NETVIBES | Exalead CloudView

CloudView CV23

ENOVIA ER Connector

Table of Contents

ENOVIA ER Connector....4
What's New....6
Supported Features....7
Incremental Indexing....7
File Filtering....8
CAS Authentication....8
Security Management....8
Logging....8
Deploying the Connector Plugin....9
Deploy the Plugin....9
Update the Plugin from the Administration Console....9
Update the Plugin from the Command Line....9
Configuring the Connectivity....10
Connectivity Modes....10
Connect to the 3DSpace....10
Connect to ENOVIA Central in HTTP REST Mode....11
Connect to ENOVIA Central in Local Context Mode (DMP)....11
Configuring the Connector in 3DSpace Mode....12
Configure the 3DSpace Mode....12
Filter Users Who Can Crawl the 3DSpace....13
Configuring the Connector in HTTP REST Mode....14
Configuring the Connector in Local Context Mode....15
Bind Exalead CloudView with ENOVIA DMP Automatically....16
Set up the Environment Variables....17
Importing a Legacy XML Configuration....21
Controlling the Connector....22
Control Document Scans....22
Adding an Event Monitor for Incremental Indexing....22
Configuring Security....24
Configure the Connector Security Parameters....24
Configure a Security Source....25
Define a Specific Aggregation Processor for Security V2....26
Appendix - Installing the ER Agent on ENOVIA....27
About Possible Architectures....27
3DSpace Architecture....27
Standalone Agent Architecture....28
Installing the ENOVIA Standalone ERAgent....28
Standalone ERAgent Security Recommendations....32
Restrict the Access to the Standalone ERAgent Port....32
Configure a Reverse Proxy with HTTPS....32
Appendix - Configuration Parameters....33
Connection Parameters....33
Query Parameters....34

Processing Parameters..... 36
Security Parameters..... 36
Advanced Parameters..... 37
Advanced Parameters Not in the UI..... 42

ENOVIA ER Connector

The Exalead CloudView ENOVIA ER connector offers several connection modes to connect and retrieve data from ENOVIA.

This connector requires a specific license. This license is included by default if you bought Exalead CloudView from the 3D#EXPERIENCE portfolio. You must purchase it additionally otherwise.

Note: If you used the old ENOVIA SBA connector, you can reuse its license for the ENOVIA ER connector, no need to purchase a new one.

You can use a single ENOVIA ER connector to manage multiple ENOVIA instances.

Audience

This document explains how to configure and deploy Exalead CloudView ENOVIA ER connector. It is assumed that the reader has working knowledge of:

- The operating system on which the Exalead CloudView server and connectors are installed.
- The ENOVIA source structure.

This document assumes that the reader is familiar with the Exalead CloudView Administration Console.

Compatibility

The ENOVIA ER connector is compatible with all ENOVIA versions from V6R2015x up to CV23. For compatibility reasons, this connector still supports the connectivity mode of the former ENOVIA SBA connector.

What Does It Index?

The ENOVIA ER connector enables Exalead CloudView to access the following:

- The ENOVIA Matrix database, to index Business Objects and Relationships.
- The ENOVIA File Collaboration Server (FCS), to index files referenced by the Business Objects.

What Is the Difference with the ENOVIA SBA Connector?

With the ENOVIA ER connector, you can manage the configuration through the connector UI, or through an imported configuration file. With the former ENOVIA SBA connector, a part of the query configuration had to be made in an XML file (ENOVIA attributes).

Where Can I Find OnePart-specific Information?

This guide applies to the Exalead CloudView product. If you want to use the ENOVIA ER connector in OnePart, from the User Assistance Table of Contents, go to .

What's New

There are no enhancements in this release.

Supported Features

The ENOVIA ER connector can crawl both Business Objects and relationships, offering a full Search experience, with thumbnails and preview. It also handles multiple-query processing.

This section describes the following supported features:

Incremental Indexing

The ENOVIA ER connector, supports two types of incremental indexing.

Important: Incremental indexing types are exclusive. If you choose to use one type, you cannot use the other.

Incremental Type	Description
By modification date	The ENOVIA ER connector identifies new or modified documents using the modification date in ENOVIA. These documents are timestamp-based on `modified` fields. **Important:** ENOVIA does not support time zones. Configure both the connector and ENOVIA with the same time zone to ensure that this comparison is valid. For deleted documents, the connector supports URIs with ID fields only (unique in a single ENOVIA), and not physical IDs (unique for all ENOVIA). **Note:** You can also use the internal ENOVIA "trace delete" feature to test incremental indexing before production.
By using an Event Monitor	The Event Monitor detects added/modified/deleted elements, and send their IDs for Exalead CloudView updates. The Event Monitor can delete objects indexed by URIs with both Id and physicalId fields. To use an Event Monitor, specify its name in the connector configuration. For more information, see Adding an Event Monitor for Incremental Indexing. This feature is compatible with ENOVIA versions from R2020x.

File Filtering

Under **Configuration > Advanced > File container handler**, you can use the **Global file filter** parameters:

- **Max. file size** to avoid indexing large files that can impact the performance.
- **Trace reason** to know why files have been included/excluded.

CAS Authentication

In 3DSpace connectivity mode, the ENOVIA ER connector supports CAS authentication for indexing only (the Search is not CAS-authentified).

All applications secured by CAS are accessible by the ENOVIA ER connector in **HTTP REST** mode.

Security Management

For non-standalone agent modes, the security level is the same as the:

- 3DSpace security for 3DSpace connectivity mode.
- Legacy ENOVIA SBA Connector for legacy modes (Local Context).

For more information, see Standalone ERAgent Security Recommendations.

Logging

The following information is available in the connector log:

- PARTIAL and COMPLETE checkpoints used for incremental indexing
- Number of indexed Business Objects
- Name and size of each file linked to a Business Object
- ID of each indexed Business Object
- Any MQL or FCS errors

You can use the **Advanced > Verbose** option for debugging the connectivity but it logs everything.

Deploying the Connector Plugin

You can deploy the connector plugin either on a Exalead CloudViewprimary or secondary server instance, or on the ENOVIA server.

Recommendation: Deploy on a dedicated server for best performance.

Deploy the Plugin

1. Open the Administration Console.
2. In **Deployment > Plugins**, upload the `enovia-er-connector-plugin-v6.zip` file.

   The ENOVIA ER connector type is now available in the Administration Console.

You are now ready to add an ENOVIA ER connector instance.

Update the Plugin from the Administration Console

1. Go to **Deployment > Plugins**.
2. Delete the `enovia-er-connector-plugin-v6` from the list of installed Custom plugins.
3. Drag the new plugin zip file onto the dedicated zone.

Update the Plugin from the Command Line

This procedure describes how to upgrade an existing installation using the `cvadmin` command-line tool.

1. Go to the `<DATADIR>/bin` directory and run the `cvadmin` tool for plugins: `./cvadmin plugins`
2. To retrieve the list of installed components, run: `list`
3. Uninstall the previous version of the connector using the installed component named from Step 2, by running the command, for example:

   ```
   remove name=enovia-er-connector
   ```
4. Install the new component:

   ```
   install file=<PATH TO enovia-er-connector-plugin-v6.zip>
   ```

Configuring the Connectivity

The creation of an ENOVIA ER connector in Exalead CloudView requires to add a new connector instance and configure it depending on the connectivity mode you want to apply.

Connectivity Modes

Configuring the Connector in 3DSpace Mode

Configuring the Connector in HTTP REST Mode

Configuring the Connector in Local Context Mode

Importing a Legacy XML Configuration

Connectivity Modes

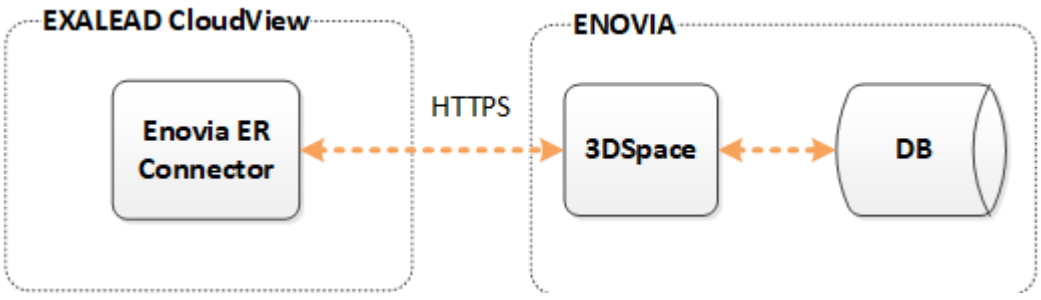
The ENOVIA ER connector succeeds the ENOVIA SBA connector. V6R2018x introduced a brand new ENOVIA ER configuration for this connector to connect to the 3DSpace, but we ensure compatibility with previous connectivity modes.

