

Implementing webMethods Content Service Platform for BPM

Version 9.0 SP1

June 2013

This document applies to webMethods Content Service Platform Version 9.0 SP1 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2010-2013 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Table of Contents

| | |
|--|-----------|
| About this Guide..... | 7 |
| Document Conventions..... | 7 |
| Online Information..... | 8 |
| Introduction to webMethods Content Service Platform..... | 9 |
| Overview..... | 10 |
| A Note About Documentation..... | 11 |
| General Concepts..... | 12 |
| Suite Integration..... | 13 |
| General Usage..... | 14 |
| Using Document Types in a Business Process..... | 15 |
| Security Considerations..... | 17 |
| Authentication..... | 17 |
| Authorization..... | 17 |
| Use of Metadata..... | 18 |
| About CSP Startup and Repository Connectivity..... | 19 |
| About Oracle Database Errors..... | 19 |
| About SharePoint Text Handling..... | 20 |
| About the Content Service Platform WebReader..... | 20 |
| Configuring and Deploying a webMethods Content Service Platform Environment..... | 23 |
| Overview of Content Service Platform Environment Configuration..... | 24 |
| About Repository Connectivity..... | 24 |
| About the Central Configuration Interface..... | 25 |
| Determining Whether the Central Configuration Interface Is Installed..... | 25 |
| Installing the Central Configuration Interface..... | 25 |
| Content Service Platform Environment Considerations..... | 26 |
| Configuring a Content Service Platform Environment..... | 26 |
| Accessing the Content Service Platform Home Page..... | 29 |
| About the Content Service Platform Home Page..... | 29 |
| Deploying a webMethods Content Service Platform Environment..... | 30 |
| Deployment Considerations..... | 30 |
| Prerequisites..... | 31 |
| Deploying the Environment..... | 31 |
| Updating a Content Service Platform Environment..... | 31 |
| Changing Host Port Numbers..... | 32 |
| Creating a Content Filter..... | 35 |
| About Content Filters..... | 36 |
| Determining Whether the Content Service Platform Configuration Interface Is Installed..... | 36 |
| Installing the Content Service Platform Configuration Interface..... | 36 |
| Creating a Content Filter..... | 37 |

| | |
|--|-----------|
| Viewing and Modifying Content Filters..... | 38 |
| Creating a Content Listener..... | 39 |
| About Content Listeners..... | 40 |
| Creating a Content Listener..... | 40 |
| Creating a Listener for Deactivated Document Types in a Repository..... | 42 |
| Viewing and Modifying Content Listeners..... | 42 |
| About the Operation of CSP Content Listeners..... | 43 |
| Configuring Portlet Applications for Use with the Content Service Platform..... | 45 |
| Configuring Content Service Platform Environment Entries for Portlet Applications..... | 46 |
| Synchronizing Document Types..... | 49 |
| Synchronizing CSP Document Types and IS Document Types..... | 50 |
| Maintaining CSP Document Types and IS Document Types..... | 50 |
| Introduction to E-forms Integration..... | 53 |
| Overview..... | 54 |
| General Concepts..... | 54 |
| Requirements..... | 55 |
| Suite Integration..... | 56 |
| About Using the Content Service Platform As an E-form Repository..... | 56 |
| Working with E-form Templates..... | 57 |
| General Usage..... | 58 |
| Using E-forms in a Business Process..... | 59 |
| Implementing E-forms..... | 61 |
| Adobe LiveCycle Implementation Considerations..... | 62 |
| Creating XFA-Compliant Forms with LiveCycle Designer..... | 62 |
| Limitations to Using XDP Files..... | 63 |
| Adobe Design Time Considerations for Task Applications..... | 63 |
| Defining the Path Between an XDP and a PDF File..... | 64 |
| Determining the CSP URL for a PDF File..... | 65 |
| Creating an XDP Form without an Associated PDF..... | 65 |
| Using an XML Schema as a Template Source..... | 65 |
| Making Adobe E-form Instances Available at Run Time..... | 66 |
| 1st Scenario: PDF E-forms..... | 66 |
| 2nd Scenario: XDP E-forms..... | 66 |
| Using LiveCycle E-forms with Adobe Reader..... | 66 |
| Behavior When Downloading an E-form from a Task..... | 67 |
| Using Traditional PDF Forms Created with Adobe Acrobat..... | 67 |
| Using Digital Signatures with Adobe LiveCycle E-forms..... | 67 |
| Microsoft Office InfoPath Implementation Considerations..... | 68 |
| Making InfoPath E-form Instances Available at Run Time..... | 68 |
| Working with InfoPath Files in Non-Windows Environments..... | 68 |
| Other Solutions for Non-Windows Environments..... | 69 |
| Field Limitations..... | 69 |

| | |
|---|-----------|
| Other Limitations and Information..... | 69 |
| Using Digital Signatures with Microsoft InfoPath E-forms..... | 70 |
| Content Service Platform Services..... | 71 |
| Content Service Platform Built-In Services Location..... | 72 |
| Connection Folder..... | 72 |
| pub.csp.connection:addConnection..... | 73 |
| pub.csp.connection:deleteConnection..... | 74 |
| pub.csp.connection:listConnections..... | 74 |
| Content Folder..... | 74 |
| pub.csp.content:createContentID..... | 75 |
| pub.csp.content:createFilter..... | 76 |
| pub.csp.content:createListener..... | 77 |
| pub.csp.content:listAllContentDefinitions..... | 78 |
| pub.csp.content:listAllIndexIDs..... | 78 |
| File Folder..... | 79 |
| pub.csp.file:addFile..... | 80 |
| pub.csp.file:getFile..... | 81 |
| pub.csp.file:retrieveFiles..... | 81 |
| pub.csp.file:updateFile..... | 82 |
| Search Folder..... | 83 |
| pub.csp.search:createMetadataMap..... | 84 |
| pub.csp.search:createSearchTerm..... | 84 |
| pub.csp.search:searchMetadata..... | 85 |
| Server Folder..... | 86 |
| pub.csp.server:createIServerForListener..... | 86 |
| pub.csp.server:listAllIServers..... | 87 |
| Index..... | 89 |

About this Guide

This guide provides information to software developers about how to integrate content using webMethods Content Service Platform within the webMethods Product Suite. You can integrate this content into business processes and user interfaces implemented with the webMethods BPMS infrastructure. This guide describes:

- General concepts and information about working with webMethods Content Service Platform.
- How to configure and deploy a webMethods Content Service Platform environment.
- How to configure a webMethods Content Service Platform content filter definition.
- How to configure and deploy a webMethods Content Service Platform listener.

For specific information about the features, functions, and operation of webMethods Content Service Platform components, refer to the documentation provided in the installation directory: *Software AG_directory/CSP/*.

Document Conventions

| Convention | Description |
|----------------|--|
| Bold | Identifies elements on a screen. |
| Narrowfont | Identifies storage locations for services on webMethods Integration Server, using the convention <i>folder.subfolder:service</i> . |
| UPPERCASE | Identifies keyboard keys. Keys you must press simultaneously are joined with a plus sign (+). |
| <i>Italic</i> | Identifies variables for which you must supply values specific to your own situation or environment. Identifies new terms the first time they occur in the text. |
| Monospace font | Identifies text you must type or messages displayed by the system. |
| { } | Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols. |

| Convention | Description |
|------------|---|
| | Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the symbol. |
| [] | Indicates one or more options. Type only the information inside the square brackets. Do not type the [] symbols. |
| ... | Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...). |

Online Information

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at <http://documentation.softwareag.com>. The site requires Empower credentials. If you do not have Empower credentials, you must use the TECHcommunity website.

Software AG Empower Product Support Website

You can find product information on the Software AG Empower Product Support website at <https://empower.softwareag.com>.

To submit feature/enhancement requests, get information about product availability, and download products, go to [Products](#).

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the [Knowledge Center](#).

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at <http://techcommunity.softwareag.com>. You can:

- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

1 Introduction to webMethods Content Service Platform

| | |
|---|----|
| ■ Overview | 10 |
| ■ General Concepts | 12 |
| ■ General Usage | 14 |
| ■ Security Considerations | 17 |
| ■ Use of Metadata | 18 |
| ■ About CSP Startup and Repository Connectivity | 19 |
| ■ About Oracle Database Errors | 19 |
| ■ About SharePoint Text Handling | 20 |
| ■ About the Content Service Platform WebReader | 20 |

Overview

Incorporating a wide range of electronic information sources, or *content*, into daily operations is a critical and ever-increasing component of today's business processes. In addition to being created within an organization, information sources outside the organization provide forms, images, and other content used by the day-to-day processes of the organization. This information must be stored and made available to the enterprise in an organized and seamless manner.

Most organizations have taken steps to organize this content in specialized content environments, such as Enterprise Content Management (ECM) systems and Document Management Systems (DMS). Over time, the number of content repositories tends to grow, creating a need for a single, integrated view into all of the available content repositories.

webMethods Content Service Platform not only provides you with a complete content storage and management environment, but also gives you the ability to connect to, write to, and read from many supported third-party content management products. webMethods Content Service Platform delivers integrated storage, access, modification, and management of a wide variety of document types across one or more content servers, and makes the content available for use by the component products of the webMethods product suite.

For example, a particular document type in a particular repository location can be monitored for changes; when a change occurs, the content can be retrieved and used to start a new instance of a business process in the webMethods Process Engine. In another situation, a particular document type or form can be used as input to or output from a task running the webMethods Task Engine.

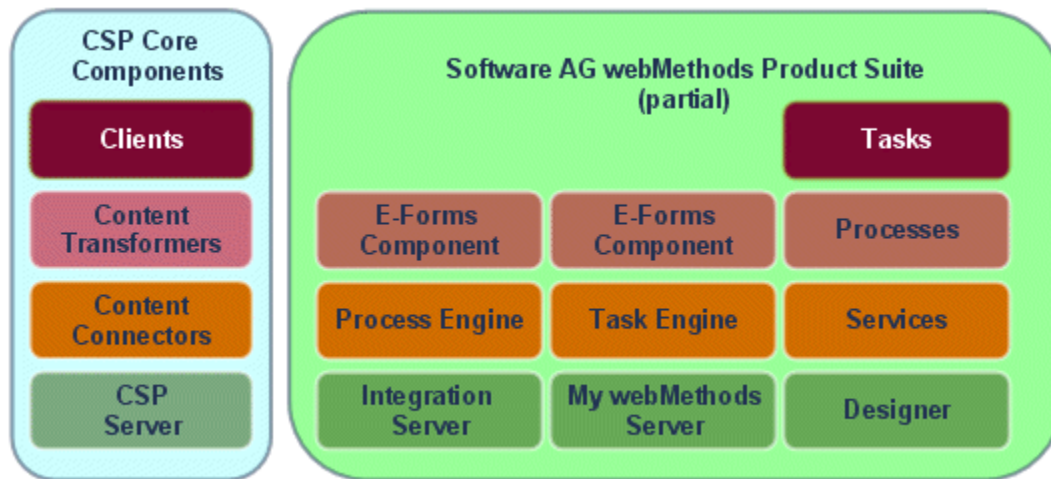
You can also use webMethods Content Service Platform to expose process and task information from the webMethods run time in the webMethods Content Service Platform client application, as well as in supported repositories.

The following components support webMethods Content Service Platform functionality (these component must be installed on network-accessible servers; for installation procedures, see the PDF publication *Installing webMethods and Intelligent Business Operations Products*):

- **webMethods Content Service Platform.** This component provides integration capabilities for content on different operating systems, databases, storage types, within certain media and application limits. It enables you to connect to a wide range of enterprise content management (ECM) repositories. It provides functions such as record, life cycle management, and reporting.
- **CSP Client.** Three client applications are available: a Windows CSP Client, a Web CSP Client, and a CSP Capture Client. These clients enable you to browse, search, and manage content in the webMethods Content Service Platform as well as in any connected repositories, providing a single, integrated view of your enterprise content environment.

- **Content Connectors.** If you plan to use webMethods Content Service Platform to integrate content from existing content repositories in your enterprise, you must have a content connector installed and configured for each content management platform you plan to connect to.

These components, plus the other webMethods suite components that interact with webMethods Content Service Platform (CSP), are shown in the figure below:



In addition, you are responsible for:

- Obtaining, installing, and configuring the server hardware and software for your supported content management servers (for example, instances of Microsoft Sharepoint or EMC Documentum).
- Enabling network connectivity from products of the webMethods product suite to your content management environment and applications.

This guide describes:

- General concepts and information about working with webMethods Content Service Platform.
- How to configure and deploy a webMethods Content Service Platform environment.
- How to configure a webMethods Content Service Platform content filter.
- How to configure a webMethods Content Service Platform listener.
- How to configure task applications to interact with webMethods Content Service Platform.

A Note About Documentation

This guide provides primarily conceptual and integration information for implementing webMethods Content Service Platform (CSP) within the webMethods Product Suite. For specific information about the features, functions, and operation of the webMethods Content Service Platform core components, including content creation

and administration, refer to the CSP HTML help, which can be found in any of these locations:

- In the product installation directory: *Software AG_directory\CSP\doc*
- If you have installed the Documentation component with the Software AG Installer:
*Software AG_directory_documentation\webMethods\Third-Party
\Content_Service_Platform*
- In the documentation area of the Empower Product Support website: <https://empower.softwareag.com> (log in required).

For additional information about working with webMethods Content Service Platform and related suite components, see the following webMethods documentation:

- *webMethods Service Development Help* (online help and PDF)
- *webMethods BPM Process Development Help* (online help and PDF)
- *webMethods BPM Task Development Help* (online help and PDF)
- *Administering webMethods Process Engine* (PDF)
- *webMethods Task Engine User's Guide* (PDF)
- *webMethods Integration Server Administrator's Guide* (PDF)

General Concepts

The webMethods Content Service Platform supports connection to over 50 content management applications and over 100 document types. Document types include such information as Microsoft Office documents, e-mail messages, rich media files, e-forms, image files, and reports.

Document types can be imported into Software AG Designer where they are converted to a standard IS document type (an IS document type is a proprietary Software AG format for exchanging information between business process components). You can use the IS document type to describe the input for a business process, as the business data for a task, or for any other supported IS document type use. For more information about creating IS document types, see the *webMethods Service Development Help*.

For document types that have an e-form associated with them (Adobe LiveCycle and Microsoft InfoPath are supported), both the e-form and the metadata are used to create the IS document. Document types that are not associated with an e-form are converted to an IS document type using only the document type's metadata.

Document types are also accessed at run time by the webMethods run-time components (Process Engine and Task Engine).

To integrate your stored document types into business processes and tasks, Software AG Designer must have network access to the storage location at design time.

During run time, the content location must be available to the Task Engine if you have implemented the ability for a user to download the content from a task instance running

in My webMethods. For more information about working with content and attachments in a task application, see the *webMethods BPM Task Development Help*.

