IBM Opportunity Detect
Version 9 Release 1.1
January 15, 2016

Administrator's Guide
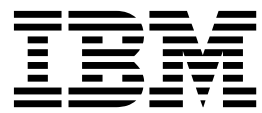
IBM

Contents

Chapter 1. About IBM Opportunity Detect

IBM® Opportunity Detect enables you to look for specified customer behaviors and patterns in your customer data. You define the transactions and patterns that Opportunity Detect looks for, and you specify the data that is written to the database or web servlet when those criteria are met.

You use Opportunity Detect components to build trigger systems in workspaces. When you run a workspace, its trigger systems apply your business logic to streams of data from your transaction and profile data feeds.

Opportunity Detect uses IBM InfoSphere® Streams technology for enhanced performance.

Streams is an advanced analytic platform that allows Opportunity Detect to quickly ingest, analyze and correlate information as it arrives from batch and real time sources. The solution can handle very high data throughput rates.

Integration with IBM Campaign

You can integrate Opportunity Detect with IBM Campaign.

You can use the Expanded Outcome data source connector to store the Outcome data in database tables that Campaign can use. See the section on Expanded Outcome tables in the *IBM Opportunity Detect User's Guide* for details about this integration.

Note: Another product, IBM Interact Advanced Patterns, provides integration with IBM Interact. With this integration, you can apply Pattern component logic to data sent from Interact in real time. See the *IBM Interact and IBM Interact Advanced Patterns Integration Guide* for details.

Trigger system basics

A trigger system is comprised of configurable components. When a trigger system runs, it produces State History and Outcome data.

State History data is the saved results of processing some components; the system uses this information in subsequent runs. Outcome data can be used by external systems to respond to customer activity.

The following diagram illustrates a basic trigger system.
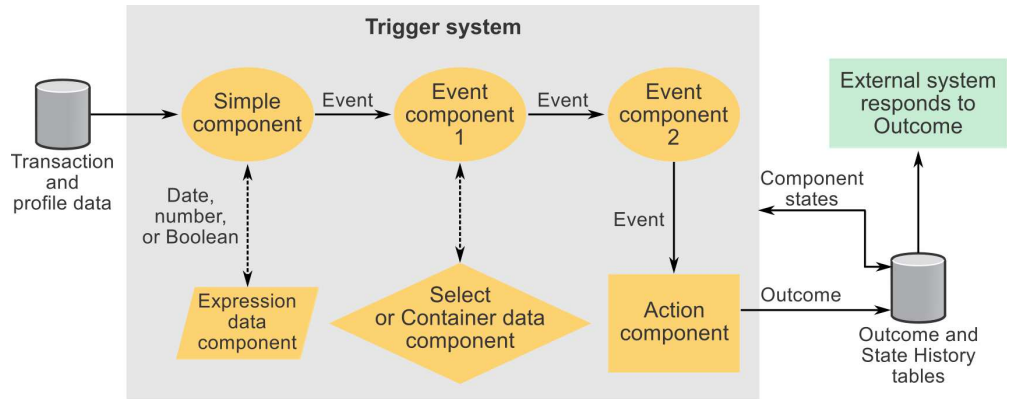
IBM Opportunity Detect architecture

IBM Opportunity Detect uses IBM InfoSphere Streams, a high-performance computing platform. Multiple instances of the Opportunity Detect engine can run in servers in a Streams instance. You can define multiple server groups and specify the data they can access. You can also specify which server group processes your Opportunity Detect workspace.

The following diagram provides a high-level overview of Opportunity Detect architecture for batch mode.
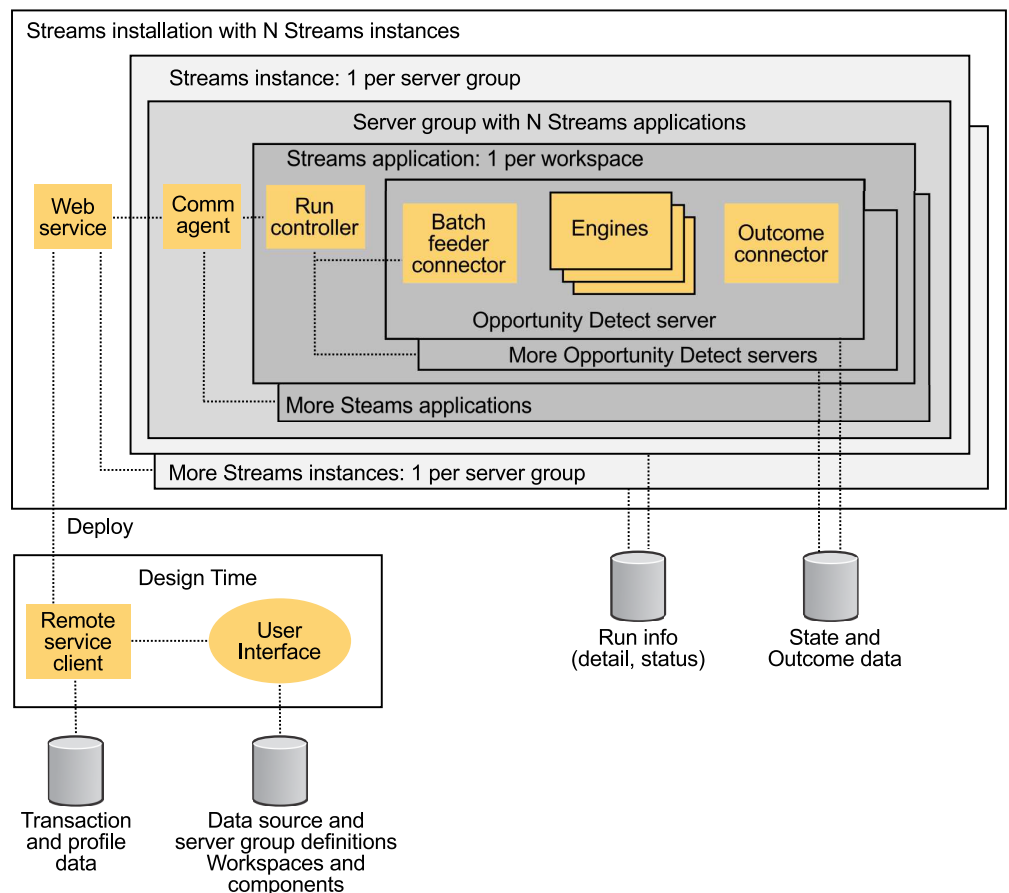
Locale in Opportunity Detect

Opportunity Detect supports several locales for transaction and profile data.

When you create a File type of data source connector, you can select a locale for the data in these files.

Note that full support for any character set in Opportunity Detect also depends on the configuration of the following.

- The Opportunity Detect system table database.
- The client machines and browsers used to access Opportunity Detect.

Chapter 2. Opportunity Detect roles and permissions

The permissions assigned to users in Opportunity Detect determine what areas of the application they can access and the actions they can perform.

You manage user application access by assigning the desired roles and permissions to individual users, or by assigning users to groups that have the desired roles and permissions. You can use the default roles, or create custom roles with the permissions that you specify. You can not create custom permissions, only custom roles.

You manage roles and permissions from the Users, User Groups, and User Roles & Permissions pages. All of these pages are available under the **Settings** menu.

Tip: For help when you work on these pages, click **Help > Help for this page**, or see the *IBM Marketing Platform Administrator's Guide*.

Permissions for Opportunity Detect

The following table describes permissions that you can assign to roles in Opportunity Detect.

All permissions that have the **Not Granted** status are treated as **Denied**.