You can select a connectivity mode in the Exalead CloudView Administration Console, using the **Connectors > YOUR ENOVIA ER CONNECTOR > Configuration > Connection parameters > Connectivity mode** parameter.

Connect to the 3DSpace

From Exalead CloudView V6R2017x FD02, you can connect to the 3DSpace interfacing the ENOVIA database directly, through HTTP.

Recommendation: Use this new connectivity mode as it is compatible with the 3D#EXPERIENCE platform.



This mode supports CAS authentication and file indexing.

To deploy it:

1. ENOVIA side - For this connectivity mode the ERAgent is a built-in feature directly deployed into the 3DSpace application.

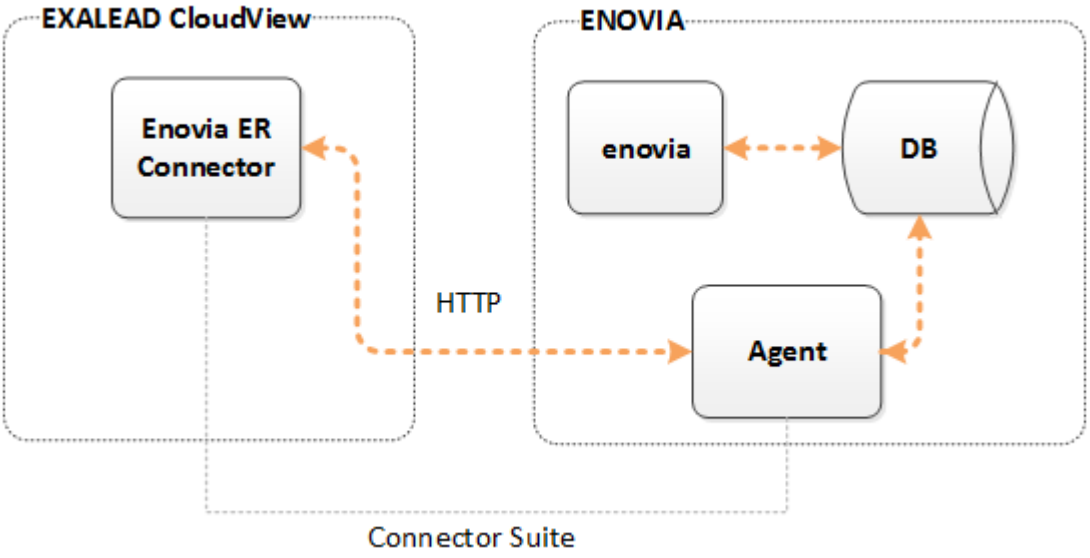
   Important: You do NOT need to deploy it.

2. Exalead CloudView side - See Configuring the Connector in 3DSpace Mode.

Connect to ENOVIA Central in HTTP REST Mode

For all Exalead CloudView versions, you can connect to an ENOVIA central called Studio Modeling Native Apps, in REST mode through HTTP REST.

It requires installing a specific agent on the ENOVIA server.



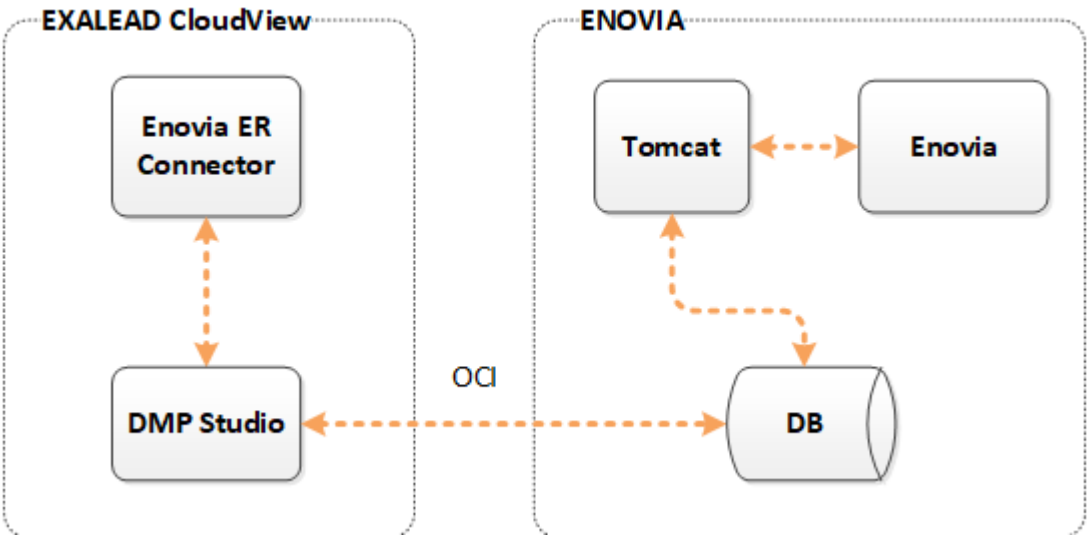
How to deploy it:

1. ENOVIA side - For this connectivity mode, you need to deploy a Standalone Agent. See Appendix - Installing the ER Agent on ENOVIA.
2. Exalead CloudView side - See Configuring the Connector in HTTP REST Mode.

Connect to ENOVIA Central in Local Context Mode (DMP)

Important: This is a legacy mode first implemented for the ENOVIA SBA connector. Use the 3DSpace connectivity mode.

For all Exalead CloudView versions, you can connect to an Exalead CloudView central called Studio Modeling Native Apps, using the Local Context mode.



To deploy it:

1. ENOVIA side - This connectivity mode uses the DMP studio agent through OCI, when ENOVIA is installed locally on the Exalead CloudView connector server.
2. Exalead CloudView side - See Configuring the Connector in Local Context Mode.

Configuring the Connector in 3DSpace Mode

This is the recommended connectivity mode. It does not require any ENOVIA jars (nor JPOs jars).

Configure the 3DSpace Mode

- Get the 3DSpace application URL.
- Make sure that the 3DSpace version is the same as the ER connector version.
- You must have an ENOVIA login and password. If using a CAS-secured 3DSpace, you need a CAS account and a role for crawling.
- Check that the ER Agent is up and running. Open the following URL:

  `https://<ER_AGENT_HOST>/3DSpace/enovia-agent/login?login=<USER_LOGIN>&password=<US`

  If you get a CAS error, this means that the agent is running. If you get a `404 Page Not Found Error`, the agent is not available in your 3DSpace installation.
- If you have a previous installation of the Legacy ENOVIA SBA Connector, remove all ENOVIA jars (including JPO jars) from the Exalead CloudView `javabin` directory and restart the Exalead CloudView instance.

- If using HTTPS with self-signed certificate, install the certificate into the Exalead CloudView java trust store before configuring the connector. For more information, see the Exalead CloudView Installation Guide.

1. Open the Administration Console.
2. Go to **Connectors** and click **Add connector**:
   a. Enter a descriptive name for the connector.
   b. From **Type**, select the **ENOVIA ER** connector.
   c. Click **Accept**.

   The connector **Configuration** tab opens.
3. Expand **Connection parameters** and for:
   a. **Component class name**, select **On premise**.
   b. **Connectivity mode**, select **3DSpace**.
   c. **URL**, enter: `https://<3DSPace_HOST>/3DSpace/enovia-agent`
   d. **Login**, enter the crawling account login.
   e. **Password**, enter the crawling account password.
4. Click **Check connectivity** to run the connector's connectivity test.

   The first **Check connectivity** action can take up to 5 minutes (succeeding ones do not take less time). If the connection is successful, a green "OK" message displays. Otherwise, a message in red indicates what is wrong.
5. Click **Save** and **Apply**.

