediate

DELIVERY OPTIONS

Combo_Barcodes_Lnx

Name: Combo_Barcodes_Lnx

Compatibility Mode: Default

Selectable Delivery Channels	
AFP240_Bit_Barcodes_Batch_Win	Express Batch
AFP300_Outline_Barcodes_Batch_Win	Express Batch
IJP300x300_Barcodes_Batch_Win	Express Batch
Immed_eMessaging_Lnx	Default Immediate
PDF300_AjanMultiLing_Archive_Lnx	Default Immediate
PDF300_AjanMultiLing_Immed	Express Immediate
PDF300_Barcodes_Batch	Default Batch
PDF300_Barcodes_Batch_Win	Default Batch
PS72_Barcodes_Batch	Express Batch
PS72_Barcodes_Batch_Win	Express Batch

Selected Delivery Channels	
PS72_Barcodes_Immed	Default Immediate
PDF300_Barcodes_Immed	Express Immediate
PDF300_Barcodes_Batch_Lnx	Default Batch
AFP240_Bit_Barcodes_Batch_Lnx	Express Batch
AFP300_Outline_Barcodes_Batch_Lnx	Express Batch
AFP300_Outline_Barcodes_Immed	Default Immediate
PS72_Barcodes_Batch_Lnx	Express Batch
AFP240_Bit_Barcodes_Immed	Default Immediate
IJP300x300_Barcodes_Immed	Default Immediate
IJP300x300_Barcodes_Batch_Lnx	Express Batch

Use the Delivery Option section of the page:

- Assign/change the name of the delivery option.

- Add/remove delivery channels from the delivery option using the arrow buttons.

Refer to [List view operations](#) on page 17 for details on icons used in the list view section of the page.

Note the following:

- The Compatibility Mode of the delivery option depends on the compatibility modes of all the delivery channels that make up the delivery option. For example, for a delivery option to be associated with an Express Batch compatibility mode, all delivery channels within the delivery option must have an Express Batch compatibility mode. Where there is a mix of Batch and Immediate compatibility modes, then the overall compatibility for the delivery is set to Default as shown above.
- The association of a template to its delivery option influences the compatibility mode associated with the template. Refer to [Compatibility modes](#) on page 84 for further information.
- Templates within a folder inherit delivery options assigned to that folder only if the template supports the delivery option.

6 - Community Administration

In this section

Document class management.....	121
Retention policy rules.....	121
Retention rules run rate.....	122
Retention history.....	123
Spell checker management.....	123
Data map management.....	124
EngageOne Communicate Integration.....	127
Vault Databases.....	129
Diagnostics.....	129



Document class management

Document classes define the metadata associated with the folders and templates.

Metadata can be used for search and retrieval purposes, for example, to help the front-office user find the right template. If you use the field **Type = Selection**, you can provide a list of choices for the template manager to select from. In this way you can provide generic options and the template manager can select from one of these instead of typing in a value each time.

After a document class is created, you can modify the definition of a document class and perform the following maintenance actions:

- rename the class
- add new fields for metadata
- remove existing metadata fields
- delete the document class if it has not been used.

The **Document Class** tab is used to define metadata field properties allocated to templates. This information is also used when you import a template so that you can add its metadata values.

1. From the **Community Administration** tab, click **Document Classes**.
2. Select an existing class for editing from the drop-down list.
3. Or, click here to create a new class.
4. You can change the existing name here or enter a new name if this is a new document class.
5. Click on the field and you can change the field name, or enter a new one.
6. Select the field type from the drop-down list.
7. If you choose **Selection**, a list of selections is provided
8. Click here to add or remove a field.

Retention policy rules

Retention policies define the time frame that a template or Active Content stored in a specific folder can be moved or deleted from the repository. If you are deploying new templates regularly, consider using retention policies as a means of managing your template usage and controlling disk space.

When you set up the name of the retention rule with its valid period you can also select different options to take when it expires. For instance you can:

- define whether the system should notify specified user(s) by e-mail that the retention rule has expired.

- move the retention rule with its contents to a pre-defined folder (the creation date will be changed to the moving date),
- or delete the retention rule and its contents from the system (including sub files).

The body of the e-mail notification will contain the community name, the template name, the template version, the date and the retention rule description.

Once you have set up a retention policy rule you can apply it to the folder or a template/Active Content – see Document templates. The retention rule is valid from the day that it is created but if you choose to move it to another folder on expiry, the start date is changed to the date the move took place. If you move a template manually from one folder to another (cut and paste, or copy and paste), the start date for the retention rule for that item only is changed to the date that it was moved.

1. from the **CommunityAdministration** tab, click **Retention Policies** followed by **Retention Rules/New**
2. Select to delete all included templates/ Active Content when the retention policy expires.
3. Enter the name of the new retention policy rule.
4. Choose the folder you want to save the templates/Active Content to when the retention policy expires.
5. Check this option to open the box for entering the e-mail address details of those you want to notify when the retention policy expires.

Retention rules run rate

To define how often the retention policy should run you must modify the respective settings in the **config-settings.xml** file.

You can configure:

- How often the retention policy will run in days (minimum is 1).
- What time of the day the retention policy should be started. If AM/PM is not specified the 24 hour clock format will be used.

Namespace	Key	Value
com.pb.openedms	RetentionCheckPeriod	an integer greater than zero
com.pb.openedms	RetentionCheckTimeOfDay	Timestamp in the form of HH:MM(AM PM). For example, 5:30PM, or 17:30.

Retention history

You can view retention rule activity for templates and Active Content within a time period specified.

For details of how to configure the retention policy sink see [Event sinks](#) on page 153

You can view retention rule activity for templates and Active Content within a time period specified. You cannot make any changes here except restore an item to its original folder if moved. The information provided is as follows:

- **Template** or Active Content folder path.
 - **Occur Time**: the date and time when the activity occurred.
 - **Delete**: a tick in the check box indicates that the item was deleted.
 - **Move**: a tick in the check box indicates that the item was moved.
 - **Notify**: a tick in the check box indicates that e-mail notifications were sent.
 - **SRC Path**: the source folder of the template, or Active Content.
 - **Destination Folder**: the target folder if the template, or Active Content was moved.
 - **Note**: displays who the notification was sent to.
1. From the **Community Administration** tab, click **Retention Policies** followed by **Retention History**.
 2. Specify the search scope by setting the start date and end date. You can use the calendar controls provided.
 3. Click on a moved folder and the Restore button will be enabled. You can restore the moved item back to its original folder.

Spell checker management

Dictionaries are automatically loaded during the EngageOne installation process and spell check management allows you to configure how you want these to be used.

You can select a dictionary language and display either a list of abbreviations, or additional words. These can be supplemented as required.

If you add a new word to the dictionary and the spell checker finds this word then the word is flagged as correctly spelled. A front-office user can only carry out a spell check on editable text.

You can also add an alternative to a regular word, for example, a common misspelling of that word, or one that is a regional variation. In such a case you enter the “correct” word for your dictionary first, leave a space and follow it on the same line with the “incorrect variation”. If the spell checker finds the second (incorrect) word it will automatically be replaced with the “correct” one.

You can import new dictionaries if they have been updated and need to be replaced. You can export a dictionary, for example, if you need to send it to EngageOne Support, or you might use this option to migrate a dictionary to a new environment. An imported dictionary is specific to the current community.

Dictionaries are community specific, so if you modify abbreviations or add additional words, those changes only affect the current community. When you create a new community the default dictionary is associated with that community.

1. From the **Community Administration** tab click **Spell Checker Management**.
2. Highlight the language you want to work with.
3. Select one of these list types.
 - **Additional words**
 - **Abbreviations**
4. Insert a space in the list as appropriate, for example, press **Return** at the end of the word after which you want to place the item. Type in an entry to add it to the dictionary. Type in two words with a space in between to set the first word as an alternative to the second. See below for an example.
5. Click **Save**.
The list is alphabetized automatically.
6. To import a dictionary file, click **Import**.
Select the check box to overwrite existing entries. Browse for the file to import and click **OK**.
7. To export a dictionary file, click **Export**.
Select a location for download and click **Save**.
8. To delete an entry, select it and click **Delete**.

Data map management

Data maps are created by importing data map files into EngageOne Server using the **Data Map** tab on the **Community Administration** tab. Create map files using an XML editor of your choice. After you import a data map, you can later update it or export it out to an XML file on your network.

EngageOne Server provides a loosely integrated, data push method for providing customer data that you use to generate interactive documents. Data push works by pushing an XML data structure from your business systems to the EngageOne Server where it is ready for an assigned user to create a document instance. The XML data structure can be pushed using a web service request or URL request to EngageOne Interactive.

Using map files that you create in an XML editor and then import to the EngageOne Server, data push passes your data in any format you define, and then maps that data to the data format used

in your EngageOne template definitions. The mapped XML data structure that is persisted on the EngageOne Server is consumed by a subsequent EngageOne Interactive session or by your own custom-written EngageOne application.

A good understanding of the data map XML schema is key to integrating data push methods between your business systems, custom applications, EngageOne Interactive and the EngageOne Server. The data map XML schema uses XPath annotation for determining source and target elements in your pushed XML data.

For more information about data map specifications, see the EngageOne Programmer's Reference Guide.

After you create the map files, import them into the EngageOne Server. EngageOne Server uses data maps to pass your data in any format you define, and then map that data to the data format used in your EngageOne template definitions.

1. From the **Community Administration** tab, click **Data Map**.
2. These are the data map choices:

Option	Description
New	Create a new data map based on an XML schema file you create.
Update	Update an existing data map. The description is editable; the map name is not editable.
Export	Export a data map to an XML schema file.
Map Name	<p>This is the name you use to reference the data map in web service requests and URL requests.</p> <p>Data map rules:</p> <ul style="list-style-type: none"> • The data map XML file must be a well-formed XML file that conforms to the data map file schema. For more information, see the EngageOne Programmer's Reference Guide. • Map names must be unique within a community. • Map names are case-insensitive, 255 maximum characters, and support the double-byte character set.
Description	Include information that will help you and others understand how the data map is used.

Option	Description
Delete	Deletes the selected data map.

A new or updated XML file is validated with the data map XML schema to ensure it is a valid data map. The data map displays in the list. After validation completes successfully, the map name and description is editable in the details panel.

EngageOne Communicate Integration

From EngageOne Administration, you can:

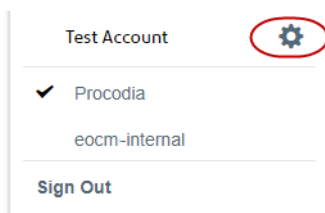
- create a connection to your EngageOne Communicate account, refer to [Retrieve Communicate account credentials](#) on page 127. Contact your support team for details on account registration if you do not already have an EngageOne Communicate account.
- import Communicate templates, refer to [Template operations](#) on page 24, for further information. Use either the **Add from EngageOne Communicate** or **EngageOne Communicate at Root** option depending on where the template is to be imported in your admin folder structure.
- generate responsive communications from templates imported from EngageOne Communicate using EngageOne Server's high speed batch processing capabilities to make the generated responsive emails content available to EngageOne Deliver for onward delivery. Refer to [About delivery management](#) on page 61 and [Batch processing](#) on page 167 for detailed information. For detailed information about EngageOne Compose integrated features refer to the EngageOne Communicate/EngageOne Compose Integrated features guide available on: <https://support.precisely.com/products/engageone-compose/>

Retrieve Communicate account credentials

Log into EngageOne Communicate to retrieve account settings required for defining your connection.

To create a connection :

1. Log into your EngageOne Communicate account.
2. Click your account profile and select the cog icon, as indicated below:



Note that the cog icon is only available to users who have team admin permissions.

3. From the Account Settings page, click on **Client Credentials**, copy **NAME**, **CUSTOMER ID**, **CLIENT ID**, and **SECRET KEY** values for subsequent use in EngageOne Server.

Teams

Users

Client Credentials

Sender Email Addresses

Procodia

eocm-internal

Client Credentials

Refresh

Add Credentials

NAME	CUSTOMER ID	CLIENT ID	SECRET KEY
Designer_Doc	sa38fa9	ca2f8d-8b6c-4c9b-b94b-69c0c0c0c0c0	*****

Integrating with an EngageOne Communicate team

You must register the Communicate credentials you have retrieved on this tab to allow for integration between EngageOne Server and EngageOne Communicate. When you have successfully created a connection to your EngageOne Communicate account, you can then import EngageOne Communicate templates.

To integrate with an EngageOne Communicate Team :

1. From the **Community Administration** tab, click **EngageOne Communicate Integration**
2. Click on the + icon, the Integrate with an EngageOne Communicate Team dialog is presented.

Integrate with an EngageOne Communicate Team

Customer ID

sa38fab

Client ID

swt88fab-gfH4nEaayngf888888888888@communicate.engageone.co

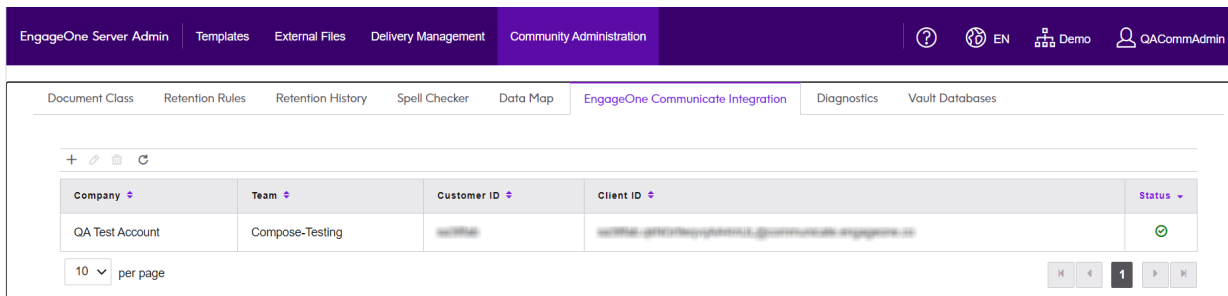
Secret key

5aWV0E3a4rHy42v4v4v4

Integrate

Cancel

- From the Account Settings page, click on **Client Credentials**, copy **NAME**, **CUSTOMER ID**, **CLIENT ID**, and **SECRET KEY** values for subsequent use in EngageOne Server.



Vault Databases

The assignment of Vault databases for use in the current communities.

A list of available Vault databases is displayed and can be selected for use in the current community.

1. From the **Communities Administration tab**, select the **Vault Databases**.
2. Use the arrow keys to select an single or all databases. You also have the option to remove a single or all databases from the list of selected databases.
3. Click **Save** to complete database assignments

Diagnostics

Note that Diagnostics can be accessed by those with both System Administrator rights and/or Community Administrator rights. Refer to [Diagnostics](#) on page 132 for further information.

7 - System administration

Note: When system administrators log into EngageOne Administration, they will only see the **System Administration** tab; unless they also have community administrator rights, the **Community Administration** tab will be visible in this scenario.

In this section

About system administration.....	131
The configuration framework.....	131
Diagnostics.....	132
Resolving composition issues.....	133
Repository configuration.....	133
Active Drive configuration.....	134
Account unlocking.....	136



About system administration

The EngageOne system administrator has access to the following:

- Diagnostics used to report on the current state of the system. Refer to [Diagnostics](#) on page 132
- Version number information of your installed applications is available from the **Status** application of each installed bundle. For example, if you have installed the core bundle on port 8080 on server core.example.com you can access version information from the URL:

`http://core.example.com:8080/status` Here you will find a section entitled **EngageOne Server Application Version Verification**. A detailed listing of deployed applications and their associated version numbers are displayed.

- The configuration framework (config-framework) is an EngageOne component that assists in accessing and maintaining the EngageOne configuration settings in a single configuration repository. Refer to [The configuration framework](#) on page 131 for further information.

The configuration framework

The configuration settings are stored in an XML file called `config-settings.xml` and the file location (typically `<bundle-root>/conf/config-settings/config-settings.xml`) is defined during the EngageOne installation process.

Its main features are:

- the config-framework has a timed reload that will refresh configuration settings when changes are made to the config-settings.xml file.
- the config-framework has the ability to separate multiple like settings by using setting groups. The setting groups are mainly used to maintain node specific settings within a multi-node environment. Config-framework will always try to retrieve a setting from a setting group according to its defined setting group level key. If a setting group is not present then the config-framework will search for settings defined at a higher, more global level that are not distinguished or associated specifically by a setting group.

It is recommended that you backup the file prior to making any changes. Although changes can be performed manually, use caution when any updates are applied to this configuration file as incorrect values, keys or namespaces may cause unpredictable behavior within the application.

Diagnostics

NA batch failed events logged and found in Diagnostics may or may not include all of the details about the failure. These events serve as an indicator that something went wrong. Review the NA batch logs for more details about the failure.

Events relating to the document cycle and administration are captured throughout the system and sent to the Event Monitor database for analysis using the diagnostics feature. See [Event Monitor](#) for details of how the Event Monitor is configured for tracing activity on your system. If, for example, an e-mail failed because the SMTP server was down, you can set up filter criteria to extract these events and re-submit the e-mails concerned.

A process type defines the type of the event, for example, **Create Document**, **Import Template**, **Login**, etc. An event type is the priority of the event. There are four event type priorities defined in the system: **Error**, **Warning**, **Informational**, and **Debug**.

1. Click **Diagnostics/ Diagnostics**.
2. Select the event process type from the drop-down list.
3. Select the required event from the list.
4. If applicable, enter the unique application identifier from which the event originated.
5. To refine your search further you can enter a date/time range.
6. Click here and the returned events from the search criteria entered will be displayed below.
7. Highlight one of the events and all available details relating to it will be displayed below.
8. These buttons are activated for Document process types which have an Error type. See below for details.
9. You can use the **Resubmit** button if you have fixed any issues relating to the document, the issue is resolved and you want to resubmit the document/work item as if it was being submitted by the document creator.

Notes:

- Only those communities to which the current Community Administrator has rights will be available in the “Community” drop-down list.
- If a community is not specified for the Event search, all events for all the communities to which the Community Administrator has rights will appear.

You can use the **Reactivate** button if you want to send the document/work item back to the document creator. This option will be used if the front-office user needs to resolve the issue relating to the document.

Resolving composition issues

The `alwaysDumpIncidentArchive` flag can be used to troubleshoot and resolve problems arising from unsuccessful composition of your communications by EngageOne Generate.

When the `alwaysDumpIncidentArchive` is set to `true` the following files are created in the incident-archive folder on the active-drive:

- All template files.
- All supporting resource files for each template. These are created during Publish for EngageOne processing in EngageOne Designer.

It is important to note that these files are created for both successful and unsuccessful composition; for this reason, setting this flag to true as a default is not recommended.

EngageOne Generate can compose communications either from batch processing requests or via SOAP/UI requests. The `alwaysDumpIncidentArchive` flag can be used for both types of requests to Generate.

Batch requests to Generate

Depending on the type of batch processing you are working with the `alwaysDumpIncidentArchive` property must be added to the batch jvm options. Update the appropriate run script, this can be found at:

```
<engageOneInstallPath>/batch/bin
```

SOAP/UI requests to Generate

Locate and run `eos-compositionw` this is found at:

```
<engageOneInstallPath>/composition/bin.
```

The EngageOne Server composition Service Properties dialog is presented.

Enter `-DalwaysDumpIncidentArchive=true` in the **Java Options** entry area.

Repository configuration

The Repository configuration allows you to change the repository alias or repository name of the Designer repository or the Content Author repository.

If you change the Designer or Content Author repository alias or name (or create new ones), you must update those values in EngageOne Administration.

To change the repository alias or repository name:

1. From the System Administration tab.
2. Click Configuration Management/Repository Configuration.
3. Select a repository in the list.
4. Edit the repository alias or repository name.
5. Click OK.

Active Drive configuration

The content repository is the content management system that provides the file storage, and version control for EngageOne. It is made up of the database and a shared filestore. The location of the content repository folder is where all customer content is stored. Active-Drive configuration allows you to specify the active-drive for your shared filestore. This is configured at installation, but you may need to move the location due to disk space issues.

Moving the active-drive

In order to move the active-drive from one location to another (or change the default directory of active-drive), you need to follow the instructions listed below:

Initial steps:

1.
 - Request all users to save their work and log out of EngageOne Interactive.
 - Stop all traffic to EngageOne SOAP services.
 - Execute EngageOne server purge process. If there are more communities present, all of them need to be purged. Just to be clear - it is not absolutely necessary to run purge, but we recommend it in order to reduce the number of rows that need to be modified. There is no need to run Event Monitor purge process.
 - Stop all EO services on all nodes.
 - Create backup of both DB and active-drive.
2. Copy the active-drive and its contents from old location to new location.
3. Update deploy.properties file, providing new path for active-drive (active.drive.dir). The file can be found in folder <release-medium>\install.
4. If you use the active-drive location for any other property (e.g. tls.trust.store.location, tls.key.store.location, eodeliver.share.dir) and you plan to use the new location, you have to update values of those properties before you proceed.
5. Open the **New query** tab and run stored procedure `usp_u_move_active_drive`.

Example for MS SQL Server:

```
exec usp_u_move_active_drive @iv_OLD_ACTIVE_DRIVE_PATH =
'<existing_active_drive_path>',@iv_NEW_ACTIVE_DRIVE_PATH =
'<new_active_drive_path>';
```

Example for Oracle:

```
set serveroutput on;exec usp_u_move_active_drive
('<existing_active_drive_path>',
'<new_active_drive_path>');
```

Before any attempt to execute the procedure, both `<existing_active_drive_path>` and `<new_active_drive_path>` have to be replaced with correct paths. After the procedure is successfully executed, something similar to the example below should appear on the console:

```
2324 rows have been updated in column Physical_Path in table
Doc_Storage_Locations
19980 rows have been updated in column AnswerFilePath in table
TDS_RequestChannels
19980 rows have been updated in column PrintFilePath in table
TDS_RequestChannels
19980 rows have been updated in column MetricsFilePath in table
TDS_RequestChannels
4 rows have been updated in column MessageText in table
TDS_RequestChannels
29970 rows have been updated in column ParameterValue in table
TDS_RequestParameters
```

The numbers can be different, but 6 lines are expected as in the example above. Four of them are related to the `TDS_RequestChannels` table, the remaining two to `Doc_Storage_Locations` and `TDS_RequestParameters`.

6. Run `eos.groovy` installation script in order to re-configure all batch, composition, conversion and core bundles present in the environment:

```
groovy eos.groovy -b <batch|composition|conversion|core> -p
<deploy.properties_path>
-t <single|primary|replica> configure
```

IMPORTANT – the script has to be executed against all specified bundles on all nodes. There is no need to re-configure the security and notification bundles (assuming that notification bundle is installed).

7. Start all EO services and perform all the sanity checks as usual.

IMPORTANT – it may be necessary to modify the configuration manually (in EO Admin application) if any delivery channel present in the environment is configured in a way that either output or journal files are created within old active-drive structure.

Account unlocking

As an OpenAM system administrator, you can unlock an account lockout caused by your account lockout policy. Use the command-line tool CURL to interact with the OpenAM RESTful API to do this.

IMPORTANT: Prior to performing the instructions in this section you must configure your system to allow for account unlocking. Refer to the Account Lockout appendix in the EngageOne Server Installation guide for detailed information.

Account unlocking using CURL

1. Install CURL on your local computer using this link (<https://curl.haxx.se/>)
2. **This step is applicable when operating in a clustered environment; otherwise, skip this step.**

You must ascertain which node the user is blocked; follow the steps listed below:

- a. Obtain `amlbcookie` from blocked-user (`amlbcookie` determines which OpenAM node is being used by the user), to do this, inspect the Request headers send by the client browser. Note that `amlbcookie` is sent in the cookie header.
- b. As an OpenAM administrator, follow the instructions in the list below to determine which node is using `amlbcookie`:
 - Login to OpenAM as an OpenAM administrator (username: `amadmin` and password entered in `deploy.properties/security.admin.password` during installation of security bundle). From the menubar select **deployment/servers**, here you will find a server listing.
 - You now need to establish which server the user has been locked-out. To do this, on each server, check its `amlbcookie` (after clicking on the server choose **Advanced** and check value of the property `com.ipplanet.am.lbcookie.value`). The property `com.ipplanet.am.lbcookie.value` should be the same as `amlbcookie`.
 - By following the instructions in the points above, you can determine which OpenAM server address and port to be used in step 4 below.

NOTE: The process listed in this step may be time-consuming, and in certain scenarios, it may be advisable to wait for the lockout time period determined in the `deploy.properties` file instead of carrying out this step.

3. Obtain session token for OpenAM administrator, as follows:

```
curl -X POST -H
"X-OpenAM-Username: amadmin" -H
"X-OpenAM-Password: <OPENAM ADMIN PASSWORD>" -H
"Content-Type: application/json" <ADDRESS>:<OPENAM
PORT>/OpenAM/json/authenticate
```

4. Use the session token obtained from the previous step to clear login attempts data, as follows:

```
curl -X PUT -H
"iPlanetDirectoryPro: <SESSION TOKEN>" -H
"Content-type: application/json" -d
"{ \"sunAMAuthInvalidAttemptsData\":[] }"
<ADDRESS>:<OPENAM PORT>/OpenAM/json/EngageOne/users/<LOCKED ACCOUNT
NAME>
```

Important: <OPENAM ADMIN PASSWORD> is the password specified in `deploy.properties/security.admin.password`

8 - Event Monitor

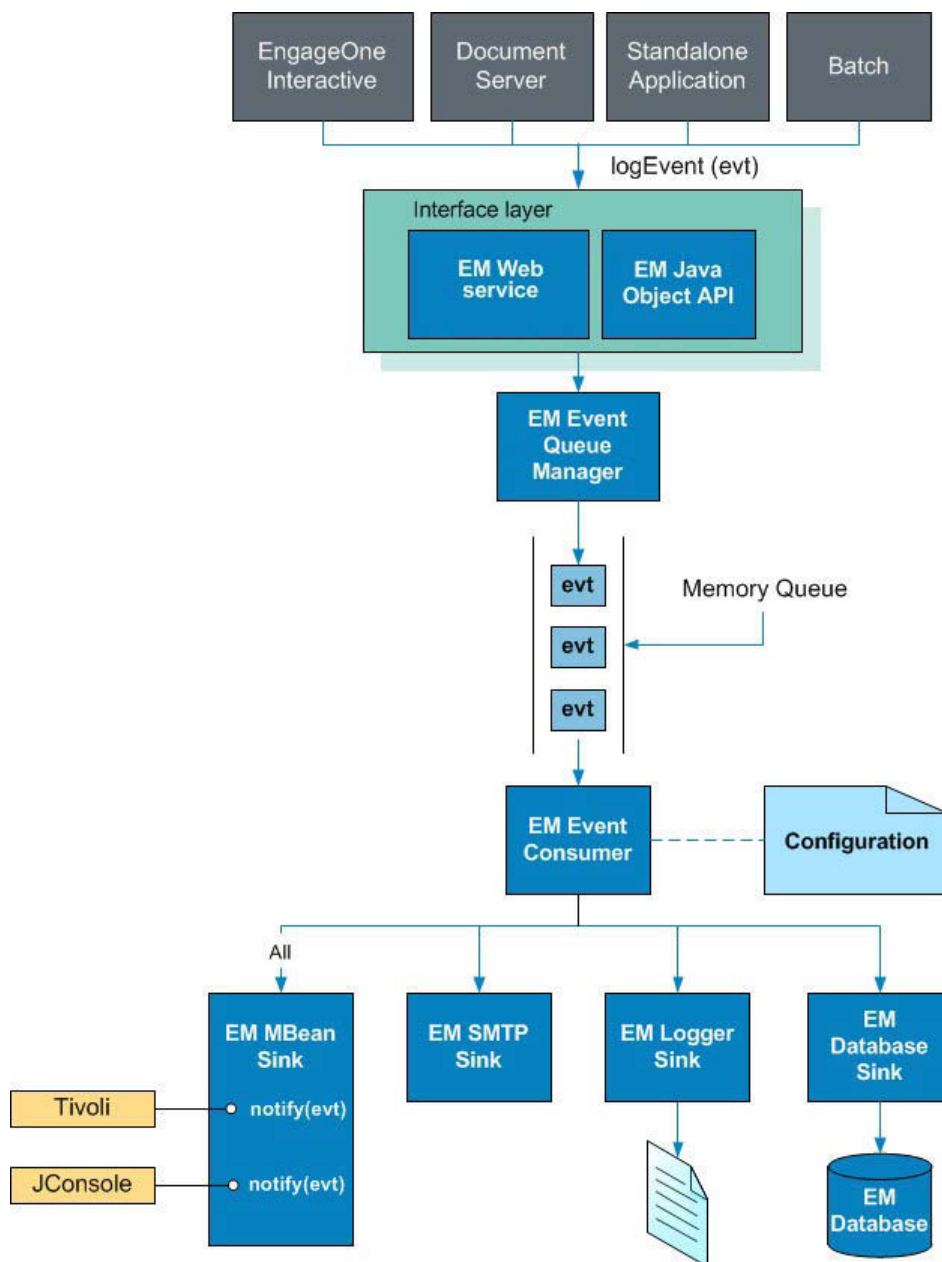
In this section

Event Monitor.....	139
EngageOne Notifications.....	140
EngageOne events.....	141
Event Monitor tables.....	148
Event configuration.....	153
Logging events.....	164
Attachment conversion process.....	166



Event Monitor

As events flow through the system, the Event Consumer component dispatches them to appropriately configured sinks. For example, the configuration could be set up to send all error level messages to an administrator mailbox via SMTP as well as to standard logs. All EngageOne events are dispatched to the Event Monitor database for reporting purposes.