You manage roles and permissions on the **Settings > User Roles & Permissions** page. You can assign roles to individual users, or you can assign roles to a group and these roles are granted to all members of the group. See the *IBM Marketing Platform Administrator's Guide for complete details.*

Table 1. Permissions in Opportunity Detect

Permission	Description
View only	Can access all of the user interface, in view-only mode.
Design triggers	• Can create workspaces and design trigger systems. • Can create, modify, and delete all trigger related resources. • Can access Workspace, Component, Audience Level, Data Source, and Named Value List pages. • Can not access the Server Groups page or the Deployment tab of a workspace. • Can not set off a batch run. • Can not administer objects that the web service creates when Opportunity Detect is integrated with Interact.
Run for testing	• Deploy deployment configurations and run batch deployment configurations on server groups not designated for production. • Can access Server Group page and the Deployment tab of a workspace, but can not designate a server group for production. • Can not deploy deployment configurations or run deployment configurations that use a production server group.

Table 1. Permissions in Opportunity Detect (continued)

Permission	Description
Run for production	• Deploy deployment configurations and run batch deployment configurations on any server group. • Perform all actions on the Server Group page and the Deployment and Batch Run tabs of a workspace, including designating a server group for production.
Administer real time	Manage objects that the web service creates when Opportunity Detect is integrated with Interact to enable real time mode. Allows the following. • Delete workspaces and components created by the web service. • Start and stop real time deployment configurations and update their log level. The user with this permission alone can not start runs for real time deployment configurations. No one, even with this permission, can do any of the following. • Delete and update audience levels, data sources, named value lists, server groups, or deployment configurations created by the web service. • Create and deploy deployment configurations created by the web service.

Related concepts:

"Built-in roles in Opportunity Detect"

Built-in roles in Opportunity Detect

Four built-in roles are included with Opportunity Detect.

In addition, you can create roles with permissions that you specify. See the *IBM Marketing Platform Administrator's Guide* for details on creating custom roles.

The following table shows the permissions assigned to each built-in role.

Table 2. Built-in roles in Opportunity Detect

Role	Permissions
OpDetectViewer	• View only
OpDetectTestDesigner	• View only • Design triggers • Run for testing
OpDetectProductionDesigner	• View only • Design triggers • Run for production

Table 2. Built-in roles in Opportunity Detect (continued)

Role	Permissions
OpDetectAdmin	• View only • Design triggers • Run for testing • Run for production • Administer real time

Related reference:

"Permissions for Opportunity Detect" on page 5

Chapter 3. Data source setup in Opportunity Detect

To set up data sources for IBM Opportunity Detect, you configure them in the user interface, including setting up connections to the databases that hold run detail, State History, and Outcome data.

To develop data feeds, you must identify the operational systems, databases, and tables that hold the feed data you want to use. If you plan to use flat files, you must extract this data into flat files in the required format, with the required file names.

The following table describes the input and output data you set up in Opportunity Detect.

Table 3. Types and formats of data used in Opportunity Detect

Type	Description	Allowed formats
Input data		
Your profile and transaction files must be accessible from the machine on which the Opportunity Detect run time component is installed.		
Transaction data	Customer activities, obtained from your organization's operational system. At least one transaction data source is required to run a workspace.	• Flat file • A message from the IBM Opportunity Detect web service
Profile data	Customer attributes. Examples of profile data are age, home address, and telephone number. Optional.	• Flat file • Database tables If the Web Service data source connector is used for transaction data, the associated profile data must be in a database table.
Output data		
State History	Data needed for component processing. You can share State History tables with multiple workspaces for development and test purposes, and set up one or more State History tables for exclusive use by the production environment.	• Database tables
Outcomes	Data produced by trigger systems, for use by external systems. You can share Outcome tables with multiple workspaces for development and test purposes, and set up one or more State History tables for exclusive use by the production environment.	• Database tables • A message from the IBM Opportunity Detect web service
Run detail	Data produced by the engine, containing information about workspace runs. The table is created in the run time database automatically during installation. When you set up data sources, you define a database connection for the run time database and associate it with the server group.	• Database tables

Data source configuration roadmap

You configure data sources in Opportunity Detect as described in this roadmap. When you have finished this process, you can build, deploy, and run trigger systems, as described in the *IBM Opportunity Detect User's Guide*.

You can access all of the configuration tools mentioned here by navigating to the **Settings > Detect Settings** page.

You must have appropriate permissions to access the configuration tools. The following built in roles allow the user to access these tools.

- **OpDetectAdmin**
- **OpDetectTestDesigner**
- **OpDetectProductionDesigner**

If you use a custom role, it must have either of the following permissions.

- **Run for testing**
- **Run for production**

Table 4. Data source configuration roadmap

Step	Where to find details
Set up your State History and Outcome database tables. Required.	"Setting up State History and Outcome tables"
On the Audience Levels page, configure audience codes. Required.	"Configuring audience level codes" on page 12
On the Named Value Lists page, create display names for pre-defined values in your transaction and profile data sources. Optional.	"Configuring named value lists" on page 13
On the Data Sources page, identify the fields in your transaction and profile data sources and specify display names. Required.	"Defining data sources" on page 15
On the Server Groups page, define database connections, data source connectors, and server groups, and map connectors to data sources. Required.	"Configuring server groups and data source connections" on page 17
Set up your transaction and profile data feeds. Required.	- If you use the flat file format, ensure that the files meet the requirements for data used in Opportunity Detect. "Naming requirements for profile and transaction files" on page 24 "Data requirements for profile and transaction files" on page 25 "Profile data" on page 26 - If you use the web service connector for transactions, see the *IBM Opportunity Detect Administrator's Guide* for details about the Java classes that must be developed to use the Web Service data source connector.

Setting up State History and Outcome tables

Use this procedure to edit the SQL scripts provided for creating your State History and Outcome tables and to run the scripts against your database.

Before you begin

You must set up a data source connection on the Server Groups page for each database where you create these tables.

About this task

You do not need to use this procedure to set up Outcome tables if you use the Opportunity Detect Web Service data source connector for your Outcome data, because the data is sent to the Java class you develop.

Procedure

1. Open the SQL scripts in a text editor and edit them as described in the comments in the files.

   The scripts are located in the `database/DB2/RunTime` directory under your Opportunity Detect run time installation. Edit the following scripts.

   - `OutcomeTable.sql`-Use this script to create the Outcome table when you use the Outcome type of data source connector.
   - `ExpandedOutcome.sql`-Use this script to create the two Outcome tables when you use the Expanded Outcome type of data source connector.
   - `StateTable.sql`-Use this script to create the State History table or tables.

   You replace variables with the names you want to give to the tables in the database. Comments in the scripts provide instructions.

2. Run the scripts against the database you have configured to hold your State History and Outcome data.

Audience level codes

Audience codes represent the various audience levels defined in your data. Opportunity Detect groups data sources based on the audience code associated with them.

About audience levels

An audience level is composed of a key or database table field that uniquely identifies a member of that audience level. For example, your organization could use the audience levels Household, Customer, and Account. Each of these levels represents a different view of your marketing data.

Audience levels are typically organized hierarchically. Using the examples above, you might have the following hierarchy.

- Household is at the top of the hierarchy, and each household can contain multiple customers.
- Customer is next in the hierarchy, and each customer can have multiple accounts.
- Account is at the bottom of the hierarchy.

Other, more complex examples of audience hierarchies exist in business-to-business environments, where audience levels may need to exist for businesses, companies, divisions, groups, individuals, accounts, and so on.

These audience levels may have different relationships with each other, for example one-to-one, many-to-one, or many-to-many. By defining audience levels,

you allow these concepts to be represented within Opportunity Detect, so users can manage the relationships among these different audiences for targeting purposes. For example, although there might be multiple customers per household, you might want to limit contact to one customer per household.