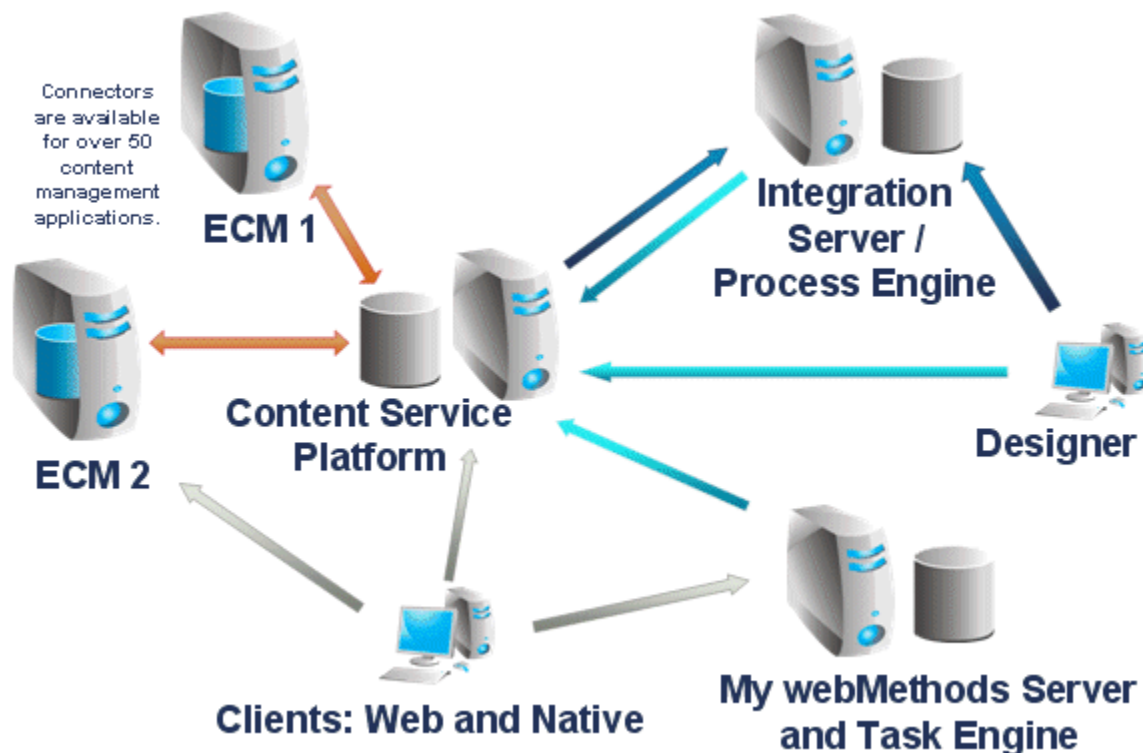
If Integration Server is connected to a Broker at the time you create an IS document type from a document type, the resulting IS document type will be publishable to the Broker and will have an associated Broker document type.

To use a document type as a run-time IS document type source that starts a business process, a webMethods Content Service Platform environment must be configured and deployed to the Integration Server host where the process will run.

For more information about configuring and deploying a Content Service Platform environment, see "[Configuring and Deploying a webMethods Content Service Platform Environment](#)" on page 23.

Suite Integration

webMethods Content Service Platform integration interacts with the following products in the webMethods Product Suite:



- **My webMethods.** You carry out the following activities related to webMethods Content Service Platform in My webMethods:
 - You use the Central Configuration interface in My webMethods to configure and deploy webMethods Content Service Platform environment definitions.

- You use the Content Configurator interface in My webMethods to create and manage content filters and content listeners.
- If you implement task applications that work with document types, you configure your deployed task applications for use with webMethods Content Service Platform by logging in to My webMethods as a system administrator.
- If you implement task applications that work with document types, users will interact with the tasks in My webMethods.
- If you implement and run business processes that work with document types, users will interact with the process instances in My webMethods.
- **Software AG Designer** You use the following Designer features when implementing and integrating content document types:
 - **webMethodsProcess Development.** This feature provides the ability to design, build, and deploy business process models, including the ability to integrate the process model with content document types.
 - **webMethodsTask Development.** This feature provides the ability to design, build, and publish business task types, including the ability to integrate the task type with content document types.
 - **webMethodsService Development.** This feature provides the ability to import content document types from the Content Service Platform into Integration Server to create a corresponding IS document type. You then use this IS document type to interact with business process models as well as task applications.
- **Integration Server and Process Engine.** You use the Integration Server and Process Engine as the run-time environment for webMethods business processes. The Integration Server host is the target for your deployed webMethods Content Service Platform environment definition, and the Process Engine receives notification when a content listener detects the creation, modification, or deletion of a document type instance in a monitored repository folder; this can be used to start a corresponding business process.
- **webMethods Task Engine.** If you have implemented task applications with document type interaction, the Task Engine manages these tasks and the document type-derived business data associated with the task. If you implement the ability to download a document type from a task instance running in My webMethods, the Task Engine formats the business data for downloading, and, after uploading, modifies and saves the task business data with the modified data from the uploaded document type.

General Usage

The most common scenarios for integrating document types from the webMethods Content Service Platform into the webMethods Product Suite are:

- To use changes in repository content status to start or provide data to a business process instance running in the Process Engine.
- To save content generated by a business process to a supported repository.
- To use repository content to provide business data to a task instance running in the Task Engine.
- To save content generated by a task to a supported repository.
- To provide users with download/upload access to forms and form data in a running task. This enables users to download a document type, modify it offline, and then upload it back to the task later.

Implementation of all of the above capabilities are supported with wizards and drag-and-drop functionality within features of Software AG Designer.

Using Document Types in a Business Process

This section provides an overview of the basic workflow for using webMethods Content Service Platform (CSP) document types with a business process; for specific information and procedures, see:

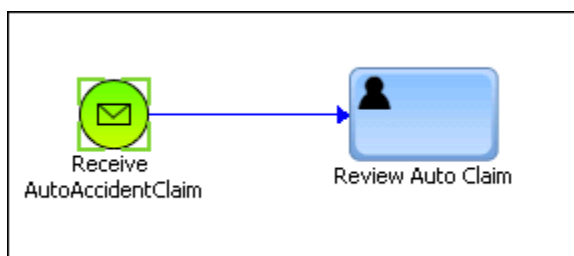
- *webMethods Service Development Help*
- *webMethods BPM Process Development Help*
- *webMethods BPM Task Development Help*
- *webMethods Task Engine User's Guide*

To use document types in a business process, you follow these general procedures:

1. Create a business process.
2. Create an IS document type from a CSP document type with the Create New Document Type wizard, as described in the *webMethods Service Development Help*.

Note: The generated document type is a fully compliant IS document type and can be used in the same ways as any other IS document type.

3. Drag the resulting IS document type onto the business process canvas to create a start message event, or drop it onto an existing start message event to update the event with the new document. For example:



Note: If the IS document is sourced from a standard CSP document type, the metadata from the CSP document type is available in the pipeline. If an e-form CSP document type is the source, both the document type metadata and the e-form metadata are available in the pipeline.

4. Specify the content listener server/repository location.

| | |
|---------------------------|---|
| Receive Protocol | Subscription (For Broker Documents) |
| Receive Document | csptest.docs:AutoAccidentClaim |
| | <input checked="" type="checkbox"/> E-form (For E-form Triggered Processes) |
| E-form Content Repository | localhost_9010;Start Automobile Claim Process |
| E-form Template Name | CMP:C:\workspace\,metadata\,plugins\com.sofi |

5. If you want to integrate the document type data with a task in the process, create a user task in one of the following ways (a process can contain multiple tasks):
 - In the Package Navigator view, use the **CAF > Generate Task** menu option from the IS document type, then:
 - i. Configure the task as required (assignment, events, etc.)
 - ii. Use the Task UI Update wizard to add the content ID, document type data fields, and content user interface components to the task interface, as described in the *webMethods BPM Task Development Help*.
 - Select an existing task, or implement an existing activity in the process as a user task, and then:
 - i. Add the IS document type as business data if not yet included.
 - ii. Configure the task as required (assignment, events, etc.)
 - iii. Use the Task UI Update wizard to add the content ID, document type data fields, and content user interface components to the task interface, as described in the *webMethods BPM Task Development Help*.
6. Build and upload the process project.
7. Create a document type instance in the specified server/repository location.

The content listener monitors the server/ repository location; when an instance of the specified document type is created or modified (depending on the listener configuration), the Process Engine automatically converts the document type instance to an IS document.

The process start message event handles the generated IS document just as it would any other IS document, and the document type data is added to the process pipeline.

If there is a user task in the process, when it is reached, a new task is queued and appears in the appropriate My webMethods inboxes, with data fields and values displayed in the Data view.

In My webMethods, the assigned task user can:

- Work with document type data in the task.
- Download the document type from the task for offline access.
- Upload the modified document type back into the task.

Security Considerations

Authentication and authorization activity is spread over several components in the webMethods Content Service Platform (CSP) environment.

Authentication

Authentication is configured on multiple layers, but generally can use the same external user store (Active Directory, LDAP, or others). Behavior for the various components is as follows:

- Enterprise Content Management (ECM) server. Typically supports an external user store.
- webMethods Content Service Platform (CSP) server. Supports an external user store and can impersonate/propagate end user identity to any ECM that supports this feature. Connects to Integration Server using a preconfigured system account.
- My webMethods Server/Task Engine. Supports an external user store and can impersonate/propagate end user identity to the CSP server. Connects to the CSP server using a preconfigured system account.
- Integration Server/Process Engine. These systems can be configured to use the Central Users feature to leverage My webMethods Server connectivity to an external user store. Connects to the CSP server using a preconfigured system account.

Authorization

Authentication is maintained by the webMethods Content Service Platform server or the Enterprise Content Management server. Behavior for the various components is as follows:

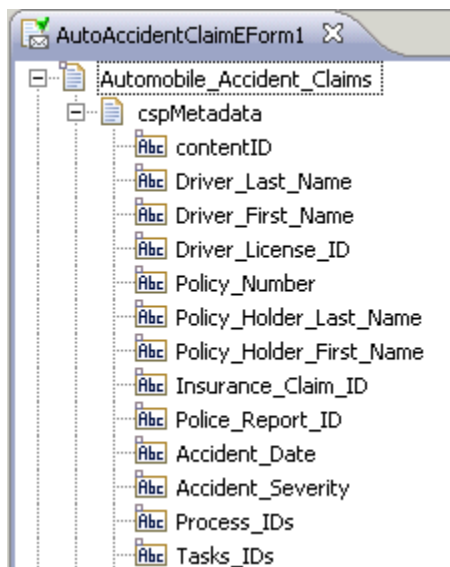
- Enterprise Content Management server. Typically has its own authorization model.
- webMethods Content Service Platform server. Can be configured to synchronize with/replicate the ECM authorization model.

- My webMethods Server/Task Engine. Delegates all content authorization to the CSP server.
- Integration Server/Process Engine. Delegates all content authorization to the CSP server.

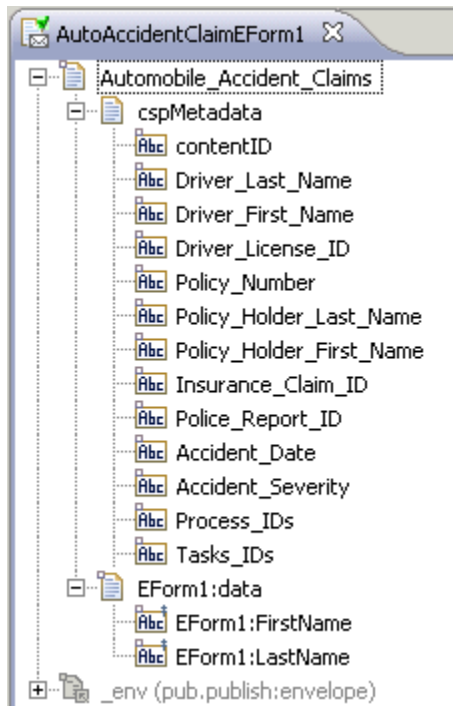
Use of Metadata

webMethods business processes and tasks cannot work directly with Content Service Platform (CSP) document types, so these document types must be imported in Software AG Designer where the information is converted into an IS document type. IS document types can be created from the following CSP document types:

- **Metadata-only Documents.** This structure is generated for Content Types that do not have associated e-form templates. The index fields are created under the cspMetadata document:



- **Metadata and E-form data Documents.** This structure is generated when an associated e-form template is found, and includes a cspMetadata document as well as a document with all data fields from the associated e-form template:



Make note of the following information:

- In both cspMetadata document examples, a contentID field is present. This field contains a unique value that identifies the document type within the Content Service Platform repository.
- The name of the e-form node in the IS document (EForm1:data in the example) comes from the name of the e-form file *within* the template stored in CSP, *not* from the name of the template itself. For example, in this case, the template can carry any name, such as "e-form sample template", but the file that was selected to be the content of that template is named "Eform.xsn" (assuming it is an InfoPath e-form).

About CSP Startup and Repository Connectivity

When you start webMethods Content Service Platform, the application checks for connectivity to the defined external repositories. If these external repositories are not found, CSP will automatically shut down. This condition can occur when there are network issues between CSP and the external repository, or if the external repository is not running. To successfully start CSP, resolve any connectivity issues and ensure that the external repository is running.

About Oracle Database Errors

When webMethods Content Service Platform is connected to an Oracle database, error messages appear in the console when you start the CSP with `csp.[bat | sh]`.

These error messages are in the form 'ERROR: [sag-cjdbc42-0011][Oracle JDBC Driver] [Oracle]ORA-01408: such column list already indexed', and are the result of a low-level API that is outside the scope of CSP.

These messages are not actually errors, but are informative messages. They do not affect functionality and can be safely ignored.

About SharePoint Text Handling

If you attach SharePoint content to a task, and then attach the same content to another task, a search for the content by taskID leads to no search results.

When SharePoint content is attached to only one task, the content is available as expected. If you attach the same content to a second task, the content is not visible in either task, and a search by the taskID of either task returns no results.

SharePoint uses the type "Multiline of Text," and if Enhanced Text is used, this returns values wrapped in <div> elements. Content is associated with a task by using an index-named taskID. For the search to work, taskIDs must contain numbers only. When these numbers are wrapped in <div> tags, the search fails.

The workaround for this is to configure taskIDs and processIDs to be "Plain Text" in SharePoint. You can make this change from the Additional Column Settings in SharePoint (this is relevant only for the type Multiline of Text).

About the Content Service Platform WebReader

You can install the Content Service Platform (CSP) WebReader, a lightweight alternative to the CSP Windows Client. The WebReader enables you to view and manipulate your CSP document types in any web browser. It provides features such as browsing, annotating, zooming, viewing thumbnails, and downloading the original files of content accessible from the CSP server. The WebReader can be installed on a web server such as My webMethods Server or Apache Tomcat.