The Event Monitor exposes a single JMX MBean which can be used by any JMX-capable application (such as Tivoli or JConsole) for monitoring of events. The JMX MBean is registered with the MBean server with the object name: `EventMonitor:name=management`.

The **Diagnostics** tab in the **System Administration** section allows you to search and report on events – see [System administration](#).

Enable or disable Event Monitor

Event Monitor can be enabled or disabled, by adjusting settings in `config-settings.xml`.

Change "value" below to TRUE to enable Event Monitor.

Change to FALSE to disable Event Monitor.

```
<setting>
<key>Event_Monitoring</key>
<value>TRUE</value>
</setting>
```

EngageOne Notifications

EngageOne version 4.4 introduced a new feature called Notifications. There are similarities between Notifications and the Event Monitor, but each has a distinct purpose.

The primary purpose of the Event Monitor is to log information about a wide range of events that occur within the system. This information can be used for reporting, auditing and diagnostic purposes. The Event Monitor is highly configurable, and it is possible to create custom event sinks to handle specific events. The Event Monitor should be used where the main requirement is for capturing system information for later analysis.

Notifications are designed to enable system-to-system integration via an industry standard message queuing infrastructure. EngageOne generates notification messages at key points in the life cycle of a communication, and publishes the notifications to a message queue. An external system can consume messages from the queue and take appropriate actions, such as updating the status of a customer record to show that a letter has been sent. The Notifications mechanism is relatively lightweight and it should be used when the primary requirement is for real-time communication of business-related events to another system.

EngageOne events

Events are either user-generated or system-generated. User-generated events include anything that you want to track that a user initiates. System-generated events are initiated by internal system processes, such as batch processing or document delivery.

The following table shows the event names as they display in the Diagnostics tab, the corresponding event type names as used in `event-monitor-sinks.xml`, and where the events originate.

All events in EngageOne have a corresponding process type, which acts as unique identifier for an event. For example, a **Create Document** event is also of the process type `create.document`. For details about the content of the table see [Event Monitor tables](#) on page 148.

The Event Monitor purge should be run whenever you run the batch purge. The frequency depends on your system's daily workload. For details about purging the Event Monitor see [Managing the Event Monitor](#) on page 232.

Event display names and process types

Process display name (event)/type	Administration	Interactive	Document Server	Batch
Add Keymap Image <code>add.keymap.image</code>	X			
Approve Document <code>approve.document</code>		X		
Approve Work Item <code>approve.work.item</code>				
Batch NA <code>batch.na</code>				
Create Data Push Map <code>create.datapush.map</code>	X			

Process display name (event)/type	Administration	Interactive	Document Server	Batch
Create Document <code>create.document</code>		X		
Create Document Class <code>assign.document.class</code>	X			
Create Keymap <code>create.keymap</code>				
Data Mapping <code>data.mapping</code>				
Delete Active Content <code>delete.active.content</code>	X			
Delete Data Push Map <code>delete.data.push.map</code>				
Delete Delivery Item <code>delete.delivery</code>			X	
Delete Document <code>delete.document</code>				
Delete Document Class <code>remove.document.class</code>	X			
Delete Keymap <code>delete.keymap</code>	X			
Delete Keymap Image <code>delete.keymap.image</code>	X			

Process display name (event)/type	Administration	Interactive	Document Server	Batch
Delete Message Content delete.message.content				
Delete Template delete.template	X			
Delete delete				
Delete Template According Retention Policy delete.template.retention	X			
Deliver Document deliver.document		X	X	X
Deliver EngageOne Deliver				
Doc1gen Composition doc1gen.composition				
Edit Document edit.document				
Export Data Push Map export.data.push.map				
Export Message Content export.message.content				
Generate Document generate.document			X	

Process display name (event)/type	Administration	Interactive	Document Server	Batch
Generic Debug generic.debug				
Generic Error generic.error				
Generic Information generic.info				
Generic Warn generic.warning				
Import Active Content import.active.content	X			
Import Keymap create.keymap	X			
Import Message Content import.message.content				
Import Template import.template	X			
License Expiration license.expiration				
License Invalid license.invalid				
Local Print Document local.print.document		X		

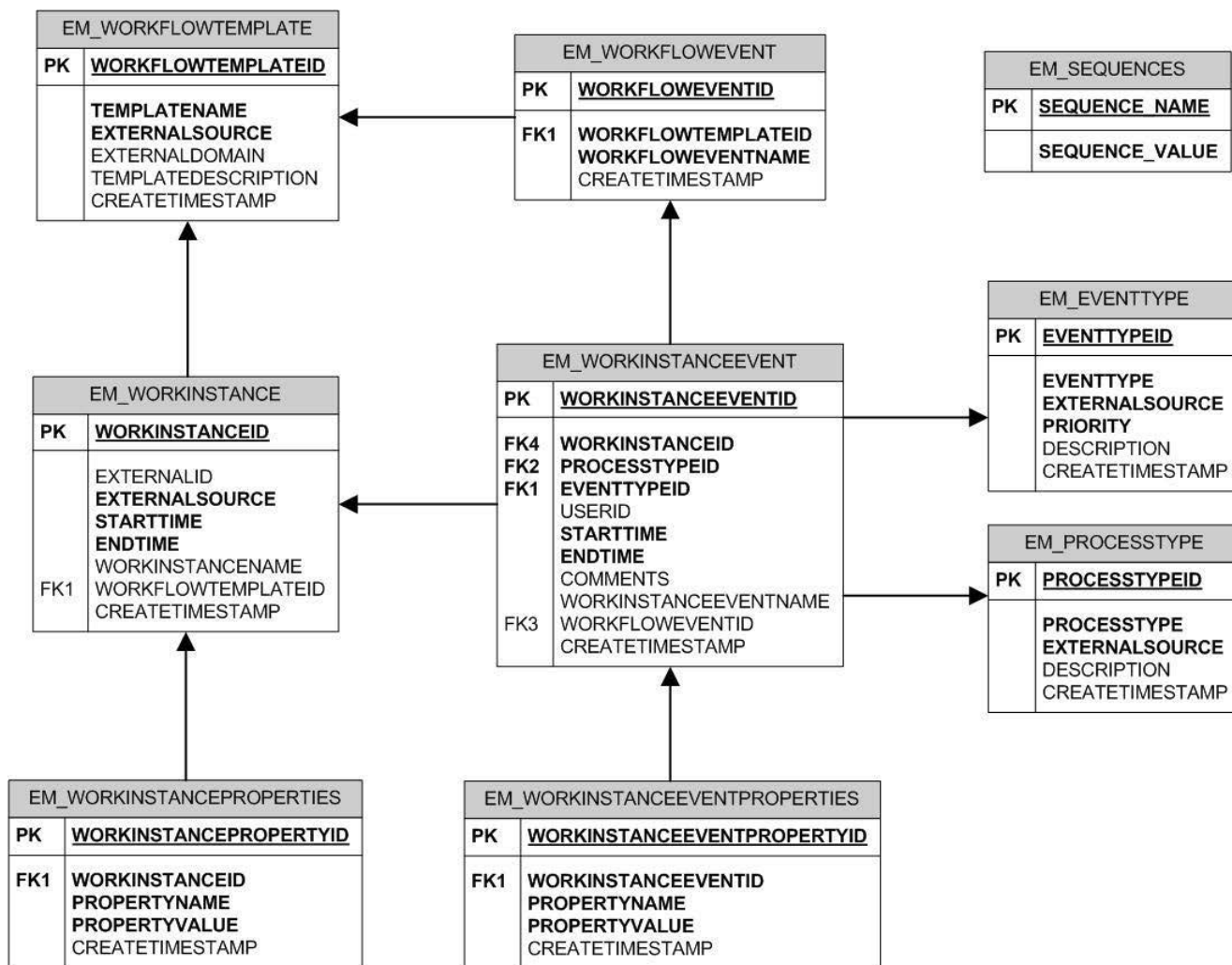
Process display name (event)/type	Administration	Interactive	Document Server	Batch
Login login	X			
Logout logout	X			
Map Data data.mapping		X		
Move Template According Retention Policy move.template.retention	X			
NA Batch batch.na				X
Preview Document preview.document		X		
Purge purge				X
Purge Batch purge.batch				X
Purge By Channel purge.channel				X
Purge By Status purge.status				X
Push Data data.push	X	X		

Process display name (event)/type	Administration	Interactive	Document Server	Batch
Reassign Document reassign.document		X		
Reassign Work Item reassign.work.item				
Reject Document reject.document		X		
Reject Work Item reject.work.item				
Restart Batch restart.batch				X
Restore Moved Template restore.template.retention	X			
Resume Batch resume.batch				X
Resume Delivery resume.delivery		X	X	
Run Channel run.channel				X
Save Data Push Map save.data.push.map				
Save Document save.document		X		

Process display name (event)/type	Administration	Interactive	Document Server	Batch
Save Key Map save.keymap	X			
Save Workflow save.workflow				
Select Delivery Option select.delivery.option		X		
Submit Document submit.document		X		
Suspend Delivery suspend.delivery		X	X	
Update Data Push Map# update.data.push.map				
Update Key Map update.keymap	X			
Update Resource Access Rights update.resource.access.rights	X			
Update Resource Delivery Options update.resource.delivery.options	X			
Update Template update.template	X			

Event Monitor tables

The data model below represents the tables that make up the Event Monitor.



The following lists describe the content of the tables in the Event Monitor:

EM_WORKINSTANCE

A work instance is the primary entity in the Event Monitor system. It wraps a group of work instance events together, for example, in EngageOne, a work instance might wrap all activity surrounding a particular document instance.

WORKINSTANCEID

Primary key

EXTERNALSOURCE	Set to EngageOne.
STARTTIME	The start time of the work instance.
ENDTIME	The end time of the work instance.
EXTERNALID	The work item ID, which provides a unique identifier in EngageOne for a document instance as it moves through the system.
WORKINSTANCENAME	Optional descriptive name for the work instance.
WORKFLOWTEMPLATEID	Optional foreign key to a workflow template.
CREATETIMESTAMP	Audit create timestamp.

EM_WORKINSTANCEEVENT

A work instance event contains the data relating to a particular event within the system. Every event is identified by a process type. For example, in EngageOne, process types include **Create Document**, **Import Template**, etc. See [EngageOne events](#) on page 141 for a complete list. An event also has what is called an event type. See EM_EVENTTYPE that follows.

WORKINSTANCEEVENTID	Primary key
WORKINSTANCEID	Parent work instance foreign key.
EVENTTYPEID	Event type foreign key.
PROCESSTYPEID	Process type foreign key.
USERID	User ID of the user who created the event.
STARTTIME	Start time of the event.
ENDTIME	End time of the event.
COMMENTS	Optional comments on the event.
WORKINSTANCEEVENTNAME	Optional name of the event.

WORKFLOWEVENTID	If this event corresponds to a workflow event, this will be the workflow event foreign key.
-----------------	---

CREATETIMESTAMP	Audit create timestamp.
-----------------	-------------------------

EM_PROCESSTYPE

A process type defines the type of the event. For example, the event **Create Document**, **Import Template**, **Login**, etc. See [EngageOne events](#) on page 141 for a list of process types.

PROCESSTYPEID	Primary key
PROCESSTYPE	Application defined process types. For example, <code>create.document</code> , <code>import.template</code> , etc.
EXTERNALSOURCE	EngageOne.
DESCRIPTION	Application defined description of the process type.
CREATETIMESTAMP	Audit create timestamp.

EM_EVENTTYPE

An event type is the priority of the event. There are four event type priorities defined in the EM system (EM_EVENTTYPE table): **Error**, **Warning**, **Informational**, and **Debug**. You can define and name as many event types as you require, but each must have a corresponding priority from one of the four mentioned above. This means an application could have multiple **Error** event types, and multiple **Warning** event types, etc.

EVENTTYPEID	Primary key.
EVENTTYPE	Identifies the type of event
EXTERNALSOURCE	EngageOne.
PRIORITY	One of four possible values: 0 = Error , 1 = Warning , 2 = Informational , 3 = Debug .
DESCRIPTION	Application defined description of the event type.

CREATETIMESTAMP

Audit create timestamp.

EM_WORKINSTANCEPROPERTIES

The work instance entity allows for properties to be assigned, ranging from standard/stock properties to any additional properties you want defined.

WORKINSTANCEPROPERTYID	Primary key.
WORKINSTANCEID	The work instance foreign key.
PROPERTYNAME	Application defined property string, for example, <code>document.id</code> .
PROPERTYVALUE	The value of the property.

EM_WORKINSTANCEEVENTPROPERTIES

Similar to work instance properties, work instance events support additional properties exceeding the standard/stock attributes of the work instance event.

WORKINSTANCEEVENTPROPERTYID	Primary key.
WORKINSTANCEEVENTID	The work instance foreign key.
PROPERTYNAME	Application defined property string, for example, <code>document.id</code> .
PROPERTYVALUE	The value of the property.

EM_WORKFLOWTEMPLATE

The Event Monitor system allows for a workflow, which can be tied to work instance events. The workflow template and workflow event tables are pre-populated by the application prior to events being logged.

WORKFLOWTEMPLATEID	Primary key
TEMPLATENAME	Name of the template.

EXTERNALSOURCE	EngageOne.
EXTERNALDOMAIN	Additional, optional criteria used to categorize the template, above and beyond the use of the external source.
TEMPLATEDESCRIPTION	Description of the template.
CREATETIMESTAMP	Audit create timestamp.

EM_WORKFLOWEVENT

A workflow event is a step within an application workflow.

WORKFLOWEVENTID	Primary key.
WORKFLOWTEMPLATEID	Workflow template foreign key.
WORKFLOWEVENTNAME	The name of the step within the workflow corresponding to this workflow event.
CREATETIMESTAMP	Audit create timestamp.

Event configuration

The EngageOne event configuration is set up to dispatch all EngageOne events to the Event Monitor database for reporting purposes. It also allows event filtering to various sinks. For example, you might want to send all **Error** events to an SMTP mailbox. For more information about filtering see [Filtering](#).

The event configuration is stored in `event-monitor-sinks.xml` and used by the Event Consumer when dispatching events. MBean operations manage the configuration on a running system. If you want to change your event configuration you can update the `event-monitor-sinks.xml` file located in: `<active.drive.dir>\config`. You will need to restart EngageOne services (bundles) for the changes to take effect.

Event sinks

The Event Monitor configuration allows for a wide variety of sinks to contain events, including log, database, e-mail, and JMX notification.

Any number of sinks may be defined. For example, multiple SMTP sinks could be defined based on their purpose. One SMTP sink could be setup to send all **Error** events to an administrator e-mail address or pager. Another SMTP sink could be setup to send all template events to a group of users who need to know when templates change (newly imported, deleted, etc.).

Note that it is not recommended to configure multiple log, database and JMX sinks.

log sink

Following is an example of how to configure the log sink within the `event-monitor-sinks.xml` file. The configuration is managed via the application server log configuration.

```
<sinks>
  <sink name="Sample log sink">
    <config type="log"/>
  </sink>
</sinks>
```

Database sink

The database sink connects to the database defined by the EngageOne data source. Following is an example of how to configure the database sink within the `event-monitor-sinks.xml` file.

```
<sinks>
  <sink name="Sample database sink">
```

```
<config type="database"/>
</sink>
</sinks>
```

e-mail (SMTP) sink

Following is an example of how to configure the SMTP sink within the `event-monitor-sinks.xml` file.

```
<sinks>
<sink name="Sample SMTP sink">
<config type="smtp">
<property name="securityMode">STARTTLS</property>
<property name="host">mail.whoever.com</property>
<property name="port">25</property>
<property name="to">somebody@somewhere.com</property>
<property name="from">engageone@somewhere.com</property>
<property name="subject">Someone logged an event!</property>
</config>
</sink>
</sinks>
```

Properties defined for an SMTP sink are listed below:

Property	Description
securityMode	Security mode. Allowed values: NONE, STARTTLS, SSLTLS. Default = NONE (optional). For STARTTLS and SSLTLS to work properly set in <code>deploy.properties</code> trust store details(<code>tls.trust.store.location</code> and <code>tls.trust.store.password</code>)
host	Specifies the SMTP server (mandatory).
port	The port number the SMTP server is listening on. Default = 25 (optional).
to	The e-mail address or addresses separated by commas to which the event should be sent (mandatory).
from	The from e-mail address of the message. Default = Event Monitor System <code>do.not.reply@event.monitor</code> (optional).
subject	The subject of the message. Default = Event Monitor Event (optional).

Note that authentication SMTP is supported if you selected this option during the install process. These SMTP settings are stored in the configuration framework file. Once `event-monitor-sinks.xml` has been edited to allow events to be processed via SMTP, all events that fit the filter criteria will be sent seamlessly as e-mails, as well as logged to the database.

JMX sink

The JMX sink sends notifications to any registered listeners. Following is an example of how to configure the JMX sink within the `event-monitor-sinks.xml` file.

```
<sinks>
  <sink name="Sample JMX sink">
    <config type="jmx"/>
  </sink>
</sinks>
```

Event filtering

Filtering is a mechanism to refine the events that are sent to a particular sink. There are different ways of configuring sink filters.

Example 1

In this example, if the process type of the event is an **Import Template**, then the event will be sent to the sink.

```
<sinks>
  <sink name="Sample SMTP sink">
    <config type="smtp">
      <property name="securityMode">STARTTLS</property>
      <property name="host">mail.somewhere.com</property>
      <property name="to">docmgrs@somewhere.com</property>
      <property name="from">templatemgrs@somewhere.com</property>
      <property name="subject">A template was changed</property>
    </config>
    <filters>
      <filter>
        <property name="processType" include="import.template"/>
      </filter>
    </filters>
  </sink>
</sinks>
```

Example 2

Filters can contain multiple properties. In this example, only informational **Import Template** events (as opposed to **Error**, **Warning**, and **Debug**) will be sent to the sink. The combination of properties within a filter grouping is a logical AND operation.

```
<filter>
  <property name="processType" include="import.template"/>
  <property name="eventType" include="Informational"/>
</filter>
```

Example 3

There can be multiple filters on a sink. This configuration states that all errors and warnings will be sent to the sink. The combination of filters is a logical OR operation.

```
<filters>
  <filter>
    <property name="eventType" include="Error"/>
  </filter>
  <filter>
    <property name="eventType" include="Warning"/>
  </filter>
</filters>
```

Example 4

The value attribute of filter property can also contain a list of values separated by commas. The combination of values in a property is a logical OR operation.

```
<filters>
  <filter>
    <property name="eventType" include="Error,Warning"/>
  </filter>
</filters>
```

Example 5

Filters can also contain wildcards. This configuration states that ALL informational template events will be sent to the sink.

```
<filter>
  <property name="processType" include="*.template"/>
  <property name="eventType" include="Informational"/>
</filter>
```

Event properties

There is a collection of standard properties that apply to all events:

Property	Mandatory Y/N	Description
processType	Y	Represents the type of event (e.g., create.document, submit.document, etc.)
eventType	Y	Represents the level of the event (e.g., Error, Informational, Warning, Debug)
userId	Y	User ID related to the event raised
externalSource	Y	Application that originated the event (e.g., EngageOne)
externalId	N	Application-defined identifier to the external entity in the application table
startTimestamp	Y	Start timestamp of the event
endTimestamp	Y	End timestamp of the event
comments	N	Optional comments
workflowName	N	Optional workflow name
workflowEventName	N	Optional workflow step name
message	N	Optional message
exception.name	N	Optional exception name (only used when eventType is an error event)
exception.message	N	Optional exception message (only used when eventType is an error event)
exception.stack	N	Optional exception stack trace (only used when eventType is an error event)

In addition, all EngageOne events contain the following mandatory property:

Property	Description
domain.name	EngageOne domain name

Finally, each specific type of event has its own set of properties, which are all mandatory.

Process type	Property	Description
*.document	document.name	Name of the document
	document.id	Internal identifier of the document (system generated)
	template.name	Name of the template
	template.id	Internal identifier of the template (system generated)
*.template	template.name	Name of the template
	template.id	Internal identifier of the template (system generated)
*.keymap.image	keymap.image.name	Name of the keymap image
	keymap.image.id	Internal identifier of the keymap image (system generated)
*.document.class	doc.class.name	Name of the document class
	doc.class.id	Internal identifier of the document class (system generated)
*.keymap	keymap.name	Name of the keymap
	keymap.id	Internal identifier of the keymap (system generated)
*.active.content	active.content.name	Name of the Active Content

Process type	Property	Description
	active.content.id	Internal identifier of the Active Content (system generated)
*.resource.access.rights		
*.resource.delivery.options		
*.batch	batch.id	Internal identifier of a batch run (system generated)
purge.batch	documents.purged	Number of documents purged
	documents.not.purged	Number of documents not purged
restart.batch	documents.delivered	Number of documents delivered
	documents.not.delivered	Number of documents not delivered
resume.batch	documents.delivered	Number of documents delivered
	documents.not.delivered	Number of documents not delivered
*.channel	channel.name	Name of the delivery channel
	channel.id	Internal identifier of the delivery channel (system generated)
purge.channel	documents.purged	Number of documents purged
	documents.not.purged	Number of documents not purged
run.channel	batch.id	Internal identifier of a batch run (system generated)
	documents.delivered	Number of documents delivered
	documents.not.delivered	Number of documents not delivered
purge.status	batch.id	Internal identifier of a batch run (system generated)

Process type	Property	Description
	status	Purge status
	documents.purged	Number of documents purged
	documents.not.purged	Number of documents not purged
select.delivery.option	document.name	Name of document
	documents.id	Internal identifier of the document (system generated)
	template.name	Name of the template
	template.id	Internal identifier of the template (system generated)
	delivery.option.id	Internal identifier of the delivery option (system generated)
*.workflow	workflow.name	Name of the workflow
delete.template.retention, move.template.retention	retention.ruleID	an ID for the policy
	retention.rule.desc	The description of the policy
	retention.fileCount	The number of files either moved or deleted
	retention.template.path	The path of the template to be moved or deleted
	smtp.to	The e-mail address 'to'
	smtp.from	The e-mail address 'from'
	template.name	The template name
	template.id	The template ID

Process type	Property	Description
	template.version	The version of the template
	effective.date	The date that the retention policy took effect
	retention.template.src.path	The src path before a move policy event
	Retention.template.dest.path	The dest path after a move policy event

In the example configuration file that follows, note the class `sink`, which allows you to define custom sinks.

