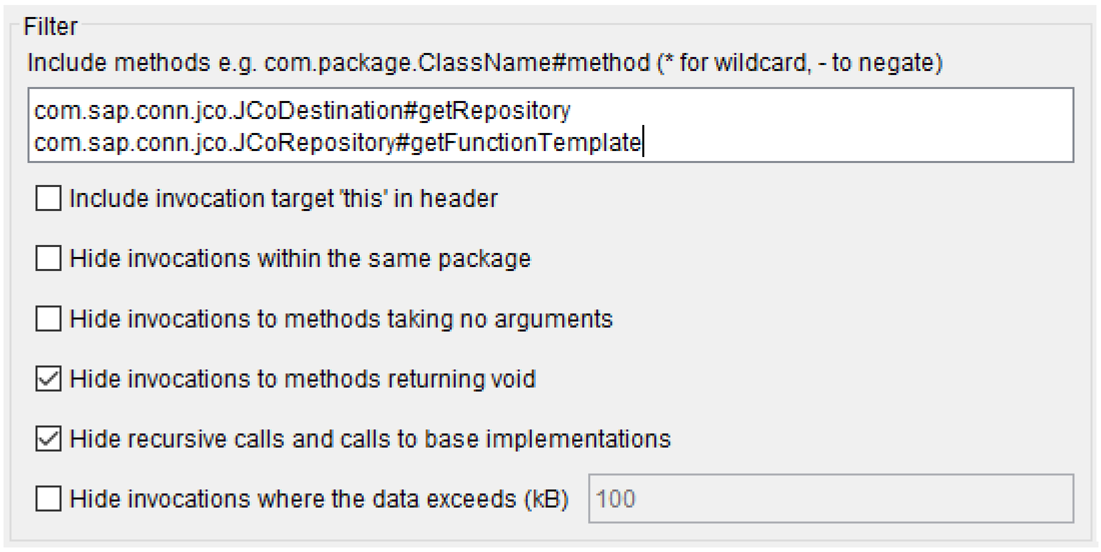
Support for virtualizing SAP RFC calls is achieved by using the Java Agent to intercept calls made by applications that use the SAP Java Connector. The application still requires a connection to the SAP server to retrieve metadata. Please see product documentation for more details.

In some circumstances, a SAP server may not be available in the environment in which virtualization is required. In this case, it is possible to record interactions where a SAP server is available for use in that environment. This requires additional configuration within the project, namely a Java Application Transport with Recording Studio settings as shown below:



Filter

Include methods e.g. com.package.ClassName#method (* for wildcard, - to negate)

```
com.sap.conn.jco.JCoDestination#getRepository
com.sap.conn.jco.JCoRepository#getFunctionTemplate
```

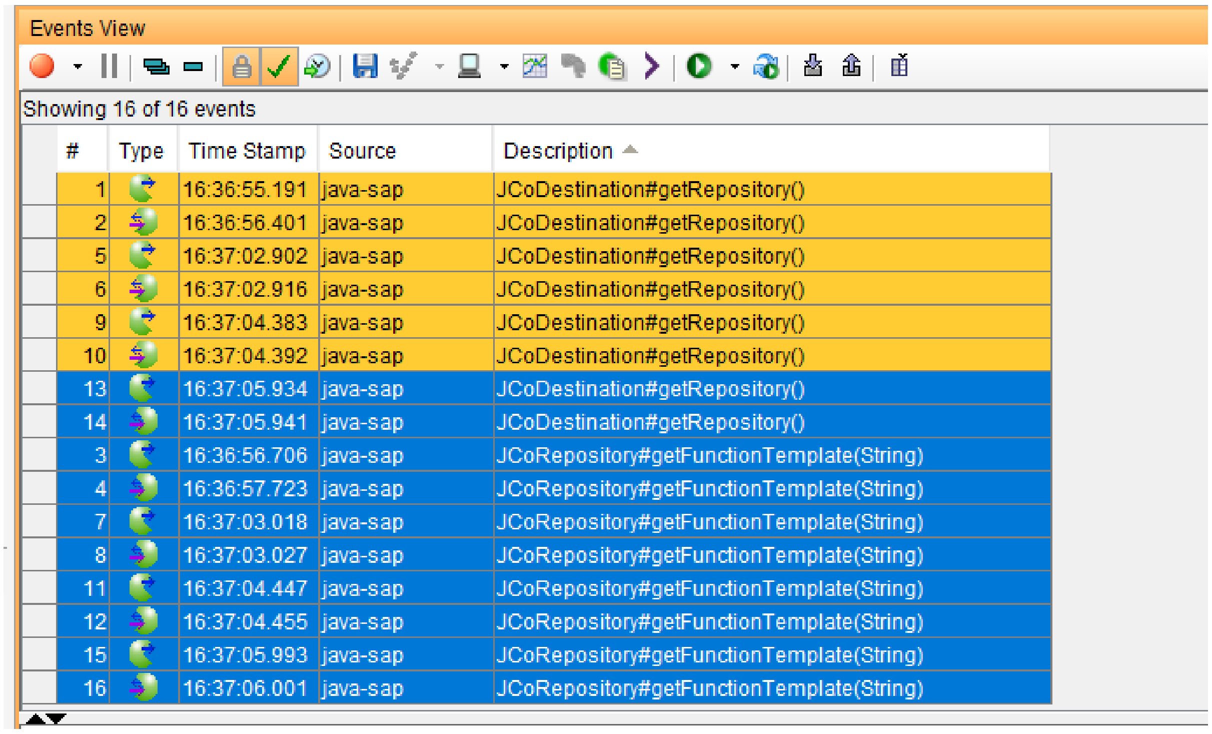
- ☐ Include invocation target 'this' in header
- ☐ Hide invocations within the same package
- ☐ Hide invocations to methods taking no arguments
- ☑ Hide invocations to methods returning void
- ☑ Hide recursive calls and calls to base implementations
- ☐ Hide invocations where the data exceeds (kB) 100