You are now ready to index. See Controlling the Connector.

Filter Users Who Can Crawl the 3DSpace

You must specify the users allowed to connect and crawl the 3DSpace using CAS Authentication, by specifying grant and deny rules.

1. Go to your 3DSpace `apache-tomcat/bin/config` directory and edit the `ERAgentRestAPICASSecurity.xml` file.
2. Uncomment and configure the grant or deny rules that you need.

```
<-- By default, all users are denied -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <casSecurityConfig>
    <denies>
      <all>true</all>
      <users/>
      <assignments/>
    </denies>
```

```
  </casSecurityConfig>

<!-- Config sample: To grant a user with crawl permissions, uncomment the followin
In this example, the user with login VPLMAdminUser will be the only one allowed to
<!--
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <casSecurityConfig>
    <grants>
      <all>false</all>
      <users>
        <login>VPLMAdminUser</login>
      </users>
      <assignments/>
    </grants>
  </casSecurityConfig>
-->
```

Configuring the Connector in HTTP REST Mode

This connectivity mode requires the deployment of a Standalone Agent on ENOVIA Central. It does not require any ENOVIA jars (nor JPOs jars).

- The **Standalone Agent must be deployed** on ENOVIA Central. See Appendix - Installing the ER Agent on ENOVIA.
- Get the ER Agent URL.
- Check that the ER Agent is up and running. Open the following URL:

  `https://<ER_AGENT_HOST>/enovia-agent/login?login=<USER_LOGIN>&password=<USER_PASSW`

  If you get an error, this means that the agent is not running or badly configured.
- You must have an ENOVIA login and password. To use the Standalone ERAgent, you need an internal account, for example, `creator`.
- If you have a previous installation of the Legacy ENOVIA SBA Connector, remove all ENOVIA jars (including JPO jars) from the Exalead CloudView `javabin` directory and restart the Exalead CloudView instance.
- If using HTTPS with self-signed certificate, install the certificate into the Exalead CloudView java trust store before configuring the connector. For more information, see the Exalead CloudView Installation Guide.

1. Register your ENOVIA security certificate:
   a. Connect to your ENOVIA server and retrieve the security certificate from your Web browser.
   b. Connect to your Exalead CloudView host and import the e4all certificate on your trusted store.

In your Exalead CloudView `<INSTALLDIR>`, run the following command:

```
<INSTALLDIR>\amd64-win64\java-jre\bin\keytool.exe -importcert
 -file passport_file -alias passport.cert -keystore dataDir/security/trusted.se
```

When asked for the trusted store password, enter `exalead`.

When asked if you really want to import the certificate, enter `Yes`.

2. Open the Administration Console.
3. Go to **Connectors** and click **Add connector**:
   a. Enter a descriptive name for the connector.
   b. From **Type**, select the **ENOVIA ER** connector.
   c. Click **Accept**.

   The connector **Configuration** tab opens.
4. Expand **Connection parameters** and for:
   a. **Component class name**, select **On premise**.
   b. **Connectivity mode**, select **REST**.
   c. **URL**, enter: `http(s)://<ER_AGENT_HOST>:<ER_AGENT_PORT>/enovia-agent`
   d. **Login**, enter the crawling account login.
   e. **Password**, enter the crawling account password.
5. Click **Check connectivity** to run the connector's connectivity test.

   If the connection is successful, a green "OK" message displays. Otherwise, a message in red indicates what is wrong.
6. Expand **Query parameters** and specify the Business Objects to index.
7. Click **Save** and **Apply**.

You are now ready to index. See Controlling the Connector.

Configuring the Connector in Local Context Mode

In Local Context mode, the ENOVIA ER Connector requires a local installation of Studio Modeling Native Apps (DMP) or LiveCollaborationServer.

Important: This mode is supported to simplify the legacy ENOVIA SBA Connector migration. In that case, the connectivity only requires to provide login / password, as we used to do with the ENOVIA SBA Connector.

To successfully deploy the ENOVIA connector locally:

- Install Exalead CloudView V6R2016x or higher.

- Install ENOVIAStudio Modeling Platform native apps on the same host as the Exalead CloudView platform. Depending on your Studio Modeling Platform version, you may need to switch to 32-bit. You must be able to execute the following MQL command and create a context without any error:

```
./mql -c "push context user USER [password PASSWORD]
```

1. Open the Administration Console.
2. Go to **Connectors** and click **Add connector**:
   a. Enter a descriptive name for the connector.
   b. From **Type**, select the **ENOVIA ER** connector.
   c. Click **Accept**.

   The connector **Configuration** tab opens.
3. Expand **Connection parameters** and for:
   a. **Component class name**, select **On premise**.
   b. **Connectivity mode**, select **Local Context**.
   c. **URL**, leave the field empty.
   d. **Login**, enter the same admin login that you set up for the ENOVIA SBA Connector.
   e. **Password**, enter the same admin password that you set up for the ENOVIA SBA Connector.
   f. **Vault**, enter the name of the vault you need to index.
4. Click **Check connectivity** to run the connector's connectivity test.

   If the connection is successful, a green "OK" message displays. Otherwise, a message in red indicates what is wrong.
5. Click **Save** and **Apply**.

Bind Exalead CloudView with ENOVIA DMP Automatically

The **Bind CloudView to ENOVIA DMP** action forces the connection through a local Matrix context. The **URL** option is then only used to build document public URL links at search time.

1. Go to the ENOVIA ER Connector **Operation** tab**Connectors** and click **More actions**.
2. In the **Bind CloudView with ENOVIA DMP** section:
   a. For **Matrix home**, enter the path to your Studio Modeling Platform installation.
   b. For **Matrix version**, specify the installation version.
   c. (Optional) For **DSLS config**, specify the ENOVIA license file full location, if it is located in a custom path.
   d. Click **Run**.

- The ENOVIA Studio Modeling Native Apps JAR files are copied from the `<OS>/docs/javaserver/` directory to the `CloudView/<DATADIR>/javabin` folder.
- In the `<DATADIR>/config/DeploymentInternal.xml` file, the `gateway` section is completed as follows automatically:

```
<envOverride>
  <KeyValue xmlns="exa:exa.bee" key="MATRIXHOME" value="[path]"> </KeyValue>
  <KeyValue xmlns="exa:exa.bee" key="TNS_ADMIN" value="[path]"> </KeyValue>
  <KeyValue xmlns="exa:exa.bee" key="DSLS_CONFIG" value="[path]"> </KeyValue>
</envOverride>
<envAppend> </envAppend>
<envPrepend>
  <KeyValue xmlns="exa:exa.bee" key="PATH" value="[path]"> </KeyValue>
</envPrepend>
```

3. There are now two possibilities:
   - If the connector is crawling ENOVIA from a Web App **URL**, there is nothing to do. The connectors-server process hosting the connector automatically restarted and you are ready to test the ENOVIA connection.
   - If you left the Web App **URL** empty, you must restart the product.

You can now set up the environment variables.

Set up the Environment Variables

1. Open the Exalead CloudViewAPI Console at http://<HOSTNAME>:<BASEPORT+1>/api-ui
2. Click **Manage > Configuration > setCloudviewDeploymentInternalConfig** to add environment variables in the `DeploymentInternal.xml` for `name="connectors.*"` and `name="gateway"`.

   For example:

```
<envOverride>
  <KeyValue xmlns="exa:exa.bee" key="MATRIXHOME" value="/data/ev6apps/V6R2011x/stu
  <KeyValue xmlns="exa:exa.bee" key="GALAXYHOME" value="/data/ev6apps/V6R2011x/stu
  <KeyValue xmlns="exa:exa.bee" key="TNS_ADMIN" value="/data/ev6apps/common/" />
  <KeyValue xmlns="exa:exa.bee" key="DSLS_CONFIG" value="/data/ev6apps/common/DSLi
</envOverride>
<envAppend>
</envAppend>
<envPrepend>
  <KeyValue xmlns="exa:exa.bee" key="LD_LIBRARY_PATH"
 value="/data/ev6apps/V6R2011x/studio/linux_a64/code/bin:/data/ev6apps/V6R2011x/st
</envPrepend>
```

   See also the code samples in Set up the Environment Variables.