Also note, `email admin`, which is set up to send all errors.

```
<?xml version="1.0" encoding="utf-8"?>
<sinks>
<!-- Setup a database sync to send all CreateDocument and SaveDocument
process types OR -->
<!-- all warnings and errors to the event monitor database. -->
<sink name="database">
  <config type="database"/>
  <filters>
    <filter>
      <property name="processType" include="CreateDocument,SaveDocument"/>
    </filter>
    <filter>
      <property name="eventType" exclude="Error"/>
    </filter>
  </filters>
</sink>
<!-- Setup an SMTP sync to send all errors to admin@whoever.com -->
<sink name="email admin">
  <config type="smtp">
    <property name="host">161.228.123.180</property>
    <property name="port">26</property>
    <property name="to">admin@whoever.com</property>
    <property name="from">engageone@whoever.com</property>
    <property name="subject">Fatal error</property>
  </config>
  <filters>
    <filter>
      <property name="eventType" include="Error"/>
    </filter>
  </filters>
</sink>
<!-- Setup an SMTP sync to send all template imports to
```

```

docmgrs@whoever.com -->
<sink name="email document managers">
  <config type="smtp">
    <property name="host">161.228.123.180</property>
    <property name="port">26</property>
    <property name="to">docmgrs@whoever.com</property>
    <property name="from">engageone@whoever.com</property>
    <property name="subject"> A template has been imported </property>
  </config>
  <filters>
    <filter>
      <property name="processType" include="ImportTemplate"/>
    </filter>
  </filters>
</sink>
<!-- Setup a log sync to send all warnings and errors to the log file.
-->
<sink name="log">
  <config type="log"/>
  <filters>
    <filter>
      <property name="eventType" include="Warning,Error"/>
    </filter>
  </filters>
</sink>
<!-- Setup a class sync to send all warnings and errors to the log file.
-->
<sink name="my event sink">
  <config type="class">
    <property
name="className">com.whoever.engageone.event.MyEventSink"</property>
    <property name="classpath">d:/blah/mylib/mysink.jar"</property>
  </config>
  <filters>
    <filter>
      <property name="processType" include="CreateDocument,SaveDocument"/>
    </filter>
  </filters>
</sink>
</sinks>

```

Event configuration file schema

The structure of `event-monitor-sinks.xml` file is defined by the following XML schema:

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:em="http://www.gl.com/EventMonitor/sinks"
targetNamespace="http://www.gl.com/EventMonitor/sinks"

```

```

xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:element name="sinks">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="em:sink" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="sink">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="em:config"/>
        <xs:element ref="em:filters" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="config">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="property" type="em:config-property" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="type" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="filters">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="em:filter" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="filter">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="property" type="em:filter-property"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="config-property">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="name" type="xs:string" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:complexType name="filter-property">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="name" type="xs:string" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

```

```

    <xs:attribute name="include" type="xs:string"/>
    <xs:attribute name="exclude" type="xs:string"/>
  </xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:schema>

```

Logging events

Logging of events can be controlled by a system configuration property.

For example, if the logging event level is set to **INFO**, all **Informational**, **Warning**, and **Error** events are logged to the system log. The logging event level can also be set to **DEBUG**, in which case all events generated by the application are logged. It is recommended to use this setting with care.

Logging configuration

The Event Monitor logging configuration is managed by the application server administrator. Each bundle has its own logging configuration, located at

```
<bundle-root>/conf/logging/logback-global.xml.
```

For more information on configuring logging, see the [Logging](#) on page 228 section in Managing your EngageOne environment.

Note: System locale is used to determine the character encoding unless you change the `<charset>` element of applicable encoders in the configuration.

Loggers

The Event Monitor uses the class of the event to look up the logger to use in logging an event.

For instance, for EngageOne, the `com.pb.engageone.event.document.CreateDocument` class is used when a **Create Document** event is to be logged. In this way, you can configure your log configuration to send all document events to a particular file. For example, add the following to `logback-global.xml`:

```

<appender name="document-events"
class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>../logs/document-events.log</file>

```

```

    <encoder>
      <pattern>[%date | %-5level | %1logger] %msg%n</pattern>
    </encoder>
    <rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">

    <fileNamePattern>../logs/document-events.%d{yyyy-MM-dd}.log</fileNamePattern>

      <maxHistory>7</maxHistory>
    </rollingPolicy>
  </appender>
  <logger name="com.pb.engageone.event.document" level="INFO"
additivity="false">
    <appender-ref ref="document-events" />
  </logger>

```

Log format

Event logs are logged with the following format:

```

<EventClassName>[name=<descriptive name for event>,
eventType=<event type>,processType=<process type>,
userId=<user id>,
externalId=<external id>,
externalSource=<external source>,
comments=<comments>,startTimestamp=<start time>,
endTimestamp=<end time>,workflowName=<workflow name>,
workflowEventName=<workflow event name>,
properties=[Pair[key=<key>,value=<value>],...]]

```

Attachment conversion process

Events related to the attachment conversion process are logged and can be viewed in the event monitor.

The following table presents the event monitor logging which tracks the attachment conversion process.

Status	Description
CreateJob (INFO)	call <code>createJob()</code> for a given attachment.
Started (INFO)	call <code>startJob</code>
Pending (INFO)	before entering the polling iteration in the <code>ConversionPoller#call</code> <code>conversionProxy.jobStatusResponse(success/error)</code> after each polling iteration in the <code>ConversionPoller#call</code>
Finished (INFO)	after <code>pollForResultURI</code> has completed
Failed (ERROR)	if any exception occurs - <code>startJob</code> or during the polling iteration

9 - Batch processing

Batch uses the configuration framework to house its configuration settings and will not require unique settings per node in a cluster. See [DBCS support](#) for more information. To change the domain controller (DC) encrypted password, see "Updating database user passwords." Note that neither of these processes run the standard Generate batch process.

Running purge is an important maintenance activity to prevent gradual performance degradation due to the build-up of data in the database tables. Make sure you implement a purge routine as part of your EngageOne Server maintenance plan.

In this section

Accumulated batch.....	168
Non-accumulated batch.....	175
NA batch setup and performance optimization.....	200
Purging using batch.....	204
Log file location.....	209
Logging information (accumulated and non-accumulated batch).....	209
Modifying log file size and count.....	210
Batch error handling.....	212
Batch threshold settings.....	214
Setting up the batch programs on a separate machine.....	217
Running batch with integrated security database connections	219



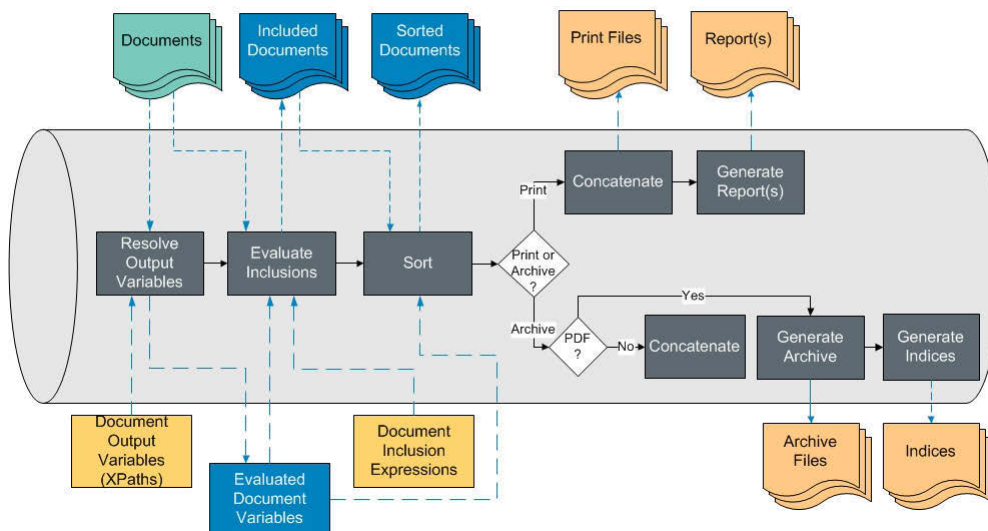
Accumulated batch

The process of submitting documents for batch delivery in EngageOne Interactive is referred to as accumulated batch.

These documents are queued up or accumulated until you start, or schedule the batch job. Output variables are resolved during composition and saved with the intermediate file in the content repository.

Using this method you have one input file and one HIP file; one pre-defined set of output files is generated. The following diagram illustrates the batch accumulation pipeline.

Batch Accumulation Pipeline



In accumulated mode an EngageOne batch job is executed for a single delivery channel. Multiple batch jobs can be triggered by a single schedule through development of a batch script, which is typically a .bat or .sh file. Any standard scheduling product can be used, for example, Autosys, CA-7, Windows Scheduler, etc.

Running the accumulated batch job

You can run the batch job from the command line or scheduler by running one of these scripts. You will find all scripts in the `bin` folder beneath the Batch bundle installation directory.

run

Function	Runs a batch process. Note that this is a general script that accepts a single command and several arguments.
-----------------	---

Command Line Interface

Script	<code>prompt> run [commands] [options]]</code>
--------	---

Examples:

```
prompt> run -domain EngageOneDevDomain -verbose
```

```
prompt> run -domain EngageOneDevDomain -channel print archive
```

```
prompt> run restart -domain EngageOneDevDomain -batch 101
```

```
prompt> run resume -domain EngageOneDevDomain -batch 101
```

Commands

resume	Resumes a batch process.
--------	--------------------------

restart	Restarts a batch process.
---------	---------------------------

Options

-help	Prints the syntax and usage information for this batch job.
-------	---

-batch	Resumes/restarts the batch ID provided as an argument.
--------	--

-channel	The name of the channel to run provided as an argument. Must be either a print, archive, or EngageOne Deliver delivery channel. Note: Multiple channel assignment can be made as required.
-domain	The name of the community for the batch process provided as an argument. (Required)
-logdoevents	Places a message in event monitor for each successful document. The default setting is not to log these messages.
-maxLogFileCount	Sets the maximum log file number of log files to keep using the argument provided. The default is 10. (Optional)
-maxLogFileSize	Sets the maximum log file size of each log file with the argument provided. The default is 10(MB). If you reach the maximum log file size, the server rolls over to a new file and deletes the oldest log file if the maximum log file count has been reached. Using the default settings, that equals a maximum of 10 log files with a maximum of 10MB each. (Optional)
-quiet	Suppresses writing log output to the console. Can be used in combination with -verbose. (Optional)
-verbose	Writes detailed information to the log. This is the default setting. A batch runs in DEBUG mode when this option is used. If -verbose is not present, A batch runs in INFO mode. Can be used in combination with -quiet. (Optional)
-recover	Used by the ONH files recovery mechanism described on ONH file recovery mechanism on page 174 (Optional)

run-channel

Function

Batch run for channels.

The syntax for `run-channel` is different than `run`. You do not specify the domain and channel as `-community {community_name} -channel {channel_name}`. See the example for the correct syntax.

Command Line Interface

Script `prompt> run-channel {domain} {channel}`

Examples:

```
prompt> run-channel EngageOneDevDomain Print_Channel
```

```
prompt> run-channel EngageOneDevDomain Print_Channel Archive_Channel
```

Options

<code>{domain}</code>	The community name to use to start the batch.
<code>{channel}</code>	The name of the channel(s) to run.
<code>-logdocevents</code>	Places a message in event monitor for each successful document. The default setting is not to log these messages.
<code>-maxLogFileCount</code>	Sets the maximum log file number of log files to keep using the argument provided. The default is 10. (Optional)
<code>-maxLogFileSize</code>	Sets the maximum log file size of each log file with the argument provided. The default is 10(MB). If you reach the maximum log file size, the server rolls over to a new file and deletes the oldest log file if the maximum log file count has been reached. Using the default settings, that equals a maximum of 10 log files with a maximum of 10MB each. (Optional)

restart-batch

Function

Restarts a batch job for the given batch ID. `restart-batch` resets the status to Composed for all records with the matching batch ID and then restarts the job.

Use `restart-batch` when you want to run a job again that initially contained errors or stopped unexpectedly and you want the output to contain all records in the job. A batch job can also be restarted by using the proper argument in the `run` function. See `run` for more information.

`restart-batch` retains the original value of output variable. If an output variable that was used during the batch run has been modified, restarting the batch still takes the original value of the variable as it was when the batch was run and the batch ID was assigned. This behavior does not apply to system variables such as system date, system time, etc.

The syntax for `restart-batch` is different than `run`. You do not specify the domain and batch ID as `-community {community_name} -batch {batch ID}`. See the example for the correct syntax.

Command Line Interface

Script	<code>prompt> restart-batch {domain} {batch ID}</code>
Example	<code>prompt> restart-batch EngageOneDevDomain 101</code>

Options

<code>{domain}</code>	The community name to use to restart the batch.
<code>{batch}</code>	Restarts the specified batch ID.
<code>-logdocevents</code>	Places a message in event monitor for each successful document. The default setting is not to log these messages.
<code>-maxLogFileCount</code>	Sets the maximum log file number of log files to keep using the argument provided. The default is 10. (Optional)
<code>-maxLogFileSize</code>	Sets the maximum log file size of each log file with the argument provided. The default is 10(MB). If you reach the maximum log file size, the server rolls over to a new file and deletes the oldest log file if the maximum log file count has been reached. Using the default settings, that equals a maximum of 10 log files with a maximum of 10MB each. (Optional)

resume-batch**Function**

Resumes a batch job for the given batch ID. `resume-batch` resets the status to Composed for all records with the matching batch ID and that were marked as errored the last time the job was run. It also processes any records that were not processed during the original run due to the job being stopped because of a system failure or an error threshold being reached.

Use `resume-batch` when you want to run a job again that stopped because of system failure or errors, but you only want the output to contain records in the job that initially failed or had not yet been processed when the job stopped. A batch job can also be resumed by using the proper argument in the `run` function. See `run` for more information.

The syntax for `resume-batch` is different than `run`. You do not specify the community and batch ID as `-community {domain_name} -batch {batch ID}`. See the example for the correct syntax.

Command Line Interface

Script	<code>prompt> resume-batch {domain} {batch ID}</code>
--------	--

Example	<code>prompt> resume-batch EngageOneDevDomain 101</code>
---------	---

Options

<code>{domain}</code>	Community name used to resume the batch.
-----------------------	--

<code>{batch}</code>	Resumes the specified batch ID.
----------------------	---------------------------------

<code>-logdoevents</code>	Places a message in event monitor for each successful document. The default setting is not to log these messages.
---------------------------	---

<code>-maxLogFileCount</code>	Sets the maximum log file number of log files to keep using the argument provided. The default is 10. (Optional)
-------------------------------	--

<code>-maxLogFileSize</code>	Sets the maximum log file size of each log file with the argument provided. The default is 10(MB). If you reach the maximum log file size, the server rolls over to a new file and deletes the oldest log file if the maximum log file count has been reached. Using the default settings, that equals a maximum of 10 log files with a maximum of 10MB each. (Optional)
------------------------------	--

Running batch on a separate machine

The batch programs can run on a separate machine other than the EngageOne Server provided that it has access to the database, the EngageOne Server JNDI, and the EngageOne Server repository.

If the servers are behind a firewall, make sure that the JNDI and database ports are open and that file sharing of the EngageOne Server repository is accessible to the user who is logged into the machine from where the batch program is being executed.

Note that to execute batch remotely, the active drive must be located either on a mapped directory or a UNC path directory.

For details on how to setup a batch client machine see the section titled "[Setting up the batch programs on a separate machine](#)".

ONH file recovery mechanism

The mechanism, which recovers missing ONH files, is part of the batch bundle and it can be invoked by running existing 'run.bat' script and providing '-recover' flag.

To recover ONH files:

1. Obtain batch IDs of failed batch processes by executing DB query:

```
SELECT distinct BatchId
FROM TDS_RequestChannels
WHERE RequestStatusId = 7
AND Immediacy = 'B'
AND BatchId is not null
AND DomainId = (select Domain_ID from System_Domain where Domain_Name
= '{community name}');
```

2. Run batch ONH recovery command for the first BatchID:

```
run.bat -domain {community name} -batch {batch process identifier}
-recover
```

3. Resume the batch process for the first BatchID:

```
resume-batch.bat {community name} {batch process identifier}
```

On completion, repeat steps 2-3 for each BatchID returned by the DB query.

Non-accumulated batch

In NA batch, which is a traditional batch process, the batch process accepts one or more input files containing document data sets, as well as optional processing instructions.

NA batch processes the document data sets using the specified templates and delivery options to compose output documents. Supported input file formats include XML, delimited, and fixed-length data files. In Designer, fixed-length files are referred to as keyed records. The primary difference between NA batch and accumulated batch is that with accumulated batch, requests are pre-composed and accumulated in a database from on-demand and interactive delivery requests.

NA batch works with both interactive templates and non-interactive templates. Interactive templates always use the interactive data model. Input files that use the interactive data model, or the data sets contained within the input files, must have a complete set of data for each data set because there is no runtime interaction to gather additional data. Non-interactive templates can use one of the following data models: keyed record, delimited, or custom XML. For more information about the data model input files, see [Non-accumulated batch input files](#).

As with accumulated batch, you can use NA batch to perform batch print, archive, report file generation with inclusion conditions, and sorting using the same configuration options in EngageOne Administration.

If Batch fails with **Time out while waiting for composition resources** you should increase the time out setting for document composition. All graphics applications will take considerable time to complete. The `doclgen.time.out` setting is located in `config-settings.xml`. The default is 120 seconds.

Express Batch

The **Express Batch** processing mode can significantly improve NA Batch performance, in this mode NA Batch skips additional configuration provided by Engage One Server during delivery channel definition, also reducing many I/O operations. For available options available for this mode refer to [Express Batch](#) on page 192.

Express Batch applies to non-interactive templates created in Designer only. During Designer's **Publish for EngageOne**. Refer to the Publishing for EngageOne section in the Designer User's Guide for detailed information on configuring templates for **Express Batch** processing.

Modes and features

Batch operating modes presented in the table below are in order of performance, from left (fastest) to right.

	Express Batch	Single composition single file	Single composition	Double composition
Type Print	✓	✓	✓	✓
Type Archive	✓	✗	✗	✓
Type Email	✗	✗	✗	✗
Type EO Deliver	✗	✗	✗	✓
Type Fax	✗	✗	✗	✗
Recipient	✗	✗	✗	✓
Inclusion conditions	✗	✗	✗	✓
Sort	✗	✗	✗	✓
Report files	✗	✗	✓	✓
Output variables	✗	✗	✗	✓
\${SYSTEM_TIME} \${INCREMENT} \${SYSTEM_DATE}	✓	✓	✓	✓
File partition number	0	0	1	0 - n
OPS config file (Symbols support)	✓	✗	✗	✗
Input file splitting	✗	✗	✓	✓

Setup

Batch capabilities are packaged in the batch bundle and are installed at the `batch.install.dir` location configured in `deploy.properties`.

If you want to run multiple batch programs in parallel, it is recommended that you run each program on a different machine. See [Setting up the batch programs on a separate machine](#) on page 217.

Limitations

- No two batch programs should use the same input directory (**-dir** option) at the same time unless they have different delivery option configured.
- No two batch programs containing the same delivery option should be processed together. Two batch programs processing the same delivery channel at the same time may cause unexpected results. For example, their print files may override each other.
- Memory and CPU utilization should be taken into account when running multiple batch programs.
- NA batch failed events logged and found in Diagnostics may or may not include all details about the failure. These events serve as an indicator that something went wrong. Review the NA batch logs for more details about the failure. For more information, see ["System diagnostics"](#).
- Only `${SYSTEM_TIME}`, `${SYSTEM_DATE}`, `${INCREMENT}` and Reserved/Journal output variables can be used in report files in single composition. Using any other output variables will force double composition mode.

Input files

NA batch works with both interactive templates and non-interactive templates. Interactive templates always use the interactive data model. Non-interactive templates can use one of the following data models: keyed record, delimited, or custom XML.

There are a number of areas which you need to consider to achieve optimum performance for your non-accumulated batch jobs. This is discussed in detail in the [NA batch setup and performance optimization](#) on page 200 section.

Interactive templates

Interactive templates use only the interactive data model. With the interactive data model, non-accumulated batch processes an XML file to generate documents. The input XML contains one or many `<records>` nodes, which can have zero or one `<template ...>` nodes, zero or one `<deliveryOption ...>` node and one or many `<answer>` nodes. It is possible to use a different template or delivery option by creating them as children of different `<records>` nodes in the XML. When combined with the command line options for non-accumulated batch, the XML format allows for much flexibility when defining the resources to be used for the data to be processed. For more information, see [Using interactive templates](#) on page 178.

Non-interactive templates

EngageOne Server also supports non-interactive templates that utilize Designer data formats such as keyed record, delimited and custom XML. These formats are in addition to the interactive data model data format, so that you can take advantage of these data formats when using the NA batch processing. Data related to non-interactive templates is validated by Generate, not by EngageOne Server. For more information, see [Using non-interactive templates](#).

Regarding non-interactive templates that use fixed length or keyed record input and that use either Recipient or Document Selector: If the value defined in EngageOne is longer than the field length defined in Designer, an error occurs. The error returned states that the value is longer than the field allows. If the condition defined in Designer is longer than the field length, Generate will not produce a document. In this case, a message may not be logged, but 0 paged documents will be generated.

Using interactive templates

Each template has its own unique answer file, which contains the data that will be used in the generated document. Within the `<answer>` nodes are CDATA blocks that contain answer file XML for the given template. Before developing your own input XML, you will need to make sure that you can produce the answer file XML with the intended data.

The samples show three possible ways that the input XML might be implemented.

- **Example A** shows an overridden record. This record uses a different template, effective date, delivery option and one answer node.
- **Example B** shows a record with no overrides and multiple answer nodes, and would rely on information defined in the command line parameters.
- **Example C** shows a record with an overridden `deliveryOption` and one answer node.

Note that the following examples are not intended to be used in a live environment as each individual EngageOne installation will be different, but they are provided to show the flexibility of the input XML format. For a live EngageOne environment, input XML will need to be developed based on the templates, delivery options, and specific configuration.

Note: The effective date can be specified in the batch XML file.

```
<?xml version="1.0" encoding="utf-8"?>

<batch xmlns="http://pb.com/EngageOne/batch-input">

  <records>

    <!--
    * Example A of an overridden record. This record uses a different
      template, effective date, and delivery option.
    * temp@name should be the logical path of the template
    -->
```

```

<template name="EngageOne Template 2"
  version="1.0"/>
<deliveryOption name="Only AFP"/>

<answer id="1">
  <![CDATA[
    <InteractiveDataModel>
      <Publication>
        <String>EngageOne Scaling String</String>
        <Integer>1234</Integer>
        <Number>12345</Number>
        <Date>2008-10-08</Date>
        <DeliveryInformation>
          <Recipient />
          <EmailToAddress />
          <EmailFromAddress />
          <EmailSubject />
          <EmailBody />
          <FaxNumber />
        </DeliveryInformation>
        <glprivate/>
      </Publication>
    </InteractiveDataModel>
  ]]>
</answer>

</records>

<records>

  <!--Example B:Non-overridden records.-->

  <answer id="2">

```

```

<![CDATA[
<InteractiveDataModel>
  <Publication>
    <String>EngageOne Scaling String</String>
    <Integer>1234</Integer>
    <Number>12345</Number>
    <Date>2008-10-08</Date>
    <DeliveryInformation>
      <Recipient/>
      <EmailToAddress/>
      <EmailFromAddress/>
      <EmailSubject/>
      <EmailBody/>
      <FaxNumber/>
    </DeliveryInformation>
    <glprivate/>
  </Publication>
</InteractiveDataModel>
]]>
</answer>

<answerid="3">
  <![CDATA[
    <InteractiveDataModel>
      <Publication>
        <String>EngageOne Scaling String</String>
        <Integer>1234</Integer>
        <Number>12345</Number>

```

```

        <Date>2008-10-08</Date>
        <DeliveryInformation>
            <Recipient/>
            <EmailToAddress/>
            <EmailFromAddress/>
            <EmailSubject/>
            <EmailBody/>
            <FaxNumber/>
        </DeliveryInformation>
        <glprivate/>
    </Publication>
</InteractiveDataModel>
]]>
</answer>
<answer id="4">
    <![CDATA[
        <InteractiveDataModel>
            <Publication>
                <String>EngageOne Scaling String</String>
                <Integer>1234</Integer>
                <Number>12345</Number>
                <Date>2008-10-08</Date>
                <DeliveryInformation>
                    <Recipient/>
                    <EmailToAddress/>
                    <EmailFromAddress/>
                    <EmailSubject/>

```

```

        <EmailBody/>
        <FaxNumber/>
    </DeliveryInformation>
    <glprivate/>
</Publication>
</InteractiveDataModel>
]]>
</answer>
</records>
<records>
<!--Example C:Another overridden record. This record specifies a
different
    delivery option.-->
<deliveryOption name="Only PDF"/>
<answer id="5">
    <![CDATA[
        <InteractiveDataModel>
            <Publication>
                <String>EngageOne Scaling String</String>
                <Integer>1234</Integer>
                <Number>12345</Number>
                <Date>2008-10-08</Date>
                <DeliveryInformation>
                    <Recipient />
                    <EmailToAddress />
                    <EmailFromAddress />
                    <EmailSubject />
                    <EmailBody />
                    <FaxNumber />

```

```

        </DeliveryInformation>

        <glprivate/>

    </Publication>

</InteractiveDataModel>

]]>
</answer>

</records>

</batch>

```

Before using your custom input XML files in your environment, you will need to use the schema below to validate the input XML. A copy of this XSD can also be found in your batch bundle installation at:

<Batch bundle installation folder>/bin/batch-input.xsd

```

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="http://pb.com/EngageOne/batch-input"
    targetNamespace="http://pb.com/EngageOne/batch-input"
    elementFormDefault="qualified">
  <xs:element name="batch">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="records" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="records">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="template" minOccurs="0" />
        <xs:element ref="deliveryOption" minOccurs="0" />
        <xs:element ref="answer" minOccurs="0" maxOccurs="unbounded" />
        <xs:element ref="data" minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="template">
    <xs:complexType>
      <xs:attribute name="effDate" type="xs:string" />
      <xs:attribute name="name" type="xs:string" use="required" />
      <xs:attribute name="version" type="xs:string" />
    </xs:complexType>
  </xs:element>
  <xs:element name="deliveryOption">
    <xs:complexType>
      <xs:attribute name="name" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>

```

```

    </xs:complexType>
  </xs:element>
  <xs:element name="answer">
    <xs:complexType mixed="true">
      <xs:attribute name="id" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="data">
    <xs:complexType mixed="true">
      <xs:attribute name="file" type="xs:string" use="required" />
      <xs:attribute name="startOfPublication" type="xs:string" />
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Using non-interactive templates

NA batch can process non-interactive templates that include an input file that contains data in any format supported by Designer and generate the appropriate content as defined by the selected output channels. A non-interactive template refers to an EngageOne template that represents a publication that is intended for non-interactive, batch processing.

This type of template is not available for use by the EngageOne Interactive application.

Templates of this type can be designed to utilize any one of the data formats that Designer supports; Keyed Record, delimited, custom XML, as well as the interactive data model.

In addition to NA batch processing, non-interactive templates are available for use with the `deliverDocument` web service. For more information about this web service, see “`deliverDocument()`” in the EngageOne Programmer’s Reference Guide.

You can also create user variables that map to the data model of a template regardless of the data format that the template implements so that these variables can be used by NA batch and OnDemand web services. Pre-defined system variables that are mapped to the interactive data model are ignored during processing of non-interactive templates. For more information, see **"Output variables"**.

The Designer file formats supported are:

- Keyed Record
- Delimited Data Format
- Custom XML

For more information about these file formats, see the *EngageOne Programmer’s Reference Guide*.

There are two options for running NA batch with non-interactive data models:

- Use an XML processing file. This file contains the control information that defines the template and delivery options. It also references the XML, delimited, or fixed-length data files.

- Insert additional processing information for template, delivery option and optional effective date directly into your XML, delimited, or fixed-length data files. For delimited and fixed-length files, only CSV and SVD files are supported. For more information about using this option see [Native Input File Support](#).

EngageOne Video PURL generation

If you are using non-accumulated batch processing with templates that contain EngageOne Video personalized URLs (PURLs), then PURL values must be generated for each of the template data fields that contain video links.

PURL generation is performed as part of the data preparation phase of the non-accumulated batch.

EngageOne Video Server is responsible for PURL generation. You must perform a bulk upload of customer data to EngageOne Video for each video that you wish to generate PURLs. The PURLs generated by EngageOne Video must then be incorporated into the non-accumulated batch input data file prior to running the batch.

EngageOne Video provides two mechanisms for bulk upload of customer data:

- Data File Uploader - suitable for uploading up to approximately 10,000 records via the management portal.
- Batch processor - suitable for uploading large numbers of records on a regular basis via SFTP.

Refer to the *EngageOne Video Service Administration Guide* for further information.

NA batch and XML processing files

The input folder contains an XML processing file just as it does for the Interactive Data Model. This file provides process-specific values for the template, template effective date, and delivery option for a given records section. The XML also identifies the data file that is to be used for the batch/records instance.

Note that referenced data files must be in a separate folder from the XML processing files.

Using the XML processing file option, you are limited in the batch job to specification of a single template, effective date, and delivery option per input file. This means you must split your input data files by these key characteristics (for example, by template and delivery option). If this option is not flexible enough, use the Native Input File Support option described in the next section.