When installed on My webMethods Server, the WebReader is integrated with the **Content** tab in CSP-enabled business tasks running on webMethods Task Engine. A **View All** button is provided to view all of the attachments to a content document type with the WebReader.

Note: If the CSP WebReader is not installed, you will receive a status 404 error message from the browser indicating that the requested resource is not available.

When you install the CSP WebReader into your Software AG installation directory with the Software AG Installer, the WebReader components are automatically installed in your My webMethods Server installation, and will be available the next time you start My webMethods Server.

If you want to deploy the WebReader to other web servers besides My webMethods Server, see the file `WebReaderDeployment.pdf`, which is installed in the directory `Software AG_directory\CSP\webclient` along with the other WebReader components. To integrate WebReader into a portlet application, see "[Configuring Content Service Platform Environment Entries for Portlet Applications](#)" on page 46.

For more information about working with the WebReader, see the CSP HTML help, which can be found in any of these locations:

- In the product installation directory: `Software AG_directory\CSP\doc`
- If you have installed the Documentation component with the Software AG Installer, in the folder `Software AG_directory_documentation\webMethods\Third-Party\Content_Service_Platform`.
- In the documentation area the Empower Product Support website: <https://empower.softwareag.com> (log in required).

2 Configuring and Deploying a webMethods Content Service Platform Environment

| | |
|--|----|
| ■ Overview of Content Service Platform Environment Configuration | 24 |
| ■ About Repository Connectivity | 24 |
| ■ About the Central Configuration Interface | 25 |
| ■ Content Service Platform Environment Considerations | 26 |
| ■ Configuring a Content Service Platform Environment | 26 |
| ■ Accessing the Content Service Platform Home Page | 29 |
| ■ About the Content Service Platform Home Page | 29 |
| ■ Deploying a webMethods Content Service Platform Environment | 30 |
| ■ Changing Host Port Numbers | 32 |

Overview of Content Service Platform Environment Configuration

The document types you work with in the webMethods Content Service Platform (CSP) are used by the webMethods product suite in the following ways:

- During design time, you use Software AG Designer to locate and select a specific document type for importation as an IS document type or as business data for a task.
- During run time, the Process Engine uses a listener to continuously monitor a designated repository location for the creation, modification, or deletion of a specific document type, using this event to start a new business process.
- Also during run time, if you implement the ability to work with a document type in a task instance interface running in My webMethods, the Task Engine enables you to upload from and download to the CSP repository. Any data associated with the document type can be used as business data within the task.

To enable these interactions, you must:

- Configure and deploy a webMethods Content Service Platform environment, which specifies a server/repository location and the credentials to connect to it. This environment is used by the various components of the webMethods product suite to access your repository servers.
- Configure a content filter definition, which defines the filter criteria you want to apply to be able to return a list of specific document types from a given repository location.
- Define a content listener for a specific document type/repository location.

When combined, these elements (the environment definition, the content filter definition, and the document type/repository location) enable you to specify a distinct set of document types that can be retrieved and managed as needed during design-time and run-time activities.

About Repository Connectivity

Your internal data network and permissions configuration must allow Software AG Designer and the webMethods run-time components (Integration Server, Process Engine, and Task Engine) to be able to access the repository folders and read the document types in them.

If the repository server is not running, or network connectivity is interrupted:

- You will not be able to import document types stored in the repository into Software AG Designer.

- No new process model instances will be started in the Integration Server/Process Engine run-time environment.
- You will not be able to upload a document type to a task instance running in My webMethods.
- You may not be able to download a document type from a task instance running in My webMethods.

About the Central Configuration Interface

This chapter provides specific information about configuring a webMethods Content Service Platform environment with the Central Configuration user interface. For additional general information about working with the Central Configuration interface, see the PDF publication *Configuring BAM*.

Determining Whether the Central Configuration Interface Is Installed

To define a Content Service Platform environment in My webMethods, you must have the Central Configuration user interface component installed with your My webMethods Server installation.

Note: Your My webMethods Server user account must be assigned the appropriate access and functional privileges to view this workspace; it is possible that the interface is installed but you do not have the correct privileges to view it. Contact your system administrator for further information.

To determine if the Central Configuration Interface is installed

1. In My webMethods: **Navigate > Applications > Administration > System-Wide > Environments > Define Environments**
2. If this application path is not available, you must use the Software AG Installer to install the Central Configuration user interface before you can create a Content Service Platform environment.

Installing the Central Configuration Interface

You need the Software AG Installer and an internet connection to install the Central Configuration interface.

To install the Central Configuration Interface

1. Shut down all webMethods products.
2. Start the Software AG Installer.

3. Enter your credentials and specify the installation directory of your current installation of My webMethods Server.
4. On the component selection panel, select **My webMethods User Interfaces >Central Configuration**.
5. Complete the installation.
6. Start My webMethods Server and verify that you can navigate to the Central Configuration page as described in the previous procedure.

Content Service Platform Environment Considerations

A webMethodsContent Service Platform environment describes the system connection information for a repository that hosts your document type folders. It also defines the Integration Server host(s) where the environment is to be deployed.

If you have multiple repositories, you will need to create and deploy a Content Service Platform environment for each one.

Obviously, your deployment architecture can become quite extensive if you have a large number of document types and business processes. The situation can become even more complex if you have multiple repositories or are working in a clustered or multiple Integration Server situation. You are advised to develop detailed deployment plans in advance of any major implementations. For further information about deployment with clustered or multiple Integration Servers, see "[Deployment Considerations](#)" on page 30.

Note: To be able to fully access your document types, you must also create one or more content filter definitions, as described in "[Creating a Content Filter](#)" on page 35.

Configuring a Content Service Platform Environment

You can create and deploy multiple Content Service Platform (CSP) environments. If your organization uses multiple CSP repository locations, you must create and deploy a Content Service Platform environment for each one. For example, if you have different repositories for different business units, you can create a separate environment for each CSP repository.

If you create multiple Content Service Platform environments, you can deploy more than one environment to a specific Integration Server host where a Process Engine is running, which will enable you to work with different CSP repositories from that host. However, a single process can be started only by events from one CSP repository. That is, there is a one to one relationship between a business process model and a CSP repository folder.

Configuring a Content Service Platform environment consists of the following general actions:

- Define a repository server by specifying the physical connection information for the repository (also referred to as a *logical server* in Central Configuration).
- Define one or more Integration Server hosts where a Process Engine is running. This defines the target location(s) for deploying the environment.
- Map the repository server to the Integration Server host(s).
- Map the endpoint information for the host(s) by specifying a transport type and port for the host system(s).
- Verify the configuration.

To create a Content Service Platform environment

1. Log on to My webMethods using an account that has permissions for configuring a Content Service Platform environment (the Administrator account has these permissions by default).
2. In My webMethods: **Navigate > Applications > Administration > System-Wide > Environments > Define Environments**
3. In the Environments window, click **Add Environment**.
4. In the Environment Information window:
 - a. Type an environment name.
 - b. Type a description for the environment.
5. Click **Save**.
6. In the Environments window, click the link created by the new environment in the **Environment Name** column.
7. On the **Design Servers** tab of the Environment Configuration window, click **Add**.
8. On the Add Logical Servers dialog box, click **Content Service Platform Server vx.x.x.x** in the **Select Logical Servers** list. This specifies that you are creating a Content Service Platform environment.
9. Click **OK**.
10. Click the **Configure Servers** tab.
11. If it is not already expanded, expand the Content Service Platform Server entry in the **Configuration** column.
12. Click the Content Service Platform Configuration link.
13. Define the server/repository connection information by typing the following:
 - **Friendly Name**—this is configured by default as a concatenation of the repository server host name and server port. However, you can enter any text value (spaces are not allowed). This value is used to display a user-friendly name in

Software AG Designer for the repository server that is hosting the content type folders. This field has a character limit of 256.

Note: If you deploy more than one environment to the same Integration Server host, the Friendly Name must be unique for each environment.

- **Server Host**—the host name or IP address of the repository server. Enter only the host name or IP address; URL descriptors are not used.
- **Server Port**—the port number for the repository server. If you are using a standard webMethods Content Service Platform repository, accept the default value of 9010. You can specify a different port number as needed.
- **Server User**—the name of an existing user account with permission to access the contents of the content type folders. This field has a character limit of 256.
- **Server Password**—the password for the specified user account. This field has a character limit of 256.

14. Click **Save**.
15. Click the **Define Hosts** tab. On this tab, you define one or more Integration Server hosts. When you deploy this environment, it will be deployed to the host(s) defined here.
16. Click **Add Host**.
17. On the Add/Edit Host dialog box:
 - Type a display name for the Integration Server host.
 - Type the host name or IP address of the Integration Server host.
18. Click **OK**. If you want to add additional Integration Server hosts, repeat steps 16 and 17.
19. Click the **Map Servers** tab.
20. Click the Edit icon in the **ACTIONS** column to map an individual logical server, or click **Map All**. This maps your repository server to the host(s).
21. Click the **Map Endpoints** tab. The default values are already configured on this tab, and the tab is automatically marked as configured. Do not change the values on this tab.

Note: Accept the default port value of 15006 for hosts where you have only one Integration Server running on the specified host (unless you have a pre-existing port conflict). If you have two or more Integration Servers running on the same host hardware, you must take care to specify a separate port number for each instance on that host. To change the port definition, see ["Changing Host Port Numbers" on page 32](#).

22. Click the **Map DB Pools** tab. There are no values to configure on this tab, and the tab is automatically marked as configured.

23. Click the **Validate** tab. If the configuration report indicates that you do not have a valid configuration, make changes to the environment configuration as needed and return to the **Validate** tab to re-validate the configuration.

Note: It is possible to obtain a valid configuration report but still have a configuration that will not work correctly. For example, a typographical error in a path name or user account name will still be accepted as valid, although it will not work correctly when deployed.


24. Click **Finish** to return to the Define Environments page.

Accessing the Content Service Platform Home Page

As a component that runs within the Integration Server environment, the WmContentServicePlatform package provides services for direct access into the Content Service Platform server, and can be accessed through the webMethods Integration Server Administrator, a Web-based interface available through your local browser.

You can start the Integration Server Administrator from the installation menu on your system, or by entering a URL in your browser's address field that represents the name of the server the Integration Server is running on, as well as its port number (the default port number is 5555). For example: `http://localhost:5555`.

To access the Content Service Platform home page

1. Log on to the Integration Server Administrator and click **Packages > Management**.
2. Locate the WmContentServicePlatform package and click the Home Page icon .

Note: You can click the **Log Off** link at the top of the page to log out of the Content Service Platform home page. You will still be logged into the Integration Server Administrator. The **Log Off** link terminates only the session for the Content Service Platform home page.

About the Content Service Platform Home Page

The Content Service Platform home page provides you with access to view, add, delete, update, and see status for connections to CSP servers. These connections are one of the following two options:

- WmContentServicePlatform usage: these connections are available for use when running the services in the WmContentServicePlatform package. A CSP connection is identified by its *friendlyName*. That *friendlyName* is used as input to services in the package, such as `createContentID` and `createFilter`.
- Process usage: these connections are used by My webMethods Server to define and deploy CSP connections to Integration Server, as described in "[Deploying a](#)

[webMethods Content Service Platform Environment" on page 30](#). They can also be created in the Define Environment area in My webMethods Server.

Deploying a webMethods Content Service Platform Environment

You must deploy your Content Service Platform (CSP) environment to the Integration Server host(s) where you will be running your business processes that consume content types. This enables Software AG Designer and the various run-time components to identify where the CSP repositories are located. You can deploy more than one environment to a specific Integration Server host where a Process Engine is running, which will enable you to work with different CSP repositories from that host.

Deployment Considerations

Some forethought and planning must be applied if you working with Content Service Platform environments in a multiple or clustered installation.

In these situations, exercise caution when deploying environments from the Central Configuration interface. If you deploy a Content Service Platform environment to more than one Integration Server/Process Engine host, you create a situation where it is possible to start multiple (duplicate) business processes when a monitored document type is updated or modified. Take care to avoid this situation.

Consider the following conditions:

- *Where are my process models running?* When you upload a content-enabled process model to an Integration Server/Process Engine host, the webMethods Content Service Platform uses the content type listener specified within the process to monitor a folder location for instances of a specified document type. If you upload the same process model to two or more Integration Server/Process Engine hosts that have the same Content Service Platform environment, a new business process instance will start on each host when the document type listener is triggered.
- *What document types am I monitoring?* When you define a content-enabled process model, you specify a content listener that monitors a specific folder location in the repository. Consider this example:
 - You create Process1 to monitor a folder in your repository for arrival of DocumentTypeA, and upload Process1 to Integration Server/Process Engine host Alpha.
 - You create Process2 to monitor the same repository folder for arrival of DocumentTypeB (that is, you have specified a different listener), and upload Process2 to Integration Server/Process Engine host Beta.
 - Both hosts monitor the same repository folder, but because the Process Engine on each host is using a different listener to monitor for a different document type, new process instances will run separately on each host, as expected.

However, if you were to upload Process1 (monitoring for DocumentTypeA) to both hosts, a process instance will start on each host when the DocumentTypeA listener is triggered.


Prerequisites

Prior to deploying a Content Service Platform environment, the following conditions must be met:

- The Content Service Platform environment must pass the Central Configuration validation step and a check mark must appear in the Ready to Deploy column in the Environments window.
- The targeted Integration Server must be running.

Deploying the Environment


To deploy a Content Service Platform environment

1. Log on to My webMethods using an account that has permissions for deploying a Content Service Platform environment (the Administrator account has these permissions by default).
2. In My webMethods: **Navigate > Applications > Administration > System-Wide > Environments > Define Environments**
3. In the Environments window, click the Deploy icon  for the environment.
4. On the Deployment Results tab, click **Deploy All**. This action deploys the Content Service Platform environment to all of the Integration Server hosts defined in the environment.
5. Results of the deployment appear in the text box on the Deployment Results tab. If the deployment is unsuccessful:
 - a. Note the messages in the results text.
 - b. Return to the Define Environments page.
 - c. Make corrections to the environment as needed.
 - d. Repeat steps 3 and 4.

Updating a Content Service Platform Environment

After you deploy a Content Service Platform environment, it is possible that some elements of the environment may change, such as port numbers or folder locations. In this case, you can enter the changes to the environment and then deploy the updates to the target hosts.

To update a Content Service Platform environment

1. Log on to My webMethods using an account that has permissions for deploying a Content Service Platform environment (the Administrator account has these permissions by default).
2. In My webMethods: **Navigate > Applications > Administration > System-Wide > Environments > Define Environments**
3. Make the necessary changes to your environment, as described in "[Configuring a Content Service Platform Environment](#)" on page 26.
4. Click **Finish** when you have entered all of your changes.
5. In the Environments window, click the Deploy icon  for the environment.

Important: The following step specifies that you click **Deploy Updates**. *Do not click **Deploy All**.* If you click **Deploy All**, you will create a new environment on the target hosts. Also note that because the Friendly Name is the unique key for an environment, if you change it and then deploy the environment to the same host, a new environment will be created on the host regardless of the deployment method (**Deploy All** or **Deploy Updates**).