Note: IBM Services can advise you on the audiences you need to define for your system.

How audiences are associated with data sources

When you define your data sources on the Data Sources page, you specify the fields you want to use in Opportunity Detect. When you specify an audience ID field, you associate that field with an audience level defined in Opportunity Detect.

Audience codes for internal use

Opportunity Detect includes a pre-defined audience code: n. The n code is used internally by the system and is read-only. You must create additional audience codes to represent the audience levels in your data.

Also, the letter i is reserved for real time mode, and signifies the audience used for the audience level used by Interact when integration is enabled. You should not use i when configuring your audience levels for batch mode.

Configuring audience level codes

Use this procedure to specify audience level codes in Opportunity Detect.

Procedure
1. Log in to Opportunity Detect and go to the **Settings > Detect Settings > Audience Levels** page.
2. Click **Add Audience Level**.
3. Give the audience level a name and a code.

   The audience code must be a single alphanumeric ASCII character, lower case (0-9, a-z except i).
4. Click **Save and Close**.

Named value lists enable display names for predefined field values

When users of Opportunity Detect work with components to build trigger systems, they can select data fields from drop-down lists in the component editors. If a data source includes a field with predefined values, you can create a named value list to assign display names to these values.

The display names are what the user sees when working with component editors. Display names make the component-building process more intuitive.

Named value lists are reusable. For example, you might have several transaction data sources that have a Boolean field where the predefined values are 0 and 1, where 1=true and 0=false. In this case, you can create a named value list that you associate with these fields in several transaction data sources.

Named value list example

For example, you might have a field in a data source that is named FREQUENCY. This field contains three predefined values, which have the following significance.

- F for customers who interact with your company frequently
- A for customers whose interaction frequency is average
- I for customers who interact with your company rarely

To help users understand what these values mean, you can map each value to a display name. You might create a named value list called Frequency and map the values as follows.

Table 5. Mapping data to display names

Value	Display name
F	Frequent customer
A	Average customer
I	Infrequent customer

After you map the values, you can associate the list with the field when you define the data source. Then, when a user selects this field in a component editor, the drop-down list that contains the predefined values shows the display names, which are more meaningful to the user than the actual values in the data source.

Predefined lists

Three lists are present when Opportunity Detect is installed, for use in the component editors. The values are mapped to display names as follows, but you can modify the values as needed to reflect your data.

Table 6. Display name mapping in predefined lists

List name	Display name mapping
NUMERIC_YES_NO	• Yes = 1 • No = 0
DAY_WEEK_MONTH	• Day = day • Week = week • Month - month
HIGH_MED_LOW	• High = H • Medium = M • Low = L

Configuring named value lists

Use this procedure to create named value lists in Opportunity Detect.

Before you begin

Obtain the following information about your data.

- Pre-defined values - You need these when you create the named value list items in this procedure.

- Name of the field in your data source that contains predefined values - You need this when you associate the named value list to the appropriate field in your data source.

Procedure

1. Log in to Opportunity Detect and go to the **Settings > Detect Settings > Named Value Lists** page.
2. Click **Add List**.
3. In the Properties panel, give the list a name and description, and specify the data type.

   The data type must match the data type of the field in your data source.
4. In the List Items panel, enter a display name and the predefined value that exists in your data source, and click **Add List Item**.

   Repeat this step for each predefined value for which you want to create a display name.
5. Click **Save and Close**.

What to do next

When you specify transaction and profile fields on the Data Sources page, associate the named value list with the appropriate field.

Deleting and modifying named value lists

Use this procedure to delete or modify named value lists in Opportunity Detect.

Before you begin

Before you can delete a list or a list item, you must either delete all of the components that use any of the list items, or edit those components to remove the reference to the list item.

The components that use items in a list are shown in the List Item and Component Association panel.

About this task

The predefined lists can be modified, but they not be deleted.

Procedure

1. Log in to Opportunity Detect and go to the **Settings > Detect Settings > Named Value Lists** page.
2. To delete a list, select the list you want to delete, click **Delete List**, and click **OK**.
3. To delete a list item, select the list to which the item belongs, select the item you want to delete, click **Delete List Item**, and click **OK**.

Setting up State History and Outcome tables

Use this procedure to edit the SQL scripts provided for creating your State History and Outcome tables and to run the scripts against your database.

Before you begin

You must set up a data source connection on the Server Groups page for each database where you create these tables.

About this task

You do not need to use this procedure to set up Outcome tables if you use the Opportunity Detect Web Service data source connector for your Outcome data, because the data is sent to the Java class you develop.

Procedure

1. Open the SQL scripts in a text editor and edit them as described in the comments in the files.

    The scripts are located in the `database/DB2/RunTime` directory under your Opportunity Detect run time installation. Edit the following scripts.

    - `OutcomeTable.sql`-Use this script to create the Outcome table when you use the Outcome type of data source connector.
    - `ExpandedOutcome.sql`-Use this script to create the two Outcome tables when you use the Expanded Outcome type of data source connector.
    - `StateTable.sql`-Use this script to create the State History table or tables.

    You replace variables with the names you want to give to the tables in the database. Comments in the scripts provide instructions.

2. Run the scripts against the database you have configured to hold your State History and Outcome data.

Defining data sources

Use the **Data Sources** page to a define each transaction and profile data source that you want to use in Opportunity Detect.

About this task

When you define the transaction and profile data sources that you want to make available in workspaces, you do the following.

- Specify the type of data source and a display name for the data source.
- Specify every field that you want to make available for use in building trigger systems.
- Optionally, associate named value lists with fields that have predefined values.

Procedure

1. Log in to Opportunity Detect and go to the **Settings > Detect Settings > Data Sources** page.
2. Click **Add Source**.
3. In the Properties panel, give the data source a name and description and specify the content type.

    The content type describes how the data is used in Opportunity Detect.

4. In the Fields panel, click **Add Fields**, complete the fields in the Add a Data Source Field window and click **OK**.

    Repeat this step for each field that you want to use as a data source.

    For every transaction data source, you must define an audience field and a date field.

Profile data sources require only an audience field. Only one profile data source per audience level is supported.

For profile data sources in database tables, the value you enter in the **Name** field must be the same as the corresponding column name in the database.

Optionally, select a named value list to associate with the field, if the field contains predefined values.

5. Click **Save and Close**.

Fields on the Data Sources page

You define data sources on the Data Sources page. If you are unsure of how to complete the fields, use the information provided here.

Table 7. Fields on the Data Sources page

Field	Description
Properties	
Content Type	Select one of the following options. • **Transaction** - Customer transactions. At least one Transaction data source is required to run a workspace. • **Profile** - Data such as address, age, and gender in an external database table.
Name	Enter a descriptive name for the data source. • For profile data sources in database tables, the name must be the same as the corresponding column name in the database. • For transaction data sources, the name does not have to match the field name in the actual flat file or web service message.
Description	Provide a description of the data source.
Add Field	
Name	Provide the physical name of the field in the source database table or flat file. Data source field names should not be assigned so that one name can be found within another. For example, the name `date` can be found within the name `firstdate`.
Display Name	Provide a descriptive name. This appears in component editors.
Description	Provide a description of the data source field.
App Type	Select the attribute type from the following options. The options available depend on the Content Type selected for the data source. • Audience Id-Select this for audience level fields. • Audience Level-If you select Audience Id as the App Type, select the audience level you want to associate with this field. • Profile Attribute-Choose this type for profile fields that are not audience or date fields. • Transaction Attribute-Choose this type for transaction fields that are not audience or date fields. • Transaction Date-Choose this type for date fields. One date field is required for each transaction data source.
Audience Level	If you selected Audience Id as the App Type, enter the audience level code that applies for the field.