Here is an example of how to process a template that implements the delimited data model using the NA-batch input files:

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
  <batch xmlns="http://pb.com/EngageOne/batch-input">
    <records>
```

```

        <template name="CSV Fed Invoice_v1"/>
        <deliveryOption name="OPTION01"/>
        <data file="linked.csv"/>
    </records>
    <records>
        <template name="Some Keyed Record File"/>
        <deliveryOption name="OPTION02"/>
        <data file="linked_2.svd"/>
    </records>
/batch>

```

Your input XML data file may contain data for multiple output documents. In this scenario, the input file will be divided into output files in as follows (in order of precedence):

1. Using the attribute, `startOfPublication` provided in the data element of the trigger file. In this attribute, the XPath to the element containing a single piece of document data should be provided. The input XML is divided into XML which contains a single instance of the element pointed to by the `startOfPublication` XPath, and the rest of the input XML is repeated in each output XML.
2. As detailed in the paragraph above, the `startOfPublication` XPath is not taken from the trigger but the template master XML file. Note that the `startOfPublication` may be missing in older templates, in which case this method of dividing output files is skipped.
3. If there is no `startOfPublication` in the trigger file or the template master XML, the input XML is divided on the first element after the root node.

The referenced data file `linked.csv` contains delimited-format records. These record sets are partitioned as they have been defined in Designer. EngageOne Server provides no way to identify a record set. This file contains the data for multiple document compositions. NA batch reads through the data file and partitions out the record sets that are then be processed as individual compositions.

```

C001, Gertrude, Forsythe, 632 West Fifth Street, , Covington, KY, 41011, Sep
1,
    Sep 30, September, 1, 30, 2005
K001, 519-346-8, 8000, 10000, 4, 5, 3905, 1905, ,
K002, Sep 2, Transfer to savings 3334411, , 500, , , ,
K002, Sep 7, Point of sale debit - JC Penney, , 100, , , ,
K002, Sep 9, Check # 1432, , 200, , , ,
K002, Sep 11, Check # 1433, , 100, , , ,
K002, Sep 15, Deposit, 1900, , , , ,
K002, Sep 16, Transfer to savings 3334411, , 500, , , ,
K002, Sep 21, Check # 1434, , 475, , , ,
K002, Sep 23, Check # 1435, , 30, , , ,
K002, Sep 30, Deposit, 2000, , , , ,
K002, Sep 30, Interest Paid, 5, , , , ,
S001, 3334411, 13990, 15000, 0, 10, 1000, 0, ,
S002, Sep 2, Transfer from checking 519-346-8, 500, , , , ,
S002, Sep 16, Transfer from checking 519-346-8, 500, , , , ,
S002, Sep 30, Interest Paid, 10, , , , ,
D001, 998123-B, 19900, 20000, 0, 100, 5, 5, "November 6, 2005", 17000, ,
H001, 233-0124, 199000, 200000, , 400000, 201000, 1000, 250000, 30, 6.5, 1400

```

```

C001,Manny,Ortiz,5424 WintercreekDrive,Suite 200,Glen Allen,VA,23060,Sep
1,
    Sep 30,September,1,30,2005
D001,168123-B,19900,20000,0,100,5,5,"November 6, 2005",17000,,
H001,933-0124,199000,200000,,400000,201000,1000,250000,30,6.5,1400

```

Based on the data model, NA batch identifies where each data block begins. In this example, the C001 record indicated the start of each data block.

NA batch and native input file support

NA batch supports the processing of keyed record and delimited-format data model types directly as input files. This eliminates the need for an XML processing file to provide control information for the non-XML data file.

The data file is placed in the input folder that is provided to NA-batch. The control record format is:

Header value, template name, <effective date>, delivery option

The header value can be any character string you choose, but must be consistent after established within a data file and does not conflict with any of the data records.

The data file must begin with a header value record when the data file is presented to NA batch in native mode. This is true even if the NA batch process is launched with parameters that identify a template and delivery option. In this case, the control record could contain only the header value:

```

XXXXXXXXXXXX, , ,

```

In this delimited-format native input file example, the first set of data is used by the `User-form` template and delivered to the `PDF-BATCH` delivery option. The second and third sets of records are used to compose the same template, but are delivered to the `EMESSAGING-BATCH` delivery option.

```

XXXXXXXXXXXX,User-form,2011-06-01,PDF-BATCH
001,Smith,"101, Princes Street",ED1 4DH
002,Debit,89.01,"Cash, charges apply"
XXXXXXXXXXXX,, ,EMESSAGING-BATCH
001,Smith,"101, Princes Street",ED1 4DH
002,Debit,89.01,"Cash, charges apply"
001,Smith,"101, Princes Street",ED1 4DH
002,Debit,89.01,"Cash, charges apply"

```

Control information can also be embedded within the input data file to provide control information to NA batch. This is useful if you want to swap template or delivery option per customer. If this information is not provided, the parameters to the batch job can be used to provide these identifiers for the entire job. NA batch uses the specified template and delivery option for all records it is processing sequentially until a another processing instruction is found.

In this keyed record example NA batch requires that the first record in the data file provide the header value to control information. Based on this, if any data record within the data file begins with

XXXXXXXXXX then that record is considered a control record and the remaining data can identify a delivery option, template name, and template effective date. All of these values are optional after they have been established. If no control record is provided between one data block and another, NA batch assumes that the same template and delivery option are to be used as previously defined. If some element of the control record is not provided, the same rule applies and the previous value is maintained pending a subsequent change.

```
XXXXXXXXXXXX,insurance-application,,PDF-BATCH
100099756345Jack Laurence  945 Elm Street
PO Box 1324    Atlanta
1500 .13  180.05 09092000  168.27  168.27
8.32100920001200099931112400 180.05.00
2000EE1.1012434 10092000 35862 09092000 34812 1050  178.0810092000
343209092000  299830
2500E145610501120145613341120 987 992 818 754 768 887 9571010
3000GG1.1007135 10092000266551 09092000266537  14 1.50  15.97
3500 45 14 35 42 45 40 30 20 10 11 9  12 12 13
5000SS .1032312 10092000  46.00 09092000  46.00 10.00  56.00-DYNA
6000HH1.0023239 10092000  30.00 09092000  30.00  30.00
8000Msg3Msg4ALL
9999
XXXXXXXXXXXX,insurance-application,2011-06-01,
100065445431Robert Jones  1431 Elm Street
Lanham
1500 .13  69.54 09092000  78.27  78.27  8.32100920001200099931112400
180.05.00
2000EE1.1012434 10092000 24500 09092000 23455 1045  1  69.5410092000
343209092000  299830
2500E16021045116013201602145412211032 956 949 852 833 957 923
3000GG1.1007135 10092000 12821 09092000 12807  14 1.50  15.97
3500 62 14 25 45 62 53 48 26 16 12 14 16 19 22
8000Msg1Msg4ALL
9999
```

Running the non-accumulated batch job

A single batch run constitutes a set of files (one to many) where each file can have multiple records.

A command-line program allows you to specify the directory where the files are located. The batch run can also be executed as a scheduled task, for example using Windows Task Scheduler, or using `cron` on Unix-like systems. Be aware of the following:

Note that The command line input only allows specification of an input folder, not an individual file.

- Categorize your input files into separate directories depending on the type of job. The purpose of separate directories is to segregate a data file that is referenced by a control file into a folder other than the input folder to avoid batch picking the data file up more than once.
- The batch program does not alter, move, or delete the input files so these files must be managed at your discretion.

- Unique answer IDs per input file are recommended if you are using the Interactive Data Model. If there is a problem with a particular XML record and there is more than one answer ID in the same input file, you will have to manually determine which of those records with the given answer ID is causing the issue. Note that Answer ID should be an integer type.

Execute the batch program on a machine that has file system access (like via SAN) to the active drive. The machine should also have access to any output directories defined for the batch delivery channels that are expected to be processed.

Note the following:

- `-template`, `-version`, and `-deliveryOption` are optional provided that they are defined in each group of records inside the NA batch input file.
- Batch processes all files with an `.xml` extension name in the directory defined in the `-dir` option.
- The `na.batch.skip.cleanup` system property cleans up all temp files created during NA batch. The default value is `true`.
- The default location for NA batch log files is `< <Batch bundle installation folder>/logs/na-batch.txt`

Command line interface

Script	<code>prompt> run-na-batch.bat [options]</code>
Example	<code>prompt> run-na-batch.bat -domain elDomain -template "EngageOne Simple Template" -deliveryOption "ARCHIVE_ONLY" -dir "c:\batch input"</code>
Java	<code>prompt> java -cp .;el-server-batch.jar;<LIBRARIES> com.pb.engageone.server.batch.na.runner.CommandLineRunner [options]</code>
Example	<code>prompt> java -cp .;el-server-batch.jar;<LIBRARIES> com.pb.engageone.server.batch.na.runner.CommandLineRunner -domain elDomain -template "EngageOne Simple Template" -deliveryOption "ARCHIVE_ONLY" -dir "c:\batch input"</code>

Options

<code>-help</code>	Prints the syntax and usage information for this batch job.
<code>-config</code>	Overrides configuration settings using the properties file in the argument. (optional)
<code>-deliveryOption</code>	Sets the delivery option name used for this batch. This option is overridden by data from input files. Note that this value is required if not present in the input files.
<code>-dir</code>	The directory (absolute path) in which the batch will gather input files. (required).

-domain	The community name for the given batch, associated templates and delivery option with this community. (required)
-logfile	Overrides the log file name with the argument provided. The default log file name is <code>na-batch.txt</code> . If old batch logging location is used, refer to Log file location on page 209 for detailed information, the default log file name is <code>log.txt</code> . When the maximum log file size is reached, a new log file is incrementally created, for example <code>na-batch.txt.1</code> , <code>na-batch.txt.2</code> , ... or <code>log.txt</code> , <code>log.txt.1</code> , <code>log.txt.2</code> , ... (optional)
-maxLogFileCount	Sets the maximum log file number of log files to keep using the argument provided. The default is 10. (optional)
-maxLogFileSize	Sets the maximum log file size of each log file with the argument provided. The default is 10(MB). If you reach the maximum log file size, the server rolls over to a new file and deletes the oldest log file if the maximum log file count has been reached. Using the default settings, that equals a maximum of 10 log files with a maximum of 10MB each. (optional)
-quiet	Suppresses writing log output to the console. Can be used in combination with <code>-verbose</code> . (optional)
-template	<p>The template name or the template logical path. Setting the template name or path, sets it for all entries in the batch input files. This option is overridden by data from input files. Where there are multiple templates with the same name, provide the template logical path to ensure the correct template is used. (required if not present in the input files).</p> <p>Format:</p> <pre>\\<folder1>\\<folder2></pre> <p><template name></p> <p>Example:</p> <pre>-template "\\mainfolder\\subfolder EngageOne Template v1.0"</pre>
-verbose	Writes detailed information to the log. NA batch runs in DEBUG mode when this option is used. If <code>-verbose</code> is not present, NA batch runs in INFO mode. Can be used in combination with <code>-quiet</code> . (optional)
-version	The version number of the template in use

Note that the template logical path is the same as the folder hierarchy you see from the template tab of EngageOne Administration, using `\\` as the folder separator.

For example, the template selected corresponds to `template\\EngageOne Template v1.0` as its template logical path. `\\template\\EngageOne Template v1.0` can be used as well.

In a scenario where there are multiple templates with the same name and the template name is used, batch will list all of templates and prompt for the template path.

Running multiple batch programs in parallel

If you want to run multiple batch programs in parallel, test the parallel jobs on the same machine to make sure you have appropriate capacity. Also, make sure that the batch thread size and DCA pool size are adjusted to accommodate the capacity of the CPU, as each batch program will introduce its own threads and DCA pool. See [NA batch setup and performance optimization](#) on page 200 for configuration details.

Note that care must be taken when configuring thread and pool sizes so as to avoid composition errors due to high CPU utilization.

Please note that non-accumulated batch is a Java program. Java must be installed on any server where non-accumulated batch is going to be executed.

Express Batch

Express Batch processing can significantly improve NA batch performance; in this mode NA batch skips additional configurations provided by Engage One Server during delivery channel definition, thus reducing I/O operations. Resources such as Keymaps, Journals and, Lookup Tables can be assigned to the template in Designer or defined/overridden via OPS settings in EngageOne Server as described below.


- Express Batch processing applies to non-interactive templates created in EngageOne Designer. Refer to [Publishing for EngageOne in Designer](#) on page 194 and the Designer User's Guide for further information.
- File assignments for the template can be made, in the EngageOne Server environment for resources such as:
 - Active content
 - Message content
 - Keymaps
 - Lookup tables

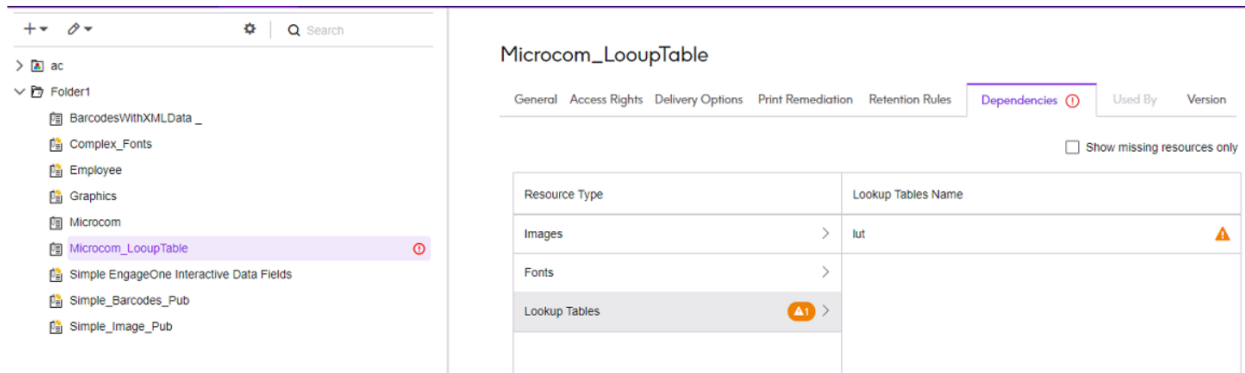
Assignments can be made either through the EngageOne Administration or, at NA-batch execution using an OPS (override) file. Refer to [Using overrides](#) on page 194 for further information.

- The XML processing file used for Express Batch processing allows you to specify the OPS file used for the NA batch run. Refer to [XML processing file](#) on page 195 for further information
- You can dynamically define parameters used in your OPS file using symbols either on the NA batch command line or, within the OPS file. Refer to [XML processing file](#) on page 195 for further information.

Points to note when running NA-batch:

- **Processing restrictions** - to allow for the enhanced NA-batch performance, The following features are not available :
 - Splitting of input data,
 - sorting criteria,
 - inclusion conditions,
 - recipient processing,
 - user defined report processing.
- **Output variable support** - only the following output variables are allowed:
 - \${SYSTEM_TIME},
 - \${SYSTEM_DATE},
 - \${INCREMENT}.

- **Missing resources** - Where templates are missing resources, a  symbol is used in the EngageOne Administration application to flag this, as shown below:



In this case, we are missing Lookup Table resources. The missing resources must be exclusively resolved ***either*** through EngageOne Administration application ***or*** at NA-batch runtime using OPS setting. Note that settings defined in the OPS file take precedence over those made in the Administration application.

- **Multiple template processing** - where multiple templates are published in EngageOne Designer, only the name of the first template will be shown in Template Management. You must specify this template name in the XML configuration used by NA-batch. On successful completion of the batch run output for all templates will be generated.
- **Multiple output device processing** - a delivery channel **must** be specified for **each** output device selected when the template was published in EngageOne Designer, for example:
A template is published using the Express Batch processing option with one PDF and one AFP output device in EngageOne Designer. In this scenario, one PDF delivery channel and one AFP delivery channel **must** be specified in the EngageOne Administration application.
- **Journals for Publishable Active Content** - Only the Structured XML Journal is supported in Publishable Active Content used with Express Batch processing templates. **Any other journal type entries will not appear in the output.**
- **Output path and file name configuration** - You can set the output path and file name via the OPS file or in the Administration application. It is important to note that assignments made in the OPS file take precedence over those set in the Administration application.

Publishing for EngageOne in Designer

While publishing for EngageOne in Designer, select the option **Express Batch**, in the **Publish for EngageOne** task. The result is a ZIP file containing resources required for onward processing by EngageOne. The ZIP file contains a *template_Master.xml* file which holds configuration information. A Express Batch compatible template will have its *Bypass OCP* field set to **true** as highlighted below.

```
<?xml version="1.0" encoding="utf-8"?>
<!--XML meta file for EngageOne Template-->
<document ocmmetafile-vers="ocm-metafile2.0" ocm-vers="OCM Template V2.0">
  <OcmMetadata>
    <documentClass>
      <metadata-field fieldName="DOC1 Repository ID" fieldValue="dabb879f7ec643158c94ff824591a74a" defaultValue="" fieldType="text" />
      <metadata-field fieldName="DOC1 Repository Name" fieldValue="DOC1_Repository" defaultValue="" fieldType="text" />
      <metadata-field fieldName="DOC1 Repository Alias" fieldValue="DOC1_Repository" defaultValue="" fieldType="text" />
      <metadata-field fieldName="DOC1 Repository SQL" fieldValue="TECHAUTHVM2\DOC1SQLEXPRESS" defaultValue="" fieldType="text" />
      <metadata-field fieldName="DOC1 Strand" fieldValue="test doc" defaultValue="" fieldType="text" />
      <metadata-field fieldName="DOC1 Strand Guid" fieldValue="DDB0607172E9449A847D5423CCF29075" defaultValue="" fieldType="text" />
      <metadata-field fieldName="DOC1 Template Type" fieldValue="non-interactive" defaultValue="" fieldType="text" />
      <metadata-field fieldName="Publish Date" fieldValue="05-Mar-2019" defaultValue="" fieldType="text" />
      <metadata-field fieldName="Published By" fieldValue="Administrator" defaultValue="" fieldType="text" />
      <metadata-field fieldName="Type" fieldValue="TEMPLATE" defaultValue="" fieldType="text" />
      <metadata-field fieldName="Design Publication Name" fieldValue="1st Template" defaultValue="" fieldType="text" />
      <metadata-field fieldName="Graphic Support" fieldValue="False" defaultValue="" fieldType="text" />
      <metadata-field fieldName="UNICODE Support" fieldValue="False" defaultValue="" fieldType="text" />
      <metadata-field fieldName="Bypass OCP" fieldValue="True" defaultValue="False" fieldType="text" />
    </documentClass>
  </OcmMetadata>
</document>
```

Where this is the case NA batch can run in **Express Batch** mode.

Using overrides

You can either specify resources used by the template in Designer in the **Publish for EngageOne** task or by using an override production settings (OPS) file; this allows you to specify supplements and alternatives to some of the job settings that were used in Designer. Note that the use of overrides is valid only for templates run Express Batch processing mode.

An example OPS file is shown below:

```
<input>
Datainput=C:\NABatch\SET2\data\Letters-ORG.xml

<KeyMap>
Showroom=ACompany.xml
Showroom=C:\key\keymap4.xml

<LookupTable>
Dealers=C:\NABatch\SET2\data\Dealerships.ETS

<Output>
Default PDF=c:\myoutputs\SET2\ACompany-KeyedImages-Lookuptables.pdf

<DIJ>
AFP300=c:\myoutputs\SET2\1.jrn

<trace>
Outputfile=c:\myoutputs\SET2\Trace.out
```

```
TraceLevel=Complete
; TraceLevel=default
; TraceLevel=verbose
; TraceLevel=off

OutputCodepage=UTF8
memlimit=0

Supporttrace=on
; supporttrace=off
```

For a complete list of OPS file settings, refer to **OPS file reference** section of the latest EngageOne Designer User's Guide.

XML processing file

The standard XML processing file has been extended to cater for the definition of the OPS file used in Express Batch process via the **additionalOptions** construct.

```
<records>
  <template name="template"/>
  <deliveryOption name="delivery option"/>
  <additionalOptions>
    <ops file="C:\ops\opsconfig.ops"/>
  </additionalOptions>
  <data file="C:\data\datafile.xml"/>
</records>
```

Using symbols

In order to dynamically assign values to OPS configuration files, symbols support is provided in Express Batch processing mode.

Symbols can be used on the command line used to run NA batch using the `-symbols` switch, as shown below:

```
run-na-batch.bat -domain sampledomain -dir C:\nabatchinput -symbols
"outputfolder=C:\output journalfilename=journal.dij"
```

Symbols can also be used in the OPS file, as shown below:

```
<Symbols>
outputfolder=C:\output
journalfilename=journal.dij
```

To reference symbols in the OPS file should be wrapped inside the % sign, as shown below:

```
<Journal>
Document=%outputfolder%\All_Sections_Message_Test.JRN
TLE_Print_Index=%outputfolder%\TLE_Print_Index.JRN
```

In the example that follows two symbols are used to dynamically resolve the paths for the input data and output using the following symbols:

- **Data** uses the value from the command-line,
- **Output** uses the same value from **Data**"

```
<Symbols>
Data=%1%
Output=%Data%

<Input>
DataInput=c:\EngageOne\GenTest\input\%Data%
ActiveContentLocation=c:\EngageOne\gentest\ActiveContent
MessagesFile=c:\EngageOne\GenTest\HIM\*.him

<Journal>
Journal2=c:\EngageOne\gentest\output\%Output%.jrn2
Journal3=c:\EngageOne\gentest\output\%Output%.jrn3

<Output>
Output1=c:\EngageOne\gentest\output\%Output%.pdf
;Output2=c:\gentest\output\%Output%.ps

<Advanced>
LogFile=c:\EngageOne\gentest\output\%Data%log

<DIJ>
Output1=c:\EngageOne\gentest\output\%Output%.xml

;<trace>
;Outputfile=c:\EngageOne\gentest\output\%Data%_trace.log
;TraceLevel=verbose
;outputcodepage=utf8
;memlimit=0
;supporttrace=on
```

Running multiple batch programs on the same server

You can concurrently execute multiple non-accumulated batch instances on the same machine without any additional configuration.

To run multiple batch programs on the same machine:

execute the **run-na-batch** script multiple times with the desired number of batch instances and with the appropriate command line parameters.

It is recommended to use the `-logfile <log file name>` command line option to have different log files for each running batch instance. This is to avoid various logging coming from different batch instances in the same log file.

Running batch on a different machine

The batch programs can run on a separate machine other than the EngageOne server provided that it has access to the database, the EngageOne server's JNDI, and the EngageOne repository.

If the servers are behind a firewall, make sure that the JNDI and database ports are open and that file sharing of the EngageOne repository is accessible to whoever user is logged in the machine from where the batch program is being executed.

SUPPORT: Note that to execute batch remotely, the active drive must be located either on a mapped directory or a UNC path directory.

For more information, see [“Setting up the batch programs on a separate machine”](#).

Batch configuration

These are the non-accumulated batch thread size settings that are used to control the configuration in the **com.pb.engageone.server.batch.na** namespace.

Configuration setting	Description
<code>jdbc.driverClassName</code>	JDBC driver class name. Use jdbc driver of com.pb.engageone.server.batch namespace
<code>jdbc.url</code>	JDBC url Use JDBC URL of com.pb.engageone.server.batch namespace

Configuration setting	Description
<code>jdbc.username</code>	Database user name Use jdbc user name of com.pb.engageone.server.batch namespace
<code>jdbc.password</code>	Database encrypted password Use jdbc password of com.pb.engageone.server.batch namespace
<code>default.file.encoding</code>	Default file encoding Use file encoding of com.pb.engageone.server.batch namespace
<code>csv.report.encoding</code>	File encoding used for delimited report/index generation. Use file encoding of com.pb.engageone.server.batch namespace
<code>fixed.report.encoding</code>	File encoding used for fixed length report/index generation. Use file encoding of com.pb.engageone.server.batch namespace
<code>dij.report.encoding</code>	File encoding used for journal index generation. Use file encoding of com.pb.engageone.server.batch namespace
<code>xml.report.encoding</code>	File encoding used for XML report/index generation. Use file encoding of com.pb.engageone.server.batch namespace
<code>document.error.level</code>	Threshold value for documents in error during batch process before the process is halted. Use value from com.pb.engageone.server.batch namespace
<code>batch.work.dir</code>	Work folder
<code>batch.cache.path</code>	cache folder where template resources, and answer files are temporarily stored
<code>batch.temp.dir</code>	Temporary folder
<code>error.folder.path</code>	Directory at which the errors are being dumped into
<code>workerThread.pool.size</code>	Number of active running threads in processing record for partial composition
<code>recomposition.workerThread.pool.size</code>	Number of active running threads in processing record for re-composition/concatenation process

Configuration setting	Description
<code>documentInfo.list.page.size</code>	Page cache size for internal DocumentInfo List
<code>batch.dca.pool.size.maximum</code>	The number of DCA adapters to use during pre-composition. Also available in the com.pb.engageone.server.batch namespace.
<code>sort.db.vendor</code>	The vendor for the sorting database - MSSQL or ORACLE
<code>sort.db.server.name</code>	The server name or IP address of the sorting database server
<code>sort.db.name</code>	The name of the sorting database - database name or Service name of the database for Oracle
<code>sort.db.port</code>	The port of the sorting database
<code>sort.db.user.name</code>	The user name of the user connecting to the sorting database
<code>sort.db.password</code>	The password of the user connecting to the sorting database
<code>sort.db.connection.pool.size</code>	The number of pooled connections to the sorting database Note: that this connection pool is separate from the pool used on the application server. Make sure that the database can handle the total number of connections you expect to use.
<code>worker.queue.size</code>	This is the maximum number of XML records (not files) that are held in memory at any given time during processing
<code>doclgen.time.out</code>	Time out value of Generate
<code>doclgen.path</code>	Directory where the document server is located

NA batch setup and performance optimization

This section describes the areas that need to be considered to attain the optimum performance for NA batch.

The following topics are covered:

- **Areas affecting NA batch performance** on page 200
- **System and hardware tuning considerations** on page 201
- **Limitations** on page 201
- **Java Virtual Machine memory allocation** on page 202
- **Scaling and performance recommendations** on page 202

Note that you can concurrently execute multiple non-accumulated batch instances on the same machine without any additional configuration. If you choose this operating scenario it is important to be aware of the areas covered in this section.

Areas affecting NA batch performance

An NA batch run can be processed in several ways, based on how the delivery channel rules are configured. Within the delivery channel configuration for your batch run, you can configure rules around inclusion conditions, sorting and partitioning of the output record.

NA batch performance is affected by the following setup items:

- The structure and the number of XML files in your input directory used to trigger batch processing.
- The output variables used for any delivery channel.
- Inclusion conditions - define if a document is included in a batch stream.
- Sorting - defines how documents are ordered in a print file, and any associated report or index.
- Partitioning - defines how the output record of an output document is separated.

The best batch performance is achieved when the following items are adhered to:

- A single XML file in the input file in directory is used to trigger batch processing.
- The XML file used to initiate batch processing contains one `records` element with associated `template`, `deliveryOption`, and `data` elements.
- Inclusion conditions and/or sorting should not be applied to any delivery channel present for the chosen delivery option.
- The only output variables that should be used in any delivery channel present for the chosen delivery option are:

- `${SYSTEM_TIME}`
- `${COMPOSITION_DATE}`
- `${INCREMENT}`
- The **File Partition Number** for any delivery channel present for the chosen delivery option should be set to the same value, either 0 or 1.

Note: Although improved performance can be achieved when the **File Partition Number** is set to 1, a value of 0 will give the best performance with the set up scenario described above.

System and hardware tuning considerations

System and hardware configuration tuning can be useful to increase the throughput of records in the NA batch process. Several observations have been made about the correlation of hardware and software tuning and NA batch throughput:

- Increasing the hardware profile of the database server from a single core, 4 GB system to a 2 core, 8 GB system showed nearly 50% improvement in pages per second (PPS) performance. Increasing to the recommended 8 core, 16 GB database server will maximize the performance of batch.
- Increasing the `workerThread.pool.size` to 30 (default is 5) and `recomposition.workerThread.pool.size` to 30, maximized performance in the system. These configurations are set in the `config-settings.xml` file. Please consult Customer Support if you need help identifying and changing these configuration settings.
- Enable quiet mode in your NA batch execution.

Limitations

Here are some important limitations you need to be aware of :

- No two batch programs should use the same input directory (**-dir** option) at the same time unless they have different delivery options configured.
- No two batch programs containing the same delivery option should be processed together. Two batch programs processing the same delivery channel at the same time may cause unexpected results. For example, the print files may overwrite each other.
- Memory and CPU utilization should be taken into account when running multiple batch programs.

Java Virtual Machine memory allocation

If performance is slow, or if you receive an **Out of Memory** error when processing a large number of records, configure the amount of heap memory that batch will be using before a batch run by adding `-Xmx<MAX_MEM>m` to the command line.

Example:

```
java -cp .;el-server-batch.jar;<LIBRARIES> -Xmx512m
com.pb.engageone.server.batch.na.runner.CommandLineRunner -domain elDomain
-template "EngageOne Simple Template" -deliveryOption "ARCHIVE_ONLY"
-dir "c:\batch input"
```

For more JVM tuning advice, see the link below:

https://docs.oracle.com/cd/E13222_01/wls/docs81/perform/JVMTuning.html

Scaling and performance recommendations

Non-accumulated batch can be configured to maintain the number of threads for scaling and performance tuning of the system. Shown below are the configuration settings in the configuration framework used for tuning performance in batch. These can be modified in **config-settings.xml**.

Namespace: `com.pb.engageone.server.batch.na`

Config setting	Description
<code>batch.dca.pool.size.maximum</code> Default value: <code>\${com.pb.dca.pool.size.maximum}</code>	The number of DCA adapters to use during pre-composition
<code>worker.queue.size</code> Default value: 1000	The maximum number of XML records (not files) that are held in memory at any given time during processing
<code>workerThread.pool.size</code> Default value: 2	The number of active running threads in processing record for partial composition

Config setting	Description
<code>recomposition.workerThread.pool.size</code> Default value: 2	The number of active running threads in processing record for re-composition/concatenation process
<code>documentInfo.list.page.size</code> Default value: 500	The page cache size for internal DocumentInfo List
<code>sort.db.connection.pool.size</code> Default value: 20	The number of pooled connections to the sort repository database.

These values can be increased depending on the machine's capability (memory and CPU). Increasing both DCA pool size and the batch thread size can increase the rate at which records are processed but should be adjusted with caution, taking CPU capacity into account.

Increasing **worker.queue.size** increases the number of documents that can be queued up for the partial composition and variable resolution process, but should be limited depending on the memory availability or the specified heap size in the JVM (as discussed in the Java Tuning White Paper link).

Increasing **documentInfo.list.page.size** increases the number of documents that can be queued up for the concatenation and the report/index generation process but should be limited depending on the memory availability, or the specified heap size in the JVM.

Test scenarios

To illustrate these improvements in version 4.4, test scenarios were run. The result was a significant increase in the number of pages an NA batch run can produce.

- In one scenario, the number of pages per second produced went from **40** to **241**.
- In another scenario, the number of pages per second produced went from **517** to **3223**.

Purging using batch

Running purge is an important maintenance activity to prevent gradual performance degradation due to the build-up of data in the database tables. Make sure you implement a purge routine as part of your EngageOne Server maintenance plan.

You can configure and run batch to purge work items for a given batch channel. This can be as part of a successful batch run. Here are some examples of how you can use the purge program:

Attention: Do not run Purge while Batch is running.

- Purge all successful work items that were processed during this batch run and specified in the run itself.
- Purge all work items for a given channel by channel name.
- Specify a status set to delete work items that have had all channels delivered. Any template that has an un-delivered channel will have none of its channels purged, even if all other channels belonging to the same template have been delivered.
- Purge the data in the database and the files in the file system when a community is deleted.

By default the purge program purges work items created or updated up to the day before the date the purge program is run. It is recommended that you validate work items that are scheduled to be purged.

The purge program supports removal of the following work items:

- Complete and delivered work items.
- Failed work items errors, for example, when work item delivery fails.
- Work items that were already submitted for channel delivery (only delivery status **Composed** and beyond).

Work items are deleted based on the command line options specified to the purge function. Any physical files that are stored in the content repository, including the TDS working folder (activeContent, delivery and preview) and sub-files associated with the work item are also deleted. It also deletes corresponding database entries of those work items that were deleted.

If none of the three options listed in the purge program table below are specified, all work items with either an error or a delivered status will be included in the purge process.

Note: Files generated by batch such as print file, report file, index file and archive file are not deleted.

When a work item is deleted, all work item-related files, work item-related delivery items, and delivery item-related working files are deleted. When a delivery item is deleted, all delivery item-related working files are deleted. If all delivery items of a work item are deleted, the work item is deleted in the purge process, including database entries and related files.

The purge program also cleans up any interim files or database entries generated by delivery failure, such as delivery items with status **New** or **Processing** and the files in the non-accumulated batch working folder.

When the purge program runs it also checks for deleted communities. If any deleted communities are located, the purge program permanently deletes the deleted communities, including all related database entries and related files.

Purge is typically scheduled to run during non-peak hours. If you choose to run purge during peak operating hours, use the `-concurrencyMode` option.

purge

Function

Executes a purge program. The purge behavior can be customized by supplying appropriate options.

- Only the domain parameter is required.
- Running purge with no status options (errored, delivered, pendingdelivery, or canceled) deletes any interim files and database entries corresponding with delivery items having New, Processing and Cancelled statuses. Files in the non-accumulated batch working folder are removed as well. This does not delete resources related to work items pending for delivery. Additionally, all delivered and errored work items are deleted.
- The following options require an argument, which are indicated below as <variables>: `-batch`, `-channel`, `-domain`, and `-logfile`.
- `numDaysRetention` defaults to zero if not specified.
- Work items created or updated on the same day up to the purge start time are not included in the purge process.

It is strongly recommended that you do not attempt to run the purge job for work items created on the same day to prevent removal of documents that might potentially still being processed by the document server.

Also, do not run purge while batch is running.

Command Line Interface:

Script	<code>prompt> purge {options}</code>
--------	---

Example	<code>prompt> purge -errored</code> <code>prompt> purge -delivered -numDaysRetention 3</code>
---------	--

Options

<code>-batch <batch></code>	the batch ID to run the purge
-----------------------------------	-------------------------------

-cancelled	deletes work items marked for removal ('soft deleted'). Does not delete New or Processing items. This does not include errored, pending delivery, or delivered work items.
-channel <channel>	name of the channel to run the purge
-concurrencyMode	allows purge to run concurrently with other EngageOne processes (EngageOne Web Services, EngageOne Interactive, etc.) without impacting the performance of those processes
-delivered	deletes delivered work items in the purge process and not the New or Processing items. This does not include errored, pending delivery, or cancelled items.
-domain <community>	name of the community to run the purge
-errored	deletes work items that have errors in the purge process and not the New or Processing items. This does not include delivered, pending delivery, or cancelled items.
-help	print this message
-logfile <file>	override log file name with given name
-maxLogFileCount	maximum log file count, default is 10MB
-maxLogFileSize	maximum log file size, default is 10
-numDaysRetention	include in the purge process all work items older than the specified number of days (relative to the current date)
-pendingdelivery	deletes items that have submitted and are pending for delivery. This does not include errored, delivered, or cancelled items.
-quiet	does not write log output to console
-verbose	write detailed information to the log

-workItemInDailyfolders

From EngageOne 4.4.7 onwards, work items are created in their own folder on the active-drive. Prior to this release work items were created in the daily folders on the active-drive.

The purge program should be run prior to upgrading to clean up the work items in the daily folders. There may however be situations where not all work items were in a purgeable state at the time of upgrade; this legacy switch is provided to cater for this scenario. It allows the purge program to clean-up any work items remaining in the daily folders.

NOTE: This switch should only be used for the purpose described above and not set by default. If set by default, the purge program will not clean up work items in the new location on the active-drive.

purge-em

Function	Purge old Event Monitor records for the dates given
Syntax	<code>purge-em -onOrBefore {yyyy-MM-dd} [-concurrencyMode]</code>
Parameters	
date	all records that were logged before the given date will be purged
-concurrencyMode	allows purge-em to run concurrently with other EngageOne processes (EngageOne Web Services, EngageOne Interactive, etc.) without impacting the performance of those processes
Example	<code>purge-em -onOrBefore 2009-11-27</code>
	<code>purge - delivered - numDaysRetention 3</code>
Java command example	<pre> jjava -cp <libraries> com.pb.eventmonitor.purge.EventMonitorPurge -onOrBefore 2009-11-27 -verbose </pre>

purge-promotion

Function	Purge all import and export promotion jobs for specific dates.
-----------------	--

Syntax	<code>purge-promotion -onOrBefore {yyyy-MM-dd}</code>
--------	---

Parameters

date	All import, export and asset export promotion-quartz jobs created before or on a specific date will be purged. If the date covers scheduled but not completed jobs, these jobs will be deleted as well.
------	---

Example	<code>purge-promotion -onOrBefore 2017-07-20</code>
---------	---

Java command example	<pre>%JAVA_HOME%\bin\java -cp <libraries> -Dviewpoint.config.externalPropertyFilePath=../conf/viewpoint.properties com.pb.engageone.server.purge.promotion.AssetPromotionPurge -onOrBefore 2017-07-15</pre>
----------------------	---

`delete-old-version`

Function	Deletes all old versions of a template and Active Content, (including database records and physical files) from the content repository.
----------	---

Command Line Interface

Script	<code>prompt> delete-old-version [options]</code>
--------	--

Example	<code>prompt> delete-old-version -verbose</code>
---------	---

Options

-help	prints this message
-------	---------------------

-domain	name of the community to run the purge
---------	--

-quiet	does not write log output to console
--------	--------------------------------------

-verbose	writes detailed information to the log
----------	--

Log file location

By default all batch logs are stored in `<installation location>\batch\logs`, each batch job has its own log file, as follows:

- a-batch.txt (includes logs from run, run-channel, resume-batch and restart-batch)
- na-batch.txt
- purge.txt
- purge-em.txt
- purge-promotion.txt

To restore the previous batch log location (`<installation location>\batch\bin`), set the `separate.log.files.in.dedicated.directory` entry in your batch config-settings.xml to false, as shown below:

```
<namespace name="com.pb.engageone.server.batch"> . . . .
<setting>
<key>separate.log.files.in.dedicated.directory</key>
<value>false</value>
</setting>
. . . .
</namespace>
```

Logging information (accumulated and non-accumulated batch)

Non-accumulated batch is a multi-threaded application that processes large numbers of documents, defined by XML records. For this reason, the NA batch log output is formatted in a specific way to make information gathering easier.

NA batch and accumulated batch each have their own log4j configuration files, by default located in:

```
<Batch bundle installation folder>/bin
```

Accumulated batch has these log files:

- log4j.properties
- log4j-quiet.properties

NA batch uses:

- `na-batch-log4j.properties`
- `na-batch-log4j-quiet.properties`

They are configured to output log messages. For audit purposes you can check the batch logging configuration information in `na-batch-log4j.properties` of the batch directory. You can also modify the `maxFileSize` setting from the **config-settings.xml** to set the maximum file size of the log file before it rolls over. The logging can also be configured to have a log file that rolls over to a new file only once per day see "Daily rolling file appender" below.

Each log entry consists of a number of fields that hold values with different meanings. The number of fields and type of data in each field will be explicitly specified as follows:

For accumulated batch:

```
<TIMESTAMP>|<LOG_LEVEL>|<CLASS_NAME>|<THREAD_NAME>|<SPECIFIC_MESSAGE>
```

For NA batch:

```
<TIMESTAMP>|<LOG_LEVEL>|<CLASS_NAME>|<THREAD_NAME>|<INPUT_ANSWER_ID>|<INPUT_FILE_NAME>|<SPECIFIC_MESSAGE>
```

As in the example above, the fields will be delimited with a "|" character. This allows one to programmatically parse log information from the log file.

Please note the following:

- The `maxFileSize` and `maxBackIndex` settings are configured in the configuration framework see [Modifying log file size and log back up count](#).
- The `log4j.appender.FILE.maxFileSize` and `log4j.appender.FILE.maxBackupIndex` properties in the `log4j.properties` are overridden by the `maxFileSize` and `maxBackIndex` settings in the **config-settings.xml**.
- The `maxFileSize` and `maxBackIndex` command line options overrides the settings in **config-settings.xml**.
- The logfile command line option overrides the log file name in the `log4j.properties` by default it is **log.txt**.

Daily rolling file appender

The logging can be configured such that the underlying log file is rolled over at a chosen frequency. See the link below on how to configure the rolling frequency.

<http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/DailyRollingFileAppender.html>

Modifying log file size and count

This feature allows you to specify the maximum size of batch log file via the configuration framework or adding a parameter when starting up batch. You can also specify the maximum amount of batch

log files to maintain via the configuration framework or adding a parameter when starting up batch (RollingFileAppender).

Use the appropriate namespace; **com.pb.engageone.server.batch** (for accumulated batch), or **com.pb.engageone.server.batch.na** (for NA batch) in the following example:

```
<namespace name="com.pb.engageone.server.batch.na"> . . . .
  <setting>
    <key>log.maxFileSize</key>
    <value>10MB</value>
  </setting>
  <setting>
    <key>log.maxFileCount</key>
    <value>10</value>
  </setting>
  . . . .
</namespace>
```

To enable this feature, you can either edit the **config-settings.xml** file entries, or override these settings by adding the **maxLogFileSize** and **maxLogFileCount** parameters:

- **maxLogFileSize** -- accepts string values, i.e. "1MB", "1GB", "1KB", etc
- **maxLogFileCount** -- accepts integer values, i.e. "1", "10", "25", etc.

Example

```
run-na-batch.bat -dir D:/temp\input1 -domain engageone -deliveryOption
"Batch Print"

-maxLogFileSize 10MB -maxLogFileCount 10
```

log4j.properties

Below is the log4j configuration for RollingFileAppender

```
log4j.rootLogger=INFO, FILE, CONSOLE
log4j.category.com.pb=INFO
log4j.appender.file.encoding=UTF-8
log4j.appender.FILE=org.apache.log4j.RollingFileAppender
log4j.appender.FILE.file=log.txt
log4j.appender.FILE.maxFileSize=10MB
log4j.appender.FILE.maxBackupIndex=9
#log4j.appender.FILE=org.apache.log4j.DailyRollingFileAppender
log4j.appender.FILE.Append=true
#log4j.appender.FILE.DatePattern='.'yyy-MM-dd
log4j.appender.FILE.layout=org.apache.log4j.PatternLayout
#log4j.appender.FILE.layout.ConversionPattern=[%d{MMM dd HH:mm:ss}] %-5p
: %m%n
log4j.appender.FILE.layout.ConversionPattern= [%d{MMM dd
HH:mm:ss}] | %-5p | %c | [%t] | %m%n
#log4j.appender.CONSOLE.encoding=UTF-8
log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender
log4j.appender.CONSOLE.Target = System.out
```

```
log4j.appender.CONSOLE.layout = org.apache.log4j.PatternLayout
#log4j.appender.FILE.layout.ConversionPattern=[%d{MMM dd HH:mm:ss}] %-5p
: %m%n log4j.appender.CONSOLE.layout.ConversionPattern = [%d{MMM dd
HH:mm:ss}] |%-5p| %c | [%t] | %m%n
```

Note that the `DailyRollingFileAppender log4j` config is commented out by default. In the above example, the `batch_` connotation has only been added on the file for clarity as there were other log4j files present in the system. This should not be appended to the name if you change it. Log formatting

The following conversion pattern is used by default in the **log4j** configuration file:

```
[%d{MMM dd HH:mm:ss}] |%-5p| %c | [%t] | %m%n
```

This pattern ensures that output is delimited by the "|" character in the event that one would programmatically process log output files. It also ensures that the fields output by NA batch are as follows:

```
<TIMESTAMP>|<LOG_LEVEL>|<CLASS_NAME>|<THREAD_NAME>|<INPUT_ANSWER_ID>|<INPUT_FILE_NAME>|<SPECIFIC_MESSAGE>
```

An example log message will look similar to the following:

```
[Dec 14 15:21:07]|INFO |BatchProgramImpl|[NA Batch Main Thread]|Input
Answer ID: N/A|Input File Name: input_1_record.xml|Done reading xml input
file: D:\dev\EngageOne_1.1\elDevEnv\el_installs\202\input\input_1_record.xml
```

Batch error handling

Error handling for accumulated and non-accumulated batch have some differences.

Error handling (accumulated and non-accumulated batch)

- Any composition errors are reported to the log file and an incident archive, which are located at `<active-drive>/incident-archive/dca`.
- If the errors are at the document level, the program will still continue until the threshold is reached. This is defined in the configuration framework file see [Batch threshold settings](#).

Additionally, error occurrences for non-accumulated batch are handled as follows:

- If the errors are at the document level, the program will still continue until the threshold that is defined in the configuration framework file is reached see [Batch threshold settings](#).
- If the error is non-document level, the batch process will fail.
- If there are errors in the first step, a log file containing the error information and resource (answer file, etc.) will be copied to the error directory `<Batch bundle installation folder>`.
- When there is an error in composition, DCA produces an incident archive.
- The log file will contain all information including the errors.

- An Event Monitor **NaBatchJobFailedEvent** is sent only when the batch job failed.
- Standard logging (commons-logging) to a file will be used to track events and errors in the application.

Batch exit codes

Use this table to understand batch exit codes. Exit codes are the same for accumulated and non-accumulated batch.

This table shows the mapping of exit codes to the error details:

Error details	Exit code value
Some document level errors (error threshold not reached)	1
DocumentInfoReachedThresholdLimitException	2
UndefinedVariableException	3
MissingPropertyException	4
ConfigFileNotFoundException	5
CLIInputParseException	6
CannotCreateOutputFileException	7
DataAccessException	8
UnableToUpdateDeliveryItemException	9
DomainNameNotFoundException	10
ChannelNameNotFoundException	11
NoChannelsAvailableException	12
Doc1LicenseDaoException	13
DeliveryOptionNotFoundException	14

Error details	Exit code value
NoTemplateDefinedException	15
NoDeliveryOptionDefinedException	16
FailedToRetrieveTemplateInformationException	17
Others/generic	100

To change exit code values:

1. Open `el-server-batch.jar` using 7zip, WinRAR or JDK's JAR -xf.
2. Modify **com/engageone/server/batch/dao/ExitCodeRetrievingDao.properties** with the appropriate values.

Batch threshold settings

The threshold level is the limit where a running batch will terminate, if the number of processed documents that have errors has been reached.

Namespace	Key	Value
com.pb.engageone.server.batch	document.error.level	an integer value. Minimum is zero

Following are occurrences of the threshold configuration in the **config-settings.xml** file for accumulated batch and non-accumulated batch. The setting has a default value of zero. The setting in the non-accumulated batch refers to the value in the batch namespace. It can be changed as well.

Accumulated batch example

```
<namespace name="com.pb.engageone.server.batch">
  <setting>
    <key>database.vendor</key>
    <value>${com.pb.bi.datalayer[database.vendor]}</value>
  </setting>
  <setting>
    <key>jdbc.password</key>
    <value>16171247f0f6b84cc894e1a71aed208</value>
  </setting>
```

```

<setting>
  <key>jdbc.url</key>
  <value>jdbc:oracle:thin:@172.16.9.51:1521:orcl</value>
</setting>
<setting>
  <key>engageOne.Guid</key>
  <value>f9ae282d19b4498fbd36fb35f99c2eab</value>
</setting>
<setting>
  <key>jdbc.driverClassName</key>
  <value>oracle.jdbc.driver.OracleDriver</value>
</setting>
<setting>
  <key>jdbc.username</key>
  <value>EO_DBA</value>
</setting>
<setting>
  <key>document.error.level</key>
  <value>0</value>
</setting>
<setting>
  <key>diJ.report.encoding</key>
  <value>UTF-8</value>
</setting>
<setting>
  <key>fixed.report.encoding</key>
  <value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
<setting>
  <key>xml.report.encoding</key>
  <value>UTF-8</value>
</setting>
<setting>
  <key>incident.archive.path</key>
  <value>${com.pb.engageone[active.drive]}/incident-archive/batch</value>
</setting>
<setting>
  <key>csv.report.encoding</key>
  <value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
<setting>
  <key>default.file.encoding</key>
  <value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
</namespace>

```

Non-accumulated batch example

```

<namespace name="com.pb.engageone.server.batch.na">
  <setting>

```

```

        <key>jdbc.username</key>
        <value>${com.pb.engageone.batch[jdbc.username]}</value>
    </setting>
    <setting>
        <key>record.cache.path</key>

<value>${com.pb.engageone.server.batch.na[batch.cache.path]}/records</value>

    </setting>
    <setting>
        <key>batch.work.dir</key>
        <value>${com.pb.engageone[server.folder]}/batch/work</value>
    </setting>
    <setting>
        <key>error.folder.path</key>

<value>${com.pb.engageone.server.batch.na[batch.work.dir]}/errors</value>

    </setting>
    <setting>
        <key>batch.cache.path</key>

<value>${com.pb.engageone.server.batch.na[batch.work.dir]}/cache</value>

    </setting>
    <setting>
        <key>jdbc.driverClassName</key>
        <value>${com.pb.engageone.batch[jdbc.driverClassName]}</value>
    </setting>
    <setting>
        <key>default.file.encoding</key>
        <value>${com.pb.engageone[default.file.encoding]}</value>
    </setting>
    <setting>
        <key>csv.report.encoding</key>
        <value>${com.pb.engageone[default.file.encoding]}</value>
    </setting>
    <setting>
        <key>document.error.level</key>
        <value>${com.pb.engageone.batch[document.error.level]}</value>
    </setting>
    <setting>
        <key>xml.report.encoding</key>
        <value>${com.pb.engageone.batch[xml.report.encoding]}</value>
    </setting>
    <setting>
        <key>workerThread.pool.size</key>
        <value>2</value>
    </setting>
    <setting>
        <key>activeContent.cache.path</key>

<value>${com.pb.engageone.server.batch.na[batch.cache.path]}/activeContents</value>

```

```

    </setting>
    <setting>
      <key>batch.temp.dir</key>

<value>${com.pb.engageone.server.batch.na[batch.work.dir]}/temp</value>

    </setting>
    <setting>
      <key>fixed.report.encoding</key>
      <value>${com.pb.engageone[default.file.encoding]}</value>
    </setting>
  </namespace>

```

Setting up the batch programs on a separate machine

The batch programs can run on a separate machine other than the EngageOne server provided that it has access to the EngageOne database. If the servers are behind a firewall, make sure that the database ports are open and that file sharing of the EngageOne repository is accessible to whoever is logged in the machine from where the batch program is being executed.

Note: To execute batch remotely, the active drive must be located either on a mapped directory or a UNC path directory.

To run batch on a different machine, refer the EngageOne Server Installation Guide for detailed information.

Properties file example: The only required override is the `doclgen.path` configuration setting. The other settings are for customization preferences.