6. On the Deployment Results tab, click **Deploy Updates**. This action deploys updates in the Content Service Platform environment to all of the Integration Server hosts defined in the environment.
7. Results of the deployment appear in the text box on the Deployment Results tab. If the deployment is unsuccessful:
 - a. Note the messages in the results text.
 - b. Return to the Define Environments page.
 - c. Make corrections to the environment as needed.
 - d. Repeat steps 3 and 4.

Changing Host Port Numbers

When you define an Integration Server/ Process Engine host as part of a Content Service Platform environment, a default port of 15006 is specified. If you have a conflict with this port number, you can specify a different port number.

Note: Do not change the default http transport type.

To specify a different host port number

1. Log on to the Integration Server Administrator.
2. In Administrator: **Security > Ports**.

3. If it is unused, delete or disable the entry for port 15006 as required by your environment. If you are using 15006 for other purposes, it can be left enabled.
4. Click the Add Port link and select a port type of webMethods/HTTP.
5. Click **Submit**.
6. In Regular HTTP Listener Configuration:
 - a. Click **Enable**.
 - b. Type the new port number.
 - c. Set the Package Name to WmPRT.
 - d. Click **Save Changes**.
7. In the Port List, click the Edit link in the **Access Mode** column for the newly created port.
8. Determine if the following services are present in the Allow List:
 - wm.prt.eforms:update
 - wm.server:connect
 - wm.server:disconnect
 - wm.server:ping
9. If any of these services are not present, click the **Add Folders and Services to Allow List** link.
10. Select the appropriate ACL to display the missing services.
11. Select the missing service and click **Append Selected**.
12. Click **Save Additions**.

Note: Be sure to verify that the port number in your e-form environment host definition matches this new port.

3

Creating a Content Filter

| | |
|---|----|
| ■ About Content Filters | 36 |
| ■ Determining Whether the Content Service Platform Configuration Interface Is Installed | 36 |
| ■ Creating a Content Filter | 37 |
| ■ Viewing and Modifying Content Filters | 38 |

About Content Filters

In an enterprise of any size, a content repository can quickly collect a very large number of files — so many that it can be difficult and time-consuming to browse through the repository to find exactly what you are looking for. webMethods Content Service Platform (CSP) provides you with the ability to define one or more content filters; when you create the filter, you specify exactly what document types are to be displayed to the filter user. This enables you to provide a customized view into the repository that shows only the document types of interest.

The user interacts with content filters in Software AG Designer. When you choose to create an IS document type from a content type in your CSP repository, the initial selection list contains all of the available content filters for that repository. After selecting the appropriate content filter, you can then locate and select the specific document type you want to work with. To assist the user with filter selection, be sure to name your content filters clearly, and to include appropriate description information.

Content filters are created in My webMethods; you must have the Content Service Platform Configuration interface installed to carry out this procedure. If this interface is not installed, you can install it with the Software AG Installer.

Determining Whether the Content Service Platform Configuration Interface Is Installed

Note: Your My webMethods Server user account must be assigned the appropriate access and functional privileges to view this workspace; it is possible that the interface is installed but you do not have the correct privileges to view it. Contact your system administrator for further information.

To determine if the Content Service Platform Configuration Interface is installed

1. In My webMethods: **Navigate > Applications > Administration > System-Wide > Content Configurator.**
2. If this application path is not available, you must install the Content Service Platform user interface before you can create a content listener.

Installing the Content Service Platform Configuration Interface

You need the Software AG Installer and an internet connection to install the Content Service Platform Configuration interface.

To install the Content Service Platform Configuration interface

1. Shut down all webMethods products.

2. Start the Software AG Installer.
3. Enter your credentials and specify the installation directory of your current installation of My webMethods Server.
4. On the component selection panel, select **My webMethods User Interfaces >Content Service Platform Configuration**.
5. Complete the installation.
6. Start My webMethods Server and verify that you can navigate to the Content Configurator page as described in the previous procedure.

Creating a Content Filter

Content filters are created in My webMethods.

To create a content filter

1. In My webMethods: **Navigate > Applications > Administration > System-Wide > Content Configurator**.
2. Available servers can be selected in the Content Service Platform Server list. This is a list of the Content Service Platform server environments you defined as described in ["Configuring a Content Service Platform Environment" on page 26](#).



The content filters configured for the selected server appear in the Content Service Platform Filter list, with the most recently created filters at the top of the list. Click the **Filter Name** column title to sort the list. If no filters are configured for the selected server, the list will be empty. To create a new filter, click **Add Filter**.
3. On the next page, select a content server in the **Server Selector** list. This is a list of the content service platform server environments you defined as described in ["Configuring a Content Service Platform Environment" on page 26](#); the filter you are creating applies to content on the selected server.
4. In the Content Service Platform Filter Information panel, type a name and a description of the filter you are creating. For clarity, choose a name that indicates the filter's intended use or the content it exposes. For example, in the description, include the document types you intend the filter to make available.
5. Click **Next**.
6. In the Content Hierarchy panel, expand the Content Hierarchy tree view if it is not already expanded. You can limit the number of entries displayed in the tree by typing text in the **Filter List** field and clicking **Go**; only document types that contain matching text will be displayed in the tree view.
7. In the tree view, select the check boxes for the content nodes you want the filter to make available. Non-selected nodes will not be visible to the filter user.
8. Click **Next**.

9. On the Content Filter Configuration Summary page, review and verify the configuration you have entered for this filter. If you want to make any changes, click **Previous** to return to the appropriate page.
10. Click **Finish** to add the filter to the list of available content filters. The new filter is added to the top of the list on the Content Service Platform Filters panel.

Viewing and Modifying Content Filters

Each content filter is associated with a particular webMethods Content Service Platform, and a specific filtered view of the platform's content hierarchy tree.

To view and modify the available content filters

1. In My webMethods: **Navigate > Applications > Administration > System-Wide > Content Configurator**.
2. Available servers can be selected in the Content Service Platform Server drop-down list. The content filters configured for the selected server appear in the Content Service Platform Filters list, with the most recently created filters at the top of the list. Click the **Filter Name** column title to sort the list. If no filters are configured for the selected server, the list will be empty.
3. In the Content Service Platform Filters panel, do any of the following:
 - **To view the filter configuration** — click  to the right of the filter name.
 - **To modify the filter configuration** — click the filter name and modify the configuration using the values described in "[Creating a Content Filter](#)" on page [37](#). Click **Save** after making your changes.
 - **To delete the filter** — click  to the right of the filter name.

4 Creating a Content Listener

| | |
|--|----|
| ■ About Content Listeners | 40 |
| ■ Creating a Content Listener | 40 |
| ■ Creating a Listener for Deactivated Document Types in a Repository | 42 |
| ■ Viewing and Modifying Content Listeners | 42 |
| ■ About the Operation of CSP Content Listeners | 43 |

About Content Listeners

Although a content repository can simply collect and hold various files relevant to your line of business, the value of that content increases tremendously when you can integrate it with your automated business processes. webMethods Content Service Platform (CSP) provides you with the ability to define one or more content listeners; when you create the listener, you specify exactly what repository location and document types you want the listener to monitor.

Within a repository, documents are stored in a hierarchical archive structure, referred to as a content hierarchy tree. Each location, or hierarchical level of the tree, can also be referred to as a *node*. Document types are usually the lowest level of the tree.

You can configure a listener to monitor the specified location and document types for any or all of these actions: creation, deletion, updating, and deactivation. By linking the listener to a webMethods business process model, you can trigger a business process when the listener detects that any of the monitored actions has occurred. For example, the creation of a new Purchase Order document type could be used to trigger an order handling process.

Content listeners are created in My webMethods; you must have the Content Service Platform Configuration interface installed to carry out this procedure. If this interface is not installed, you can install it with the Software AG Installer. For more information, see "[Determining Whether the Content Service Platform Configuration Interface Is Installed](#)" on page 36.

Creating a Content Listener

Content listeners are created in My webMethods.

To create a content listener

1. In My webMethods: **Navigate > Applications > Administration > System-Wide > Content Configurator**.
2. Available servers can be selected in the Content Service Platform Server list. The content listeners configured for the selected server appear in the Content Service Platform Listener list, with the most recently created listeners at the top of the list. Click the **Listener Name** column title to sort the list. If no listeners are configured for the selected server, the list will be empty. To create a new listener, click **Add Listener**.
3. On the next page, select a content server from the **Server Selector** list. This is a list of the content service platform server environments you defined as described in "[Configuring a Content Service Platform Environment](#)" on page 26; the listener you are creating applies to content on the selected server.
4. In the Content Service Platform Listener Information panel, type a name and a description of the listener you are creating. For clarity, choose a name that indicates

the listener's intended use or the content it targets. For example, in the description, include the document types you intend the listener to monitor, and the listener actions you plan to implement for them.

5. Click **Next**.
6. In the Content Hierarchy panel, expand the Content Hierarchy tree view if it is not already expanded. You can limit the number of entries displayed in the tree by typing text in the **Filter List** field and clicking **Go**; only document types that contain matching text will be displayed in the tree view.
7. In the tree view, select the check boxes for the content nodes you want the listener to monitor.
8. Specify the index information that will be used to identify the document types you want the listener to monitor:

Note: If you are creating a listener to detect a deactivated document type in a Sharepoint or Alfresco repository, special conditions apply. For more information, see ["Creating a Listener for Deactivated Document Types in a Repository" on page 42](#).

- **Index**—select the index you want to define for the document type you want to monitor.
 - **Operator**—select the operator that you want to apply to the index value (for example, +, >, <).
- Important:** If you are specifying a date value, valid operator values are limited to "<", ">", "<=", and ">=". Selection of any other operator results in an error.
- **Value**—type the value that you want the selected operator to evaluate against. A calendar control is provided for date values.
 - **Join**—select a join type (for example, AND, OR), if you want to add additional criteria for defining the monitored document types.
 - **Add/Remove**— click **Add** to add another row of index definition information; click **Remove** to delete an existing row of index information.
9. In the Monitored Events panel, select any combination of the check boxes to identify the events you want to monitor for the defined document types: **Create**, **Delete**, **Update**, and **Deactivate**.

10. Click **Next**.
11. Select an Integration Server from the drop-down list. The selected server is the target environment for the listener; when a monitored event occurs on a defined content object, the event notification is sent to the specified Integration Server.

You can add an Integration Server if it is not available in the list by clicking **Add New Integration Server**. On the Add New Integration Server dialog box, specify:

- **Server Name**—the name by which the server will be identified in the interface.
 - **Host**—the host name of the server.
 - **Port**—the Integration Server port number, usually 5555.
 - **Login**—the name of an Integration Server user account with sufficient privileges to support CSP functionality.
 - **Password**—the password for the specified user account.
12. Click **Next**.
 13. On the Content Listener Configuration Summary page, review and verify the configuration you have entered for this listener. If you want to make any changes, click **Previous** to return to the appropriate page.
 14. Click **Finish** to add the listener to the list of available content listeners. The new listener is added to the top of the list on the Content Service Platform Listeners panel.

Creating a Listener for Deactivated Document Types in a Repository

Special considerations apply to the creation of a listener for detecting the deactivation of a document type in a Sharepoint or Alfresco repository. The listener must contain only one query definition, which must conform to these requirements:

- The **Index** value must be set to “Document title”.
- The **Operator** value must be set to “=”.
- The **Value** must be set to “*”.

That is: `Document title = *`



Do not specify any other values or query definitions. This configuration is required because when a document type is deactivated, document type data is no longer available.

Viewing and Modifying Content Listeners

Each content listener is associated with a particular webMethods Content Service Platform, a specific location in the platform’s content hierarchy tree, and the content contained there.

To view and modify the available content listeners

1. In My webMethods: **Navigate > Applications > Administration > System-Wide > Content Configurator**.

2. Available servers can be selected in the Content Service Platform Server drop-down list. The content listeners configured for the selected server appear in the Content Service Platform Listener list, with the most recently created listeners at the top of the list. Click the **Listener Name** column title to sort the list. If no listeners are configured for the selected server, the list will be empty.
3. In the Content Service Platform Listeners panel, do any of the following:
 - **To view the listener configuration** — click  to the right of the listener name.
 - **To modify the listener configuration** — click the listener name and modify the configuration using the values described in "[Creating a Content Listener](#)" on [page 40](#). Click **Save** after making your changes.
 - **To delete the listener** — click  to the right of the listener name.

About the Operation of CSP Content Listeners

When you configure a receive step in a business process model to use an IS document type created from a Content Service Platform (CSP) document type, you specify the following information:

- The name of the CSP document type being used with the receive step.
- The location of the folder where the monitored CSP document type exists.

For an example, see the illustration in "[Using Document Types in a Business Process](#)" on [page 15](#).

After you build and upload a CSP-enabled business process, the Content Service Platform uses a content listener to continuously monitor the designated repository location for the creation, modification, deactivation, or deletion of the specified CSP document type. For information about creating content listeners, see "[Creating a Content Listener](#)" on [page 39](#).

The content listener is enabled when the process model is enabled. Detection of a monitored event causes Process Engine to start a new instance of the CSP-enabled business process.

After the business process starts, a flow service begins the process of converting the specified CSP document type into the correct IS document type, according to the relationship defined when the CSP document type was initially imported. The resulting IS document type is then passed to the receive step.

If a content listener is not running, no new processes instances will be started even if a CSP document type is created or modified in the monitored folder.

Be aware of the following key points:

- Use caution if you configure two or more process models to use the same content listener. For more information, see "[Deployment Considerations](#)" on [page 30](#).

- If the monitored CSP repository server stops running, or if network connectivity to the repository server is lost, the content listener will be unable to detect document type events and business processes will not be started.

5 Configuring Portlet Applications for Use with the Content Service Platform

- [Configuring Content Service Platform Environment Entries for Portlet Applications](#) 46

Configuring Content Service Platform Environment Entries for Portlet Applications

If you have configured one or more task or other portlet applications for Content Service Platform (CSP) support, you must ensure that global CSP-related environment entries are correctly set so each task application can connect to a CSP server at run time.

When you define these global CSP environment entries, they are applied to all portlet applications published to My webMethods Server, including all task applications.

Setting CSP global defaults provides a convenience for connecting all portlet applications to a particular CSP repository. While it is possible to define specific CSP environment entries for an individual task application, you are advised to apply these environment entries globally to ensure consistency across all portlet applications.

In a more complex installation, you might need to connect to two or more CSP repositories, or to connect a single application to two or more repositories. In these cases, you can:

- Define CSP environment entries on a per application basis; environment entries applied at the application level override the global entries.
- Define additional/new CSP environment entries on a per application basis, enabling the application to connect to two or more repositories.

Note: After you modify any of the environment variable values, you must restart My webMethods Server (or the affected portlet applications) for the changes to take effect.