Table 7. Fields on the Data Sources page (continued)

Field	Description
Data Type	Select the type of the data in the field. The options available depend on the App Type selected for the field, and also on the custom data types defined on the Custom Data Types page, if any. The following are the default options. • Boolean • Currency • Date • Double • Integer • String The application supports profile data sources in database tables in a variety of database types. The data types shown in this field are generic types that may be different in your database. The system converts the type you select into the correct data type for your database.
Name Value List	Named value lists with a data type that matches the field you are working with are displayed here. If the field contains predefined values, you can associate it with a named value list.

Deleting and modifying data sources

Use this procedure to delete or modify data sources in Opportunity Detect.

Before you begin

If you want to edit a data source that is being used by a component, you must either delete the components that use the data source, or edit those components to remove the reference to the data source.

The components that use the data source are shown in the Source Field and Component Association panel.

Procedure

1. Log in to Opportunity Detect and go to the **Settings > Detect Settings > Data Sources** page and select the data source you want to work with.
2. To delete the whole data source, click **Delete Source** and click **OK**.
3. To add a field, click Add Field and click **OK**.
4. To delete a field, select the field, click **Delete Field** and click **OK**.

Configuring server groups and data source connections

Use this procedure to define server groups and connect them to data sources you have configured.

About this task

Server groups are comprised of the following.
• Streams servers that are set up during installation
• Database connections that you create

- Data source connectors that you create and map to data sources

Typically, organizations use different server groups for their development, test, and production environments. Development and test server groups often share Outcome and State History tables, while the production server group normally has its own, exclusive Outcome and State History tables.

Procedure

1. Log in to Opportunity Detect and go to the **Settings > Detect Settings > Server Groups** page.
2. The first time you use the Server Groups page, and whenever you add a new server, click **Synchronize with changed streams configuration** on the Servers tab to refresh the list of servers.
3. On the Database Connections tab, configure a connection for every database containing the following tables that you plan to use with Opportunity Detect.
   - Run time tables
   - Outcome tables
   - State History
   - Profile

   You need the database name.
4. On the Data Source Connectors tab, configure connectors for your transaction, profile, Outcome, and State History data.

   You do not have to set up a connector for your run time tables.

   For transaction and profile data in file format, you specify the encoding, locale, and currency format used in the file. Opportunity Detect constructs the required file name, and you must ensure that your data files use this name.
5. On the Server Groups tab, select a server group to open a panel with four tabs. Configure server groups as follows.
   a. On the Properties tab, select an available Streams server and instance, and specify whether the server group is used for production.

      You can also view a list of workspaces that have deployment configurations associated with the server group.
   b. On the Servers tab, double-click in the **# of Engines** column to set the number of Opportunity Detect engines to run on this machine.
   c. On the Database tab, select a run time database and select the database connections you want to make available for table connector mapping.
   d. On the Data Source Mapping tab, map transaction and profile data sources and your Outcome and State History tables to data source connectors.

      Table connectors and flat file connectors enable the system to connect to your Outcome and State History database tables and your transaction and profile files.

Related concepts:

"About data source connectors" on page 22

Related reference:

"Fields on the Server Groups page" on page 19

Fields on the Server Groups page

If you are unsure of how to complete the fields on the Server Groups page, use the information provided here.

Table 8. Server Groups page: Servers tab

Field	Description
Synchronize with changed streams configuration	Click to refresh the list of servers and stream instances.

Table 9. Server Groups page: Database Connections tab.
Tip: If a database connection error occurs, verify that the information entered in these fields is correct.

Field	Description
Add	Click to open a panel where you can add a database connection.
Name	Enter a descriptive name for this database.
Database Type	Select a database type from the drop-down list of supported databases.
Database Name	Enter the name of the database as shown in your database management client. This must exactly match the name as shown in the client. Case-sensitive.
Server Name	Enter the fully qualified name or IP address of the machine that hosts the database server. For example, machine.mycompany.com.
Port	Enter the port on which the database listens. The default port for DB2 is 50000.
User Id	Enter the user name of the database account you want Opportunity Detect to use to access to this database.
Password, Confirm Password	Enter the password for the account you entered in the **User ID** field.

Table 10. Server Groups page: Data Source Connectors tab

Field	Description
Add	Click to open a panel where you can select the type of connector to add. The options are `Table Connector` and `File Connector`. If you have installed Fix Pack 9.1.1.2 for IBM Opportunity Detect and IBM Interact Advanced Patterns, this button is different, as follows: When you click the button, you select the type of connector to add. The options are: • `File` • `Queue` • `Table` **Note:** For additional information on setting up queue data source connectors, see Chapter 5, "Real time processing in Opportunity Detect," on page 35.
Table Connector	
Name	Enter a descriptive name for this table connector. It is a good practice to align this name with the name of the database you plan to associate with the connector.

Table 10. Server Groups page: Data Source Connectors tab (continued)

Field	Description
Type	Select a type from the drop-down list. The options are: • State • Profile • Outcome • Expanded Outcome
Table Name	Enter the name of the table as shown in your database management client. This must exactly match the name as shown in the client. Case-sensitive.
Description	Enter a description of the table.
Sharable	Select this checkbox if you want to be able to use this table connector on different server groups.
File Connector	
Name	Enter a descriptive name for the file connector.
File Name	Click the link in this field to open a pop-up window, and complete the following fields. • Name - Enter the base file name for this flat file. Permitted characters are ASCII letters, numbers, and underscore. Do not use spaces or any other special characters. • Audience Level - Select from a drop-down list of available audience levels. Opportunity Detect constructs a complete file name from the base name you provide. The name of your file must match the name that Opportunity Detect constructs.
Description	Enter a description of this file.
File Encoding	Select the encoding used in the file from the drop-down list. Options are: • Chinese Traditional (Big 5) • Unicode (Little endian) • Unicode (Big endian) • Western European (ISO) • Central European (ISO) • Latin 3 (ISO) • Latin 9 (ISO) • Korean (EUC) • Chinese (GB 18030) • Unicode (UTF-7) • Unicode (UTF-8)

Table 10. Server Groups page: Data Source Connectors tab (continued)

Field	Description
Date Locale	Select from a drop-down list of supported locales. Options are: • English (United States) • Chinese (China) • English (United Kingdom) • French • German • Italian • Japanese (Japan) • Korean • Portuguese (Brazil) • Russian • Spanish • vThai
Currency	Select from a drop-down list of supported locales. Options are the same as for **Date Locale**.

Table 11. Server Groups page: Server Groups tab

Field	Description
Add	Click to open a panel where you can define a server group.
Properties tab	
Name	Enter a descriptive name for the server group.
Stream Instance	Select a stream instance ID. See your administrator if you are not sure which one to select.
Usage	Enter a brief description of how this server group is used. For example, Production or Development.
For Production	Select this checkbox if you want to restrict user's ability to run workspaces on this server, based on their permissions.
Servers tab	
Fully Qualified Name	Double-click in the **# of Engines** column to set the number of Opportunity Detect engines to run on this machine.
Database tab	
Select Runtime database connection	The run time database holds the tables where your run, run details, and run status data is stored. Select the run time database that you want this server group to be able to access.
Select database connections for table connector mapping	The database connections you check here are the ones that are available when you map a table data source to a connector in the Table Data Source Connector Mapping window.
Data Source Mapping tab	
List of datasources	Click the name of a data source to open a pop-up window where you can map the data source to a connector. Only sharable table connectors are available for mapping in the server group page. A connector can be mapped to only one data source.
Data Source Mapping tab: Table data source connector mapping	

Table 11. Server Groups page: Server Groups tab (continued)

Field	Description
Data Source name	This is a read-only field that contains the name of the data source you clicked to open this window.
Connector	Select from a list of previously configured connectors.
Database connection	Select from a list of previously configured database connections.
Data Source Mapping tab: File data source connector mapping	
Data Source name	This is a read-only field that contains the name of the data source you clicked to open this window.
Connector	Select from a list of previously configured file data source connectors.