3. Click **Save** and then **Apply**.
4. Restart Exalead CloudView.

You are now ready to:

- Add an ENOVIA connector in the Administration Console. Make sure that the **URL** configuration parameter is empty.
- Import your ENOVIA SBA connector legacy XML configuration.

Example 1. Environment Variables

Define the following environment variables on the Java connector server hosting the ENOVIA connector.

Use the following table to determine the environment variables to add to the `<DATADIR>/config/DeploymentInternal.xml` file.

Environment variable	Context	Action	Description
`MATRIXHOME`		override	Path to ENOVIA Studio
`GALAXYHOME`	Linux with ENOVIA versions before V6R2011x only	override	Path to the ENOVIA Studio `/galaxy/linux` folder
`DB2INSTANCE`	IBM DB2 only	override	DB2 instance name
`TNS_ADMIN`	Oracle only	override	Override the path to the folder containing the `tnsnames.ora` file. This variable is not required if you are not referencing `tnsnames.ora` entries in your `MATRIX-R` file.
`LD_LIBRARY_PA`	Linux only	prepend	Add the following paths to the variable: • `<path_to_studio>\<OS>\code\bin` • `<path_to_studio>\<OS>\code\lib` You may also need to add the path to the SQL client for your database.
`PATH`	Windows only	prepend	Add the following paths to the variable: • `<path_to_studio>\win_b64\code\bin` • `<path_to_studio>\win_b64\code\lib`

Environment variable	Context	Action	Description
			You may also need to add the path to the SQL client for your database. For example, for DB2: `C:\DATA\DB2\IBM\SQLLIB\BIN`
`DSLS_CONFIG`		override	In case the `DSLicSrv.txt` license file is not installed in the default location, specify the absolute path to the folder containing the file, including the file_name. Default locations are: • Windows: `C:\ProgramData\DassaultSystemes\Licenses\DSLicSrv.txt` • Linux: `/var/DassaultSystemes/Licenses/DSLicSrv.txt`

The following samples are for an ENOVIA environment:

- Linux 64-bit
- Bound to an ENOVIA installation
- Custom license server config file location

Where `name="connectors.*"` matches all java connectors

```
<ProcessInternalConfig xmlns="exa:exa.bee.deploy.v10"
name="connectors.*" runtime="java" httpMinThreads="5"
httpMaxThreads="500" httpMaxStopTime="100" httpMaxGETQuerySize="32768"
httpMaxFormPOSTQuerySize="10000000" httpMaxIdleTimeMS="86400000"
httpAcceptorThreads="1" httpAcceptQueueSize="1200">
  <params>
  </params>
  <args>
    <StringValue xmlns="exa:exa.bee" value="-Dfile.encoding=UTF-8" />
    <StringValue xmlns="exa:exa.bee" value="-XX:+UseConcMarkSweepGC" />
    <StringValue xmlns="exa:exa.bee" value="-Xmx2048m" />
    <StringValue xmlns="exa:exa.bee" value="-Xms48m" />
    <StringValue xmlns="exa:exa.bee" value="-XX:MaxPermSize=128m" />
    <StringValue xmlns="exa:exa.bee" value="-XX:PermSize=48m" />
    <StringValue xmlns="exa:exa.bee" value="-Djava.awt.headless=true" />
    <StringValue xmlns="exa:exa.bee" value="-Dmmap_thrshold=262144" />
    <StringValue xmlns="exa:exa.bee" value="-Xmx512m" />
  </args>
  <envOverride>
    <KeyValue xmlns="exa:exa.bee" key="MATRIXHOME" value="/data/ev6apps/V6R2011x/stud
```

```
    <KeyValue xmlns="exa:exa.bee" key="GALAXYHOME" value="/data/ev6apps/V6R2011x/stud
    <KeyValue xmlns="exa:exa.bee" key="TNS_ADMIN" value="/data/ev6apps/common/" />
    <KeyValue xmlns="exa:exa.bee" key="DSLS_CONFIG" value="/data/ev6apps/common/DSLic
  </envOverride>
  <envAppend>
  </envAppend>
  <envPrepend>
    <KeyValue xmlns="exa:exa.bee" key="LD_LIBRARY_PATH"
 value="/data/ev6apps/V6R2011x/studio/linux_a64/code/bin:/data/ev6apps/V6R2011x/studi
  </envPrepend>
</ProcessInternalConfig>
...
```

Where `name="gateway"` for the security source

```
<ProcessInternalConfig xmlns="exa:exa.bee.deploy.v10"
 name="gateway" runtime="java" httpMinThreads="5"
 httpMaxThreads="500" httpMaxStopTime="100" httpMaxGETQuerySize="32768"
 httpMaxFormPOSTQuerySize="10000000" httpMaxIdleTimeMS="86400000"
 httpAcceptorThreads="1" httpAcceptQueueSize="1200">
  <params>
  </params>
  <args>
    <StringValue xmlns="exa:exa.bee" value="-Dfile.encoding=UTF-8" />
 <StringValue xmlns="exa:exa.bee" value="-XX:+UseConcMarkSweepGC" />
 <StringValue xmlns="exa:exa.bee" value="-Xmx2048m" />
 <StringValue xmlns="exa:exa.bee" value="-Xms48m" />
 <StringValue xmlns="exa:exa.bee" value="-XX:MaxPermSize=128m" />
 <StringValue xmlns="exa:exa.bee" value="-XX:PermSize=48m" />
 <StringValue xmlns="exa:exa.bee" value="-Djava.awt.headless=true" />
 <StringValue xmlns="exa:exa.bee" value="-Dmmap_thrshold=262144" />
 <StringValue xmlns="exa:exa.bee" value="-Xmx512m" />
  </args>
  <envOverride>
    <KeyValue xmlns="exa:exa.bee" key="MATRIXHOME" value="/data/ev6apps/V6R2011x/stud
    <KeyValue xmlns="exa:exa.bee" key="GALAXYHOME"
 value="/data/ev6apps/V6R2011x/studio/linux_a64/code/bin" />
    <KeyValue xmlns="exa:exa.bee" key="TNS_ADMIN" value="/data/ev6apps/common/" />
    <KeyValue xmlns="exa:exa.bee" key="DSLS_CONFIG"
 value="/data/ev6apps/common/DSLicSrv.txt" />
  </envOverride>
  <envAppend>
  </envAppend>
  <envPrepend>
    <KeyValue xmlns="exa:exa.bee" key="LD_LIBRARY_PATH"
value="/data/ev6apps/V6R2011x/studio/linux_a64/code/bin:/data/ev6apps/V6R2011x/studio
  </envPrepend>
</ProcessInternalConfig>
...
```

Important: This sample ENOVIA configuration must appear before the standard java process configuration in the `DeploymentInternal.xml` file. This is because the first "internal process configuration" that matches the process is used.

Importing a Legacy XML Configuration

This section describes what you can do if you previously used the former ENOVIA SBA connector and want to migrate the connector configuration from it.

This is useful to avoid recreating the same configuration for the new connector. It reuses the XML Configuration to populate the new ENOVIA ER configuration with the same set of Business Objects and Relationships.

1. Open your ENOVIA ER connector.
2. Go to **Operations** and click **More actions**.
3. In the **Import legacy XML connector config** section:
   a. For **File path**, browse to find your legacy xml file.
   b. Click **Run**.

   If you expand the **Configuration > Query parameters** section, you have a set of "Business Object queries" and "Relationship queries" corresponding to the ones you had in your `config.xml`.
4. Click **Save** and **Apply**.
5. Go to the **Home** page > **Connectors** section, and click **Scan** for your ENOVIA ER connector.

Controlling the Connector

The ENOVIA ER connector can crawl both Business Objects and relationships, offering a full Search experience, with thumbnails and preview. It also handles multiple-query processing.

Control Document Scans

In the Exalead CloudView Administration Console, the **Home > Connectors** section provides status information and the following actions for managed connectors.

Note: You can also find these actions under **Index > Connectors > YOUR CONNECTOR > Operation**.

Action	Description
Scan	Performs the scan for this connector, full or incremental, depending on connector configuration and previous scan operations. Scan requests are processed one after the other. When a scan request is launched while another is already being processed, it is added to a queue persisted to disk, and started once the current scan is finished.
Abort scan	Stops the scan for a specific connector. **Note:** It does not delete the documents already processed.
Clear documents	Deletes all connector documents from the system and resets the connector state.
More actions	Most connectors include more actions than **Scan**, **Abort scan** and **Clear documents**. For example, you can have actions such as **Force rescan**.