To configure global environment variables for portlet applications

1. Log on to My webMethods Server as a system administrator (sysadmin).
2. In My webMethods: **Administration Dashboard > Configuration > CAF Application Runtime Configuration**.
3. Click **Configure Global Defaults**.
4. On the CONFIGURE GLOBAL DEFAULTS panel, click **Environment Entries**.
5. For each Content Service Platform-specific environment entry in the **Name** column, type a value in the **Value** column as follows:

| Environment Entry | Value |
|-------------------------|--|
| csp/default/name | Identifies the name of the Content Service Platform server the portlet is to connect to. Specify the server name. The default value is <code>csp_master</code> . |

| Environment Entry | Value |
|-----------------------------------|--|
| csp/default/host | Identifies the host to use to connect to the Content Service Platform server. Specify the host name as an IP address or DNS name, for example, <code>10.140.00.12</code> or <code>localhost</code> . The default value is <code>localhost</code> and it is resolved to the host name. |
| csp/default/port | Identifies the port to use to connect to the Content Service Platform server. Specify the port number. The default value is <code>9010</code> . |
| csp/default/streaming#port | Identifies the streaming port to use to connect to the Content Service Platform server. Specify the streaming port number. The default value is <code>9011</code> . |
| csp/default/username | Identifies the user name to supply to the Content Service Platform server for authentication. Specify a valid administrative-level user name for the Content Service Platform server. This user account must have the CSP privilege for impersonating other users. The default value is <code>csp</code> . |
| csp/default/password | Provides the password for the user name. This password is supplied, along with the user name, to the Content Service Platform server for authentication of the configured user account. The value for this parameter is case-sensitive. The default value is <code>operating</code> . |
| csp/default/webreader-url | Provides the URL of the CSP WebReader. This URL is used to open attachments from user tasks in a web browser. The value for this parameter is case-sensitive. The default value is <code>http://localhost:8585/SAGWebReader/</code> . |
| is/host | Identifies the host name to use to connect to the Integration Server. The default value is <code>localhost</code> . |
| is/port | Identifies the port to use to connect to the Integration Server. Specify a port number. The default value is <code>5555</code> . |
| is/login | Identifies the user name to supply to the Integration Server for authentication. Specify a valid |

| <u>Environment Entry</u> | <u>Value</u> |
|--------------------------|---|
| | administrative-level user name for the Integration Server. The default value is <code>Administrator</code> . |
| is/password | Provides the password for the user name. This password is supplied, along with the user name, to the Integration Server for authentication of the configured user account. The value for this parameter is case-sensitive. The default value is <code>manage</code> . |

6. Click **Apply**. Do one of the following to apply the changes:
 - Restart My webMethods Server.
 - Restart all CSP-enabled applications using the CAF Runtime Configuration page; this method does not require restart of My webMethods Server. For each application:
 1. Search for the application with the keyword search mechanism.
 2. In the list of results, click the application name to open it.
 3. Click **Restart Application**.

6 Synchronizing Document Types

- Synchronizing CSP Document Types and IS Document Types 50
- Maintaining CSP Document Types and IS Document Types 50

Synchronizing CSP Document Types and IS Document Types

Because Content Service Platform (CSP) document types are converted into IS document types, you must be aware that any modification to either the CSP document type or to the IS document type will result in mis-matched (out of sync) documents.

- After a CSP document type is imported and an IS document type is created from it, any modifications to the content or structure of the IS document type will cause it to be out of sync with the CSP document type from which it was created. When an instance of the CSP document type is received, it will no longer match the IS document type, and therefore will be unusable by the Process Engine.
- The same situation will occur if you change the format of the CSP document type, and content objects arrive in the repository in the new format. The instances will no longer match the IS document type created from the initial CSP document type.

In both cases, you must delete the IS document type and the associated IS schemas. Then, recreate the IS document type from the latest version of the CSP document type.

If there are changes to the IS document type that will affect your process model (such as data structure changes), additional procedures are required. You must:

- Delete the existing process model receive step that uses the IS document type.
- Drag the recreated IS document type to the process canvas to create a new receive step in process model and configure the step accordingly.
- Build and upload the process model.

Note: Changes to field attributes in the CSP document type do not require these activities.

For more information about working with IS document types, see the *webMethods Service Development Help*.

Maintaining CSP Document Types and IS Document Types

After you import a Content Service Platform (CSP) document type and create an IS document type with it, you must maintain that CSP document type in its original import location for as long as the resulting IS document is used by the webMethods product suite.

When the CSP document type and its resulting IS document type is no longer used by the webMethods product suite, you can safely delete or remove the CSP document type from its import location.

Similarly, do not delete or remove the IS document type created when you import a CSP document type until that IS document type is no longer used by the webMethods

product suite. For more information about working with IS document types, see the *webMethods Service Development Help*.

7 Introduction to E-forms Integration

| | |
|---------------------------------------|----|
| ■ Overview | 54 |
| ■ General Concepts | 54 |
| ■ Working with E-form Templates | 57 |
| ■ General Usage | 58 |

Overview

Electronic forms, or *e-forms*, are replacing paper processes in paper-intensive industries such as government, financial services, and education. Both small and large organizations often use dozens or even hundreds of forms to facilitate various aspects of their business processes—mortgage applications, employment applications, time sheets, invoices, order forms, and so on. Process participants use these e-forms as a primary user interface for entering and modifying process-related information.

The primary goal of webMethods Content Service Platform electronic forms integration is to enable the use of e-forms in business processes and user interfaces implemented using the webMethods BPMS infrastructure. This chapter describes general concepts and information about working with e-forms.

You are responsible for obtaining and installing the supported e-form applications for creating, modifying, and managing your electronic forms; Software AG products do not provide this functionality.

For additional information about working with e-forms, see the following documentation:

- *webMethods Service Development Help*
- *webMethods BPM Process Development Help*
- *webMethods BPM Task Development Help*
- *webMethods CAF Development Help*
- *webMethods Task Engine User's Guide*

General Concepts

The webMethods Content Service Platform (CSP) supports BPM suite integration for e-forms created with the following applications:

- Adobe LiveCycle
- Microsoft Office InfoPath

Each of these applications enables you to create the following objects:

- An e-form *template*—one or more files that describe the form structure, appearance, and behavior. A template carries no user-entered form data (although sample data may be present).
- An e-form *instance*—a published version of the template, containing specific form data in a single file.

For more information about templates and instances, see the documentation for your e-form application.

Supported e-form templates can be imported into Software AG Designer where they are converted to a standard IS document type. You can use the IS document type to describe the input for a business process, as the business data for a task that expects an e-form document, or for any other supported IS document type use. Software AG Designer must have network access to the webMethods Content Service Platform repository at design time.

webMethods Content Service Platform also provides the ability to define filtered views of a CSP repository, enabling you to provide custom repository views to your users. For more information, see ["Creating a Content Filter" on page 35](#).

E-form templates and instances are also accessed at run time by the webMethods run-time components (Process Engine and Task Engine). During run time, the template and instance location must be available to the Task Engine if you have implemented the ability for a user to download an e-form from an e-form enabled task instance running in My webMethods.

If Integration Server is connected to a webMethods Broker at the time you create an IS document type from an e-form template, the resulting IS document type will be publishable to the webMethods Broker and will have an associated webMethods Broker document type.

Requirements

To use an e-form instance as a run-time IS document type source that starts a business process, the following requirements apply:

- A webMethods Content Service Platform (CSP) environment must be configured and deployed to the Integration Server host where the process will run. For more information about configuring and deploying CSP environment, see ["Configuring and Deploying a webMethods Content Service Platform Environment" on page 23](#).
- Your e-form instances must be available as document types from a folder in the repository defined in the deployed CSP environment.
- Your e-form templates must be available as document types in the folder *Software AG_directory/CSP/system/templates*.
- You must define a content listener to monitor the repository folder containing your e-form instances. For more information, see ["Creating a Content Listener" on page 39](#).
- If you are working with CSP-enabled tasks, you must configure global variables to enable the task applications to connect to the CSP repository (see ["Configuring Portlet Applications for Use with the Content Service Platform" on page 45](#)).
- Although multiple e-form templates can be associated with a document type, only one can be active if the document type is to be used with a business process.
- Microsoft InfoPath e-forms must contain only one root node; InfoPath e-forms with multiple root nodes are not supported.

For more information about working with IS document types, see the *webMethods Service Development Help*.

Suite Integration

E-form integration interacts with the following products in the webMethods product suite:

- **My webMethods.** You use the Central Configuration interface in My webMethods to configure and deploy CSP environment definitions, and you use the Content Configurator interface to define content filters and content listeners. If you implement e-form enabled task applications, users will interact with the tasks in My webMethods; if you have implemented the ability to do so, they can download and upload e-form documents from within the task user interface.
- **Software AG Designer** You use Software AG Designer to import an e-form template and create a corresponding IS document type from it. You then use this IS document type to create e-form enabled business process models as well as e-form enabled task applications.
- **Integration Server/Process Engine.** You use the Integration Server/ Process Engine as the run-time environment for webMethods business processes. The Integration Server host is the target for your deployed CSP environment definition. When the Content Service Platform detects the creation of or a change in an e-form instance specified in a process model, the Process Engine uses that information to start a corresponding business process.
- **webMethods Task Engine.** If you have implemented e-form enabled task applications, the Task Engine manages these tasks and the e-form business data associated with the task. If you implement the ability to download an e-form from an e-form enabled task instance running in My webMethods, the Task Engine formats the e-form business data for downloading using the associated template, and, after uploading, modifies and saves the task business data with the modified data from the uploaded e-form.

About Using the Content Service Platform As an E-form Repository

E-form integration requires communication with a webMethods Content Service Platform (CSP) repository configured with an e-form *templates folder* as well as content folders containing e-form *instances*. webMethods Content Service Platform must have network access to Software AG Designer and the webMethods run-time components (Integration Server, Process Engine, and Task Engine).

- Document types with e-form templates must be stored in the folder *Software AG_directory/CSP/system/templates*.
- Document types with e-form instances can be stored in any folder (node) in the repository.

After you set up a CSP repository, you must configure one or more CSP environments in My webMethods and deploy these environments to the Integration Server host(s) where e-form enabled processes will be running (see ["Configuring and Deploying a webMethods Content Service Platform Environment" on page 23](#)).

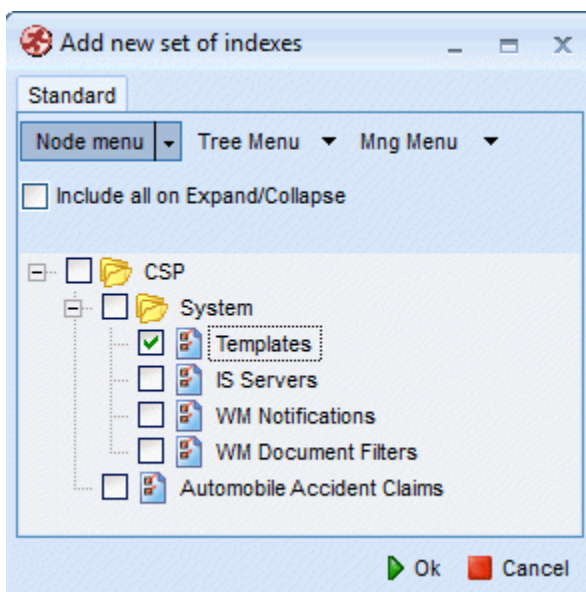
By deploying a CSP environment, you are specifying the CSP repository location of your e-form templates and e-form instances to the various components, so the templates and instances can be retrieved as needed during design time and run-time activities:

- During design-time activities with Software AG Designer, the deployed CSP environment enables you to locate and select an e-form template for importation as an IS document type or as business data for a task.
- During run-time activity, the deployed environment identifies the e-form instance folder; the Content Service Platform continuously monitors this folder for the creation of or modifications to instances of an e-form to start a new business process.
- Also during run time, if you implement the ability to download/upload an e-form from a task instance running in My webMethods, the Task Engine must have access to the e-form template location.

For more information about creating and deploying an e-form environment, see ["Configuring and Deploying a webMethods Content Service Platform Environment" on page 23](#).

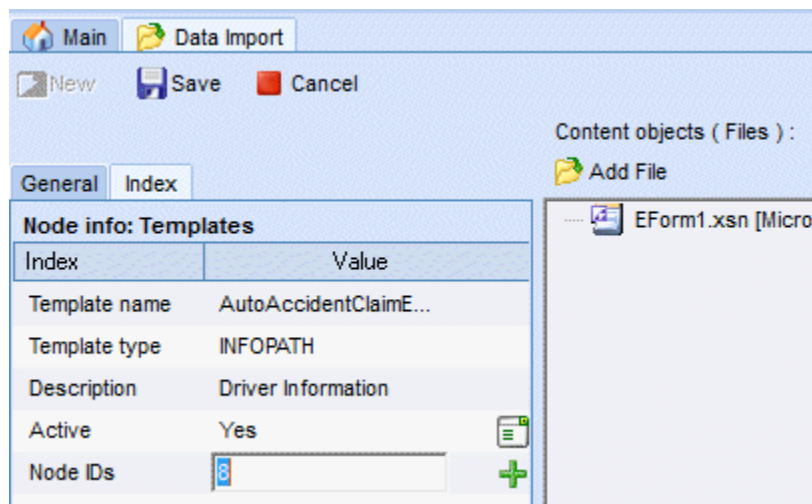
Working with E-form Templates

E-form instances can exist as document types in any folder (node) of a webMethods Content Service Platform (CSP) repository. However, e-form template document types must be stored in the *Software AG_directory/CSP/system/templates* folder, as shown in the figure below:



The following information is required, as shown in the figure below:

- Template Name and Description
- Template Type: InfoPath or LiveCycle
- Association with Content Hierarchy nodes that can use this template
- E-form template file as an attachment



The following conditions also apply:

- There can be one active e-form template per document type.
- There can be only one e-form attachment per e-form template document.
- The e-form template has to be associated with the content hierarchy node before the IS document is generated for this document type.

General Usage

The most common scenarios for integrating e-forms into the webMethods product suite are:

- To start or provide data to a business process running in the Process Engine (just as with any IS document type).
- To provide business data to a task running in the Task Engine (just as with any IS document type).
- To provide users with download/upload access to forms and form data in a running task. This enables users to download an e-form, modify it offline, and then upload it back to the task later.

Implementation of all of the above capabilities are supported with wizards and drag-and-drop functionality within Software AG Designer.

Using E-forms in a Business Process

This section provides an overview of the basic workflow for using e-forms with a business process; for specific information and procedures, see:

- *webMethods Service Development Help*
- *webMethods BPM Process Development Help*
- *webMethods BPM Task Development Help*
- *webMethods CAF Development Help*
- *webMethods Task Engine User's Guide*

To use e-forms in a business process, you follow these general procedures:

1. Create a business process. For more information, see the *webMethods BPM Process Development Help*.
2. Create an IS document type from an e-form template with the Create New Document Type wizard. For more information, see the *webMethods Service Development Help*.