Related concepts:

"About data source connectors"

"Audience level codes" on page 11

Related tasks:

"Configuring server groups and data source connections" on page 17

"Configuring audience level codes" on page 12

About data source connectors

Data source connectors enable the system to connect to the transaction, profile, State History, and Outcome data sources that you have defined. The types of data source connectors are described in this section.

Note: For Outcomes, the data source connector you use determines the format of the Outcome data.

File Connectors

Use flat file data source connectors to connect to transaction and profile data that is in file format.

Table Connectors

Use table data source connectors to connect to the following data sources.

• Profile data that is in database table format

• All State History tables

• Outcome data that you want to be written to database tables

You have the option to make table connectors sharable. The sharable feature works as follows.

• A sharable connector can be used in more than one server group.

• Sharable connectors can be mapped on the Data Source Mapping tab of the Server Groups page and on the Deployment tab of a workspace. Table connectors that are not sharable can be mapped only on the Deployment tab of a workspace.

For example, you might want to have connectors for Outcome and State History tables that are not shared and can be used only by the deployment configuration

used for production. You might also want sharable table connectors so you can map them to the same Outcome and State History tables for all test workspaces.

Regardless of whether a connector is sharable, a table connector can be mapped to only one data source.

There are four types of table data source connectors.

State

Used to connect to your State History tables.

Profile

Used to connect to your profile data that is in database table format.

Outcome

Used when you want Outcome data to be stored in database tables in the XML format used by previous versions of Opportunity Detect.

Expanded Outcome

Used when you want Outcome data to be stored in database tables in a form that external systems or IBM Campaign can use more easily than the format provided by the Outcome connector.

Web Service

You can use the Web Service connector to connect to transaction and Outcome data coming from or being sent to the Opportunity Detect web service.

If you use the web service for transaction data and you also use profile data, the profile data must be in database table format.

You do not have to create this type of connector. It exists in the system by default.

For details about the Java classes that must be developed to use the Web Service data source connector, see "Web Service data source connector for input and output in Opportunity Detect" in the *IBM Opportunity Detect Administrator's Guide*.

Queue

If you have applied Fix Pack 9.1.1.2 for IBM Opportunity Detect and IBM Interact Advanced Patterns, you can use this connector for real time operation.

For additional information on setting up Queue data source connectors, see "Real time processing in Opportunity Detect" in the *IBM Opportunity Detect Administrator's Guide*.

Default TCP Connector

Used by the system with IBM Interact Advanced Patterns when integration with IBM Interact is implemented.

You do not have to create this type of connector. It exists in the system by default.

Example of data source connector configuration

The following diagram illustrates a possible data source configuration, showing the relationship between database connections, data source connectors, and the data sources used in Opportunity Detect.
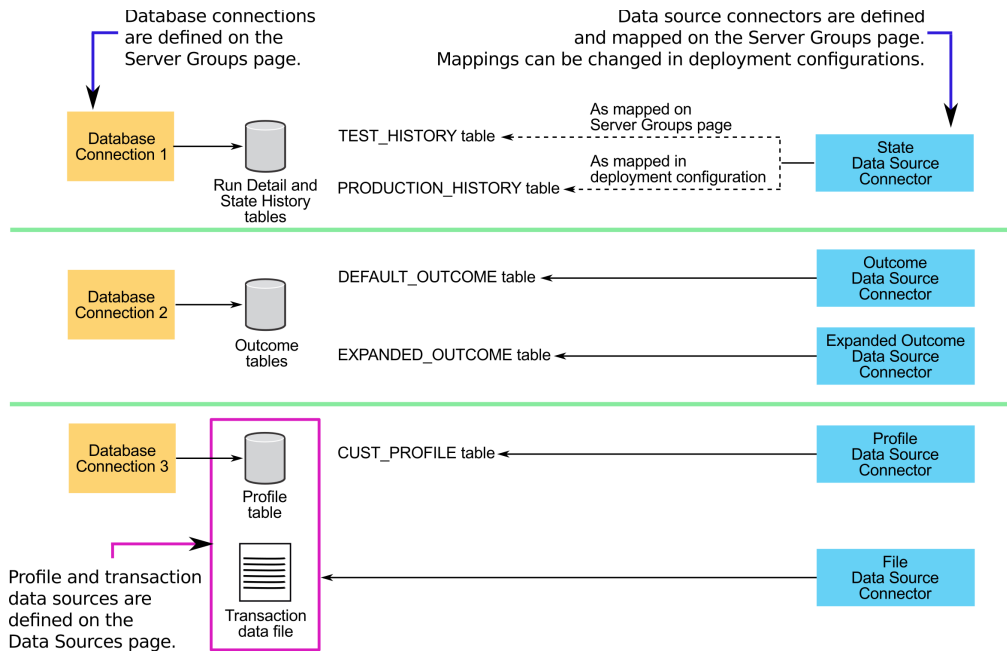
Related concepts:

Chapter 6, "Web Service connectors for input and output in Opportunity Detect," on page 45

Related tasks:

"Configuring server groups and data source connections" on page 17

Related reference:

"Fields on the Server Groups page" on page 19

Naming requirements for profile and transaction files

As part of the process for setting up server groups, you configure a file connector on the Data Source Connectors tab on the Server Groups page. The system uses the name you specify to generate a file name. Ensure that the names of your transaction and profile feed files exactly match the name shown on the Data Source Connectors tab on the Server Groups page.

When you define file data source connectors for server groups, you supply a name. The system expects the names of transaction and profile feed files to contain this name plus additional information, as follows. In this example, *data_source_name* is the name you enter when you define the data source connector.

`Detect.`*audience_code*`.`*data_source_name*`.`*date*

You must use this name format for the transaction and profile feed files that correspond to the data sources that you configure in Opportunity Detect.

Permitted characters for the name you enter are ASCII letters, numbers, and underscore. Do not use spaces or any other special characters

Date portion of the name

The *date* portion of the file name must have this format: `YYYYMMDDhhmmss`

Names of flat files must contain a date.

The date determines the order in which the files are processed. Files with the same timestamp are processed simultaneously. If you use a profile, you must create one for each set of files with a different timestamp.

During ramp-up, you might process multiple transaction files with different dates in a single run. If you are using profile files, you must include different profile files named with dates that match the date in the names of each of the transaction files.

How flat files are processed

Opportunity Detect uses the *data_source_name* portion of the file name to identify multiple instances of a transaction data source.

You might have a series transaction files, all with exactly the same fields, containing batches of transactions created at different times and using different timestamps in their names. Every time you run a trigger system that uses this data, a file with a different date stamp and/or timestamp would be used.

Data requirements for profile and transaction files

Profile and transaction files have a specified format.

Format requirements

Flat files must have the following format.
- One header record followed by zero or more records.
- The first field must be the audience code.
- All record lines must use the pipe (vertical line: |) delimiter between fields, but not before the first field in the line or after the last field in the line.
- A pair of delimiters next to one another indicates the absence of data for the field positioned between those two delimiters. If the first field of the record contains no data, then the line begins with the first delimiter, and if the last field contains no data, then the line ends with the last delimiter.
- The pipe delimiter must not be used within data fields.

Date field requirement

For transaction files only, one field must be a timestamp field.