```

#Number of active running threads in processing record for partial
composition
workerThread.pool.size=2
#Number of active running threads in processing record for
re-composition/concatenation process
recomposition.workerThread.pool.size=2
#Page cache size for internal DocumentInfo List
documentInfo.list.page.size=500
#The number of DCA adapters to use during pre-composition
batch.dca.pool.size.maximum = {new dca pool size setting}
#Work directory of the batch instance
batch.work.dir= {new work directory location}
#This is the maximum number of XML records (not files) that are held in

```

```

memory at any given time
#during processing
worker.queue.size=10000
#batch logging config
log.maxFileSize= {new log file size setting with size designation. For
example, 5MB}
log.maxFileCount= new log file count setting}
#doclgen settings for NA Batch
doclgen.path= {new doclgen directory location}
. . . .
. . . .
. . . .

```

Rules for Properties file example:

- Use "/" instead of "\" for file or folder separators.
- Settings should only be literals. Config-framework style variables (\${....}) are not supported.
- Only settings specified in the properties will be used as overrides.
- Settings that are not specified in the properties file will have their default value retrieved from the config-framework.

config-settings.xml example:

```

<namespace name="com.pb.engageone.server.batch.na">
  <setting-group description="{batch machine1}">
    <key>{your new value from item#3}</key>
    <setting>
      <key>documentInfo.list.page.size</key>
      <value>500</value>
    </setting>
    <setting>
      <key>recomposition.workerThread.pool.size</key>
      <value>{desired thread size for final composition}</value>
    </setting>
    <setting>
      <key>workerThread.pool.size</key>
      <value>{desired thread size for partialcomposition}</value>
    </setting>
    <setting>
      <key>batch.dca.pool.size.maximum</key>
      <value>{desired DCA pool size for this machine}</value>
    </setting>
    <setting>
      <key>worker.queue.size</key>
      <value>10000</value>
    </setting>
    <setting>
      <key>batch.work.dir</key>
      <value>{new work directory location}</value>
    </setting>
    <setting>
      <key>doclgen.time.out</key>

```

```

<value>{new doclgen time out value}></value>
</setting>
<setting>
<key>doclgen.path</key>
<value>{new doclgen directory location}</value>
</setting>

. . . .
. . . .
. . . .
</setting-group>
</namespace>

```

Running batch with integrated security database connections

There are two preferred ways of running batch processes if your environment is configured to use Integrated Authentication for MS SQL Server:

1. Run all batch processes with the exactly the same domain user as the one configured to access EO databases. No further configuration steps are necessary.
2. Configure Credential Manager to store credentials used for database access:
 - a. Open **Credential Manager**.
 - b. Click **Windows Credentials**, then **Add a Windows credential**.
 - c. Provide the appropriate values in fields **Internet or network address**, **User name** and **Password**.
 1. **Internet or network address** must contain the DB Server URL in format `mysqlserver.mydomain.com:1433`.
 2. **User name** must contain the name of user provided in the `deploy.properties` file, found in property `os.service.username`
 3. **Password** must contain the password of user provided in field **User name**

10 - Managing your EngageOne environment

EngageOne Administration does not, and cannot, provide user interface features for all tasks necessary to completely manage the system performance and security of your EngageOne systems. Guidance is provided throughout this chapter to help you devise the best management practices for your EngageOne systems. Some of the practices incorporate EngageOne Administration features. Other practices require a combination of plan development, installed scripts, and process engineering.

EngageOne Administration includes comprehensive features for:

- creating and organizing folders that will contain templates and Active Content.
- dynamically mapping Designer keys to image resources so that images may be updated without updating every template.
- defining devices for viewing and defining delivery options.
- setting up, configuring, and maintaining EngageOne communities, users, document classes, retention policies, database passwords, product licensing, spell checkers, and system diagnostics.

In this section

Managing disk space.....	221
System performance recommendations.....	225
Tuning the system for performance.....	234
System interactions and problem solving.....	244
Monitoring the Interactive Editor's WebSocket connections.....	260



Managing disk space

As with any application you must monitor and assess your EngageOne disk space regularly. Make best use of disk space using the provided Administration features and scripts.

If you deploy new templates regularly, implement a process to manage your templates. Consider your business needs and use a combination of the following actions to implement your disk space management policy.

Comprehensive disk space management includes these activities.

Activity	Description
Batch purging	<p>Validate and purge work items after running batch jobs and immediate composition requests. For example, purge the previous day's work items every day at the start of the next day's job.</p> <p>For information about configuring batch working directories, see Batch work directories and clean-up on page 222.</p>
Cache management	<p>The Interactive Editor ActiveX controls use a caching mechanism to manage resources in publications and documents. While caching resources when previewing a publication in Designer does not result in a huge performance benefit, it assumes significance when ActiveX is used in EngageOne Interactive.</p> <p>For detailed information, see Client cache management on page 223.</p>
DCA incident archive management	<p>The DCA incident archive is a means of collecting information about potential issues during document composition.</p> <p>DCA incident archives can be deleted after they have been reviewed and considered for possible reasons of document composition failure.</p> <p>For detailed information, see Document Composition Adapter.</p>

Activity	Description
Retention policies	Use retention policies to notify users when templates controlled by retention rules have expired, move expired templates to a pre-defined folder, and delete older template versions. Consider how often the retention policy will run and what time of the day the retention policy should be started.
Template version management	For batch processing, remove all previous template versions including the related physical files in the repository. Warning: Never manually delete any files in the active-drive or records from the database. Use the purge and clean-up utilities provided.
Back-up and restore	EngageOne Interactive does not perform backups for any of the system components. All backups and recoveries must be scheduled and performed by the EngageOne system administrator. For detailed information, see Backup and restore on page 225.

Batch work directories and clean-up

Non-accumulative batch uses a temporary work directory, which it uses to cache template information and resources as well as record files necessary for composition. The default location of this work directory is: `<install-dir>/<bundle-root>EngageOne/server/batch/work`.

On a batch run startup, the program creates a folder with a unique name under the `work/cache` directory. NA batch deletes this directory upon completion of a batch run.

The work directory can be configured through the configuration framework. You can change the value of the **batch.work.dir** setting from the **com.pb.engageone.server.batch.na** namespace.

Please see the following example:

```
<namespace name="com.pb.engageone.server.batch.na">
. . .
<setting>
<key>batch.work.dir</key>
<value><install dir>/EngageOne/server/batch/work</value>
</setting>
<setting>
<key>error.folder.path</key>
<value>${com.pb.engageone.server.batch.na[batch.work.dir]}/errors</value>
```

```

</setting>
<setting>
<key>batch.cache.path</key>
<value>${com.pb.engageone.server.batch.na[batch.work.dir]}/cache</value>
</setting>
. . . .
</namespace>

```

Client cache management

The ActiveX Editor client cache in EngageOne Interactive is independent of document server cache management.

The Interactive Editor ActiveX control uses a caching mechanism to effectively manage resources in publications and documents. While caching resources when previewing a publication in Designer is straightforward and does not result in a huge performance benefit, it assumes significance when the ActiveX is used in EngageOne Interactive.

Note that the following instructions are for Internet Explorer 8.0. Use these as a guide if you are working with a different version of Internet Explorer.

To use Internet Explorer cache options:

Warning: Internet Explorer manages the cache entries for the ActiveX Editor and manual deletion of any cache entries may lead to unexpected results. This is usually due to missing references within Internet Explorer, where Internet Explorer attempts to use a cache file that has been deleted manually. This option is not recommended. If performed accidentally, follow up with an Internet Explorer clear cache action as explained previously.

1. In Internet Explorer, choose **Tools > Internet Options**.
2. On the **General** tab, click **Delete** in the **Browsing history** section. A new window opens showing the **Delete Browsing History** settings.
3. Click the **Delete files** option. Check the **Preserve Favorites website data** and **Temporary Internet files** boxes, and then click **Delete**.

Viewing ActiveX editor resources

The resources used the Interactive ActiveX editor can be viewed in Internet Explorer.

1. In Internet Explorer, choose **Tools > Options**.
2. On the **General** tab, click **Settings** in the **Browsing history** section. A new window opens showing the **Temporary Internet Files and History Settings**.
3. Click **View files**. A Windows Explorer window opens.
4. Browse to the files prefixed with **ipe.resource** or sort by Internet address column to view the same.

ActiveX caching in EngageOne Interactive

The EngageOne server does not allow the ActiveX Editor direct access to its resources but allows it to download them to the client machine where the ActiveX Editor can cache them for further use.

Note that the interactive editor actively uses Internet Explorer to manipulate and control its cache for it.

- The server allocates a unique SRID (Server Resource ID) for a resource, which is used by the ActiveX Editor to enforce a mild form of version control. Any change to a resource triggers a new SRID from the server, which then downloads the new resource to the client.
- The ActiveX control downloads the resource files to the client machine and these files become candidates for caching.
- The ActiveX control uses Internet Explorer (IE) to create a cache entry along with a new file (typically a wrapper file with **ipe.resource** as prefix to the name and a time stamp as suffix) for the resource.
- The resource file is downloaded into this wrapper and Internet Explorer makes a reference to it and stores the reference.
- Any new file download will be checked against the Internet Explorer cache entry and reference resource list. If the resource entry has been identified and located, the cached entry will be used rather than launch a request to the server for a download.

The following are candidates for caching when available:

- Publication file
- XML files: content properties, instance XForm, instance, speller, and key map
- Instance schema file
- Active Content fragments
- Images - from publication, Active Content and keyed images
- Speller dictionaries - Lexicon file, Index file, Abbreviation file, History Files per language used

Client cache management folder location

The session folder contains folders that have timestamps for folder names. The files under this folder will be reused as long as the session is valid. The event log file is a log file and not a cached file. It logs file downloads for the session. Document fragments used by Active Content and the speller dictionary files that have been downloaded once will remain and are used independently of the session. These document fragments are subject to repeat download only when changes occur to them, which EngageOne tracks using SRIDs. The folder structure of files subject to caching is shown in the following example:

The files that are stored in the **ie_cache** folders are copies of the files in the **Temporary Internet files** folder. These files are copied here to help understand the reference created by Internet Explorer to the resource file being used for that particular session. These copy files have no real significance in relation to caching.

Backup and restore

EngageOne Interactive does not perform backups for any of the system components. All backups and recoveries must be scheduled and performed by the EngageOne system administrator. The following diagram shows a typical setup of the EngageOne environment.

Frequent backups are recommended for the EngageOne database, the active-drive folder, and the installation directory.

Contact the system administrators or vendor(s) for more information about the recommended backup and recovery methods for middleware, hardware, operating systems and other products, or devices not bundled with EngageOne Interactive.

System performance recommendations

The primary factors in achieving optimal EngageOne performance include the following, in the order of importance:

1. Disk Drive IOPS: Persistent storage performance is vital in achieving optimum results. High Performance RAID storage with Serial Attached SCSI (SAS) drives, such as 15K RPM Hard Drives or High Performance SLC SSD is highly recommended.
2. CPU see the EngageOne Server 4.3 Software and Hardware Requirements guide.
3. RAM see the EngageOne Server 4.3 Software and Hardware Requirements guide.
4. Multiple disks: If separate and independent high-performance drives are available, configure each of the following directories (configured in config-settings.xml) on a separate drive. If only one high-performance drive is available, keep all directories on that drive:
 - a. active-drive
 - b. batch/temp
 - c. delivery channel output
 - d. Java I/O temp

Configuring for optimal performance

The following settings should be reviewed and adjusted to achieve optimal performance in your environment. Each new environment is different and should be looked at for potential adjustment. For detailed information on what these settings are and how to tune them, see the EngageOne Server Performance Benchmarks white paper available on the EngageOne Server software media.

- The A-batch namespace (`com.pb.engageone.server.batch`) settings within `config-settings.xml`.
 - `concurrent.channel.processing.count`
 - `workerThread.pool.size`
 - `java.io.tmpdir`
- The NA-batch namespace (`com.pb.engageone.server.batch.na`) settings within `config-settings.xml`.
 - `batch.work.dir`
 - `answer.xml.validation.enabled`
 - `workerThread.pool.size`
 - `recomposition.workerThread.pool.size`
 - `java.io.tmpdir`
- The DCA namespace (`com.pb.dca`) setting within `config-settings.xml`.
 - `pool.size.maximum`
- The working directories defined in `config-settings.xml` were separated to local isolated partitions so that I/O processing performance is not a bottleneck. The three partitions hold the following:
 1. Batch work directory
 2. Output files and temp directory
 3. Log file
- The script for running A-batch (`run.bat`).
 - Expanded JVM memory space
 - Do not use the `-verbose` parameter. This removes the debug messages.
- The script for running NA-batch (`run-na-batch.bat`).
 - Expanded JVM memory space
 - Do not use the `-verbose` parameter. This removes the debug messages.
- When executing the batch scripts, the `-quiet` parameter was added. This is a critical step to get optimal performance.

Incorporating Purge into your maintenance plan

Running **purge**, **purge-em**, and **purge-promotion** is an important maintenance activity to prevent gradual performance degradation due to the build-up of data in the database tables. Purge is typically scheduled to run during non-peak hours. If you choose to run purge during peak operating hours,

use the `-concurrencyMode` option. Do not run purge while batch is running. Make sure you implement a purge routine as part of your EngageOne Server maintenance plan.

Accumulated batch and hardware caching

If you run accumulated batch after rebooting performance will initially be degraded. As hardware caching by the operating system is established, performance will improve. Do not disable disk read and write caching. Also, a universal power supply is required to maintain hardware cache in the event of a power outage.

Accumulated batch and hardware caching

If you run accumulated batch after rebooting performance will initially be degraded. As hardware caching by the operating system is established, performance will improve. Do not disable disk read and write caching. Also, a UPS is required to maintain hardware cache in the event of a power outage.

NA-Batch

There are four modes in which you can execute the NA-Batch process; refer to [Modes and features](#) for detailed information. The Express Batch processing mode provides optimum performance, but not all EngageOne Server features are supported. However, all customers moving from standalone Generate (Doc1Gen) to EngageOne Server should strongly consider redesigning their templates and making them compatible with Express Batch. Refer to [Express Batch](#); this section contains an in-depth description of this processing mode. Whether it is possible to meet the criteria of Express Batch processing or not, it's worth reviewing the information present in section [NA batch setup and performance optimization](#). The section covers all the areas you need to consider to attain the optimum performance for NA-batch.

Concurrent users

EngageOne can support 250 concurrent EngageOne Interactive users on a single application server node, provided that the physical servers in the environment meet the performance requirements for this type of load. EngageOne (along with the application server), and the database (Microsoft SQL Server or Oracle) being used should all have their own dedicated physical resources. Not only should these applications be separated from each other, but for optimal performance, they should not share their individual server resources with other resource-intensive applications. Optimal performance in any environment requires that the physical server's resources are dedicated to the EngageOne related applications and components.

Note that after your system is operational or after adding new projects or applications, it is important to test performance and tune the system to achieve optimal performance.

Port range and anticipated load

The number of available ports required during run-time depends on the concurrent number of users. These ports are used when handling Web Service requests. For an anticipated load of more than 100 concurrent users, a range of 25,000 ports must be allocated. The range of ports can be set accordingly:

- On Windows, the range of ports can be increased via the `MaxUserPort` registry value.
- On Unix, at least 30,000 ephemeral ports are set by default, which is the ideal number during run-time.

For further information see:

<http://marc.info/?l=axis-user&m=118912788410969&w=2>

<http://support.microsoft.com/kb/q196271/>

http://www.ncftp.com/ncftpd/doc/misc/ephemeral_ports.html#AIX

Logging

EngageOne logging is configured through several different touch points, depending on the logging domain in question. This is due in part to the construction of the system stack. The embedded container relies on `java.util.logging` (JUL), the web applications in EngageOne rely on Logback and the batch applications rely on Log4J.

Application logging

Each bundle has a global logging configuration, meaning all applications in a given bundle share the same configuration (appenders, loggers, log levels, and so on), with two exceptions. The first exception is the security bundle. This bundle is essentially just OpenAM, which provides its own logging mechanisms, configurable through the OpenAM administrative console. The other exception is the batch bundle. A-batch and purge share a configuration, but NA-batch has its own.

Component	Configuration file
Embedded container	<code><bundle-root>/conf/logging/logging.properties</code>
Bootstrap application	<code><bundle-root>/conf/logging/logback-global.xml</code>

Component	Configuration file
Web applications	<bundle-root>/conf/logging/logback-global.xml
NA-batch	<bundle-root>/bin/na-batch-log4j.properties <bundle-root>/bin/na-batch-log4j-quiet.properties
A-batch	<bundle-root>/bin/log4j.properties
Purge	<bundle-root>/bin/log4j-quiet.properties
Embedded container: access	<bundle-root>/conf/logging/logback-access.xml

The logging framework for web applications and bootstrap applications is Logback. Logback configuration is very powerful, and recommended only when the default settings are insufficient. For detailed information about Logback configuration, refer to the documentation on their website (<http://logback.qos.ch>).

In addition to the fine-grained control provided by the configuration files, EngageOne Server provides a few course-grained options in <bundle-root>/conf/deploy.properties

logging.default.level

This property determines the initial logging level of the system. The setting takes effect during installation, but can be overridden later by modifying the configuration files in table 4, above. By default, the logging level is set to WARN. This property has no effect on batch and purge applications.

Note: In this case, "default" denotes the initial value found in deploy.properties. With a logging level of WARN, several applications generate no logs; this is desirable, but can cause alarm for administrators who were expecting to see more logs. Setting the log level to INFO or DEBUG will likely degrade performance. The default configuration should be left for production environments and only modified in extenuating circumstances.

logging.stdout.enabled

This property determines whether log messages also go to the console. By default, log messages go only to the logfiles. This property has no effect on logging to files.

Container logging

The embedded container maintains its own log, provided by java.util.logging (JUL). Refer to this [website](#) for documentation on configuring JUL.

Access logging

Access logging consists of recording each request to the container, and is disabled by default. To enable it, modify the following properties in `<bundle-root>/conf/bundle.properties`:

logging.access.enabled

This property determines whether access logging is performed at all. By default, access logging is disabled.

Note: In this case, "default" denotes the value assumed by the system if the property is not defined in `bundle.properties`.

logging.access.configuration.path

This property allows you to override the location of the access logging configuration. The default value points to the generic configuration included with EOCS. That configuration file can be modified in place, but this property is provided as a convenience.

Note: In this case, "default" denotes the value assumed by the system if the property is not defined in `bundle.properties`.

logging.access.configuration.quiet

This property configures whether logback should print information about its own configuration to the console during startup (a feature designed to debug errors in the access logging configuration file). By default, this property is true (the debug information is disabled).

Note: In this case, "default" denotes the value assumed by the system if the property is not defined in `bundle.properties`.

To configure the appenders for access logging, modify the appropriate configuration file or files according to the table below.

Component	Configuration file
Embedded container: access	<code><bundle-root>/conf/logging/logback-access.xml</code>

Security bundle logging

There are additional logs available from OpenAM in the security bundle. The configuration of the settings for these logs is handled separately.

Server install log

This is an additional log produced during the initial configuration of the security bundle. By default, this log file is named `install.log` and is located at `${security.install.dir}/conf/OpenAM` in your security bundle installation.

Administration logs

These logs recorded information on OpenAM events. A detailed breakdown of these logs can be found in OpenAM's online documentation.

By default, these logs are located within the `${security.install.dir}/logs/OpenAM` folder of your security bundle installation. The default log level is INFO. The configuration settings can be modified by logging in to your OpenAM server console and navigating to **Configuration > System > Logging**.

Debug logs

These logs contain information useful for troubleshooting OpenAM problems. A detailed breakdown of these logs can be found in OpenAM's online documentation.

By default, these logs are located within the `${security.install.dir}/conf/OpenAM/OpenAM/debug` folder of your security bundle installation. The default log level is ERROR. The configuration settings can be modified by logging in to your OpenAM server console and navigating to **Configuration > Servers and Sites > <server_name>** and modifying the "Debugging" section.

You can also modify the value of these settings by appending the following properties to your `${security.jvm.settings}` property in your `deploy.properties` configuration file and rerunning the "configure" target to apply the changes.

```
-Dcom.ipplanet.services.debug.level=error
-Dcom.ipplanet.services.debug.directory=C:\mydirectory
```

Agent logs

Each bundle has an agent log containing information about the bundle's OpenAM agent.

By default, this log is named `debug.out` and is located within the `${bundle.install.dir}/logs/openam-agent/debug` folder of each bundle. The default log level is ERROR. The configuration settings can be modified by logging in to your OpenAM server console and navigating to **Access Control > <top_level_realm> > Agents > J2EE > EngageOneAgent** and modifying the "Agent Debug Level".

Managing the Event Monitor

Event Monitor sinks contain events, including log, database, e-mail, and JMX notification. Any number of sinks may be defined. For example, multiple SMTP sinks can be defined with a single purpose for each sink.

One SMTP sink can be used to send all **Error** events to an administrator e-mail address or pager. Another SMTP sink can be used to send all template events to a group of users who need to know about template changes (newly imported, deleted, etc.).

Note that if SMTP information is supplied in `event-monitor-sinks.xml`, in either the `email.admin` or Retention Policy Sink, these values will override the SMTP settings in the `config-settings.xml`.

You can filter events so that you do not overload the Event Monitor tables with unnecessary data, but still allow the reports you need to be generated. For details about configuring sinks and filtering, see [Event Monitor](#) on page 138.

Sinks define a destination for an event. The benefit of a sink depends on the nature of the destination sink type.

Database sink: In a database sink, the events are stored in the database.

SMTP sink: In an SMTP sink, events are sent to e-mail addresses or even pagers to alert on-call personnel of potential issues with the system.

JMX sink: In a JMX sink, live events are sent to a monitoring application (for example, Tivoli) so that operations personnel can be notified of potential issues with the system.

What are the consequences of having too many sinks?

The Event Monitor processes events asynchronously. Events are received in a queue and are processed sequentially. Some events take longer to process depending on the sink configuration.

If an event is set for delivery to a log, database, SMTP, and JMX, it will take a while for the Event Monitor consumer to process that event. Adding more sinks increases the processing time. The more sinks, the more CPU time required to process each event.

Limiting the number of events

In order to limit the number of events in the queue, certain event priorities can be disabled in Event Monitor, and will not be entered in the queue.

There are four types of event priorities:

- **DEBUG**

The highest priority and most verbose.

DEBUG events include information for INFO, WARN, and ERROR events.

- **INFO**

INFO events include information for WARN, and ERROR.

- **WARN**

WARN events include information for ERROR events.

- ERROR

ERROR events detail only information for errors.

For example:

```
<setting>
<key>eventLoggingLevel</key>
<value>DEBUG</value>
</setting>
```

Performance indications of event logging

To handle performance implications of logging events, configure `event-monitor-sink.xml` to log only “critical” events and exclude “trivial” events. For example, the sample filter below shows exclusion of the **debug** event logging:

```
<filters> <filter> <property name="eventType" exclude="Debug" /> </filter>
</filters>
```

Event Monitor tables comprise the largest part of the EngageOne database and can become out-of-date, providing little value past a certain age. EngageOne includes an Event Monitor purge function that works much like batch purge. The Event Monitor purge is run using a command line script; **purge-em.bat** for Windows and **purge-em.sh** for Unix. It can also be run in parallel with the **purge.bat** script, however the **-numDaysRetention <offset days>** must correspond to the **-onOrBeforeDate** of this script. Make sure that you have them scheduled correctly.

Note that you cannot specify specific events to purge. You can only purge all events that are logged before a user-specified date.

Purged events are not shown in EngageOne Administration **Diagnostic** panel and once deleted are no longer traceable. To remove all events related to a pending item, remove the pending items using batch purge with the `-pendingdelivery` option.

Run Event Monitor purge at the same frequency as batch purge. The run frequency depends on your system’s daily workload.

To launch Event Monitor purge from a batch file:

use `purge-em.bat/purge-em.sh`.

purge-em

Function	Purge old Event Monitor records for the dates given
Syntax	<code>purge-em -onOrBefore {yyyy-MM-dd}</code>
Parameters	

date	all records that were logged before the given date will be purged
Example	<pre>purge-em -onOrBefore 2009-11-27</pre>
	<pre>purge -delivered -numDaysRetention 3</pre>
Java command example	<pre>java -cp <libraries> com.pb.eventmonitor.purge.EventMonitorPurge -onOrBefore 2009-11-27 -verbose</pre>

Tuning the system for performance

Java memory tuning

The Java Runtime memory settings are configured in the `deploy.properties` file. The default values are tuned to a scenario of 250 concurrent users per node.

- `security.jvm.settings`
- `core.jvm.settings`
- `composition.jvm.settings`
- `conversion.jvm.settings`
- `batch.jvm.settings`

Perform the following steps to modify the JVM settings:

1. Update the setting in your `deploy.properties`.
2. Stop the associated service bundle you wish to tune.
3. Run the 'configure' target on the server.
For example, if you modify the `security.jvm.settings` execute the `eos.groovy` on the security bundle servers:

```
groovy eos.groovy -b security -p deploy.properties -t primary configure
```

Tuning recommendations

Please see `deploy.properties` for recommended default configurations. In addition, the following guidelines should be applied when tuning memory settings.

All bundles

The following should be considered for all bundle settings:

- The most common parameters used to tune performance are the heap sizes defined by `-Xms` (minimum/initial allocation) and `-Xmx` (maximum allowed allocation). Configure the maximum and minimum heap sizes to equal values (`-Xmx` and `-Xms`)

Note: Default values are not tuned in this manner. This is to reduce RAM consumption for installs with a concurrency less than 250 users.

- To maximize your compute resources set the `-Xmx` value to leverage as much memory as possible. Some memory should be left free for use by the Operating System (see your OS documentation for general recommendations / guidelines), but the remainder may be allocated to the installed bundles by increasing the `-Xmx` value.
- Java garbage collection cycles consume system resources and can slow down the system. If possible, set the `-Xms` and `-Xmx` values the same to minimize the frequency of garbage collection cycles.
- A minimum of 512M should be allocated to the perm gen space (`-XX:MaxPermSize=512M`)
- Outside of heap size allocations, JVM defaults generally work well with EngageOne Server. Generally, rely on defaults and modify configurations judiciously.

Security

- Use the concurrent mark sweep garbage collector and disable the explicit garbage collection. (`-XX:+UseConcMarkSweepGC -XX:+DisableExplicitGC`)
- Allocate a third of the heap size to 'young' generation. (set the `-XX:NewSize` to at least one third of the `-Xmx` value)

Additional resources for JVM tuning

For additional information on JVM tuning please see these resources.

- JVM options
 - <http://www.oracle.com/technetwork/java/javase/tech/vmoptions-jsp-140102.html#G1Options>
- JVM performance consideration and heap sizing
 - http://www.oracle.com/technetwork/java/javase/gc-tuning-6-140523.html#generation_sizing.total_heap

Database connection pooling

The default database connection pool values are tuned to a scenario of 250 concurrent users per node (see Configure Connection Thresholds section). Database connection pool tuning may improve overall system performance. See the “Connection Pool (CP) Sizing” section in the following properties files:

- `${core.install.dir}\conf`
 - `engageoneDS.properties`
 - `jackrabbitDS.properties`
 - `flowableDS.properties`
- `${composition.install.dir}\conf`
 - `engageoneDS.properties`

These files contain specific recommendations and best practices. Pay special attention to the inline documentation as any adjustments to this pool should be made with the entire system in mind.

JMS queue pooling

The composition bundle manages the asynchronous process with a Java Messaging Service (JMS) queue. Increasing the JMS listener count may improve overall throughput of the composition process.

The listener count configuration is not available in the `deploy.properties` file. Rather, after the install and configure process:

1. Open the `${composition.install.dir}\conf\viewpoint.properties` file.
2. Modify the `${composition.jms.listener.count}` property.
3. Stop and restart the composition bundle service.

Attention: When increasing this value, an associated adjustment must be made to the database connection pool. See recommendations in `${composition.install.dir}\engageoneDS.properties` file. If the connection pool is not adjusted accordingly you may experience system errors as the connection pool is too small to handle all the parallel listeners.

Session Management

Session number

The session number is the maximum number of allowable concurrent sessions; it is set to 5000 by default. If the volume of users is expected to exceed 5000, you are recommended to increase this value accordingly. Use the:

The session number value can be configured using optional property defined in the `deploy.properties` file under key:

```
security.max.sessions.number
```

Session timeout

The session timeout setting tracks for EngageOne Administration, EngageOne Web Service API and EngageOne Compose, the following:

- Authenticated user inactivity. The inactivity timeout for the Core and Security bundles are managed independently. These timeouts should be the same, or the timeout for the Security bundle higher than the Core bundle, otherwise a user will be logged out of EngageOne Compose prematurely.
- Authenticated user total session time.

Bundles

To configure the login session idle time or maximum session time through the bundles:

1. Go to the `deploy.properties` file in install folder
2. Edit the file
3. Uncomment **`security.max.idle.time`** property and set the value.
4. Uncomment **`security.max.session.time`** property set the value.
5. Re-configure primary Security bundle and Core/Composition bundles on each node in the cluster.

Both properties are optional and do not have to be changed together. The commented property remains default value.

The Maximum Idle Time attribute accepts a value (in minutes) equal to the maximum amount of time of inactivity before a session expires and the user must reauthenticate. Enter a value of 1 or higher, the default is 30.

The Maximum Session Time attribute accepts a value (in minutes) equal to the maximum number of minutes that a session can remain active before a user is required to re-authenticate, the default is 480.

Note: A recommendation to balance security and convenience requirements would be to set the “Max Session Time” interval to a relatively higher value, and the “Max Idle Time” interval to a lower value.

UI

To configure the login session idle time through the UI on Core bundle, repeat the following steps below for each node in the cluster.

1. Stop EngageOne Core bundle service.
2. Navigate to the application WAR file.

For example, navigate to:

```
C:\Program Files\EngageOne\Server\core\deployments\client.war
```

Make a backup copy.

3. Extract `client.war` to a temporary folder. Edit the following files:

- `apps\projects\js\config.js`
- `apps\admin\js\config.js`
- `apps\interactive\js\config.js`

4. Add the following key to the files above:

- `// how long (in ms) until session timeout from inactivity`

```
inactivityTimeout: 20 * 60 * 1000
```

Note: the time above is in milliseconds.

- The valid range is any integer.
- Default value is 20 minutes.
- If other properties precede this key, use a trailing "," (comma) after the previous property.

5. Optional configuration for a session timeout notification:

- `// how long (in ms) before session timeout to notify user`

```
inactivityNotificationBuffer: 2 * 60 * 1000
```

Note: default value is two minutes.

- The notification displays two minutes before session timeout. However, the notification count down starts at 60 seconds.

The default: a notification message displays after 18 minutes of user inactivity.

For example, a notification message will display two minutes before session timeout. In this case, 20 minutes.

The counter starts at 60 seconds, after that time the user is logged out. In this case, the session lasts 19 minutes, not 20 minutes.

- To keep the notification buffer and counter in sync, change the counter value:

Edit `shared\js\framework.js`, and search for `"defaultSeconds:"`.

For example, `"defaultSeconds:2*60"`.

The value is in seconds. The default is 60 seconds.

The formula for user inactivity (in seconds) to automatically log the user out:

```
inactivityTimeout/1000 - inactivityNotificationBuffer/1000 +
defaultSeconds
```

6. Repackage the `client.war` file.

Copy the file back into the `\deployments` directory.

7. Navigate to the application deploy directory, and delete the client folder.

For example:

```
C:\Program Files\EngageOne\Server\core\tmp\webapps
```

Start EngageOne Core bundle service.

Note: before logging into EngageOne Compose, delete all cached files from your web browser.

doc1gen tuning

The following properties modify the interaction between the system java processes and the underlying composition engine (doc1gen). These options apply to both the composition and batch bundles.

doc1gen instance pool

The system maintains an open pool of doc1gen processes. This pool may be tuned by modifying the following properties in the `deploy.properties` file:

- `dca.pool.size.minimum` (default 1) - minimum number of doc1gen instances in the pool.
- `dca.pool.size.maximum` (default 5) - maximum number of doc1gen instances in the pool. Set this property to the number of CPU cores available on the batch or composition server.

Attention: There is an additional pool allocation check setting that can be changed but must be made directly in `config-settings.xml` (`pool.allocation.check.minutes`). See the Document Composition Adapter (DCA) section for the additional details regarding this setting.

doc1gen timeout

There is a document generation timeout setting defined in `deploy.properties`. This property specifies the amount of time (in seconds) the batch or composition bundle will wait until the doc1gen process responds. This property should be increased according to the size and content of the documents being composed.

- `doc.generation.timeout` (default 120 seconds)

To modify these configurations on your composition or batch bundle install:

1. Update the setting in your `deploy.properties`.
2. Stop the associated service bundle you are tuning.
3. Run the 'configure' target on the server.

