Using IBM FileNet with Adobe® LiveCycle® ES

Adobe LiveCycle ES Connector for IBM FileNet (the FileNet Connector) provides seamless integration between LiveCycle ES solution components and your FileNet system. It enables developers to rapidly and easily develop a content-driven transactional or persuasive application. Examples of those applications include new account opening, invoice processing, and litigation support—or industry-specific processes like medical records, mortgage origination, claims processing, or loan approval. By combining LiveCycle ES technologies with the FileNet ECM capability, you can transform your document-intensive business processes into dynamic interactions that enhance customer service, accelerate transaction cycles, reduce document management costs, maximize the value of your investment in your FileNet system, and improve productivity throughout the enterprise.

The FileNet Connector enables developers to manage LiveCycle ES design assets, including forms, fragments, images, and other content assets, that are stored in the IBM FileNet system. It enhances the collaboration and increases efficiency when developing a content-centric application by allowing developers to access those assets directly within LiveCycle Workbench ES.

The FileNet Connector includes service operations to enable LiveCycle ES applications and processes to retrieve the content objects, including forms or documents as well as their properties, directly from the FileNet Content Engine. These service operations also enable LiveCycle ES applications and processes to archive contents, such as user submitted form data and converted PDF files, directly into the FileNet Content Engine, with full control over their properties and relationships. Moreover, the FileNet Connector includes service operations for LiveCycle ES processes to interact with FileNet workflows by allowing developers to visually map values between LiveCycle data and FileNet workflow step parameters.

With service operations provided by the FileNet Connector, you can flow your data directly into other business processes, enable full-text or index search capabilities through the ECM system, meet regulatory and compliance requirements, and let your end users participate in business processes inside or outside of the FileNet system.

This technical guide provides guidance and tips on how to effectively leverage the capabilities that the LiveCycle ES Connector for IBM FileNet offers to build your content-centric applications.
Deployment

Typical deployment
LiveCycle ES Connector for IBM FileNet is a service component hosted by the LiveCycle service container. Therefore, it is deployed along with other solution components in the same LiveCycle ES server or clustering environment. While LiveCycle ES supports many combinations of operating systems, databases, and application servers, you may want to run IBM FileNet Content Manager on a different server using platforms supported by FileNet. The FileNet Connector enables you to deploy your systems in a heterogeneous environment where LiveCycle ES and FileNet Content Manager run on totally different operating systems, databases, and application servers.

Examples: Table 1 shows a few examples of deployment combinations that the FileNet Connector supports. Refer to product documents for LiveCycle ES and IBM FileNet for detailed information on their supported platforms respectively.

<table>
<thead>
<tr>
<th>LiveCycle ES</th>
<th>IBM FileNet Content Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX® 5.3/WebSphere 6.1/DB2</td>
<td>Windows 2003/WebSphere/DB2</td>
</tr>
<tr>
<td>Red Hat® 4.6/WebLogic 9.2/Oracle</td>
<td>Solaris 10/WebLogic 9.2/Oracle</td>
</tr>
<tr>
<td>Solaris™ 10/WebSphere/DB2</td>
<td>Red Hat 4.0/IBoss/Oracle</td>
</tr>
</tbody>
</table>

Table 1. Examples of platform combinations between LiveCycle ES and FileNet.

Choose and configure the FileNet transport protocol
LiveCycle ES Connector for IBM FileNet relies on the FileNet Content Engine Java™ API. In FileNet P8 4.0, the Content Engine supports two basic transport protocols: EJB and Content Engine Web Services (CEWS). The EJB protocol (typically RMI-IIOP-based) is language-independent and passes serialized Java objects across the network for requests and responses to and from the Content Engine server. The Content Engine Java API, configured to run over the EJB transport, is the fastest interface available in FileNet P8 4.0. However the EJB client is typically required to run in the same type of application server as the EJB server does. On the other hand, the CEWS transport to the Content Engine server is based on the SOAP protocol and allows the client to run in a totally different type of application server than the server does.

Tip: By specifying the correct connect URL in LiveCycle Administration Console, you can configure how you want the FileNet Connector to talk to the FileNet Content Engine. Please refer to Table 2 to choose the transport in your environment.