Adding an Event Monitor for Incremental Indexing

Optionally, you can manage incremental indexing through an Event monitor. If you do not, the connector uses the default incremental scan based on dates.

The Event Monitor detects added/modified/deleted elements, and sends their IDs for Exalead CloudView updates.

This feature is compatible with ENOVIA versions from R2020x.

To add an Event Monitor:

1. Configure the ENOVIA ER connector connectivity mode. For more information, see Configuring the Connectivity.
2. From the **Configuration** tab, expand **Connection parameters**.
3. In **Event monitor name**, enter the name of the Event Monitor you want to create.
4. Select the **Has write access** option.
5. Click **Apply**.

The Event Monitor is created in the ENOVIA database.

Note: In **Operation > More actions > Manage event monitor**, you can also select **Delete** to delete the Event Monitor from the database, or **Clear** to purge its events from the database.

Configuring Security

During indexing, Exalead CloudView indexes the document along with all associated roles and ACLs. At search time, when the user connects to the search application, Exalead CloudView queries ENOVIA to retrieve its roles. When a document matching the user request is found, Exalead CloudView verifies that at least one of the user's roles or the user's UID is in the access control list (ACL) of the index.

Configure the Connector Security Parameters

First select a security mode at the connector level.

It assumes that you have already configured your connector according to the required connectivity mode.

1. Open your ER Connector instance.
2. In the **Configuration** tab, expand **Security parameters**.
3. Click **Add item**, and from **Component class name** select a security mode.

Connector Security Modes and Parameters

Mode	Description
Default Security	This mode allows the definition of the two following parameters only. **Note:** These parameters are common to all security modes, except for **Project Based Security**. • **All BusinessObjects visible by default**: If no security token is generated, this parameter adds the Everybody token on all BOs. It makes documents visible by everyone. If a security handler generates a security token, this parameter has no effect. • **All Connections visible by default**: If no security token is generated, this parameter adds the `Everybody` token on all Connection documents. It makes documents visible by everyone. If a security handler generates a security token, this parameter has no effect.
ENOVIA **SBA Security**	This mode comes from the legacy SBA connector. It is based on the ENOVIA policy security model. See Standalone ERAgent Security Recommendations.

Mode	Description
Project Based Security	A simple security, based on the `project` attribute. Users can only see the documents that are in their projects (that is, Collaborative Spaces, for the 3DSpace). To distinguish Collaborative Spaces that may have the same name in different 3DSpaces, you can specify a **prefix** to apply to security tokens.
Security V2	• **Access list URI**: Choose whether to build AccessList URI compatible with the SBA mode, for version V6R2018x and higher. • **Prefix**: Specify a URI prefix when you want to index data with the same Consolidation Server as another ENOVIA connector. • **(Required) Link with business objects**: Builds a graph out of Access Lists.

Configure a Security Source

A security source is provided with the ENOVIA ER connector. Whenever a user authenticates to Exalead CloudView, the security source verifies the groups this user belongs to.

1. Open the Exalead CloudView Administration Console.
2. Go to **Access > Security Sources** and click **Add security source**.
   a. For **Name**, enter the security source name, for example `ersecurity`.
   b. For **Type**, select **ER Connector**.
   c. Click **Accept**.
3. Configure the security source properties.

Property	Description
Agent URL	The ENOVIA ER agent URL: `http://<enoviaer_host>:<PORT NUMBER/enovia-agent` **Note:** If empty, the security source uses the local context.
Login	ER client pool login. It must be the same login as the ER connector login.
Password	ER client pool password. It must be the same password as the ER connector password.

Property	Description
Security mode	Select the same security mode that you applied to the connector (see Configure the Connector Security Parameters): • **SBA Connector**. • **Project Based**. • **Access List**. If you select this mode, proceed with Define a Specific Aggregation Processor for Security V2.
Ldap users	Uses LDAP users. Passwords are not taken into account in this case, as ENOVIA does not know the user passwords. If this option is not selected, LDAP users who try to connect will get error messages.
Space prefix	For project-based security only, specify the same prefix that you defined for the connector.

4. As administrator or super-user, you can test user authentication on the ENOVIA server.
   a. In **Test user authentication**, click **Test**.
   b. Enter a user login and password (do not use the admin account credentials), and then click **Test**.

      If successful, the user ID, display name, and list of security tokens displays. If not, verify the parameters and contact your system administrator.

Define a Specific Aggregation Processor for Security V2

To use the Security V2 mode, install an aggregation processor to manage ACLs.

1. Go to **Index > Consolidation**.
2. Add an **aggregation** processor:
   a. Select **Java** as format.
   b. For **Name**, enter the name of your choice.
   c. For processor, select **Plugin enovia-er-connector-plugin-v6 > ACLs Security Aggregator**.
   d. Click **Accept**.
3. Choose the security level to apply on ACLs.

   You can choose to keep all positive ACLs (default option with all **yes level authorized** check boxes selected), or select more precisely to keep **Yes level**, **Maybe yes level**, or **Maybe yes local level**.

Appendix - Installing the ER Agent on ENOVIA

The aim of the ER Agent is to provide an indexing API for ENOVIA.

This API:

- Supports both native (DMP, Server) connectivity and REST remote connectivity.
- Is independent from the connectivity mode, that is to say that the connector does not handle content differently if the content is local or remote.
- Is read-only, which means that the connector, whether in native or REST mode, cannot write anything into ENOVIA.

The REST endpoints are implemented over a version of the API in native mode.

About Possible Architectures

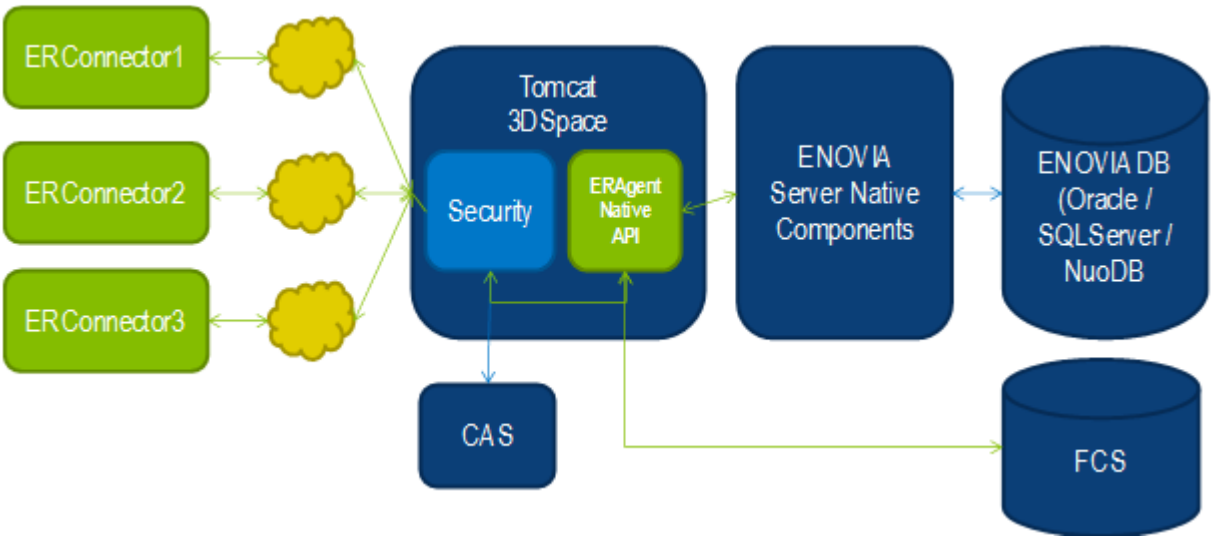
Installing the ENOVIA Standalone ERAgent

Standalone ERAgent Security Recommendations

About Possible Architectures

The following diagrams focus on REST server components. ENOVIA components are displayed in blue. ER Agent components are displayed in green.