```
groovy eos.groovy -b composition -p deploy.properties -t primary
configure
```

LDAP to database sync

EngageOne Compose maintains its own index of authorized users and groups for fast searching and type-ahead fields. Comparing this stored index to the latest LDAP data can be costly, so the `ldap.syncIntervalSeconds` property in `deploy.properties` configures how often the two are synchronized. By default, this value is 2 hours (7200 seconds).

To modify this value:

1. Update the setting in your `deploy.properties`.
2. Stop the associated service bundle you are tuning.
3. Run the 'configure' target on the server.

```
groovy eos.groovy -p deploy.properties -t primary -b core configure
```

This value should be modified based on your business need and how frequently users are added to LDAP. A smaller value will create additional load on the core primary node, but make the LDAP data appear more quickly in EngageOne. A smaller value will decrease load but may delay a new LDAP user's ability to use the system.

Note: Changing user and group entities in LDAP affects EngageOne role mapping functionality after *two minutes*.

For example, if the `CommAdm` group is defined in LDAP as Community Administrator of `community1`. When `user1` is added to this group, after two minutes `user1` will be able to use EngageOne as a Community Administrator of `community1`.

EngageOne Notifications

EngageOne Notifications is a mechanism for real-time communication of business-related events. By default it makes use of an instance of Apache ActiveMQ embedded in the Notification bundle, although it can be configured to use a different JMS-compliant message queue. A number of factors should be considered regarding the performance and resource utilization of the Notifications mechanism.

- Notifications should only be enabled if a message consumer is present. If there is no consumer the notification messages have no value and should not be produced.

The Notification mechanism is turned off by default and can be enabled by setting `notification.publisher.enabled=true` in `deploy.properties` at installation or re-configuration time.

- If Notifications are not enabled or if the Notifications mechanism is configured to use an external message queue the Notification bundle should not be started up.

Starting this bundle will use system resources unnecessarily.

- Specific Notification types can be suppressed if they are not required by any consumer.

Suppressing messages reduces the system resources required by the message queue and also reduces the number of messages that a consumer has to process.

Individual Notification messages types can be suppressed using `notification.type.communication.omitted.actions`, `notification.type.batch.omitted.actions` and `notification.type.workflow.omitted.actions` properties in `deploy.properties`.

For example, the "StatusChanged" action for "Communication" notifications could be suppressed.

- The JMS implementation persists messages until they are consumed by a client application.

The persistence mechanism requires disk space and this is by default located on a drive on the Notification bundle's server.

The amount of space required can be minimized by ensuring that at least one message consumer is consuming each type of message. Otherwise messages are left in the queue indefinitely unless a non-zero value is specified for `notification.message.timeToLive` in `deploy.properties`.

This property can be set to a specific value (for example one hour), so that unprocessed messages are automatically purged from the queue after that time.

Note that the number is in milliseconds, and therefore an hour would be 360000.

SOAP API

If you wish to integrate your services with EngageOne Server you need to be aware of the optional `transientSession` flag. This flag is present in the `UsernameToken` section of each SOAP request and should be set according to the needs of the environment integrating with EngageOne Server.

If you do not provide the `transientSession` flag explicitly, the default behavior depends on property `security.soap.transient.session.default`. This optional property is set in your `deploy.properties` file, and it can take one of two values:

- `ACTIVE_WHEN_NOT_SPECIFIED`
- `INACTIVE_WHEN_NOT_SPECIFIED`

If the property is not provided, the system behaves as if `ACTIVE_WHEN_NOT_SPECIFIED` was set.

OpenAM sessions related to SOAP web service calls are not created if:

- `transientSession` flag is set to `TRUE` in the incoming SOAP request;
- `transientSession` flag is not provided in the SOAP request and property `security.soap.transient.session.default` is set to `ACTIVE_WHEN_NOT_SPECIFIED`.
- `transientSession` flag is not provided in the SOAP request and property `security.soap.transient.session.default` is commented.

Each SOAP request will create a new OpenAM session if:

- `transientSession` flag is set to `FALSE` in the incoming SOAP request;
- `transientSession` flag is not provided in the SOAP request and property `security.soap.transient.session.default` is set to `INACTIVE_WHEN_NOT_SPECIFIED`.

Consequently, there's no need to set `transientSession` flag explicitly unless it is absolutely necessary, as the same can be achieved through the default setting available in `deploy.properties` file.

The only scenario where `transientSession` needs to be provided (and set to `FALSE`) is when there is a need to reuse the session token returned in the SOAP response. In all other cases, we recommend setting `transientSession` flag to `TRUE` or not providing it at all, as it results in a significant reduction of the data processed and replicated between security nodes.

System interactions and problem solving

This section helps you understand system interactions and solve common problems:

- maintain the active-drive

- install and repackage ActiveX
- upgrade the Generate component in EngageOne Server
- update database user passwords
- resolve document composition incidents and failures
- change the default DBCS encoding scheme
- activate trusted connections between EngageOne and custom applications
- modify OpenAM configuration settings

Working with the active-drive

The active-drive contains many system critical artifacts. These artifacts include things like licensing information, temporary work folders and files, and configuration settings. The temporary working folders and files should never be manually altered or moved as they must maintain parity with other critical areas of the system, such as the database.

ActiveX editor repackaging

If you need to plug in a new version of the Interactive Editor ActiveX control to EngageOne Interactive, follow the steps below.

Note: Pre 6.6.11 versions of the editor are not supported. You are advised to uninstall any earlier editor version before performing any installation activity.

1. On the EngageOne Server, back up the installed `<EngageOneInstallPath>\core\deployments\Client.war` file.
2. Stop the core service.
3. Edit the installed `<EngageOneInstallPath>\core\deployments\Client.war` file to update the following:
 - a. Copy `eoeditor.cab`, and `eoeditor.ver` to the `Client.war/shared/activeX` directory in the deployable archive.
 - b. Edit `client.war\shared\js\framework.js`
 - c. Find the string "editorVersion:" in `framework.js` file.

Replace the existing version with the one in `eoeditor.ver` file added in **step a**.

Note: The format of the "editorVersion" string updated in `framework.js` must use the comma-separated format present in `eoeditor.ver`.

You may need to use `jar` or another archiving utility to open and re-archive the file.

4. Run the `eos.groovy` script to re-configure and restart the core service.

5. On a Clustered environment repeat steps 1-4 on each server node.
6. On the EngageOne Interactive Client, edit the communication.

When prompted, allow the new Interactive Editor to be installed.

Generate repackaging

If you need to plug in a new version of Generate to EngageOne Server, follow the steps below.

Note:

- The *default location* below is in the format `<bundle>/bin/doc1gen/<platform>/`
Where `<bundle>` corresponds to both `<composition>`, and `<batch>`.
- In addition, `<platform>` corresponds to the operating system.
 - For Linux, use `<composition>/bin/doc1gen/<lin>/`
 - For Windows, use `<composition>/bin/doc1gen/<win>/`

1. Back up the original Generate directory.

The *default location* is Linux `<bundle>/bin/doc1gen/<lin>/` or Windows `<bundle>/bin/doc1gen/<win>/`.

2. Remove all content in the *default location*.
3. Copy the entire contents of `doc1gen` to the *default location*.

The final content of the `doc1gen` directory should have the new DLL and executable files directly under the *default location*.

4. Copy any saved files from step 2 to the *default location*.

Replace any existing files.

Configuring Generate

You can optionally modify the standard behavior of EngageOne Generate when invoked by EngageOne Server to suit your specific operating requirements.

To do this, you must manually create a file called `eo-server.config` using your preferred text editor adhering to the following guidelines:

- Options are coded as keywords and associated parameters within sections

- Sections must be introduced with the relevant name within angle brackets, for instance: <Trace>, <Message>, etc.
- If you want to include comments in the configuration file, prefix the comment line with a semicolon character
- No sections or keywords are compulsory and you should code only those options that suit your requirements.
- All file references can include both path and file name as required. Ensure that you code all such references in a format suitable to the operating system under which you are running EngageOne Generate.

This configuration file (`eo-server.config`) must be placed in the EngageOne Server Generate (doc1gen) execution directory.

Note: that this feature is currently restricted to the Windows and Linux production platforms only.

References:

- Refer to [eo-server.config file reference](#) on page 248 for detailed information on setting related to this configuration file.
- Refer to the **OPS file reference** section in the Designer User guide for a full list of override settings.

eo-server.config file reference

Syntax

```

<trace>
    Outputfile=Filename
    TraceLevel={off|default|verbose}
    outputcodepage={UTF8|default}
    memlimit=Memory
    publication=Number

    <Messages>
    MandatoryNotPlaced=Stop|Continue|Warn
    MandatoryMessageError=Stop|Continue
    OptionalMessageError=Stop|Continue
    CampaignDate=String
    Cycle=String
    MessageProcessing={Yes|No}
    NoMessages=Stop|Warn

    <Advanced>
    ErrorFile=Filename
    LogFile=Filename
    Checkpointfile=Filename
    CPconsole= {0|1}
    ConstantShapeOffPage=Abort|Warn|Ignore
    DynamicShapeOffPage=Abort|Warn|Ignore
    RangeOfPublications=n
    Workspace=Filename
    SystemTempFiles={Yes|No}
    SuppressMessages={NONE|ALL|INFORMATION|<comma separated message
IDs>}

    <Overflow>
    OverflowFile=Filename
    OverflowSize=Memory

    <Custom>
    Name=Parameter
    Name=Parameter
  
```

The following `e-server.config` file settings are relevant to EngageOne Generate when invoked from EngageOne Server.

Sections, keywords and parameters	Usage
<Trace>	
OutputFile	<p>Filename is the name of the file for the trace information. If no file is specified the output is sent to the standard output medium of the Generate host environment (e.g. command prompt window, system log, etc.)</p> <p>If you specify %d in the file name, for example:</p> <pre>OutputFile=traceoutput%d.txt</pre> <p>then %d in the file name will be replaced with the date and time the file was generated.</p>
TraceLevel	<p>This controls the amount of trace information that is output. Off writes nothing i.e. turns trace off. Default shows the path that Generate took when the error occurred. This is in the form of a tree structure. This enables the error to be tracked down to a particular object in the publication design. Verbose includes additional information such as internal references and the instructions that are being executed. This can be used by Customer Support</p>
OutputCodePage	<p>This allows you to change the code page for the output file from the default host code page to the general Unicode UTF8 code page.</p>
MemLimit	<p>Controls the maximum amount of memory that the trace module can use for buffering trace information. A value of '0' means no limit.</p>
Publication	<p>Allows the user to specify the number of a publication to be traced in addition to the first publication that causes the error. This will only be traced if it occurs before the publication with the error (that causes the trace to stop processing).</p>
<Messages>	
MandatoryNotPlaced	<p>Select the action you want Generate to take when a mandatory message cannot be included in a document for which it was intended. The default is to Stop processing, otherwise you can ignore the error and Continue or issue a Warning and continue.</p>

Sections, keywords and parameters	Usage
MandatoryMessageError	Select the action you want Generate to take when a mandatory message has unresolved links – typically when a font used by the message is not included in the resource pack or when a data field used has not been mapped. The default is to Stop processing or you can ignore the error and Continue processing.
OptionalMessageError	As above but for non-mandatory messages.
CampaignDate	Specifies the date to be used when selecting messages using the activation and expiration attributes as defined within the Message1 and Content Author environments. Options are: Auto – the current system date. Auto+ -<n><d w m> – the current system date plus or minus the specified number of days, weeks or months, e.g Auto+10d, Auto-3wdd/mm/yyyy or mm/dd/yyyy – a specific date
Cycle	String defines which cycle (defined in the Message1 and Content Author environments) to use. If defined, only messages belonging to this cycle will be selected.
MessageProcessing	Allows details for message rejections to be output to the standard output medium for the system on which the program is running. The default is No .
NoMessages	Specify the Generate action to be taken if messages are expected, but none can be found in the Messages file. The default is to Stop processing, or you can ignore the error, issue a Warning and continue.
FontMappingFromHip	Setting this to Yes will ensure that any mappings that have been applied to fonts used in a publication in the Designer (i.e. are in the HIP file) will also be applied to the same fonts used in messages created in Content Author or Message1. The default is No .
<Advanced>	
ErrorFile	This option is used to specify the file that will receive any publication datasets that cannot be processed by Generate when a production job is run. It will override the file specified in the Data record file option in the Publish Wizard. The filename must be in the required format for the operating system.

Sections, keywords and parameters	Usage
LogFile	This option is used to specify the file that will receive any error or warning messages issued by Generate. The filename must be in the required format for the operating system.
Checkpointfile	This option is used to specify the file that will receive the messages that indicate which publication data set is currently being processed.
CPconsole	Checkpointing messages can be reported to the standard output medium of the Generate host environment (e.g. command prompt window, system log). Set to 0 – do not report messages (default) 1 – switch reporting on.
#Restart	When this is included after Generate has failed, it will restart and continue processing from the last checkpoint using the information in the checkpoint file. See also the Publish Wizard checkpoint progress option in the Designer User's Guide. This can be included anywhere in the OPS file.
RangeOfPublications	<p>This is used if you want Generate to process only a subset of the publication data sets available in the input data file. You may want to do this if you need to rerun portions of a production job without creating a new input data file. You can indicate the sequential numbers of the publication data sets to be processed as follows:</p> <p>27, 280, 674 – specific publications</p> <p>100-1000 – publications between 100 and 1000</p> <p>1000+ – all publications after the first 1000</p> <p>366, 500-1000, 2000+ – combinations.</p>
ConstantShapeOffPage	<p>Defines the action Generate should take if graphic objects (including text boxes) positioned using constant values for both X and Y offsets are positioned all or in part outside the active logical page area. Options are:</p> <p>Abort – Generate aborts immediately. Any output files that have been created by the job are deleted (if this is permitted by the host operating system).</p> <p>Warn – a warning message is issued for each object that is found to be positioned outside the logical page area. Processing of the job continues as normal. The off page object is included in the output datastream; the effect of this in the printer/browser environment will depend on the device type. Ignore – as above but no warning message is issued. This is the default for objects placed using constant values.</p>

Sections, keywords and parameters	Usage
DynamicShapeOffPage	As above but this option applies to graphic objects positioned using variable data for either offset.
WorkSpace	<p>This option specifies a file template used by Generate to create temporary files at runtime. Refer to the Creating a host object section in the Designer User Guide.</p> <p>The section on creating a host object in the Designer User's Guide for further information Use either the %1 or %2 placeholders to create unique filenames, refer to the Specifying a file template section in the Designer User Guide.</p> <p>The Publish Wizard checkpoint progress option in the Designer User's Guide for further information. This option is not for use on z/OS. However you can define a temporary file explicitly when specifying the output file, refer to the Specifying files for Generate section in the Designer User Guide.</p>
SystemTempfiles	When set to Yes the host operating system will allocated temporary files for Generate to use at runtime. Note that either this option or the Workspace option should be used to manage temporary files.
SuppressMessages	<p>Use this setting to indicate the level message suppression used by Generate for your production job. You can either indicate the category of messages to be suppressed or indicate specific messages that you do not want to be reported. Choose from one of the options that follows:</p> <ul style="list-style-type: none"> • NONE - No messages suppressed • INFORMATION - All information messages suppressed • ALL - All warning and information messages suppressed • A list of comma separated message IDs to suppress (e.g. 121,420)When using this option you must specify only the identifier that appears before each message when issued.
<p><Overflow></p> <p>These options are used to override the Limit composed pages in memory settings on the Memory Handling page in the Publish Wizard. For details, see the section on error handling in the Publish Wizard options in the Designer User Guide.</p>	
OverflowEnabled	When set to No – the default value – the OverflowFile and OverflowSize settings only take effect if the Limit composed pages in memory option is specified in the Publish Wizard. When set to Yes , the OverflowFile and OverflowSize settings always take effect.

Sections, keywords and parameters	Usage
OverFlowFile	This option designates the temporary file to which the composed files are written.
OverFlowSize	This option specifies the memory limit at which the process of writing to the overflow file begins. The default memory value is 4 megabytes. You can use the suffix K to indicate kilobytes, M to indicate megabytes or if no suffix is used the value will be in bytes.
<Custom> This section allows you to specify any temporary settings that may be required as part of problem resolution. The keyword Name and associated Parameter will be provided directly by Customer Support as required. You may code as many entries in this section as necessary.	

Example

```

<Trace>
  Outputfile=trace%d.out
  TraceLevel=default
  outputcodepage=utf8
  memlimit=0
  publication=3

  <Messages>
  MandatoryNotPlaced=Warn
  MandatoryMessageError=Continue
  OptionalMessageError=Continue
  CampaignDate=12/07/2007
  Cycle=AC02
  MessageProcessing=Yes
  NoMessages=Warn

  <Advanced>
  ErrorFile=doc\backups\june21err.txt
  LogFile=trace04.out
  Checkpointfile=check.out
  CPconsole=1
  RangeOfPublications=100-350
  ConstantShapeOffPage=Warn
  DynamicShapeOffPage=Warn
  Workspace=d:\process\work\b%1xml
  SystemTempfiles=yes

  <OverFlow>
  OverFlowFile=doc\memerror.txt
  OverFlowSize=48m

```

```
<Custom>
PTF5690=""Type1""
```

Updating the configuration

In some cases, you might need to change configuration for one or more bundles. For example, if your database security policy requires periodic password changes, you will need to update the database credentials each time they change. To update the configuration, modify the same `deploy.properties` file used during installation, replacing the old values with the new, correct values. Then, for each bundle (as appropriate), execute the following command:

```
groovy eos -b <bundle-name> -p </path/to/deploy.properties> -t <node-type>
configure
```

Document Composition Adapter

The Document Composition Adapter (DCA), is a component that interacts with the document server to compose documents in the context of EngageOne. DCA is responsible for passing EngageOne document composition requests to the document server processes. The number of active the document server processes, as well as the life cycle for each process are managed by DCA via its configuration settings.

Namespace	Key	Value
com.pb.dca	pool.size.minimum	an integer greater than 0
com.pb.dca	pool.size.maximum	an integer greater than or equal to pool.size.minimum
com.pb.dca	pool.allocation.check.minutes	minutes, greater than 0 – how often DCA checks to make sure it's pool is within the defined parameters. Use with caution – see below.

The `pool.size.minimum` and `pool.size.maximum` settings determines the minimum and maximum active document server processes used in the EngageOne Server for document composition.

The underlying purpose of the `pool.allocation.check.minutes` setting is to manage the number of DCA instances and the associated resources that the document server allocates. If your system is not heavily used, then it allows unused instances of DCA to be shut down and releases those resources. Checking the pool too often however, can also hinder performance when the pool is under heavy load. This becomes more important depending upon the content and complexity of the documents being generated, as more memory and other resources are needed in certain situations. You will need to adjust this setting accordingly, depending on your hardware configuration, as well as the concurrency, scalability, and document composition requirements for your particular environment.

Incident recovery

When document composition fails the server gathers available information and resources and puts those items in the specified archive location.

Document Composition Adapter (DCA) Incident Archive

The DCA incident archive is a means of collecting information about potential issues during document composition.

DCA incident archives can be deleted once they have been reviewed and considered for possible reasons of document composition failure. These files are typically found in the `<ACTIVE-DRIVE>/incident-archive/...` of your installation directory. If you have customized this location you may need to refer to your configuration settings for the location of these files.

Following is an example of a possible entry.

```
D:\Program Files (x86)\EngageOne\EngageOne
Compose\active-drive\incident-archive\dca\2009-11-12-113937125-19_1
```

Accumulated Batch Incident Archive

There is another type of incident archive for accumulated batch. Similar to the DCA incident archive, batch gathers available information and resources and puts them in a specific location whenever there is an error in the final composition.

```
<ACTIVE-DRIVE>/incident-archive/batch/...
```

The sub-folder and naming convention is similar to the DCA incident archive:

```
D:\Program Files (x86)\EngageOne\EngageOne
Compose\active-drive\incident-archive\batch\2009-11-12-113937125-0_1
```

Both incident archives have an **incident.log**, which contain information and error logs from the application.

DBCS support

EngageOne supports the following encoding schemes for reporting files and exported files.

Encoding	Description
ASCII	American Standard Code for Information Interchange.
UTF-8	Eight-bit UCS Transformation Format.
UTF-16	Sixteen-bit UCS Transformation Format, byte order identified by an optional byte-order mark.
UnicodeBigUnmarked	Sixteen-bit Unicode Transformation Format, big-endian byte order.
UnicodeLittleUnmarked	Sixteen-bit Unicode Transformation Format, little-endian byte order.
UnicodeBig	Sixteen-bit Unicode Transformation Format, big-endian byte order, with byte-order mark.
UnicodeLittle (default)	Sixteen-bit Unicode Transformation Format, little-endian byte order, with byte-order mark.

The default encoding scheme set on installation is UTF-16LE. It has the default name `UnicodeLittle` and is set in the `config-settings.xml` file under the namespace `com.pb.engageone` as `default.file.encoding`. The namespaces `com.pb.engageone.server.batch` (reporting files) and `com.pb.openedms` (exported files) automatically inherit the `default.file.encoding` value – see the example below for accumulated batch.

If you require another encoding scheme you only need to change the `default.file.encoding` value in the `com.pb.engageone` namespace.