<table>
<thead>
<tr>
<th>LiveCycle ES Connector for IBM FileNet</th>
<th>WebSphere 5.1/6.0</th>
<th>WebSphere 6.1</th>
<th>WebLogic 8.1</th>
<th>WebLogic 9.2</th>
<th>JBoss 4.0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CEWS or EJB</td>
<td>CEWS or EJB</td>
<td>CEWS</td>
<td>CEWS</td>
<td>CEWS</td>
</tr>
<tr>
<td>WebLogic 8.2</td>
<td>CEWS</td>
<td>CEWS or EJB</td>
<td>CEWS</td>
<td>CEWS or EJB</td>
<td>CEWS</td>
</tr>
<tr>
<td>WebLogic 10.x</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>JBoss†</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
</tr>
</tbody>
</table>

† When you use the EJB transport on WebSphere, global security must be turned on in LiveCycle ES.
† The FileNet Connector cannot run on WebLogic 10.x server since the FileNet Content Engine Java API 4.0.1 does not support this version.
† The FileNet Connector cannot run on JBoss server since the FileNet Content Engine Java API 4.0.1 does not support the JBoss version that LiveCycle certifies.

Table 2. Understanding the login format.
User management and authentication

LiveCycle ES Connector for IBM FileNet includes an authentication provider service that you can use when you set up a LiveCycle enterprise domain to authenticate the access of LiveCycle applications, such as Workbench or Workspace, through your FileNet user management. With this, end users will have the same login username and password for both LiveCycle ES and FileNet. If you have a directory server set up for FileNet, you can let LiveCycle ES synchronize with the same directory server.

Examples: In a typical configuration, you create an enterprise domain in the Domain Management settings of LiveCycle Administration Console. In this domain, you add a custom authentication provider—IBM FileNet Authentication Provider (see Figure 1). You may also want to add the directory server that FileNet uses to the same enterprise domain. You can skip the directory server setup if you have only local users who are not in any directory server and will access LiveCycle ES and FileNet. In this case, you need to make sure that you have the exact same credentials (username and password) for the same user in both LiveCycle ES and FileNet. Once you set up the enterprise domain for FileNet, you can log into LiveCycle Workbench ES, or any other LiveCycle application, to access FileNet object stores.

In the configuration setting of the FileNet Connector in LiveCycle Administration Console, you must provide a system username and password for the FileNet Content Engine. This credential is used as the system credential when LiveCycle services or processes require system access and when the property editor of the Content Repository Connector (CRC) operations (see Figure 6 later in this guide) needs to get ECM metadata, such as class types or repositories, from FileNet.

Tip: You can log into Workbench ES and other LiveCycle ES applications in two ways. First, you can log in with a username. This gives you access to the default object store. In Workbench, this lets you see the default object store in the Resources view. Another way to log in is to use the format username@objectstore. This format allows you access to that specific object store. In Workbench, the Resources view shows the folders for that particular object store.
Table 3 shows how login format, authentication provider, and directory server are connected.

<table>
<thead>
<tr>
<th>Login format</th>
<th>Credentials used</th>
<th>Object store in the Resources view or to be accessed</th>
<th>Authentication provider required?</th>
<th>Directory provider required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>LiveCycle super admin user</td>
<td>System credential</td>
<td>Default ObjectStore</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LiveCycle super admin user/ObjectStore1</td>
<td>System credential</td>
<td>ObjectStore1</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Local user</td>
<td>User credential</td>
<td>Default ObjectStore</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>LDAP user</td>
<td>User credential</td>
<td>Default ObjectStore</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Local user/ObjectStore1</td>
<td>User credential</td>
<td>ObjectStore1</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>LDAP user/ObjectStore1</td>
<td>User credential</td>
<td>ObjectStore1</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Note: The system credential is the username and password used in the FileNet Connector configuration in the Administration Console.

Table 3. Understanding the login format.

**Managing LiveCycle ES development assets in FileNet**

LiveCycle ES Connector for IBM FileNet enables developers to access a FileNet object store directly within LiveCycle Workbench ES. This feature is enabled by setting the IBM FileNet Repository Provider as your current repository service provider (see Figure 2).

---

**Figure 2. Choose the repository provider.**
Once you have done this setup, you can log into Workbench and access and manage your LiveCycle assets in the FileNet object store through the Resources view. Depending on how you log in (see Table 3) the designated FileNet object store is shown under the server name. You can browse all folders and content objects within that object store (see Figure 3).