Note: The generated e-form document type is a fully compliant IS document type and can be used in the same ways as any other IS document type.

3. Drag the resulting IS document type onto the business process canvas to create a receive step, or drop it onto an existing receive step.
4. Specify the receive step CSP repository e-form folder location.
5. If you want to integrate the e-form data with a task in the process, create a task step in one of the following ways (a process can contain multiple e-form enabled tasks):
 - Use the **CAF > Generate Task** menu option from the e-form document type, then:
 - i. Configure the task as required (assignment, events, etc.).
 - ii. Use the Task UI Update wizard to add the e-form fields and upload/download buttons to the task interface.
 - Select an existing task, or implement an existing step in the process as a task, and then:
 - i. Add the e-form IS document type as business data if not yet included.
 - ii. Configure the task as required (assignment, events, etc.).
 - iii. Use the Task UI Update wizard to add the e-form fields and upload/download buttons to the task interface.
6. Build and upload the process project.

7. Deliver an e-form instance to the repository instance folder (sourced from the correct e-form template).

The Content Service Platform monitors the specified instance folder in the repository; when an event monitored by the content listener occurs in the folder, the Process Engine automatically converts the e-form instance to an IS document based on the previously defined e-form template-to-IS document type relationship.

The receive step handles the e-form based IS document just as it would any other IS document, and the e-form data is added to the process pipeline.

If there is a task step in the process, when it is reached, a new task is queued and appears in the appropriate My webMethods inboxes, with e-form fields and e-form instance values in the Data view.

In My webMethods, the assigned task user can:

- Work with e-form data in the task.
- Download the e-form from the task for offline access (in actual e-form format).
- Upload the modified e-form back into the task.

8 Implementing E-forms

- Adobe LiveCycle Implementation Considerations 62
- Microsoft Office InfoPath Implementation Considerations 68

Adobe LiveCycle Implementation Considerations

The webMethods product suite supports interactive Adobe forms that comply with XML Forms Architecture (XFA). Interactive XML forms can be deployed as PDF or as an XML Data Package (XDP). An XDP file is an XML file that packages a PDF file in XML, along with XML form and template data.

The webMethods product suite requires that the form elements and data be presented in XDP format to satisfy both its design time and run-time requirements.

Adobe Forms come in two categories:

- XFA-compliant, XML-based forms created, modified, and managed with Adobe LiveCycle Designer. These e-forms are the primary implementation for use with the webMethods product suite.
- Acrobat forms created, modified, and managed with Adobe Acrobat. These forms must be imported into Adobe LiveCycle Designer before they can be used with webMethods product suite. For more information, see ["Using Traditional PDF Forms Created with Adobe Acrobat" on page 67](#).

Creating XFA-Compliant Forms with LiveCycle Designer

Adobe LiveCycle Designer is a design tool for creating rich interactive e-forms that comply with XFA (XML Forms Architecture, sometimes referred to as XFA forms). You can design e-form templates and save them in PDF and XDP format. The target format of the forms depends upon who the end users are and how they will interact with the document and submit their data.

- E-forms saved and distributed in the PDF format are worked upon by end users using the Adobe Reader software.
- The XDP format provides greatly enhanced capabilities for rendering and submitting user data, and requires the Adobe Forms Server for its rendering and data submission.

The webMethods product suite requires e-form definition in XDP format. The typical steps involved in creating the design-time XDP from a PDF document are as follows:

1. Form creation—Create the e-form in LiveCycle Designer and save it as PDF (Static or Dynamic).
2. Enable Usage Rights—To be able to save and export form data, you must set the appropriate usage rights for the PDF file using Adobe Reader Extension. Note that usage rights may also be enabled using Acrobat Professional software.
3. Preview and Export—Open up the rights-enabled PDF form using Adobe Reader and then enter sample data into all the form fields. This step is particularly important because the exported data file is used to infer the XML schema of the form fields. The quality of the input data dictates how well the form fields are represented

in the schema and it is advised that conforming data be entered for each of the form fields.

Additionally, care should be taken to ensure that the values match the data-type and format expected from the end users. For example, area code, phone number, and date fields may allow both formatted and non-formatted data; in this case, during preview, you are advised to enter the data in the format that is most likely to be used by the end-users. Be sure to enter data for non-required fields as well.

4. Save and export in XDP format—Finally, review the filled-in form and make sure that valid data is entered for all editable fields. Save as "XML Data package (xdp)" and export the form data as an XML file (the default extension of the exported file is .xdp). If you plan to use the e-form with a task application so the task user can upload and download e-forms, save the file in PDF format as well.

You can now place the new e-form template (the exported .xdp file from step 4 above) in the appropriate webMethods Content Service Platform repository templates folder. From there, the template can be imported into Software AG Designer as an IS document type or as business data for a task.

Limitations to Using XDP Files

There are some limitations to creating an IS document type using an XDP file. Software AG Designer examines the form data within the XDP and derives an XML schema from it. This inferred schema is then used to generate the IS document type.

An XML file is restricted in that it does not contain specific data type and formatting information for its elements, so the resulting document type is also less restrictive in terms of allowed data values. There are two ways to mitigate risks involved when creating IS document type using XDP:

- Include sample data that best represents the formatting and type restrictions for each of the form fields. Also, make sure that data is supplied even for non-required form fields to eliminate ambiguity when schema elements are derived.
- The user can optionally review the generated IS document type and edit document fields manually.

For more information on importing e-form templates and working with IS document types, see the *webMethods Service Development Help*.

Adobe Design Time Considerations for Task Applications

If you plan to configure a task application so the task user can upload and download Adobe LiveCycle e-forms, it is important to keep the following points in mind.

The XDP data file is used at design time to create the JCR template provider for the task download function. During run time, at the time of download, the XDP data file requires an associated PDF file so the e-form can be successfully opened in a PDF reader. You can save the template in PDF format at creation time, as noted in "[Creating XFA-Compliant Forms with LiveCycle Designer](#)" on page 62. You must store the PDF in a location

accessible to Software AG Designer and the run-time components (Integration Server, Process Engine, and Task Engine).

The XDP stores the location of its associated PDF file as an href attribute inside the XDP. That href attribute must be a fully-resolvable URL; for example:

- http://
- ftp://
- Absolute path: C:\FolderA\FolderB\File.pdf or /forms/folderA/folderB/File.pdf
- Relative path: ../FolderA/FolderB/File.pdf.

Otherwise, the download will fail at run time.

The value of the href is set at export time (although you can manually edit the href after completing the export), and is derived based on the target location of the XDP, relative to the location of the PDF.

If an XDP file is moved from its original location, then the relative link to its associated PDF will be broken. In this case, the XDP file will not open in a PDF reader (in most cases, the PDF reader will prompt the user to browse for the PDF) and a JCR provider cannot be created.

Defining the Path Between an XDP and a PDF File

Use the following method to ensure the proper definition of a path to the PDF associated with an XDP file.

To manually edit the path to the PDF

1. The template folder in webMethods Content Service Platform (that is, the location where both the PDF and the exported XDP will reside) is *Software AG_directory /CSP/system/templates*.
2. Place the PDF file in this location, and export the XDP file into the same location.
3. Open the XDP file in a text editor and modify the href value. At this point, the href will contain only the name of the PDF file; for example, href="filename.pdf". Change the value to conform to the following example:

```
New Value: href =
http://CSPServerName:port/smi://sagwmcsp82_retrieval/C:/softwareag/CSP/
server/JBX/1012/1023/57//10121023572703962829f0d089ecf113.1.1
```

Where *CSPServerName:port* defines the name and port number of the CSP server containing the PDF file. The remainder of the URL can be determined as described in ["Determining the CSP URL for a PDF File" on page 65](#).

4. Save the file.

Determining the CSP URL for a PDF File

As described in [Defining the Path Between an XDP and a PDF File](#), you must enter a webMethods Content Service Platform (CSP) URL in the XDP file. Use this procedure to determine the URL.

To determine the CSP URL of a PDF File

1. Open any CSP-enabled task instance.
2. Click the **Content** tab.
3. Click **Attach Existing Content**.
4. Search for the template content in the System/Template folder.
5. Attach the found content to the task instance.
6. Refer to the **Attachments List** panel that shows content objects for the selected content.
7. Locate and copy the download URL so you can paste it into the XPD file.

Creating an XDP Form without an Associated PDF

The form author can choose to save a newly created form in XDP format without having to create a PDF. Such forms typically have a button that a user can click to submit their data in XDP format.

After the form design is finalized, the author can preview the document within LiveCycle Designer.

- In the preview mode, the form author enters sample data for every field in the form.
- The form author clicks the submit button and submits the sample data to the specified folder location.
- This submitted data is captured as an XDP file and saved into a location that is accessible to Software AG Designer.

Using an XML Schema as a Template Source

During form creation, it is possible to use an XML schema (as a .xsd file) as a template source. The form fields are bound to the XML elements at design time. If the form is designed entirely from an XML schema then it is possible to use the XML schema. There are some distinct advantages of using this method:

- It eliminates the need to provide the design time XDP file with the sample form data, which is used to infer a schema. At design time, when using the e-form import wizard to create an IS document type within Software AG Designer, the user selects the XML schema file (.xsd) that was used to create the form instead of the .xdp file.

- When an .xsd file is used to create an IS document type, the resulting document type fully encapsulates content and type restrictions for each of its elements per the schema.

Making Adobe E-form Instances Available at Run Time

To start a business process at run time, the webMethods product suite expects an XDP file that contains the user-submitted data to appear in the monitored instances folder (that is, any webMethods Content Service Platform folder that is monitored by a Content Listener).

Important: Adobe LiveCycle e-form instances placed in a monitored e-form instances folder *must* have a file name extension of .xdp.

When an e-form is ready for distribution, (either PDF or XDP), the form data can be submitted as described in the following two scenarios:

1st Scenario: PDF E-forms

1. Deploy the PDF e-form and make it available to the end users.
2. Users access the e-form using Acrobat or Reader software and fill in the form data.
3. After the form is filled in, the user must export the form data in XDP format. The XDP file serves up as the run-time artifact that must be copied into the instances folder to trigger a business process.
4. The user may optionally save the file with the data local to their environment.

2nd Scenario: XDP E-forms

1. Deploy the e-form to the Forms Server for general user access.
2. Customers access it via the web, download it, fill it in, and click the e-form's submit button.
3. The filled-in data is captured in XDP format and copied into the instances folder to trigger a business process.

Note: As described above, the run-time XDP file is an XML file that consists of form data and a reference to the original PDF document. At run time, when a user downloads an e-form instance, the href tag will link the form to its original PDF document.

Using LiveCycle E-forms with Adobe Reader

For proper operation with LiveCycle e-forms, you are advised to configure your PDF reader application to display the PDF file in the reader application itself, and not within a browser (that is, as a plug-in). The following procedures are for Adobe Reader; other

PDF reader applications have not been tested with the webMethods product suite e-form implementation.

To configure the Adobe Reader application

1. Close all browsers.
2. Open Adobe Reader.
3. In Adobe Reader, **Edit > Preferences > Internet**.
4. Clear the **Display PDF in browser** check box.
5. Click **OK**.
6. Close Adobe Reader.

Behavior When Downloading an E-form from a Task

The user:

1. Logs into My webMethods.
2. Locates and open the task.
3. Clicks the download button.

Adobe Reader starts separately (that is, as a standalone application) and displays the e-form PDF. At this point, the user can save the file locally in PDF format, or export the data in XDP format. The file must be in XDP to be uploaded back to the task.

Note: If you are using Mozilla Firefox, you may see a different behavior, depending on the version of Adobe Reader installed. With Adobe Reader 8.x.x, clicking the download button displays the File Save As dialog box, giving you the option to either open the file using Adobe Reader or to save the file locally as an .xdp. If you select the 'open with reader' option, the Adobe Reader application opens and displays the PDF.

Using Traditional PDF Forms Created with Adobe Acrobat

Forms that are created using the older Acrobat suite of products are not XFA compliant. Users who need to leverage such pre-existing PDF forms may be able to do so by importing these documents into LiveCycle Designer; however, depending upon the complexity and the version involved, some cleanup might be required.

For more details related to importing PDF forms into LiveCycle, refer to the LiveCycle documentation. After the document is imported into LiveCycle and the form design is finalized, then the e-form can be distributed as an XFA-compliant form.

Using Digital Signatures with Adobe LiveCycle E-forms

Digitally signing LiveCycle e-forms has two aspects:

- Document signatures that secure the look and feel of the form.
- Data signatures that secure form data at submission time.

The suggested approach to integrate signed LiveCycle forms with the webMethods product suite is to use document signatures.

Microsoft Office InfoPath Implementation Considerations

Microsoft Office InfoPath is an integrated application for developing and publishing e-forms from within the Microsoft Office suite of products. InfoPath e-forms are XML based and can be saved as XML files and published as XSN files. The webMethods product suite imports only XSN files for use as e-form templates.

Making InfoPath E-form Instances Available at Run Time

To start a business process at run time, the webMethods product suite expects a .xml file that contains the user submitted data to appear in the monitored instances folder (that is, any webMethods Content Service Platform folder that is monitored by a Content Listener).

Important: Microsoft Office InfoPath e-form instances placed in a monitored e-form instances folder *must* have a file name extension of .xml.

Working with InfoPath Files in Non-Windows Environments

InfoPath is a Windows-only application. If your organization is working in a Windows-only environment, this should pose no problems. However, if portions of your webMethods product suite are running on other operating systems, such as UNIX or Linux, you may face some limitations.

For example, it should be possible to trigger a business process running on a UNIX system with an InfoPath e-form that arrives in the instances folder of your repository, assuming you have correctly imported the e-form template, created an IS document type, and configured the business process to use it.

However, if a user working on a non-Windows operating system downloads an InfoPath e-form from a task, it will not be possible for the user to actually work with the e-form in that environment, as the InfoPath application can only run in a Windows environment.

Therefore, if you plan to enable users to download InfoPath e-forms, work with them in their local environment, and then upload them back to a task, you must ensure that those users log on to My webMethods from a Windows operating system.

Software AG Designer and the run-time components (Integration Server, Process Engine, and Task Engine) can be implemented on non-Windows operating systems even when used with InfoPath e-forms.

Important: If you are running your installation of My webMethods Server/Task Engine on a UNIX or Linux platform, you must install version 1.2 or later of the `cabextract` utility into `/usr/bin` (or create a link to it within `/usr/bin`) on your My webMethods Server system to enable proper operation between the Task Engine and InfoPath e-forms. For further information about this utility, see <http://www.cabextract.org.uk/>.

Other Solutions for Non-Windows Environments

If you expect that some of your users will be working on non-Windows computers and those users will be downloading and interacting with InfoPath e-forms, consider developing a process that will translate the e-form data in the pipeline into an Adobe form. That form can then be downloaded from a task onto a non-Windows platform, worked with in Adobe Reader, and then uploaded back into a task in the process.