Audience ID field requirement

One field in all data sources must be an ID field used for an audience level, and this must be the first field in each record in the file.

Sort order requirement

The records in a flat file must be sorted by audience ID first, then by transaction timestamp.

Additional considerations

Note the following considerations.
- Opportunity Detect cannot match fields with NULL values when performing comparisons. You must substitute either a single printable character or a character string for the NULL values if you want to use NULL as a condition in comparisons.
- Date fields must be in system-readable date format even if you want those dates to be empty (NULL). You can represent NULL dates as either very old dates or very future dates .
- The date format in date fields must match the date format that is associated with the date locale used in the file connector.
- Data source field names should not be assigned so that one name can be found within another. For example, the name `date` can be found within the name `firstdate`.

Transaction and profile file examples

Note: The following examples are formatted with constant widths for readability. Actual transaction and profile files must not use constant widths.

Here is an example of a simple transaction file.
```
ID      |NAME    |CALLED_NUMBER|CALL_LENGTH|TRAN_DATE_TIME
001234|David   |732-123-4567 |15         |2012-02-10 09:12:33
001234|David   |732-111-5555 |48         |2012-02-10 10:11:50
002941|Jeremiah|732-777-8888 |40         |2012-02-10 11:22:44
005555|Anthony |732-333-4444 |27         |2012-02-10 03:01:02
005555|Anthony |732-32-8945  |121        |2012-02-10 10:12:30
005555|Anthony |973-597-0022 |2          |2012-02-10 19:00:21
006789|Tom     |732-111-2222 |4          |2012-02-10 06:54:01
```

Here is an example of a simple profile file.
```
ID      |AGE|ZIP
001234|25 |11111
002941|55 |22222
005555|31 |33333
006789|60 |44444
100382|18 |55555
```

Profile data

The following requirements apply to all profile data sources used in trigger systems.
- Profile data is not required. However, if profile data is not defined, all the data used in a trigger system (including any required profile data) must be included in the transaction data.
- When profile data is defined, for every customer represented in your transaction file, there must be a single record for that customer in your associated profile.

When a transaction file is associated with a profile flat file, the system ignores transaction records without an associated profile record, and ignores profile records without an associated transaction record. In both cases, these orphan records affect processing efficiency.

Chapter 4. Automating tasks using the RemoteControlCLI utility

The Opportunity Detect command line utility, RemoteControlCLI (CLI), allows you to automate the management of deployment configurations and batch runs. You can use the CLI to perform the same actions you can perform on the Deployment & Batch Run tab of a workspace.

The RemoteControlCLI utility and sample batch or shell scripts are all located in the cli directory under your Opportunity Detect design time installation.

The following sample scripts are provided with your installation.

Table 12. Command line scripts

Script	Usage
encode	Encrypt the password that the CLI uses to connect to the design time database.
deploy	Deploy a deployment configuration.
stop	Stop the deployment of a deployment configuration.
start	Re-start the deployment of a deployment configuration.
startBatch	Start a batch run by starting a deployed deployment configuration.
stopBatch	Stop a batch run by stopping a deployed deployment configuration.

Prerequisites

The following are prerequisites for using the Opportunity Detect command line scripts.

- The computer where you run the scripts must have network access to machines where the Opportunity Detect design time and run time components are installed.
- Java version 6 must be installed on the machine where you run the scripts.
- You must set the JAVA_HOME system environment variable. You can do this either on the machine where you run the scripts, or by editing the scripts to set it temporarily when the scripts run.

Password security

The information the CLI uses to connect with the design time database is saved in the RemoteControlCLI.xml file, located in the cli directory. This information includes the user name and password for an account in the database.

You can encrypt the password that is stored in this file, to avoid storing the password in clear text.

Setting up the database connection for the Opportunity Detect CLI

You edit the RemoteControlCLI.xml file to enable the CLI to connect with the Opportunity Detect design time database.

Before you begin

Obtain the following information about your design time database.

- The name and URL of the database used with your design time installation.
- The user name and password for an account with Administrator privileges in the design time database.
- The URL of your design time installation
- The URL and class name for the database driver used with your design time database.

Procedure

1. Open the `RemoteControlCLI.xml` file, located in the `cli` directory under your Opportunity Detect design time installation.
2. Run the `encode` script to encrypt the password for the database account. Copy the resulting string and use it as the value for the `ConnectionPassword` key in the `RemoteControlCLI.xml` file.

   This step is optional, but provides the best security. If you do not encrypt the password, it is stored in clear text in the `RemoteControlCLI.xml` file.

   If you use an encrypted password, set the value of the `ConnectionPasswordEncrypted` key to `True`.
3. Use the database information you obtained to complete the values in the rest of the keys in the `RemoteControlCLI.xml` file.
4. Save and close the `RemoteControlCLI.xml` file.

Results

You can now run the CLI commands.

Related reference:

"RemoteControlCLI XML reference"

RemoteControlCLI XML reference

The values you enter in the `RemoteControlCLI.xml` file enable the CLI to connect with your Opportunity Detect design time database.

Table 13. Reference for CLI XML keys

XML key	Value
RemoteControlURL	The URL for your Opportunity Detect design time installation. Example: `http://example.com:8080/axis2/services/RemoteControl`
ConnectionDriverName	The class name of the database driver used with your Opportunity Detect design time installation. Example: `com.ibm.db2.jcc.DB2Driver`
ConnectionURL	The URL of the driver used with your Opportunity Detect design time installation. Example: `jdbc:db2://example.com:50000/` `Detect91:retrieveMessagesFromServerOnGetMessage=true;`

Table 13. Reference for CLI XML keys (continued)

XML key	Value
ConnectionUserName	The user name of an account in the Opportunity Detect design time database. Example: `example_user_name`
ConnectionPassword	The password of the same account used for the ConnectionUserName. Example: `example_password`
ConnectionPasswordEncrypted	A flag that indicates whether the password value used for ConnectionPassword has been encrypted using the `encode` script. If you use an encoded password, set this value to `True`.
Schema	The URL of the Opportunity Detect design time database. Example: `http://example.com:8080/axis2/services/RemoteControl`

Related tasks:
"Setting up the database connection for the Opportunity Detect CLI" on page 29

RemoteControlCLI command reference

The RemoteControlCLI utility supports the following commands and options.

Syntax

`RemoteControlCLI deploy -d *deployment configuration ID* -v *version number*`

`RemoteControlCLI start -d *deployment configuration ID* -v *version number*`

`RemoteControlCLI stop -d *deployment configuration ID* -v *version number*`

`RemoteControlCLI startBatch -d *deployment configuration ID* -v *deployment configuration version number* -w *workspace ID* -fp *path to feed files*`
`[-am *Off|EndOfDay|EndOfRun* ]`
`[-ll *Off|Fatal|Error|Warn|Info|Debug|Trace*]`
`[-n *notification file name*]`
`[-r] [-ri] *ID of the batch run to recover*`

In addition, if you have applied Opportunity Detect FixPack 9.1.1.2, the following options are available with the `startBatch` command.
`[-im *Off|On*]`
`[-id *inactivity date in YYYY-MM-DD HH:MM:SS format*]`

`RemoteControlCLI stopBatch -d *deployment configuration ID* -v *version number*`

You can obtain the help for this utility by entering **-h** by itself or with any of the above commands.

Where to find the deployment configuration ID and version number

You can obtain the version number and ID of the deployment configuration on the Deployment tab of the workspace. Select the deployment configuration and select

the History tab in the panel that opens. Look under the Version and Message columns for the version and configuration ID. When you run the command, increment the deployment configuration version by 1.

For example, if the deployment version is currently n, you would specify a deployment configuration version number of n+1.