![Figure 3. FileNet objects shown in the Resources view.](image)

From here, you can leverage your FileNet object store when you design and develop your LiveCycle ES application.

Here are some examples of what you can do:

- View and edit your form design, stored in the FileNet object store, using LiveCycle Form Design perspective.
- Create and save your form design directly into the FileNet object store.
- Share your form design with other developers and business owners, abiding by FileNet’s access control.
- Create and use the form fragments library in your FileNet object store.
- Use content assets, such as images or text, created by non Workbench tools from other members when you design your forms.
- Access the FileNet object store in the resource picker, which is used in property setup for some variables and service operations (see Figure 4).
Developing LiveCycle applications with contents in FileNet

When you are developing an application using LiveCycle and FileNet, you have many choices. Each has its pros and cons. You need to consider several factors in order to decide which one to choose:

- Build your own integration layer using FileNet’s Java APIs. For example, you can write your own operation and its property editor to be used in LiveCycle Workbench. Another example is that you can write a set of Java APIs to reflect a content transaction by calling the FileNet API directly.
  - Pros: You will have access to FileNet’s API in its lowest granularity level.
  - Cons: You must maintain a team who has expertise in both LiveCycle and FileNet. You need to build your own LiveCycle components for FileNet so you can visually design and configure them during LiveCycle process and application design.

- Use a repository URL through IBM FileNet Repository Provider. The Repository Provider is one of the major components in LiveCycle ES Connector for IBM FileNet. It enables developers to use a repository URL (repository://) to represent the location in the FileNet repository from which a content object is accessed. You can use the provider with or without the Content Repository Connector or Process Engine Connector. See the “Integrating with FileNet Process Engine” section on page 13 for more information.
  - Pros: You will be able to use many LiveCycle ES data types, such as xfaform, which inherently use a repository URL as their resource property. You can also leverage fragments in your forms as they are also built to use a repository URL. You can use a repository URL to represent form designs or other contents in some of the service operations, such as renderPDFForm.
  - Cons: You can only access the latest version of forms or contents through the repository URL. You will not be able to access or set properties for those contents stored in the FileNet object store.

- Use operations in the Content Repository Connector and/or Process Engine Connector. This is another major component in LiveCycle ES Connector for IBM FileNet. This component provides a set of service operations you can use in Workbench to integrate the FileNet Content Engine and Process Engine into a LiveCycle process or application.
  - Pros: You have full control over how and what you want to access from the FileNet object store. This includes the versioning, the relationship, content properties, and the authentication option. These operations give developers a visual design and configuration on the content services provided by FileNet. You can also visually map LiveCycle data to the FileNet Process Engine step parameters (or vice versa) and enable the FileNet workflow to leverage the rich features that LiveCycle ES provides.
• Cons: You may not have access to all the FileNet functions you want to use, and you may need to augment your application with separate FileNet APIs.

It’s worth noting that you can choose to use the Repository Provider, Content Repository Connector, or Process Engine Connector separately or in any combination to fit your needs.

**Using Repository Provider**

Repository URL syntax

If you use IBM FileNet Repository Provider as your current repository provider, you can use a repository URL to access the latest version of documents in the FileNet object store. The root node of the URL path is the name of the FileNet object store. The rest of the nodes represent folders and objects. For example: /Samples/Forms/SimpleLoanApplication.xdp

In this example, the URL points to a content object that has a document named SimpleLoanApplication.xdp. This object is in the Forms folder in the Samples object store.

**Workspace ES application**

Without worrying about how to get the right assets from FileNet, you can quickly build a Workspace application using a repository URL for the needed resources. These resources are stored in a FileNet object store, but developers do not need to worry about the lower level of integration mechanism between LiveCycle and FileNet.

Tip: If you follow the Creating Your First LiveCycle ES Application tutorial ([www.adobe.com/support/documentation/en/livecycle](http://www.adobe.com/support/documentation/en/livecycle)), the only additional step needed to make the application work with forms in FileNet is to set the current repository provider to IBM FileNet Repository Provider. No code change and no recompilation are needed.

**Content Repository Connector**

LiveCycle ES Connector for IBM FileNet includes a set of service operations to enable developers to visually design and configure the FileNet integration within a LiveCycle process or application. These operations are grouped as the Content Repository Connector as shown in the Services view in Workbench (see Figure 5).