```
<namespace name="com.pb.engageone">
. . .
<setting>
<key>default.file.encoding</key>
<value>UnicodeLittle</value>
</setting>
. . .
```

```

</namespace>
<namespace name="com.pb.engageone.server.batch">
  . . .
  <setting>
    <key>csv.report.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  <setting>
    <key>diJ.report.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  <setting>
    <key>xml.report.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  <setting>
    <key>fixed.report.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  . . .
</namespace>
<namespace name="com.pb.openedms">
  . . .
  <setting>
    <key>resource.export.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  . . .
</namespace>
  . . .
  <setting>
    <key>resource.export.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  . . .
</namespace>

```

You can also tailor your requirements to an individual key, leaving the rest defaulted as in the following example where the `DIJ` report has an over-ride value of `UTF-8`.

Example for accumulated batch

```

<namespace name="com.pb.engageone.server.batch">
  . . .
  <setting>
    <key>csv.report.encoding</key>
    <value>${com.pb.engageone[default.file.encoding]}</value>
  </setting>
  <setting>
    <key>diJ.report.encoding</key>

```

```

<value>UTF-8</value> <!-- If desired/required, for example -->
</setting>
<setting>
<key>xml.report.encoding</key>
<value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
<setting>
<key>fixed.report.encoding</key>
<value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
. . .
</namespace>
<namespace name="com.pb.openedms">
. . .
<setting>
<key>resource.export.encoding</key>
<value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
. . .
</namespace>

```

Example for non-accumulated batch

```

<namespace name="com.pb.engageone.server.batch.na">
. . .
<setting>
<key>default.file.encoding</key>
<value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
<setting>
<key>di.j.report.encoding</key>
<value>${com.pb.engageone.batch[di.j.report.encoding]}</value>
</setting>
<setting>
<key>csv.report.encoding</key>
<value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
<setting>
<key>xml.report.encoding</key>
<value>${com.pb.engageone.batch[xml.report.encoding]}</value>
</setting>
<setting>
<key>fixed.report.encoding</key>
<value>${com.pb.engageone[default.file.encoding]}</value>
</setting>
. . .
</namespace>

```

Trusted connection

The trusted connection Web Service feature is used where there is a higher level of trust between EngageOne and a custom application. It can be configured setting proper properties in `deploy.properties` file. Refer to the `EngageOne_Configuration_Checklist.pdf` distributed with the release media for details.

See the Programmer's Reference Guide for further background and details about this Web Service.

Updating OpenAM configuration

The following OpenAM configuration settings can be modified in the `config-settings.xml` file.

Namespace	Configuration Setting	Description
<code>com.pb.engageone</code>	<code>OpenAm.url</code>	OpenAM server URL
<code>com.pb.engageone</code>	<code>OpenAM.userid</code>	user ID used to connect to OpenAM services
<code>com.pb.engageone</code>	<code>OpenAM.password</code>	the password of the above user
<code>com.pb.engageone</code>	<code>OpenAm.password.encrypted</code>	indicates whether or not the the password is encrypted

Note: By default the value of `OpenAM.password` is encrypted, so the default value of `OpenAM.password.encrypted` is `TRUE`. When you modify the password setting with plain text, you need to modify the value of `OpenAM.password.encrypted` to `FALSE` as well. The system would automatically encrypt the password value when application server is restarted.

Monitoring the Interactive Editor's WebSocket connections

The relay-service-client provides the ability to check and monitor the relay-service to ensure that messages can be sent between both sides involved in communication.

This section describe the process you need to follow to:

- enable the relay-service-client
- test and monitor communications between both parties.

Enabling and using the relay-service-client

You can enable the relay-service-client by configuring the following property found in the `deploy.properties` file:

```
relay.service.client.enabled
```

Once relay-service-client is enabled it is deployed in core bundle at which point it can be accessed via:

```
http/https:<core bundle url>:<core bundle port>/relay-service-client
```

For example:

```
https://eo-core-bundle-node1:8080/relay-service-client
```

It is important to note that where the core bundle is in a clustered setting, accessing the relay-service-client is performed under a loadBalancer url e.g.:

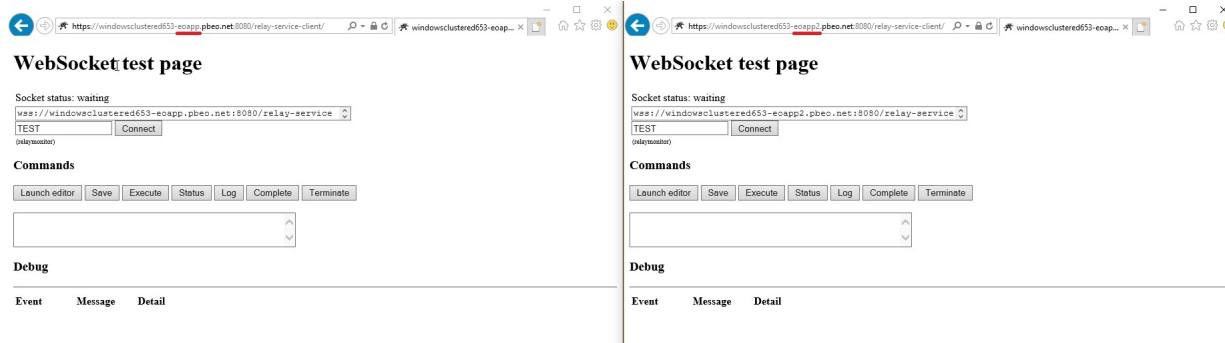
```
https://eo-core-bundle-loadBalancer:8080/relay-servcie-client
```

This will cause the relay-service-client to monitor or diagnoses websocket traffic on one node depending on the loadBalancer redirect. To investigate issues in the environment effectively you will need to connect to all core bundles in cluster. The example below illustrates a scenario where the core bundle has two nodes:

```
https://eo-core-bundle-node1:8080/relay-service-client
```

and

```
https://eo-core-bundle-node2:8080/relay-service-client
```



To send message from one relay-service-client instance to another:

- Open two relay-service-client instances.
- Specify randomly named sessionID (next to Connect button)
- Click Connect.

When the websocket channel has been establish you can send messages between clients by executing the appropriate command button e.g. **Save, Execute, Status, Log, Complete**.

Monitoring websocket communication

If you wish to monitor websocket communication you can run the relay-service-client in **Monitor Mode**. In this mode of operation the Monitor Mode relay-service-client allows you to view all communication exchanges between EngageOne Interactive and the Interactive Editor for all sessions.

To enable Monitor Mode:

1. Set the following property in `viewpoint.properties` file of core bundle set to `true`:

```
relay.service.client.monitor.mode.enabled
```

2. On the relay-service-client page type **relaymonitor** in **sessionID** text field (next to the **Connect** button) .
3. Click **Connect**.

At this point the relay-service-client should present all websocket traffic on the specific node.

WebSocket test page

Socket status: OnMessage | sessionId: TEST | session.getid: 2 | messageType: 3 [Terminate]

ws://windowclustered653-eoapp.pbeo.net:8080/relay-service

relaymonitor Connect

Debug

Event	Message	Detail
Info		Connected to: ws://windowclustered653-eoapp.pbeo.net:8080/relay-service/ica/relaymonitor
Info		OnOpen sessionId: relaymonitor session.getid: 1
Info		OnOpen sessionId: TEST session.getid: 2
Info		OnOpen sessionId: TEST session.getid: 3
Info		OnMessage sessionId: TEST session.getid: 2 messageType: 1 [Execute]
Info		OnMessage sessionId: TEST session.getid: 2 messageType: 3 [Terminate]

WebSocket test page

Socket status: Connected to: ws://windowclustered653-eoapp.pbeo.net:8080/relay-service/ica/TEST

ws://windowclustered653-eoapp.pbeo.net:8080/relay-service

TEST Connect

Commands

Launch editor Save Execute Status Log Complete Terminate

Debug

Event	Message	Detail
Info		Connected to: ws://windowclustered653-eoapp.pbeo.net:8080/relay-service/ica/TEST
Msg Tx	Execute	[Length=18 Type=1 SeqNo=1 SID=TEST] {}
Msg Tx	Terminate	[Length=16 Type=3 SeqNo=2 SID=TEST] {}

WebSocket test page

Socket status: Connected to: ws://windowclustered653-eoapp.pbeo.net:8080/relay-service/ica/TEST

ws://windowclustered653-eoapp.pbeo.net:8080/relay-service

TEST Connect

Commands

Launch editor Save Execute Status Log Complete Terminate

Debug

Event	Message	Detail
Info		Connected to: ws://windowclustered653-eoapp.pbeo.net:8080/relay-service/ica/TEST
Msg Rx	Execute	[Length=18 Type=1 SeqNo=1 SID=TEST] {}[69,88,80,0,18,1,0,0,0,1,0,4,84,69,83,84,0,0]
Msg Rx	Terminate	[Length=16 Type=3 SeqNo=2 SID=TEST] {}[69,88,80,0,16,3,0,0,0,2,0,4,84,69,83,84]

11 - EngageOne Server system monitoring

In this section

Use case 1: Monitoring with Oracle's JConsole.....	264
Use case 2: Monitoring with commercial monitoring tools - example: AppDynamics.....	267



Use case 1: Monitoring with Oracle's JConsole

You can change the EngageOne Server configuration to enable Java Management Extensions (JMX) and monitor key resources using Oracle's JConsole – which is freely available with the Java Development Kit (JDK). The resources that can be monitored in this way include:

- Memory usage
- CPU usage
- Threads
- Classes loaded
- VM summary

The next section describes the configuration changes you must make and a brief overview of JConsole.

Enabling JMX (no security)

The following describes how to enable JMX instrumentation in a non-secure manner. Depending on your environment this may be sufficient and is relatively straightforward (in a production environment however, it is strongly recommended that you Enable JMX with the secure option).

Perform the following steps to enable JMX (non-secure):

1. Stop the bundle service you wish to instrument.
2. Update the '<bundle>.jvm.settings' property in the deploy.properties file.
In the deploy.properties file each bundle has its own jvm.settings (such as core.jvm.settings, conversion.jvm.settings). You need to add the following settings from Oracle:

- -Dcom.sun.management.jmxremote
- -Dcom.sun.management.jmxremote.port=PORT_NUMBER

where PORT_NUMBER is an open port of your choice.

- -Dcom.sun.management.jmxremote.authenticate=false
- -Dcom.sun.management.jmxremote.ssl=false

For example:

```
core.jvm.settings=-Xms1024m -Xmx4g -XX:MaxPermSize=512M
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9994
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
```

3. Run the `configureBundle.groovy` script to apply configuration.
4. Restart the bundle service.

For more details on the `deploy.properties` and the `configureBundle.groovy` script, see the *EngageOne Server Installation Guide*.

Enabling JMX (with security)

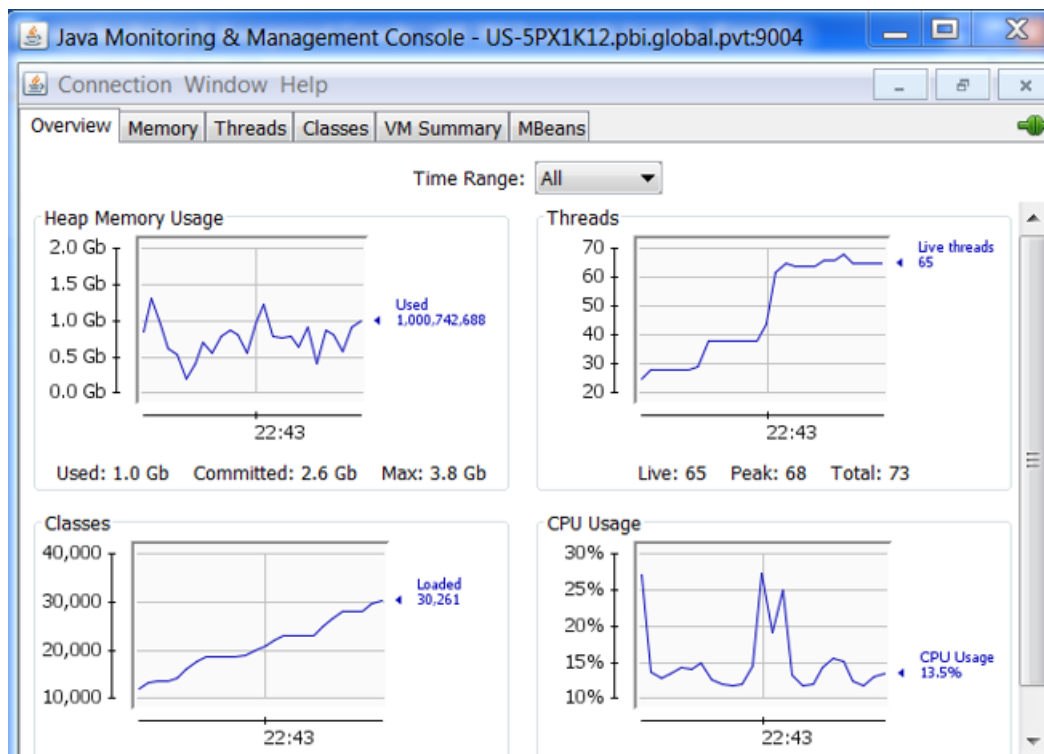
There are two ways to enable secure access to JConsole remotely. Password based authentication and SSL enabled security. If you are interested in configuring JMX in a secure manner please see Oracle's documentation. This link discusses remote access options and best practices. Additional JVM settings are required and should be applied in the same manner as described above.

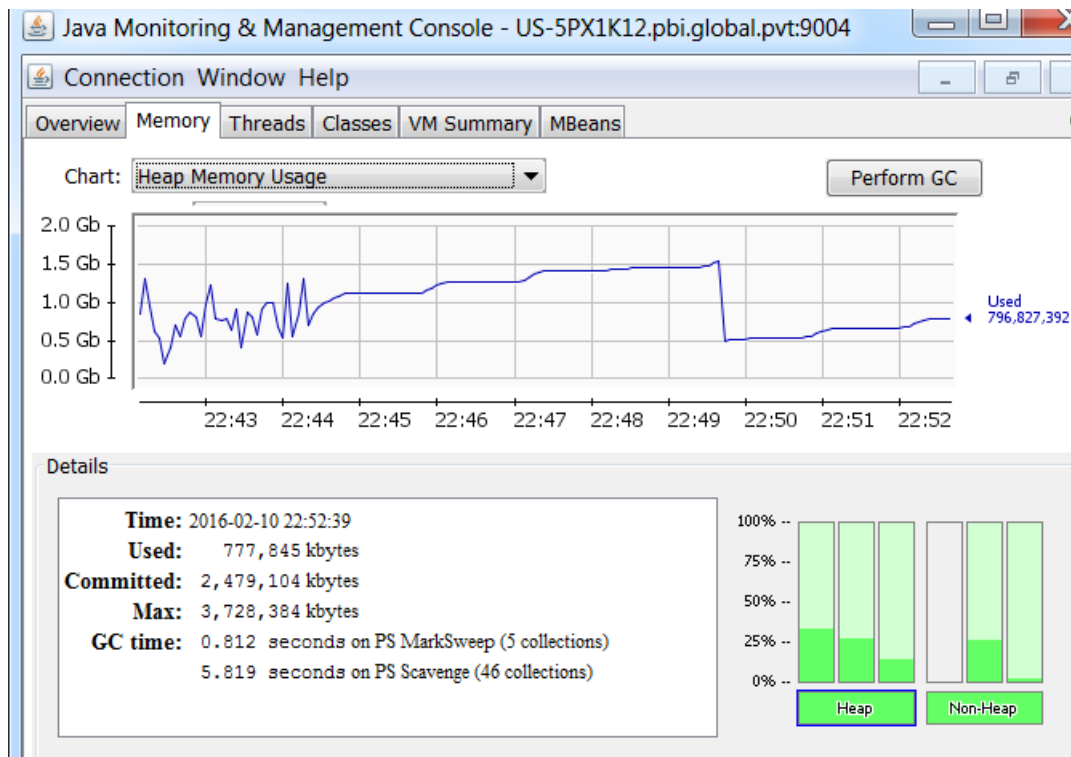
Note: For the overall security of the EngageOne application, it is recommended that JConsole is enabled in a secured option.

Using JConsole

JConsole shows many metrics including memory usage, threads, CPU usage, class loaders and a VM summary. More information on each tab and definition of key terms can be found from the [Oracle JConsole page](#).

Sample JConsole screens.





Use case 2: Monitoring with commercial monitoring tools - example: AppDynamics

EngageOne Server, like all web applications, can be monitored with other third-party tools. Below is an overview of how you can configure the system for monitoring by AppDynamics. AppDynamics is a commercial IT monitoring tool not included with the EngageOne Server.

Note: The configuration of AppDynamics may differ based on the AppDynamics version and installation.

Enabling AppDynamics

The following procedure assumes you are knowledgeable in the AppDynamics system, and that the proper license and AppDynamics agent are installed on the server.

Perform the following steps to enable AppDynamics:

1. Stop the bundle service you wish to instrument.
2. Update the '<bundle>.jvm.settings' property in the deploy.properties file.
In the deploy.properties file each bundle has its own jvm.settings (such as core.jvm.settings, conversion.jvm.settings). You need to add the following settings from AppDynamics:
 - -javaagent:APP_DYNAMICS_INSTALL/javaagent.jar
where APP_DYNAMICS_INSTALL is the install location of AppDynamics
 - -Dappdynamics.agent.uniqueHostId=UNIQUE_HOST_ID
where UNIQUE_HOST_ID is your chosen host ID for this bundle/installation
 - -Dappdynamics.cron.vm=true

For example:

```
core.jvm.settings=-Xms1024m -Xmx4g -XX:MaxPermSize=512M
-javaagent:C:\:\appdynamics-agent\AppServerAgent\javaagent.jar
-Dappdynamics.agent.uniqueHostId=EngageOneCoreNode1
-Dappdynamics.cron.vm=true
```

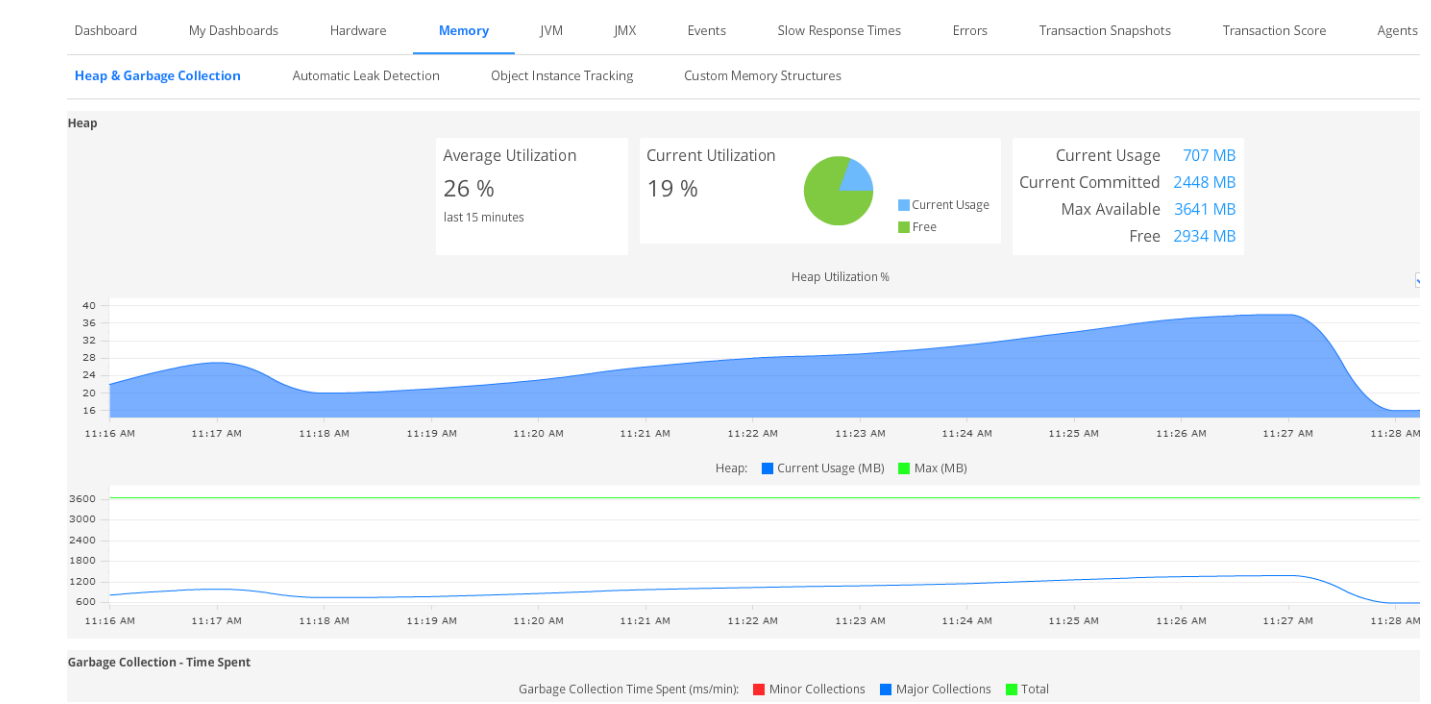
3. Run the configureBundle.groovy script to apply configuration.
4. Restart the bundle service.

For more details on the deploy.properties file and the configureBundle.groovy script, see the *EngageOne Server Installation Guide*.

Using AppDynamics

AppDynamics is a robust monitoring tool with many features. Consult your AppDynamics user guide on how to launch the AppDynamics dashboard. One of the most useful pieces of monitoring information provided by AppDynamics is the 'Memory Usage' graph, shown below.

Sample AppDynamics screen.



Notices



Copyright ©2008, 2021 Precisely. All rights reserved.

This publication and the software described in it is supplied under license and may only be used or copied in accordance with the terms of such license. The information in this publication is provided for information only, is subject to change without notice, and should not be construed as a commitment by Precisely. To the fullest extent permitted by applicable laws Precisely excludes all warranties, representations and undertakings (express or implied) in relation to this publication and assumes no liability or responsibility for any errors or inaccuracies that may appear in this publication and shall not be liable for loss or damage of any kind arising from its use.

Except as permitted by such license, reproduction of any part of this publication by mechanical, electronic, recording means or otherwise, including fax transmission, without the express permission of Precisely is prohibited to the fullest extent permitted by applicable laws.

Nothing in this notice shall limit or exclude Precisely liability in respect of fraud or for death or personal injury arising from its negligence. Statutory rights of the user, if any, are unaffected.

*TALO Hyphenators and Spellers are used. Developed by TALO B.V., Bussum, Netherlands Copyright © 1998 *TALO B.V., Bussum, NL *TALO is a registered trademark ®

Encryption algorithms licensed from Unisys Corp. under U.S. Patent No. 4,558,302 and foreign counterparts.

Security algorithms Copyright © 1991-1992 RSA Data Security Inc

Copyright © DL Technology Ltd 1992-2010

Barcode fonts Copyright © 1997 Terrapin Solutions Ltd. with NRB Systems Ltd.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

Artifex and the Ghostscript logo are registered trademarks and the Artifex logo and Ghostscript are trademarks of Artifex Software, Inc.

This product contains the Regexp++ library Copyright © 1998-2000 Dr. John Maddock

PostScript is a trademark of Adobe Systems Incorporated.

PCL is a trademark of Hewlett Packard Company.

Copyright (c) 2000 - 2015 The Legion of the Bouncy Castle Inc. (<http://www.bouncycastle.org>)

ICU License - ICU 1.8.1 and later Copyright (c) 1995-2006 International Business Machines Corporation and others All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

Matra 0.8.2b (<http://matra.sourceforge.net/>) The contents of this documentation are subject to the Mozilla Public License Version 1.1 (the "License"); you may not use this documentation except in compliance with the License. You may obtain a copy of the License at <http://www.mozilla.org/MPL/>. Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License. Otherwise all product names are trademarks or registered trademarks of their respective holders.

This product contains Sycamore, version number 0.5.0, which is licensed under the MIT license. The license can be downloaded from <https://github.com/ryexley/sycamore/blob/master/dist/requester.js>. The source code for this software is available from <https://github.com/ryexley/sycamore>.

This product contains Underscore, version number 1.13.1, which is licensed under the MIT license. The license can be downloaded from <https://github.com/jashkenas/underscore/blob/master/LICENSE>. The source code for this software is available from <http://underscorejs.org/>.

This product contains Flowable, version number 6.4.2, which is licensed under the Apache license. The license can be downloaded from <https://github.com/flowable/flowable-engine/blob/master/LICENSE>. The source code for this software is available from <https://github.com/flowable/flowable-engine>.

This product contains Bootstrap, version number 3.4.1, which is licensed under the MIT license. The license can be downloaded from <http://getbootstrap.com/getting-started/#license-faqs>. The source code for this software is available from <http://getbootstrap.com/getting-started/#download..>

This product contains Commons-Configuration, version number 1.10, which is licensed under the Apache license. The license can be downloaded from <http://www.apache.org/licenses/>. The source code for this software is available from <http://commons.apache.org/proper/commons-configuration/>.

This product contains jQuery, version number 3.5.1, which is licensed under the MIT license. The license can be downloaded from <https://jquery.org/license/>. The source code for this software is available from <http://jquery.com/download/>.

This product contains Knockout-AMD-Helpers, version number 0.7.4, which is licensed under the MIT license. The license can be downloaded from <https://github.com/rniemeyer/knockout-amd-helpers/blob/master/LICENSE>. The source code for this software is available from <https://github.com/rniemeyer/knockout-amd-helpers>.

This product contains Knockout, version number 3.5.1, which is licensed under the MIT license. The license can be downloaded from <https://github.com/knockout/knockout/blob/master/LICENSE>. The source code for this software is available from <http://knockoutjs.com/downloads/>.

This product contains bootstrap-datetimepicker, version number 4.17.49, which is licensed under the MIT license. The license can be downloaded from <https://github.com/myactionreplay/bootstrap-datetimepicker/blob/master/LICENSE>. The source code for this software is available from <https://github.com/myactionreplay/bootstrap-datetimepicker>.

This product contains Knockout-DelegatedEvents, version number 0.6.1, which is licensed under the MIT license. The license can be downloaded from

<https://github.com/rniemeyer/knockout-delegatedEvents#license>. The source code for this software is available from <https://github.com/rniemeyer/knockout-delegatedEvents>.

This product contains Moment.js, version 2.29.1, which is licensed under the MIT license. The license can be downloaded from <https://github.com/moment/moment/blob/develop/LICENSE>. The source code for this software is available from <http://momentjs.com/>.

This product contains Quartz-Scheduler, version number 2.3.2, which is licensed under the Apache license. The license can be downloaded from <http://quartz-scheduler.org/>. The source code for this software is available from <http://quartz-scheduler.org>.

This product contains RequireJS Text, version number 2.0.15, which is licensed under the BSD and MIT licenses. The license can be downloaded from

<https://github.com/requirejs/text/blob/master/LICENSE>. The source code for this software is available from <https://github.com/requirejs/text>.

This product contains RequireJS, version number 2.3.6, which is licensed under the BSD and MIT licenses. The license can be downloaded from

<https://github.com/jrburke/requirejs/blob/master/LICENSE>. The source code for this software is available from <http://requirejs.org/docs/download.html>.

This product contains Apache ActiveMQ, version number 5.15.9, which is licensed under the Apache license, version number 2.0. The license can be downloaded from

<http://www.apache.org/licenses/>. The source code for this software is available from <http://activemq.apache.org>

This product contains Apache NMS version 1.7.2, which is licensed under the Apache license, version number 2.0. The license can be downloaded from <http://www.apache.org/licenses/>. The source code for this software is available from: <http://activemq.apache.org/nms>

This product contains Apache Commons DBCP2, version number 2.6.0, which is licensed under the Apache license, version number 2.0. The license can be downloaded from

<http://www.apache.org/licenses/>. The source code for this software is available from <https://commons.apache.org/proper/commons-dbcpl/>.

This product contains OWASP Encoder, version number 1.2.2, which is licensed under the BSD license. The license can be downloaded from <https://opensource.org/licenses/BSD-3-Clause>.

The source code for this software is available from

https://www.owasp.org/index.php/OWASP_Java_Encoder_Project.

This product contains Narayan, version number 5.2.13.Final, which is licensed under the LGPL license, version number 2.1. The license can be downloaded from

<http://www.gnu.org/licenses/old-licenses/lgpl-2.1.txt>. The source code for this software is available from <http://narayana.io/>.

This product contains Logback, version number 1.2.3, which is licensed under the EPL and LGPL licenses, version numbers 1.0 and 2.1. The license can be downloaded from

<http://logback.qos.ch/license.html>. The source code for this software is available from <http://logback.qos.ch/>.

This product contains JBoss Weld, version number 3.1.0.Final, which is licensed under the Apache license, version number 2.0. The license can be downloaded from <http://www.apache.org/licenses/LICENSE-2.0>. The source code for this software is available from <http://weld.cdi-spec.org>.

This product contains Hibernate, version number 5.4.25.Final, which is licensed under the Apache and LGPL license, version numbers 2.0 and 2.1. The license can be downloaded from <http://hibernate.org/community/license/>. The source code for this software is available from <http://hibernate.org/orm/>.

This product contains Apache Tomcat, version number 9.0.43, which is licensed under the Apache license, version number 2.0. The license can be downloaded from <http://www.apache.org/licenses/LICENSE-2.0>. The source code for this software is available from <http://tomcat.apache.org/>.

This product contains Apache Procrun, version number 1.1.0, which is licensed under the Apache license, version number 2.0. The license can be downloaded from <http://www.apache.org/licenses/>. The source code for this software is available from <http://commons.apache.org/proper/commons-daemon/procrun.html>.

This product contains FasterXML Jackson, version number 2.9.8, which is licensed under the Apache license, version number 2.0. The license can be downloaded from <http://www.apache.org/licenses/>. The source code for this software is available from <https://github.com/FasterXML/jackson>.

This product contains Log4net. The license for log4net can be downloaded from <https://www.apache.org/licenses/LICENSE-2.0>. The source code for this software is available from https://logging.apache.org/log4net/download_log4net.cgi.

Support

Click [here](#) for full EngageOne Compose documentation and access to your peers and subject matter experts on the Knowledge community.



1700 District Ave Ste 300
Burlington MA 01803-5231
USA

www.precisely.com

© 2008, 2021 Precisely. All rights reserved.