A pre-configured transport is available in **sap-metadata-recording.zip**, the following steps details how to import this into an existing project, use it to record metadata, and generate a stub:

Pre-requisite: a project that is configured to record SAP RFC interactions using the Java Agent.

1. Import the **sap-metadata-recording.zip** file:
   a. **Project** -> **Import Resources…**
   b. Select **sap-metadata-recording.zip**
   c. Select default options and complete wizard
2. Configure the physical Java Application transport
   a. In Logical View right-click on the **sap-metadata-recording** transport
   b. From the context menu: **Set Binding In** -> <environment> -> **Create new JVM**
   c. In most cases, just save with the default values (refer to product documentation for further details on JVM configuration)
3. Record the SAP RFC calls and metadata
   a. In Logical view, select the **sap-metadata-recording** transport and the existing SAP transport.
   b. Click the record button to add these as monitors and take you to the Recording Studio perspective
   c. Start recording
4. Perform interactions with the SAP system, when done stop the recording
   a. Click the stop recording button in Recording Studio
5. Generate a stub to provide SAP metadata
   a. In Recording Studio, select the **sap-metadata-recording** monitor in the Monitor Configuration.
   b. Click on the Description column to sort it in ascending order

c. Select the last 2 “JCoDestination#getRepository()” events and all of the “JCoRespository#getFunctionTemplate(String)” events. (e.g. select the event representing the final call to getRepository and use Ctrl-Shift-End to select the remaining events):


Events View

Showing 16 of 16 events

#	Type	Time Stamp	Source	Description
1		16:36:55.191	java-sap	JCoDestination#getRepository()
2		16:36:56.401	java-sap	JCoDestination#getRepository()
5		16:37:02.902	java-sap	JCoDestination#getRepository()
6		16:37:02.916	java-sap	JCoDestination#getRepository()
9		16:37:04.383	java-sap	JCoDestination#getRepository()
10		16:37:04.392	java-sap	JCoDestination#getRepository()
13		16:37:05.934	java-sap	JCoDestination#getRepository()
14		16:37:05.941	java-sap	JCoDestination#getRepository()
3		16:36:56.706	java-sap	JCoRepository#getFunctionTemplate(String)
4		16:36:57.723	java-sap	JCoRepository#getFunctionTemplate(String)
7		16:37:03.018	java-sap	JCoRepository#getFunctionTemplate(String)
8		16:37:03.027	java-sap	JCoRepository#getFunctionTemplate(String)
11		16:37:04.447	java-sap	JCoRepository#getFunctionTemplate(String)
12		16:37:04.455	java-sap	JCoRepository#getFunctionTemplate(String)
15		16:37:05.993	java-sap	JCoRepository#getFunctionTemplate(String)
16		16:37:06.001	java-sap	JCoRepository#getFunctionTemplate(String)

d. Click on “Create stubs from recorded events” button, complete the Wizard selecting where to create the stub and what to call it (e.g. under the pre-existing SAP system with a name of “metadata”).

Having created this stub, you will be able to virtualise calls to any of the RFCs for which you have virtualised getFunctionTemplate from applications that use the SAP Java Connector without having to connect to a SAP server. Consult the product documentation for more details on creating stubs for SAP RFC calls.