3DSpace Architecture

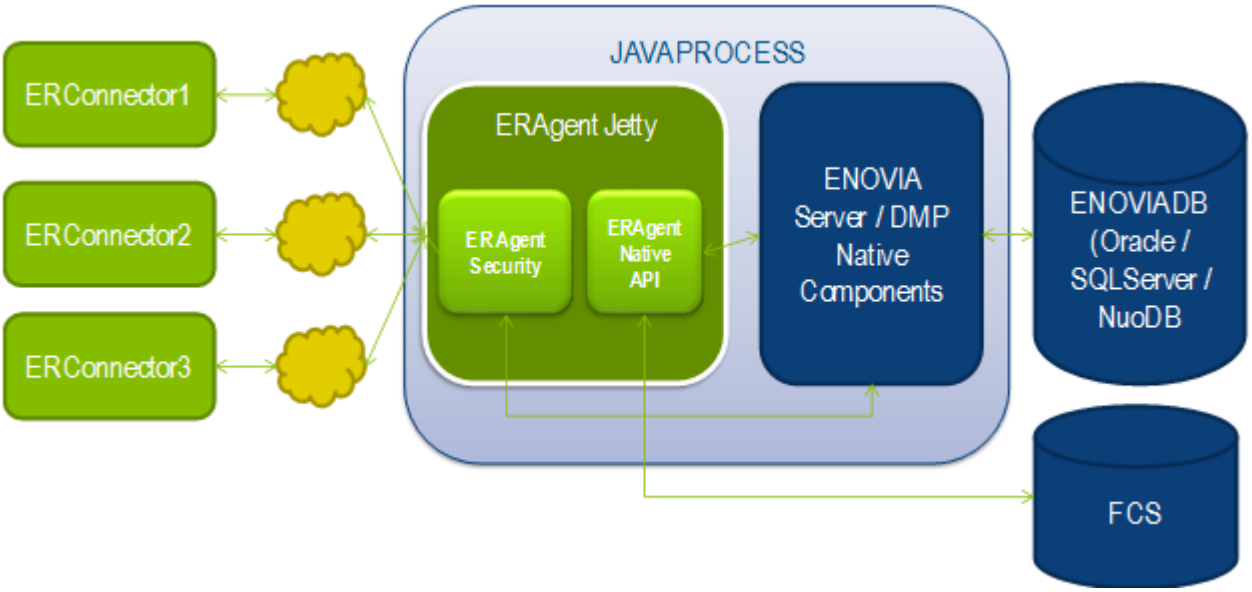


In a 3DSpace architecture:

- The ERAgent is fully embedded into the 3DSpace web application. There is no need to deploy it.

- The ERAgent native API handles queries and forwards them to the ENOVIA API and the FCS API.
- Security is delegated to the 3DSpace security component that provides ENOVIA connections to the ERAgent native API.

Standalone Agent Architecture



In a standalone mode:

- Deploy the ERAgent. See Installing the ENOVIA Standalone ERAgent.
- The ERAgent components are embedded into a Jetty server provided with the Standalone agent package.
- The ERAgent native API is strictly the same as the 3DSpace mode one.
- The ERAgent manages security and delegates authentication to the ENOVIA API.
- All ENOVIA native components run inside the java process of the Jetty server. There is no other process for the ERAgent.

Installing the ENOVIA Standalone ERAgent

This section describes how to install the Standalone ERAgent on ENOVIA.

Important: The Standalone ERAgent Jetty server does not support SSL encryption. Yet, it is possible and highly recommended to map the application behind a reverse proxy with SSL encryption.

The Standalone ERAgent requires:

- A native ENOVIA installation:
  - Either a server installation
  - Or a LiveCollaborationServer installation
  - Or a StudioModelingPlatformRichClient installation
- A free HTTP port
- The jetty JVM works with 512m Xmx. It does not require a large amount of memory. The JVM does not convert any files and does not require a lot of CPU.

Note: The connector (Exalead CloudView host) makes:

- An HTTP connection to the Standalone ERAgent Jetty.
- An HTTP(S) connection to all FCS servers to index.

1. Unzip the `enovia-standalone-agent.zip` file in your installation directory.

   The standalone agent package contains the following directories:
   - `lib` - all required libs to run the Standalone ERAgent
   - `scripts` - all scripts used to start the Agent up
2. Run the install script:
   - `install.sh` for Linux.
   - `install.bat` for MS Windows.

   Important: Both Linux and Windows scripts use the same command-line arguments.

Argument	Description
`--matrix-home` (mandatory)	The path to the folder that contains the MATRIX-R file. It is a FOLDER, not a FILE.
`--galaxy-home`	The path to the folder that contains the ENOVIA installation binaries. For example, `E:\ENOVIA\Studio\win_b64\code\bin` This argument is required if the `MATRIX-R` file is not stored into the ENOVIA server / studio installation. It is not required if `--matrix-home` points to the ENOVIA installation itself.
`--tns-admin`	The path to the folder that contains the `tnsnames.ora` file. It is a FOLDER, not a FILE.
`--dsls-config`	The path to the `DSLicSrv.txt` file. It is a FILE, not a FOLDER.

Argument	Description
	This argument is not mandatory if the `DSLicSrv.txt` file is available in `C:\ProgramData\DassaultSystemes\Licenses` or `/var/DassaultSystemes/Licenses`
`--agent-port` (mandatory)	The HTTP port on which the Standalone ERAgent listens. This port is used in the URL provided to the connector in the connectivity configuration. **Note:** The +1 port must also be available for the future stop mechanism.

When you run the script, it creates files in the `scripts` directory.

File	Description
`agentMxEnv.properties`	Contains environment variable definitions
`agentMxEnv.sh`	Called on startup
`agent-version.txt`	Contains the version of the component itself
`enovia-version.txt`	Contains the ENOVIA version
`startup.sh` or `startup.bat`	Scripts for start
`shutdown.sh` or `shutdown.bat`	Scripts for stop

Note:

- The Standalone ERAgent writes its logs into the `startup/run` directory.
- The Standalone ERAgent has a cache stored in the `install/build` directory.

  The cache size does not exceed 50MB and contains only an image of the ENOVIA data model (no content is cached).

3. If the Standalone ERAgent installation is successful, from the `scripts` directory, you can:
   - Start it with the `startup.sh |startup.bat` script.
   - Stop it with the `shutdown.sh | shutdown.bat` script.
4. To test the Standalone ERAgent connection, run a ping call:

   `http://<host>:<port>/enovia-agent/ping`

   The response must be:

   ```
   {"status":"success","payloadClass":"com.exalead.enovia.api.rest.common.json
   {"status":false}}
   ```

Note: To verify that the login/ password work correctly, go to the Exalead CloudView Administration Console, and in your connector instance, click **Check connectivity**.

Example 2. Examples

Example	Description
Studio or Live Collaboration Server	Context: • The Studio has been installed into the `/data/johndoe/enovia/enoviaV6R2015x/studio directory` • The `MATRIX-R` file is available at `/data/johndoe/enovia/enoviaV6R2015x/studio/MATRIX-R` • The `DSLicSrv.txt` file is available at `/data/johndoe/enovia/DSLicSrv.txt` • The `tnsnames.ora` file is available at `/data/johndoe/ora11g/network/admin/tnsnames.ora` The install script is called as follows: `./scripts/install.sh --matrix-home /data/johndoe/enovia/enoviaV6R2015x/studio --tns-admin /data/johndoe/ora11g/network/admin --dsls-config /data/johndoe/enovia/DSLicSrv.txt --agent-port 9393` **Note:** The `--galaxy-home` argument is not specified because it is under `/data/johndoe/enovia/enoviaV6R2015x/studio`, which is the `matrix-home` directory. In this case, the `galaxy-home` corresponds to the `/data/johndoe/enovia/enoviaV6R2015x/studio/linux_a64/code/bin` directory.
e4all server	Context: • The server is installed in `/home/data/RTV/R421relalodevpril234/server` • The `MATRIX-R` file is available at `/home/data/RTV/R421relalodevpril234/server/MATRIX-R` • The `DSLicSrv.txt` file is available at `/var/DassaultSystemes/Licenses/DSLicSrv.txt` • The `tnsnames.ora` file is available at `/home/data/ora121/network/admin/tnsnames.ora`

Example	Description

	The install script is called as follows:
	`./scripts/install.sh`
	`--matrix-home /home/data/RTV/R421relalodevpril234/server`
	`--tns-admin /home/data/ora121/network/admin`
	`--agent-port 9393`

Standalone ERAgent Security Recommendations

This section focuses on the Standalone ERAgent security, if you choose the HTTP REST connectivity mode.