Command and option details

RemoteControlCLI deploy -d *deployment configuration ID* **-v** *deployment configuration version number*

Deploy a deployment configuration to the Streams server. For a first time deployment, you must use the Deployment & Batch Run tab of the workspace. You can perform subsequent deployments using the CLI.

RemoteControlCLI start -d *deployment configuration ID* **-v** *deployment configuration version number*

Start a deployment configuration.

RemoteControlCLI stop -d *deployment configuration ID* **-v** *version number*

Stop a deployment configuration.

RemoteControlCLI startBatch -d *deployment configuration ID* **-v** *deployment configuration version number* **-w** *workspace ID* **-fp** *path to feed files*

Start a batch run.

The non-required options for this command allow you to set the same parameters that are available on the Deployment & Batch Run tab, as follows.

- **-am** *Off|EndOfDay|EndOfRun*

  Set the artificial transaction mode
- **-ll** *Off|Fatal|Error|Warn|Info|Debug|Trace*

  Set the logging level for all Streams components
- **-n** *notification file name*

  Set the file used to send notifications about run success or failure
- **-r**

  Run in recovery mode (requires **-ri**)
- **-ri** *ID of batch run*

  Set the ID of the batch run to recover (used with **-r**)

In addition, if you have applied Opportunity Detect FixPack 9.1.1.2, the following options are available.

- **-im** *Off|On*

  Turn inactivity mode on or off. If you do not set this mode, the default is `off`.
- **-id** *inactivity date in YYYY-MM-DD HH:MM:SS format*

  Set the date for the inactivity mode (used with **-im**). The date that you specify is the date that the system uses to look for any Forward Looking Inactivity events that need to be triggered.

RemoteControlCLI stopBatch -d *deployment configuration ID* **-v** *version number*

Stop a batch run.

Chapter 5. Real time processing in Opportunity Detect

The type of data source connectors that you use for your workspace determines whether it is processed in batch or real time mode. For real time mode, use a Queue or Web Service type of data source connector for transaction and Outcome data. Queue data source connectors are available if you have applied Fix Pack 9.1.1.2 for IBM Opportunity Detect and IBM Interact Advanced Patterns.

Data source connectors are mapped to data sources in two places. The default mappings are set when a server group is configured. However, these mappings are commonly changed when a deployment is configured.

Queue connectors allow Opportunity Detect to operate with either the IBM WebSphere or Active MQ queue server.

Consider the following points when you are deciding whether to use queue connectors or the Web Service connector.

- The Queue type of connector retains messages in the event of a network or machine failure, or if you deploy a new version of a configuration deployment. In contrast, the Web Service drops messages in those cases.
- Queues handle spikes in demand more efficiently than the Web Service does.
- Both choices require some development and IT effort on the part of your organization.
  - To use the Web Service data source connector, your organization must develop code to receive and send the transaction and Outcome data. This is described in Chapter 6, "Web Service connectors for input and output in Opportunity Detect," on page 45.
  - To use the queue data source connector, your organization must develop code to send the transaction data, and you must install and maintain a queue server.

Related concepts:

Chapter 3, "Data source setup in Opportunity Detect," on page 9

Chapter 6, "Web Service connectors for input and output in Opportunity Detect," on page 45

Related reference:

"Transaction message example for the sending program"

"Outcome message example" on page 37

Transaction message example for the sending program

This section describes the data in the message that your sending program must produce when you use a queue data source connector for transaction data. It also provides a code sample for creating a transaction message in the required format.

Information required for the transaction message

When you use a queue connector for transaction data, the sending program must provide the following information in the data sent to the queue server.

Table 14. Required information for transaction data

Name	Value
AUDHASH	Hash for the audienceId generated using a standard hash function. The function must generate a positive value. Used in the header of a JMS message.
deploymentConfigurationId	You can obtain the ID of the deployment configuration on the Deployment tab of the workspace. Select the deployment configuration and select the History tab in the panel that opens. Look under the Message column for the configuration ID. Used in the header and the body of a JMS message.
inputVersion	The input version number entered on the Properties tab of the deployment configuration. Used in the header and the body of a JMS message.
AudienceLevel	The code assigned to the audience level on the **Settings > Detect Settings > Audience Levels** page. Used in the body of a JMS message.
audienceId	The customer ID associated with this transaction. Used in the body of a JMS message.
inputDataSourceName	The name of the data source to which the queue connector is mapped. Data sources are defined on the **Settings > Detect Settings > Data Sources** page. Used in the body of a JMS message.

transaction	Fields from your Transaction data source, as defined on the **Settings > Detect Settings > Data Sources** page.
	
	Used in the body of a JMS message.
	Note: The date must be expressed in milliseconds.
	
	Example:
	<pre>{"FLAG":true,
	"DATE":1408077610000,
	"AMT":1000.00,
	"NUMBER":10,
	"TRANCODE":"ATMD",
	"CURRENCY":1009.1,
	"ID":"cm00411"}</pre>

Sample code for creating a transaction message using the JMS API

Here is a sample code snippet for creating a JMS message for transaction data using the JMS API. Line breaks have been added for readability.

```
//JMS Header Fields
  MapMessage message = session.createMapMessage();
  message.setLongProperty("AUDHASH",hashcode);
   message.setStringProperty("deploymentConfigurationId",
       "eecdc1b4-5333-4076-9af6-dc6c2f932c0e");
   message.setLongProperty("inputVersion", 5);

//JMS Body Fields
   message.setString("deploymentConfigurationId",
```

```
            "eecdc1b4-5333-4076-9af6-dc6c2f932c0e");
    message.setInt("inputVersion", 5);
    message.setString("audienceLevel", "audienceLevel");
    message.setString("audienceId", "cm00409");
    message.setString("inputDataSourceName", "DataSourceNam");
    message.setString("transaction", "{\"Date\":1408077610000,\"Amount2\":1000.00,\
        "Type\":\"E1a\",\"Amount1\":1009.1,\"ID\":\"cm00409\"}");
```

Example of a transaction message

This is sample input message that was sent to an Active MQ queue server. Line
breaks have been added for readability.

```
ActiveMQMapMessage {commandId = 0, responseRequired = false,
messageId = ID:ADMINIB-3C6H892-58046-1428399823708-0:0:1:1:1,
originalDestination = null,
originalTransactionId = null,
producerId = null,
destination = queue://amqinput,
transactionId = null,
expiration = 0,
timestamp = 1428399824137,
arrival = 0,
brokerInTime = 0,
brokerOutTime = 0,
correlationId = null,
replyTo = null,
persistent = false,
type = null,
priority = 4,
groupID = null,
groupSequence = 0,
targetConsumerId = null,
compressed = false,
userID = null,
content = org.apache.activemq.util.ByteSequence@60945fea,
marshalledProperties = null,
dataStructure = null,
redeliveryCounter = 0,
size = 0,
properties = {inputVersion=5,
    deploymentConfigurationId=eecdc1b4-5333-4076-9af6-dc6c2f932c0e,
    AUDHASH=13670},
    readOnlyProperties = false,
    readOnlyBody = false, droppable = false}
ActiveMQMapMessage
{ theTable =
    {audienceLevel=audienceLevel,
    audienceId=cm00409,
    inputVersion=5,
    transaction={"FLAG":true,
       "DATE":1408077610000,
       "AMT":1000.00,
       "NUMBER":10,
       "TRANCODE":"ATMD",
       "CURRENCY":1009.1,
       "ID":"cm00411"}
  }
```

Outcome message example

This section describes the data in the message that the system produces when you
use a queue data source connector for Outcomes.

Outcome message format using the queue connector

When you use a queue connector for the Outcome data source, the message has the format shown in the following table.