Field Limitations

You might encounter the following issues when working with Microsoft Office InfoPath:

- IS document types generated from an InfoPath XSD may contain field names that webMethods Broker will not propagate to non IS-clients.
- InfoPath template fields intended to be mandatory are imported into an IS document type with the fields shown as optional. This applies to InfoPath template fields with the "Cannot be blank" property selected. When InfoPath renders the form into an XML Schema, these fields are shown as optional, which is how they are treated when they are imported into an IS document type.

This is caused by the way InfoPath handles attributes. Attribute elements are optional by default; in this case, the "use" attribute for the field in question should be set to "required." If this value is not specified, then the default value is "optional" and it is imported as such.

For more information, you can refer to these sources:

http://www.w3schools.com/Schema/el_attribute.asp

<http://www.infopathdev.com/blogs/greg/archive/2004/09/16/The-Difference-Between-Optional-and-Not-Required.aspx>

Other Limitations and Information

The following limitation apply to Microsoft Office InfoPath e-forms:

- InfoPath e-forms with more than one root node are not supported.
- By default, the root node in an InfoPath e-form is `myFields`. The designer of the InfoPath e-form can set a different name for the root node. Although this is supported, it is the user's responsibility to contact the e-form developer to learn the alternate name for the root node prior to attempting to import the e-form.

Using Digital Signatures with Microsoft InfoPath E-forms

When designing an InfoPath e-form, there are two generic actions involved when creating e-forms enabled for digital signatures:

- You can enable digital signatures for the entire form or for specific parts of the form. When users sign the form, the digital signature locks the data so that it cannot be changed, and stores the view with the digital signature.
- You can optionally configure a prompt to the user to sign the form upon submittal if the user attempts to submit the form without a digital signature (this feature exists only for signatures that sign the entire form). If you do not enable this prompt, it is possible for a user to save the form without signing it.

During design time, when an InfoPath e-form enabled for digital signatures is imported as an IS document type, the signature field is set to "required." If an instance of this e-form is submitted at run time without a digital signature, a run-time exception will appear in the Integration Server error log, indicating that the form data "does not conform to the Publishable Document Type."

When an InfoPath template associated with signature-enabled form is imported as an IS document type, you are advised to examine the properties of the signature field in the document. The signature field is typically set to be a required field. If you want to accept e-forms that are enabled for digital signature but are not digitally signed, you must manually change the field type in the IS document type to "optional" to avoid errors at run time.

9 Content Service Platform Services

| | |
|---|----|
| ■ Content Service Platform Built-In Services Location | 72 |
| ■ Connection Folder | 72 |
| ■ pub.csp.connection:addConnection | 73 |
| ■ pub.csp.connection:deleteConnection | 74 |
| ■ pub.csp.connection:listConnections | 74 |
| ■ Content Folder | 74 |
| ■ pub.csp.content:createContentID | 75 |
| ■ pub.csp.content:createFilter | 76 |
| ■ pub.csp.content:createListener | 77 |
| ■ pub.csp.content:listAllContentDefinitions | 78 |
| ■ pub.csp.content:listAllIndexIDs | 78 |
| ■ File Folder | 79 |
| ■ pub.csp.file:addFile | 80 |
| ■ pub.csp.file:getFile | 81 |
| ■ pub.csp.file:retrieveFiles | 81 |
| ■ pub.csp.file:updateFile | 82 |
| ■ Search Folder | 83 |
| ■ pub.csp.search:createMetadataMap | 84 |
| ■ pub.csp.search:createSearchTerm | 84 |
| ■ pub.csp.search:searchMetadata | 85 |
| ■ Server Folder | 86 |
| ■ pub.csp.server:createISServerForListener | 86 |
| ■ pub.csp.server:listAllISServers | 87 |

Content Service Platform Built-In Services Location

The built-in services in this chapter are installed on the Integration Server as part of the WmContentServicePlatform package. These Java services can be found in the indicated folder location in the Package Navigator view of Designer, or in the Package Management link in the Integration Server Administrator.

This chapter describes the services and supporting elements found in the `\pub` folder. You can use these services to perform a wide variety of actions on the Content Service Platform (CSP), such as adding a connection, creating a new content object, and adding a file to a content object.

The `WmContentServicePlatform\pub\csp` folder is subdivided into five folders whose names also represent the services contained therein:

- [Connection Folder](#)
- [Content Folder](#)
- [File Folder](#)
- [Search Folder](#)
- [Server Folder](#)

For additional information about working with services in Designer, see *webMethods Service Development Help*.

Related Topics

[Configuring and Deploying a webMethods Content Service Platform Environment](#)

Connection Folder

The following elements are available in this folder:

| WmContentServicePlatform Package Element | Description |
|---|--|
| pub.csp.connection:addConnection | This service adds a new CSP connection against which the other services in the package can be executed. The <i>friendlyName</i> identifies the added connection. |
| pub.csp.connection:deleteConnection | This service deletes the CSP connection identified by the <i>friendlyName</i> parameter. |

| WmContentServicePlatform Package Element | Description |
|--|--|
| pub.csp.connection:listConnections | This service returns an array of IData containing all connections available for use in this package. |

pub.csp.connection:addConnection

Adds a new CSP connection against which the other services in the package can be executed. The *friendlyName* identifies the added connection.

Tip: The connections created with the addConnection service are similar to the Content Service Platform Server environment connections in My webMethods Server: **Administration > System-Wide > Environments > Define Environments**. For more information, see "[Configuring a Content Service Platform Environment](#)" on page 26.

Input Parameters

| | |
|---------------------|--|
| <i>host</i> | String The CSP host name or IP address to which you want to connect |
| <i>port</i> | String The CSP port. For the typical CSP installation, this value should be 9010. |
| <i>userName</i> | String The CSP user connecting to the CSP |
| <i>password</i> | String The password associated with the CSP <i>userName</i> |
| <i>friendlyName</i> | String The identifier associated with the CSP server connection being added |
| <i>overwrite</i> | String Optional. Deletes and replaces an existing <i>friendlyName</i> |

Important: There is no undo when you *overwrite* .

Output Parameters

| | |
|---------------|--|
| <i>status</i> | String Indicates success or failure |
|---------------|--|

pub.csp.connection:deleteConnection

Deletes the CSP connection identified by the *friendlyName* parameter.

Input Parameters

friendlyName **String** The CSP connection to delete

Output Parameters

status **String** Indicates success or failure

pub.csp.connection:listConnections

Returns an array of IData containing all connections available for use in this package.

Input Parameters

None.

Output Parameters

connections **String Array** CSP *friendlyName* s. If there are no connections, this is null.

connectionDetails **IData Array** Each element in the array is a connectionDetail object, and each connectionDetail object contains the *friendlyName*, *host*, *port*, and *login* (*userName*). The list contains the key-value pairs used in each IData element.

If there are no connections, there are no *connectionDetails* .

Content Folder

The following elements are available in this folder:

| WmContentServicePlatform Package Element | Description |
|---|---|
| pub.csp.content:createContentID | This service creates a new content object in the CSP indicated by the |

| WmContentServicePlatform Package Element | Description |
|--|--|
| | <i>friendlyName</i> . After you create the content object, you can add document files to it using <code>addFile</code> . |
| <code>pub.csp.content:createFilter</code> | This service creates a new document filter. |
| <code>pub.csp.content:createListener</code> | This service creates a new CSP listener, which is used to trigger a Designer process based on monitored activity. |
| <code>pub.csp.content:listAllContentDefinitions</code> | This service lists all the content definitions on a supplied CSP server <i>friendlyName</i> . The content definitions are returned as a <code>HashMap</code> where the key is the <i>indexName</i> and the value is the <i>indexID</i> . |
| <code>pub.csp.content:listAllIndexIDs</code> | This service lists all <i>indexIDs</i> on a specified CSP server in <code>Map <String,Integer></code> format. The service output, <i>indexMap</i> , provides data that can be used in many other CSP services. |

pub.csp.content:createContentID

Creates a new content object in the CSP indicated by the *friendlyName* . After you create the content object, you can add document files to it using `addFile`.

Input Parameters

| | |
|---------------------|--|
| <i>nodeID</i> | String The ID of the node to which the content will be added |
| | Tip: You can see an example of <i>nodeID</i> usage in the sample included with webMethods Content Service Platform. The <i>nodeID</i> of “Automobile Accident Claims” is 8. |
| <i>friendlyName</i> | String The CSP server where the <i>nodeID</i> is located |
| <i>metadataMap</i> | HashMap <String,String> or String Optional. Contains key-value pairs that represent metadata about the created node. |

This can either be a String containing comma-separated key=value pairs (key1=value1, key2=value2) or a HashMap <String,String> with key-value pairs at each map index/position. You can associate any desired metadata key-value pairs with the created *contentID*.

Output Parameters

| | |
|------------------|--|
| <i>contentID</i> | String The <i>contentID</i> of the object in the CSP with which files are associated. Represents a globally unique ID, or <i>GUID</i> . |
| <i>status</i> | String Indicates success or failure |

pub.csp.content:createFilter

Creates a new document filter.

Tip: The connections created with the `createFilter` service are similar to the Content Service Platform Server environment connections in My webMethods Server. For more information, see ["Configuring a Content Service Platform Environment" on page 26](#).

Input Parameters

| | |
|------------------------------|---|
| <i>friendlyName</i> | String The identifier of the CSP server where the filter is added |
| <i>cspFilterFriendlyName</i> | String The user-specified "friendly" name of the new filter |
| <i>nodeIDs</i> | String The comma-separated list of IDs of the nodes on which to filter |

Tip: You can see an example of *nodeID* usage in the sample included with webMethods Content Service Platform. The *nodeID* of "Automobile Accident Claims" is 8.

Output Parameters

| | |
|---------------|--|
| <i>status</i> | String Indicates success or failure |
|---------------|--|

pub.csp.content:createListener

Creates a new CSP listener, which is used to trigger a Designer process based on monitored activity.

Tip: The listeners created with the `createListener` service are similar to the listeners in My webMethods Server. For more information, see "[Configuring a Content Service Platform Environment](#)" on page 26.

Input Parameters

friendlyName **String** The identifier of the CSP server on which the listener is created

cspListenerFriendlyName **String** The user-specified "friendly" name of the new listener

eventType **String** The comma-separated list of one or more of the following event types for the listener to monitor: CREATE, UPDATE, DEACTIVATE, and DELETE.

nodeIDs **String** The comma-separated list of IDs of the nodes to monitor

Tip: You can see an example of *nodeID* usage in the sample included with webMethods Content Service Platform. The *nodeID* of "Automobile Accident Claims" is 8.

searchFilter **String** Indicates which specific *indexID* value is being listened to. *searchFilter* must be formatted like this: (*{indexID}* operator value) <join clause> OR (*indexName* operator value) <join clause>. Two examples:

- "Driver Last Name ~ * and"
- "({\$200000009} > 06.08.2012) AND"

Since the *indexID* for Driver Last Name is the same for all accident claims, use *searchFilter* to specify which specific Driver Last Name is of interest.

isName **String** The name of the IS where the listener is created

description **String** Optional. Description of the new listener.

Output Parameters

status **String** Indicates success or failure

pub.csp.content:listAllContentDefinitions

Lists all the content definitions on a supplied CSP server *friendlyName*. The content definitions are returned as a HashMap where the key is the *indexName* and the value is the *indexID*.

Input Parameters

friendlyName **String** The identifier of the CSP server

Output Parameters

size **Integer** Indicates the number of elements included in the *contentDefinitionMap* HashMap.

contentDefinitionMap **HashMap <String,String>** Lists content definitions in the form: {*indexName=indexID*, *indexNameA=indexID2*}. For example: {Automobile Accident Claims=8, IS Servers=1000011000}.

status **String** Indicates success or failure

pub.csp.content:listAllIndexIDs

Lists all *indexIDs* on a specified CSP server in Map <String,Integer> format. The service output, *indexMap*, provides data that can be used in many other CSP services.

Input Parameters

friendlyName **String** The identifier of the CSP server

nodeID **String** The ID of the node

Output Parameters

indexMap **Map <String,Integer>** Uses an *indexName*-to-*indexID* mapping: {*indexName=indexID* }. For example: {Driver Last Name=236 , Policy Number=1107 }.

File Folder

The following elements are available in this folder:

| WmContentServicePlatform Package Element | Description |
|--|---|
| pub.csp.file:addFile | This service adds a new file into the CSP indicated by the <i>friendlyName</i> . The service creates a new <i>contentID</i> if one is not provided. Also, one or both of <i>contentID</i> and <i>nodeID</i> fields can be provided, but at least one needs to be provided; that is why BOTH are listed as required. If both are provided then the <i>nodeID</i> takes precedence. If both are null then, a service error results. |
| pub.csp.file:getFile | This service retrieves a single file from the CSP, based on the <i>contentID</i> and <i>fileID</i> pair. |
| pub.csp.file:retrieveFiles | This service returns information about all files associated with the supplied <i>contentID</i> . The CSPContentObject contains the input <i>contentID</i> ; a <i>fileID</i> value, which is an index into the array of files associated with this <i>contentID</i> ; and a URL, which provides direct access to the file in the CSP. |
| pub.csp.file:updateFile | This service updates the content and metadata of an existing CSP file associated with the supplied <i>contentID</i> . |

pub.csp.file:addFile

Adds a new file into the CSP indicated by the *friendlyName* and creates a new *contentID* if one is not provided. One or both of *contentID* and *nodeID* fields can be provided, but at least one needs to be provided; that is why BOTH are listed as required. If both are provided then the *nodeID* takes precedence. If both are null, a service error results.

Input Parameters

| | |
|------------------------|--|
| <i>contentID</i> | String The <i>contentID</i> of the object in the CSP with which the file is associated. Represents a globally unique ID, or <i>GUID</i> . If a <i>contentID</i> is not provided, one is created at the supplied <i>nodeID</i> . |
| <i>nodeID</i> | String The ID of the node |
| <i>fileName</i> | String The name of the file to be added |
| <i>fileInputStream</i> | String (Base64 encoded) Either a Base64 encoded byte stream of the file to be added, or the full path to the file |
| <i>friendlyName</i> | String Identifies the CSP server where the file is located |
| <i>metadataMap</i> | HashMap <String,String> or String Optional. Contains a collection of key-value pairs that represent data about the created node. This can be either a String in the form of comma-separated key=value pairs or a HashMap with key-value pairs entered at each map index/position. This allows you to associate any desired metadata key-value pairs with the created <i>contentID</i> . |

Output Parameters

| | |
|------------------|---|
| <i>contentID</i> | String The supplied <i>contentID</i> , or the created <i>contentID</i> if one was not supplied |
| <i>fileID</i> | String The ID of the file |
| <i>status</i> | String Indicates success or failure |

Usage Notes

You can supply a *contentID* and a *nodeID*, or only one of these. Both fields are considered required. At least one must be supplied. If both are supplied, *nodeID* takes precedence. If both are null, an error results.

pub.csp.file:getFile

Retrieves a single file from the CSP, based on the *contentID* and *fileID* pair.