Important: For the other connectivity modes, the security level is the same as the:

- 3DSpace security for the 3DSpace connectivity mode.
- Legacy ENOVIA SBA Connector for legacy mode (Local Context).

Restrict the Access to the Standalone ERAgent Port

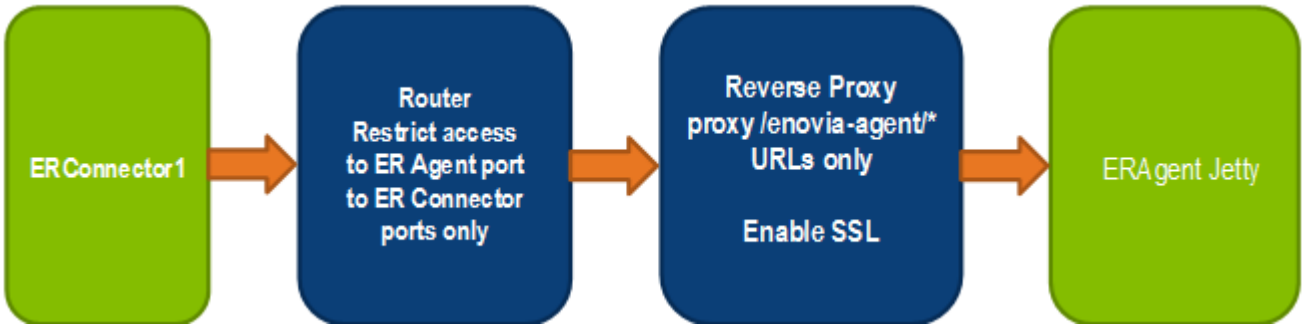
It is better to restrict the IP addresses that can access the Standalone ERAgent (`/enovia-agent` URI) to prevent other machines than the ER connector host to get access to ENOVIA data.

Configure a Reverse Proxy with HTTPS

The standalone agent does not offer an HTTPS connectivity by default.

As login and passwords are sent to the agent in HTTP POST bodies, it is better to secure it using an HTTPS reverse proxy. The reverse proxy with HTTPS is available on the ENOVIA server. You only need to add some entries to filter the ERAgent. To proxy pass, reverse the `/enovia-agent/*` URLs. When using an encrypted reverse proxy, all messages exchanged from the ER connector to the Standalone ERAgent host are encrypted.

The following diagram shows the basic security workflow.



Appendix - Configuration Parameters

The following tables detail the parameters that you can use to configure the ENOVIA ER connector.

Connection Parameters

Parameter	Description
Component class name	Select **On premise**.
Connectivity mode	Select the connectivity mode to connect to ENOVIA. • **Local Context** - Use this mode to connect to ENOVIA using the DMP studio agent through OCI. This RMI In-Process (RIP) mode uses the Live Collaboration Server implementation classes. However, instead of going through a registry and invoking a separate process, the client's address space loads the server classes directly. Database communication is directly between the ENOVIA client and the database. This is more efficient in situations that involve processing an extremely large number of transactions. This mode reduces the load on the HTTP ENOVIA server. • **REST** - In this mode, you can crawl data from the ENOVIA Tomcat server in HTTP Rest mode. It allows you to connect remotely on a CAS-secured instance. • **3DSpace** - New mode to connect to ENOVIA through the 3DSpace. • **Legacy XML Servlet** - Do not use this mode.
URL	Specify the base URL of the **3DSpace**, or leave empty if you use **Local Context**.
Login	3DSpace | XML Servlet | Local Context user login for indexing documents. Read-access required.
Password	3DSpace | XML Servlet | Local Context password for the ENOVIA user.
Role	Role that can access ENOVIA.
Vault	Optionally, you can specify the name of the ENOVIA vault you want to index. If you do not specify any vault, the connector scans all vaults.

Parameter	Description
	Important: It is possible to use different connectivity modes on several vaults in parallel.
Event monitor name	Optional. Specifies the Event monitor name, if you want to create an Event monitor to manage incremental indexing. By default, the connector uses an incremental scan based on dates. **Important:** Defining the Even monitor name is one of the steps in the Event monitor configuration. To get the complete procedure, see Adding an Event Monitor for Incremental Indexing.
Has write access	Allows the connector to modify the content of the database. It is typically useful for the following use cases: • If you use an Event Monitor, the connector can purge consumed events out of the database. If the write access is not enabled, events remain in the database and you must purge them manually. • If the FCS is remote, the connector needs to update file synchronization information.
Tenant	This is for Cloud deployments. For Exalead experts only.

Query Parameters

Parameter	Description
Component class name	Configure query using: • **UI** - that is to say through the Exalead CloudView Administration Console, by listing Business objects and Relationships below this property. This configuration type applies to the 3DSpace connectivity mode only. It is based on 3DSpace NLS files. ◦ **Translate into**: Translates workflow fields (for example, `STATE`). ◦ **Translate all selects**: Translates all fields. ◦ **Names conform with data-model**: Adapts the meta name to conform with the name rules of a data model. Names must conform with the regular expression `[A-Za-z0-9_]+`

Parameter	Description
	• **file** - through a configuration file (same configuration mode than the legacy ENOVIA SBA connector). ◦ **Query configuration file path**: Specify the file path of the configuration file. It must be on the same server than your ENOVIA instance. The xml nodes correspond to the UI options that found in this table. See the Configuration File Example below.
Business objects	
Enovia type name	Name of the Business Object type.
Make a query by subtype	Useful when data is very split. For example, when a Designer Central contains all CAD types and several subtypes, we can make a query for each subtype. This option gives you multithreading capabilities. **Important:** Leave empty if you have more than one object type to index.
Where clause	Filters objects. For example, `owner == <login>`
Index basics	Indexes all ENOVIA basics attributes. For example, all IDs. **Recommendation:** Avoid using this option as it has an impact on performance. It is better to specify **Selectables** in the configuration (see below).
Index all attributes	Indexes all ENOVIA attributes. **Recommendation:** Avoid using this option as it has an impact on performance. It is better to specify **Selectables** in the configuration (see below).
Index files	Indicates whether to index files for this Business object type.
Trace tenant	If your server vault has multiple tenants, it traces the source tenant by storing this info in a specific tenant meta.
Trace history	Indicates whether to trace history events as documents linked to the Business Object.
Trace type lineage	Creates a taxonomy of Business object types. It stores the hierarchy of the object using a `|` (pipe) separator. For example, `MCAD Model|3DPart|CATIA Part`

Parameter	Description
Preserve empty values	Shows attribute names in search results, even when these attributes have null or empty values. Without this option, when an attribute is not valuated, Exalead CloudView cannot retrieve it and cannot produce a document with a meta containing the attribute.
CV type names	The first element in this item list must be the Data Model class and the hierarchy type root node of the Consolidation Server.
Include policies	Includes the specified Business object policies only.
Exclude policies	Excludes the specified Business object policies only.
Selectables	Use selectables to use basic or other ENOVIA attributes without attribute `[<ATTRIBUTE_NAME>]` curly braces inclusion. You can use the target meta name to copy basics or attributes into one or several metas.
Programs	For legacy mode we need to support JPO, TCL or MQL. The connector sends the code to the ENOVIA kernel directly. From V6R2017x, it also supports JPO servers.
File filter	This node allows you to specify formats, extensions, etc. to accept or exclude during indexing. For example, you may want to index PDF and exclude XML files when indexing the content of a repository. The **Trace reason** option allows you to log why Exalead CloudView did not index files (for debug purpose).
Relationships	The properties and options are the same that the Business Object ones.

Processing Parameters

Parameter	Description
Component class name	Select processing policy. This is typically useful to get rid of items you do not want to index. Using a processor allows you to go further than using Include/Exclude policies as you can customize it to your own needs.

Security Parameters

See Table 1.