![Figure 5. Service operations in the Content Repository Connector.](image-url)
Each service operation has an extensive property editor to enable developers to control and configure different aspects of the operations. Figure 6 shows the property editor for the Store Content operation.
Authentication in the service operations

You have three authentication choices for each service operation (see Figure 7).

- Use the credential from the LiveCycle process context if this operation is part of a long-lived LiveCycle process, and the operation should be subject to access control from FileNet for the owner of the current activity. This is a typical option for CRC operations in a LiveCycle Workspace application.

- Use the user credential that needs to be input using the username and password fields in the property editor. You may want to use this, for example, when you want all FileNet accesses to be under a single common user representing a system process.

- Use the FileNet session ticket when you want current activity to be executed under the access control list of the current FileNet session. This is a typical option when you call the LiveCycle process through a FileNet client, such as Workplace or the FileNet workflow process.
Control of content versions
In the Store Content operation, you can specify the versioning behavior (see Figure 8).

![Figure 8. Versioning options in the Store Content operation.](image)

Accessing and setting values of content properties
Using service operations, developers have direct access to the properties of FileNet content objects. You have two ways to access the content properties:

- Visual mapping shows properties for an object type when you select the object store name in the property editor and select the object type from that object store (see Figure 9). Once you see the list of content properties, you can associate process variables by opening up the XPath Builder when you click the grid in the value column. In this mode, you need to use the literal value for both Object Store and Document Class Type. You can either select from the drop-down list or type in the literal value of the object store name or object type. This mode is best used when the object store and object type are known and predetermined within the application.

![Figure 9. Visual mapping of content properties.](image)
Dynamic mapping enables developers to access and set content properties based on the object store and object type used during the runtime (see Figure 10). It enables developers to build a versatile process to handle a complex logic based on real-time business rules. In this mode, a process variable of map data type is used to contain the mapping between process variables and content properties. You can use the setValue operation to construct the mapping dynamically.

Example: Figure 11 shows how to set the value of the CurrentStatus field in an XDP form in the formstatus property of a FileNet content object. The first line sets the value of that form field to a process variable. The second line sets the value of the process variable to the value of a key-value pair in a map variable where the key is the name of the FileNet content object property. It is worth noting that you need to put quotes around the expression /process_data/@formCurrentStatus. This is necessary so that the connector operation can interpret it correctly. This example also shows how you can map an XDP form field to a content object property.

Figure 11. Map form field to content property.

Figure 12 shows how to get the value of the content object property—formstatus—and set it into the process variable—formCurrentStatus. The process variable attributeMap is used in the Attribute Name Value Map property of the retrieve content operation.

Figure 12. Set value of a content property to a process variable.
Associating submitted data to the form design

In a typical data capture application, submitted data from a form is stored separately as an XML object in FileNet. This XML object can then be used as part of the other FileNet application or workflow process. To render the submitted XML data correctly, you need to maintain a proper relationship with the form design. You can use Create Relationship (see Figure 13) to establish the relationship between two FileNet content objects, and you have full control of the version behavior of this relationship. The Get Related operation, on the other hand, allows you to get the related documents from the source content object (see Figure 14).

![Figure 13. Create a relationship between two content objects.](image)

![Figure 14. Get a list of IDs of the related content objects.](image)
Classifying form data into the appropriate folders
Starting from ES Service Pack 1, LiveCycle ES Connector for IBM FileNet introduces a service operation for dynamically creating a folder tree in FileNet (see Figure 15). Combining this service operation with other solution component operations, you can develop a process that stores form data in an appropriate folder based on the value of the data. One business example is when customer requests are sent in, a LiveCycle process can extract the data from the submitted form, store the form data in a regional folder based on the zip code that customers filled in on the form, and trigger a process to the right regional call center.

Accessing LiveCycle ES processes within FileNet
Accessing from Workplace
All LiveCycle processes generate, by default, both SOAP and EJB endpoints. You can choose either protocol to call LiveCycle processes from the FileNet Workplace application.

Example: In a typical data capture application, you would browse to the Workplace folder, open a form, fill it out, and submit it back. The submitted form, then, is archived in the appropriate places inside FileNet. One way to implement this is to construct two LiveCycle processes—render service and submit service. The two processes then are exposed as web services or EJB endpoints.