Input Parameters

| | |
|---------------------|---|
| <i>fileID</i> | String The position of the file in the <i>fileList</i> returned in the <i>retrieveFiles</i> service. This value is not constant. The <i>fileID</i> can be obtained from the <i>fileList</i> , which is one of the outputs of <i>retrieveFiles</i> . |
| <i>friendlyName</i> | String Identifies the CSP server where the file is located |
| <i>downloadDir</i> | String The directory into which the file is to be downloaded |
| <i>contentID</i> | String The <i>contentID</i> of the object in the CSP with which files are associated. Represents a globally unique ID, or <i>GUID</i> . |

Output Parameters

| | |
|-----------------|--|
| <i>fileName</i> | String The name of the file |
| <i>status</i> | String Indicates success or failure |

pub.csp.file:retrieveFiles

Returns information about all files associated with the supplied *contentID*. The *CSPContentObject* represents the content object.

Note: Each content object has its own content properties as well as a list of content objects. The structure is organized as a tree.

The *CSPContentObject* contains the input *contentID*; a *fileID* value, which is an index into the array of files associated with this *contentID*; and a URL, which provides direct access to the file in the CSP.

Input Parameters

| | |
|---------------------|--|
| <i>friendlyName</i> | String The identifier of the CSP server from which the files are being retrieved |
| <i>contentID</i> | String The <i>contentID</i> of the object in the CSP with which files are associated. Represents a globally unique ID, or <i>GUID</i> . |

Output Parameters

| | |
|-----------------|--|
| <i>fileList</i> | List <CSPContentObject> The CSPContentObject contains the input <i>contentID</i> ; a <i>fileID</i> value, which is an index into the array of files associated with this <i>contentID</i> ; and a URL, which provides direct access to the file in the CSP. |
| <i>status</i> | String Indicates success or failure |

pub.csp.file:updateFile

Updates the content and metadata of an existing CSP file associated with the supplied *contentID*.

Input Parameters

| | |
|------------------------|--|
| <i>fileID</i> | String Represents the index into the file list that comes back in the <i>retrieveFiles</i> service. The value used here depends on what position in the <i>contentID</i> it is located. The value of the <i>fileID</i> is not constant, so you must check the values returned in the <i>fileList</i> from the <i>retrieveFiles</i> service. |
| <i>contentID</i> | String The <i>contentID</i> of the object in the CSP with which files are associated. Represents a globally unique ID, or <i>GUID</i> . |
| <i>fileInputStream</i> | String (Base64 encoded) Either of two options: the full path to the file to be updated, or a Base64 encoded byte stream of the file to be updated. |
| <i>fileName</i> | String The name of the file to update |
| <i>friendlyName</i> | String The identifier associated with the CSP server on which the file is being updated |

metadataMap **HashMap <String,String>** or **String** Optional. Contains a collection of key-value pairs that represent data about the created node. This can be either a String in the form of comma-separated key=value pairs or a HashMap with key-value pairs entered at each map index/position. This allows you to associate any desired metadata key-value pairs with the created *contentID*.

Output Parameters

fileID **String** The supplied ID of the file

status **String** Indicates success or failure

Search Folder

The following elements are available in this folder:

| WmContentServicePlatform Package Element | Description |
|--|--|
| pub.csp.search:createMetadataMap | Translates the key-value pairs in the supplied <i>IData</i> of an Integration Server document into a HashMap called <i>metadataMap</i> . This output can be used as input to other services in the package, such as <i>createContentID</i> , <i>addFile</i> , and <i>updateFile</i> . |
| pub.csp.search:createSearchTerm | Creates a <i>searchTerm</i> object from the supplied values. This object can be used as input to <i>searchMetadata</i> . When this service is invoked with Java code, multiple invocations of the service can be made in order to produce <i>searchTerm</i> objects that can then be joined using the <i>join</i> input parameter. |
| pub.csp.search:searchMetadata | Searches the CSP for specific metadata. You must use the <i>createSearchTerm</i> service and place the resulting <i>searchTerm</i> into the <i>searchTerms</i> list to use this service. |

pub.csp.search:createMetadataMap

Translates the key-value pairs in the supplied IData of an Integration Server document into a HashMap called *metadataMap*. This output can be used as input to other services in the package, such as *createContentID*, *addFile*, and *updateFile*.

Input Parameters

metadata **IData** IData is expressed in key-value pairs

Output Parameters

metadataMap **HashMap <String,String>** Contains a collection of key-value pairs that represent the supplied IData of an IS document.

pub.csp.search:createSearchTerm

Creates a *searchTerm* object from the supplied values. The *searchTerm* object can be used as input to *searchMetadata*. When this service is invoked with Java code, multiple invocations of the service can be made in order to produce *searchTerm* objects that can then be joined using the *join* input parameter.

Input Parameters

indexID **String** Contains a single *indexID* value in the form of a CSP indexID, some examples of which can be seen in the Automobile Accident Claims sample included with the CSP.

For instance, the Index Name "Driver Last Name" has an *indexID* value of 200000001, and the Index Name "Driver First Name" has an *indexID* value of 200000002.

operator **String** The operator to use in the comparison.

- ~ (contains)
- !~ (does not equal)
- < (less than)
- < (less than)
- <= (less than or equal to)

- >= (greater than or equal to)

| | |
|--------------------|---|
| <i>searchValue</i> | String User-defined value to use in the comparison. You can use the asterisk (*) character to indicate all values. |
| <i>join</i> | <p>String Set to either AND or OR. The default value is AND. This value is used to join multiple <i>searchTerm</i> objects. When this service is invoked from the launch menu, you can provide only one input value. AND should be used in that case.</p> <p>When this service is invoked with Java code, multiple invocations of the service can be made in order to produce <i>searchTerm</i> objects that can then be joined using the <i>join</i> input parameter.</p> |

Output Parameters

| | |
|---------------------|---|
| <i>searchTerm</i> | Object Can be used as an input to the <i>searchMetadata</i> service |
| <i>searchFilter</i> | String A toString() representation of the <i>searchTerm</i> output: input fields are encapsulated into a single formatted string |

pub.csp.search:searchMetadata

Searches the CSP for specific metadata. You must use the *createSearchTerm* service and place the resulting *searchTerm* into the *searchTerms* list to use this service.

Input Parameters

| | |
|---------------------|---|
| <i>nodeID</i> | String The ID of the node being searched |
| <i>friendlyName</i> | String The identifier of the CSP server being searched |
| <i>searchTerms</i> | List <SearchTerms> The list of SearchTerms can be built by calling the <i>createSearchTerm</i> service and adding the <i>searchTerm</i> output to the list of <i>searchTerms</i> . |

Output Parameters

| | |
|--------------------|--|
| <i>contentList</i> | String List Contains a list of <i>contentID</i> s that can be used in other <i>WmContentServicePlatform</i> services, such as <i>addFile</i> and <i>retrieveFiles</i> . |
|--------------------|--|

status **String** Indicates success or failure

Server Folder

The following elements are available in this folder:

| WmContentServicePlatform Package Element | Description |
|--|--|
| pub.csp.server:createISServerForListener | This service creates an Integration Server connection for listener events. The IS connection is created on the CSP indicated by the supplied <i>friendlyName</i> . |
| pub.csp.server:listAllISServers | This service returns a list of all IS servers associated with the supplied <i>friendlyName</i> of the CSP server. |

pub.csp.server:createISServerForListener

Creates an Integration Server connection for listener events. The IS connection is created on the CSP indicated by the supplied *friendlyName*.

Input Parameters

| | |
|---------------------|--|
| <i>friendlyName</i> | String The identifier of the CSP server to which the IS listener is being added |
| <i>host</i> | String IS host name or IP address |
| <i>port</i> | String IS port |
| <i>isName</i> | String The “friendly” name of the IS |
| <i>userName</i> | String The user to connect to the IS |
| <i>isPassword</i> | String The IS password for the supplied <i>userName</i> |

Output Parameters

status **String** Indicates success or failure

pub.csp.server:listAllIServers

Returns a list of all IS servers associated with the supplied *friendlyName* of the CSP server.

Input Parameters

friendlyName **String** The identifier of the CSP server

Output Parameters

IServerMap **HashMap <String,String>** Formatted as [isName1, userName1@host1][isName2, username2@host2]

status **String** Indicates success or failure

Index

A

- Adobe Acrobat forms, using 67
- Adobe LiveCycle
 - Acrobat forms, using with 67
 - Adobe Reader interaction 66
 - digital signature considerations 67
 - implementation considerations 62
 - limitations with 63
 - starting processes with e-forms 66
 - task application considerations 63
 - typical usage 62
 - XML schema as template source 65
- Adobe Reader, interaction with e-forms 66, 69
- applications, e-form, supported 54

C

- Central Configuration 25
 - determining if present 25
 - installing 25
- cluster considerations, Integration Server/Process Engine 30
- configuration
 - CSP, overview of environment 24
- connections, webMethods Content Service Platform managing 29
- connectivity requirement for CSP 19
- Content Configurator
 - content filter, creating with 37
 - content listener, creating with 40
 - determining if present 36
 - installing 36
- content filter
 - modifying 38
 - viewing 38
- content filters
 - about 36
 - creating 37
- content integration
 - general concepts 12
 - suite behavior 13
- content listener
 - about 40
 - creating 40
 - for deactivated document types 42
 - deleting 38, 42
 - modifying 42

- viewing 42
- Content Service Platform
 - built-in services location 72
 - configuration of environment, overview 24
 - connectivity requirements 19
 - determining if interface is present 36
 - environment
 - changingport number 32
 - configuring 26
 - considerations 26
 - deploying 30
 - updating 31
 - error messages from Oracle database 19
 - installing My webMethods interface for 25, 36
 - listener components 43
 - starting 19
 - WebReader 20
- Content Service Platform server, accessing 29

D

- deactivated document types, listener requirements 42
- deleting templates and IS document types 50
- deploying a CSP environment 30
 - considerations of 30
 - deployment procedure 31
 - prerequisites for 31
- Designer
 - content integration with 14
 - integration with e-forms 56, 56
- digital signature considerations
 - Adobe LiveCycle 67
 - Microsoft Office InfoPath 70
- document types
 - general usage description 14
 - maintaining life of 50
 - synchronizing with IS document types 50
 - viewing attachments with WebReader 20
- documentation
 - using effectively 7
- downloading an e-form from a task 67

E

- e-forms
 - Adobe LiveCycle information 62
 - Adobe Reader interaction with 66
 - downloading from a task 67
 - general concepts 54

- general usage description 58
 - instances, location of in CSP repository 56
 - introduction to 54
 - Microsoft Office InfoPath information 68
 - repository overview 56
 - suite integration 56
 - templates
 - working with in CSP 57
 - templates, location of in CSP repository 56
 - environment
 - configuration overview 24
 - considerations for CSP 26
 - deploying 30
 - overview of configuration 24
 - port number, changing 32
 - updating a deployed 31
 - environment entries, configuring for CSP repository 46
 - error messages from Oracle database 19
- F**
- field limitations, Microsoft Office InfoPath 69
- G**
- global environment entries, configuring 46
- H**
- home page, Content Service Platform
 - accessing 29
 - logging out of 29
 - home page, webMethods Content Service Platform
 - managing connections 29
- I**
- instances
 - defined 54
 - folder
 - location in repository 56
 - monitoring considerations 30
 - Integration Server
 - changing port number for 32
 - integration with e-forms 14
 - Integration Server Administrator, starting 29
 - IS document type
 - maintaining life of 50
 - synchronizing with e-form templates 50
- L**
- listener components
 - key points 43
 - overview and key points 43
- M**
- mandatory fields, limitations of 69
 - metadata, use in document types 18
 - Microsoft Office InfoPath
 - digital signature considerations 70
 - field limitations 69
 - implementation considerations 68
 - other limitations and information 69
 - root node requirements 69
 - starting processes with e-forms 68
 - My webMethods
 - content integration with 13
 - Content Service Platform, interface for 36
 - environment entries, configuring 46
 - integration with e-forms 56
- O**
- Oracle database, error messages from 19
- P**
- package, WmContentServicePlatform 29
 - PDF file
 - defining a path from an XDP file 64
 - href attribute for 64
 - not needed with an XDP file 65
 - using Acrobat forms 67
 - port numbers, changing 32
 - portlet applications, configuring for CSP repository 46
 - Process Engine
 - content integration with 14
 - defining services for 32
 - integration with e-forms 56
 - process models, requirements for 13, 55
 - processes
 - listener component considerations 43
 - starting with Adobe LiveCycle e-forms 66
 - starting with Microsoft InfoPath e-forms 68
 - use with deployed environments 30
 - using document types with 15
 - using e-forms with 59
- R**
- removing templates and IS document types 50

repository
 network connectivity to 24
 overview of e-forms 56
 stopped, effects of 24

S

services, defining for Content Service Platform 32
 SharePoint text handling 20
 Software AG Designer
 content integration with 14
 integration with e-forms 56, 56
 specification, addConnection 73
 specification, addFile 80
 specification, createContentID 75
 specification, createFilter 76
 specification, createISServerForListener 86
 specification, createListener 77
 specification, createMetadataMap 84
 specification, createSearchTerm 84
 specification, deleteConnection 74
 specification, getFile 81
 specification, listAllContentDefinitions 78
 specification, listAllIndexIDs 78
 specification, listAllISServers 87
 specification, listConnection 74
 specification, retrieveFiles 81
 specification, searchMetadata 85
 specification, updateFile 82
 start-up requirement for CSP 19
 suite integration 13, 56
 supported e-form applications 54

T

task
 applications, configuring for CSP repository 46
 behavior when downloading an e-form 67
 integration with WebReader 20
 using document types with 16
 using e-form with 59
 Task Engine
 content integration with 14
 integration with e-forms 56
 templates
 CSP requirements for 57
 defined 54
 folder, location in repository 56
 maintaining life of 50
 synchronizing with IS document types 50

working with in CSP 57
 XDP files 62
 XSN files 68
 text handling in SharePoint 20

W

webMethods product suite, content integration with 13
 WebReader, CSP 20
 WmContentServicePlatform package 29
 WmContentServicePlatform\pub\csp
 connection folder 72
 content folder 74
 file folder 79
 search folder 83
 server folder 86
 WmContentServicePlatform\pub\csp folder 72

X

XDP
 defining a path to a PDFfile 64
 file limitations 63
 files as e-form templates 62
 without an associated PDF 65
 XML schema, using as a template source with Adobe LiveCycle 65
 XSN files, as e-form templates 68

Symbols

<\$nopage>filters. See content filters 36
 <\$nopage>InfoPath. See Microsoft Office InfoPath 68
 <\$nopage>listener. See content listener 40