Advanced Parameters

Parameter	Description
Pushed documents before checkpoint	Minimum number of documents pushed between 2 consecutive checkpoints. It is better to specify a low value when Exalead CloudView documents are sorted (default behavior), and take time to be processed (for example, documents in OnePart).
Connect to ENOVIA on apply configuration	The connector connects to ENOVIA automatically when you click **Apply**, to verify that the configuration is correct. For example, to verify that you there are no references to types that do not exist.
Push API thread count	Number of threads used to push data into the Exalead CloudView Push API Server. It is useful when converting documents using PushAPI filters.
Push API queue size	Maximum number of intermediary documents stored between the ENOVIA database and the Exalead CloudView Push API Server. When the number of documents reaches this limit, the connector stops reading the ENOVIA database, to stop receiving more documents.
Verbose	Useful to debug the connector configuration. Logs each event sent to the Exalead CloudView Push API. For internal use by Support teams only.
Multithreading strategy	Choose how to organize your thread pool. • **Query**: for on-premises deployment, Business Object and Relationship queries are executed in parallel on a single ENOVIA source. • **Tenant**: for cloud deployment, it allows you to index multiple ENOVIA sources at the same time.
Enovia indexing thread count	Number of threads used by Query or Vault Tenant Producers.
Refresh the data model at each scan	Each time a scan starts, it looks for changes in the data model. This is a costly operation. Use it only when your data model changes regularly. Do not use it if the data model never changes, or very rarely.
Graph management mode	• **Default E/R graph manager** - This is the most simple mode as it allows you to simplify the naming of URIs. You can use the Trace

Parameter	Description
	tenant option for cloud deployments, to trace the ENOVIA vault and tenant. • ENOVIA **SBA graph manager** - The connector manages graphs like the legacy SBA connector (Exalead CloudView processes URIs and Consolidation Server object graph the same way).
File container handler	Specifies how to attach ENOVIA files as binary parts to Exalead CloudView documents.
Incremental scan options	• **Incremental timestamp offset**: Specify the number of minutes to add to the last `modified` date checkpoint. • **Force strict ordering**: Specify the number of seconds between 2 consecutive scans. If the previous scan happened before this duration, then the connectors make a partial ordering over the `modified` date, else this is a strict ordering. ◦ Negative value disables the partial ordering ◦ `0` forces the partial ordering ◦ Positive value gives the duration in seconds such as: T_n - T_{n-1} `this value`, the where clause for query indexing is: `modified previous checkpoint`. T_n - T_{n-1} > `this value`, the where clause for query indexation is: `modified > previous checkpoint`.
Delete options	• **Delete timestamp offset**: Specify the number of minutes to add to the last `delete` date checkpoint. • **Disable Where clause**: If you do not select this option, the connector deletes all the documents corresponding to the reverse of the `where` clause, for each query made to the ENOVIA database. This can be costly. **Recommendation:** For better performance, select this option to avoid querying the ENOVIA database.
Send data model in consobox	• **Send the data model to consobox**: Shows options to send documents describing the ENOVIA data model in the Exalead CloudView Consolidation Server.

Parameter	Description
	• **Send dimension**: Use it to see the dimensions referenced by the pushed attributes. • **Arcs to interfaces**: Draw arcs between the business objects or connections, and the interface types they reference. The connector requires this configuration to send interfaces to the Consolidation Server. • **Arcs to relationships**: Draw arcs between the connections and the relationship types they reference. The connector requires this configuration to send relationships to the Consolidation Server. • **Arcs to types**: Draw arcs between business objects and the types they reference. The connector requires this configuration to send types to the Consolidation Server. • **Metadata type to send to the consobox** can be of 3 types: **Type**, **Relationship**, or **Interface**. The **Send attribute** option sends the attributes attached to the Type. Select the **User name for URI** option to use the type name instead of the type id, to identify the documents stored in Exalead CloudView.
Error management	Specifies how to handle retry on errors for Business Objects retrieved from the ENOVIA database. It does not handle errors that may occur to the document afterward, for example, a faulty Push API filter or a Consolidation Server error. • **Enable error management**: Enables retry on error. If disabled: ◦ The connector fails at the first error when retrieving data. ◦ It does launch a retry, even if there is a major error requiring one. • **Maximum number of retries**: Specifies the maximum number of consecutive fetch retrieval errors a single object can trigger before being ignored. The connector launches each retry only once per scan, provided the query that fetched the object is still present. Possible values: ◦ A negative value means that there is no upper limit. ◦ `0` means that no retry is launched. The connector discards and logs every error. ◦ A positive value (above `0`) specifies the number of retries.

Parameter	Description
	Note: If the scan of a Business Object fails as many times as the maximum number of retries, but matches the WHERE clause in the following scan, the cycle starts once again. • **Maximum number of errors**: Specifies the maximum number of errors tolerated for a single scan. Once this number is reached, the connector generates an error. Possible values: ◦ A negative value means that there is no limit in the number of errors. ◦ `0` means that the first error stops the entire scan. ◦ A positive value (above `0`) specifies the number of tolerated errors for a given scan. This value may sometimes be crossed as the query can be executed in parallel.

Example 3. Configuration File Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<query-config>
  <businessobject-meta-groups>
    <metas>
      <name>attributes</name>
      <selects>
        <select translated="false" select="Title">
          <target-metas>
            <meta-name>name</meta-name>
          </target-metas>
          <data-type>STRING</data-type>
        </select>
        <select translated="false" select="Originator">
          <target-metas>
            <meta-name>description</meta-name>
          </target-metas>
          <data-type>STRING</data-type>
        </select>
      </selects>
      <programs/>
    </metas>
  </businessobject-meta-groups>
  <businessobject-queries>
    <query>
      <name>MCAD Model</name>
      <cloudview-types>
        <name>mcad_model</name>
```

```
      <name>model</name>
    </cloudview-types>
      <index-all-basics>true</index-all-basics>
      <index-all-attributes>false</index-all-attributes>
      <sorted>true</sorted>
      <index-files>true</index-files>
      <make-a-query-by-subtype>false</make-a-query-by-subtype>
    <includes>
      <name>attributes</name>
    </includes>
    <metas/>
    <file-filter>
      <max-file-size>-1</max-file-size>
      <trace-reason>false</trace-reason>
    </file-filter>
  </query>
</businessobject-queries>
<relationship-meta-groups>
  <metas>
    <name>attributes</name>
    <selects>
      <select translated="false" select="IsFinalized">
        <target-metas>
          <meta-name>name</meta-name>
        </target-metas>
        <data-type>STRING</data-type>
      </select>
    </selects>
    <programs/>
  </metas>
</relationship-meta-groups>
<relationship-queries>
  <query>
    <name>VersionOf</name>
    <cloudview-types>
      <name>instance</name>
    </cloudview-types>
    <index-all-basics>false</index-all-basics>
    <sorted>false</sorted>
    <index-all-attributes>false</index-all-attributes>
    <make-a-query-by-subtype>false</make-a-query-by-subtype>
    <includes>
      <name>attributes</name>
    </includes>
    <metas/>
  </query>
</relationship-queries>
<translation-locale>en</translation-locale>
```

```
  <translate-all-selects>false</translate-all-selects>
  <datamodel-metaname-conformant>false</datamodel-metaname-conformant>
</query-config>
```

Advanced Parameters Not in the UI

For specific use cases, you can also edit other advanced parameters manually in the `connector.properties` file.

This file is present in:

- The Exalead CloudView `<DATADIR/connector-java0/config` directory.
- The ER Agent, either in the `/config` directory or in the Tomcat `/config/bin` directory if you have a 3DSpace configuration mode.

Note: This section does not give an exhaustive list of the `connector.properties` file parameters.

Parameter	Description
`relationship.invali`	Use this Boolean parameter if the query sent to the ENOVIA database returns an object for which either the `id` or the `PhysicalId` attribute is empty: • If `false` (default value), the connector generates an exception and stops the indexing of the relationship. • If `true`, the connector generates an error message in the ER Agent (or in the connector logs if you use the Local Context mode), and continue its indexing job.
`businessobject.inva`	Use this Boolean parameter if the query sent to the ENOVIA database returns an object for which either the `id` or the `PhysicalId` attribute is empty: • If `false` (default value), the connector generates an exception and stops the indexing of the business objects. • If `true`, the connector generates an error message in the ER Agent (or in the connector logs if you use the Local Context mode), and continue its indexing job.