Tip: In FileNet, you can associate a servlet with a MIME type. In the example above, you can create a MIME type for XDP form design, write a servlet to call LiveCycle processes, and associate the MIME type to the servlet. In this servlet, you can pass all the necessary information to the LiveCycle render process, such as Workplace session authentication, form design location, URL for the servlet that is called when a user submits the form, and default form data location.

Integrating with the FileNet Process Engine
Many customers are using the FileNet Process Engine to execute ECM-related workflows. The FileNet workflow consists of steps, resources, and routing logic needed to complete a business process. The FileNet workflow relies on step parameters to pass information from one step to another. LiveCycle ES Connector for IBM FileNet enables you to leverage the workflows defined in the FileNet Process Engine. Through Process Engine Connector service operations, developers can visually establish mappings between LiveCycle process variables and FileNet workflow parameters. Developers can also advance a FileNet workflow step once a LiveCycle process is finished. (Technically, it advances the work object. A step can be assigned to multiple users, and each of them will have the same step but different work object numbers.)
Example: A popular use case from our customers is a review and approval process that runs within the FileNet Process Engine and uses LiveCycle ES to give end users a rich, engaging experience with a popular client platform, such as Adobe Reader® or Adobe Flash® Player software. In this case, contents and data stored in the FileNet Content Engine can be presented to end users in PDF, HTML, or guided Flash form. When users submit the data back, this data is processed by LiveCycle ES and flows back to the FileNet workflow to drive it to different resources and follow-up actions.

Launch the FileNet workflow

Typically a form is used to launch the FileNet workflow. You can use techniques described in the “Accessing from Workplace” section earlier in this guide to designate a servlet and MIME type to call out LiveCycle ES to render the form and trigger the FileNet workflow once the data is submitted back.

Advance the FileNet workflow

One important thing for the FileNet workflow to call out LiveCycle ES processes is to develop a step processor (see Figure 16). This step processor serves the same role as the servlet we mentioned earlier to execute the LiveCycle ES processes through endpoints, such as SOAP or EJB. Developers can use operations provided as part of the Process Engine Connector to get the step parameters, use the values in other LiveCycle activities, set the values back to step parameters in the end, and advance the FileNet workflow step so that the generated data flows back into the FileNet workflow and drives the follow-up actions.

Figure 16. A custom step processor calling LiveCycle services.
Example: Figure 17 shows a typical LiveCycle process participating in a FileNet workflow step to process the data submitted from end users.

![Diagram](image-url)
Accessing and setting values of FileNet step attributes

Step parameters serve the important role of carrying the information within the FileNet workflow. Process variables serve a similar role within a LiveCycle process. LiveCycle ES Connector for IBM FileNet includes operations to allow you to visually link the process variables to the step parameters.

Both operations—Retrieve Workflow Step Parameters and Set Workflow Step Parameters—have similar property editors (see Figure 18) to allow developers to access the FileNet workflow step parameters within LiveCycle Workbench ES.

Figure 18. Access FileNet workflow step parameters.
Example: Figure 19 shows how you can set up a map variable in LiveCycle to get the values of step parameters defined in FileNet (see Figure 20).

Figure 19. Set up a map variable for accessing FileNet workflow step parameters.

Figure 20. A sample FileNet workflow.

Tip: You can use these techniques to map values of data in a LiveCycle form directly to a step parameter in FileNet.
Conclusion
LiveCycle ES Connector for IBM FileNet provides a unified development experience by extending LiveCycle Workbench ES to access and manage content assets stored in the FileNet content server. Developers can create, modify, and store forms and other contents in FileNet directly. It makes development collaboration much easier than before.

The FileNet Connector also includes service operations to help developers build a robust, content-centric application. To provide a rapid development environment, the FileNet Connector provides basic content repository services, such as content retrieval, content storage, content relationship, and content properties access, through service operations. Without any coding, developers can easily set and retrieve values between the properties of a FileNet content object and LiveCycle ES assets attributes, such as form fields in Adobe LiveCycle Forms ES control attributes of LiveCycle Reader Extensions ES, or properties of LiveCycle PDF Generator ES.

In addition, the FileNet Connector provides operations to enable developers to conveniently design applications to leverage both LiveCycle ES and FileNet, including its Process Engine. Whether you want to use FileNet as a content repository or as a workflow engine to execute ECM-related processes, LiveCycle ES Connector for IBM FileNet can help you quickly build a robust application to enhance the engagement experiences of your end users.