In addition, the following rules apply.
- System-defined field names are camel case.
- Data types start with a capital letter.
- Valid data types are String, Integer, Double, Currency, Boolean, Date, and Tabular.
- Values for Date and String data types are surrounded with double quotes.
- Values for Integer, Double, Currency, Boolean, and Tabular data types are **not** surrounded with double quotes.

Table 15. JMS message format for Outcome data

Name	Value
producerName	OpportunityDetect
clientID	If you have multiple Opportunity Detect installations, enter the ID of the installation where this queue connector is used, as defined in the **clientId** property under the **IBM Opportunity Detect And Interact Adv Patterns	System** category on the **Settings > Configuration** page. If you have a single Opportunity Detect installation, this value should be `default`.
actionName	The name of the Action component that produced the Outcome.
workspaceName	The name of the workspace.
deploymentName	The name of the deployment that is running the workspace.
outputVersion	The output version number entered on the Properties tab of the deployment configuration.
data	The data you have specified in the Action component for your Outcome, plus the data that is included in every Outcome.
	Outcome data always includes the audience ID, audience level, the ID of the Action component, and a time-stamp.
	Time-stamps have this format:
	`yyyy-MM-dd HH:mm:ss`
	Additional fields that you specify in the Action component are output with the label you give the field in the Action component, the value, and the data type. This data can be a single value (scalar), or a row in a data record stored in a Container or Select component (tabular).

Example of the Outcome message using the queue connector

Here is an example of a message produced by the system when you use a Queue data source connector for an Outcome. Line breaks have been added for readability.

```
producerName: OpportunityDetect,
clientID :default,
actionName:  "Name of Action component"
workspaceName :  "Name of workspace"
deploymentName : "Name of deployment"
outputVersion: "Version",

data = [
  {
    "message": {
        "value": "Pass",
        "dataType": "String"
        }
  },
  {
    "audienceId": {
        "value": "acc333",
        "dataType": "String"
        }
  },
  {
    "audienceLevel": {
        "value": "audienceLevelvalue",
        "dataType": "String"
        }
  },
  {
    "componentId": {
        "value": "AA123344555666",
        "dataType": "String"
        }
  },
  {
    "firingtime": {
        "value": "2014-04-01 05:40:01",
        "dataType": "Date"
        }
  },
  {
    "fieldName1": {
        "value": 333,
        "dataType": "Integer"
        }
  },
  {
    "fieldName2": {
        "value": "2014-04-01 05:40:01",
        "dataType": "Date"
        }
  },
  {
    "fieldName3": {
        "value": {
            "header" : [
                {
                    "tablefield1": {
                     "dataType": "String"
                    },
                }
                {
                    "tablefield2": {
                     "dataType": "Date"
                    }
                },
            ]
            rows : [
                {
```

```
                                  "values" : ["value1","value2"]
                               },
                               {
                                  "values" : ["value1","value2"]
                               },
                               {
                                  "values" : ["value1","value2"]
                               },
                          ]
                  },
                  "dataType": "Tabular"
               }
          }
     ]
```

Queue processing setup roadmap

This roadmap describes how to set up real time processing using queue data source connectors in Opportunity Detect.

Table 16. Real time processing setup roadmap

Step	Where to find details
Ensure that you have met the prerequisites.	"Prerequisites for using queue connectors"
Decide how many queue data source connectors to create and how you want to map them.	"Planning your queue connectors" on page 41
Copy client libraries from your queue server to your IBM InfoSphere Streams machine and set environment variables.	"Installing MQ client libraries" on page 41
Set up data sources in Opportunity Detect for the transaction data that you want to process in real time mode.	Chapter 3, "Data source setup in Opportunity Detect," on page 9
Create queue data source connectors for the transaction and Outcome data sources that you want to process in real time mode.	"Creating queue connectors" on page 42
Map your queue data source connectors to your transaction and Outcome data sources.	"Mapping queue connectors to data sources" on page 43

Prerequisites for using queue connectors

Ensure that the following prerequisites are met when you use queue connectors.

1. You must have a functioning IBM WebSphere or Active MQ queue server with queues configured as required for your deployment configuration.
2. You must set up one queue for each queue data source connector you plan to use.
3. The queue server must be accessible from the Opportunity Detect run-time server (the Streams server).
4. The Streams system user, typically the streamsadmin user, must have read and write permissions on the queue server.
5. Your organization must develop the necessary sending program to push transaction data into the queue.
6. If you have multiple Opportunity Detect installations, set an ID for each one in the **clientId** property under the **IBM Opportunity Detect And Interact**

Advanced Patterns | System category on the **Settings > Configuration** page. If you have a single Opportunity Detect installation, you do not have to set any values for this property.

Planning your queue connectors

Consider the following points when you create and map queue data source connectors.

- Your queue server must not be configured to use SSL, because Opportunity Detect does not support SSL communication with the queue server.
- Ensure that at least one queue is created on your queue server for each queue connector that you plan to use.
- A workspace has only one Outcome data source, regardless of how many Action components it contains.
- You must map one queue connector to each transaction and Outcome data source. Optionally, you can map more than one queue connector to the transaction and Outcome data sources.
- A non-shared queue connector can be used by only one deployment. The system enforces this constraint.
- A non-shared connector cannot be mapped to a data source when you configure a server group. It can be mapped in a deployment configuration only.
- A shared queue connector can be used across multiple deployment configurations.
- Shared connectors are mapped when you configure a server group. You can change this mapping in a deployment configuration.
- For Outcomes, you would typically use multiple non-shared queues if you have multiple consumers of the Outcomes, and use a shared queue if you have only a single consumer of the Outcomes.
- For testing, it is typical to use shared data source connectors for your transaction data, to reduce the time required to set up separate queues. Because the message includes the configuration ID, the transaction data is sent to the appropriate queue. For production, you should generally use non-shared connectors for your transaction data.

Installing MQ client libraries

To use the queue data source connector, you must install the message queue client and set the library path in an environment variable on your Streams server.

Procedure

1. Obtain the client for your message queue server.
   - For WebSphere, download the client here:

     http://www-01.ibm.com/software/integration/wmq/clients/
   - For Active MQ, download the client here:

     http://activemq.apache.org/download.html
2. Install the message queue client one every machine where IBM InfoSphere Streams is installed.

   See the documentation for your queue server for full instructions on installing the client.
3. On all machines where Streams is running, edit the Streams `.bashrc` file for the streamsadmin user as follows.

The `.bashrc` file is typically located in home directory for streamsadmin user.

a. Set the `JAVA_HOME` environment variable to use Streams Packaged Java. For example:

```
export JAVA_HOME=/home/streamsadmin/InfoSphereStreams/java
```

b. Set additional environment variables, as follows.

- Websphere MQ

  Set the `STREAMS_MESSAGING_WMQ_HOME` environment variable to the path where you installed the client library. For example:

  ```
  export STREAMS_MESSAGING_WMQ_HOME="/opt/mqm"
  ```

- ActiveMQ

  Set the `STREAMS_MESSAGING_AMQ_HOME` environment variable to the location where Apache ActiveMQ is installed. For example:

  ```
  export STREAMS_MESSAGING_AMQ_HOME="/home/streamsuser/
  ApacheActiveMQ"
  ```

4. Restart the Streams instance.

Creating queue connectors

Follow this procedure to create a queue connector.

Before you begin

You must have transaction data sources defined on the **Settings > Detect Settings > Data Sources** page.

Procedure

1. Navigate to the **Settings > Detect Settings > Server Groups** page.
2. On the Data Source Connectors tab, click **Add a new Data Source Connector** and select **Queue**.
3. Complete the fields in the panel that opens.

Related concepts:

Chapter 3, "Data source setup in Opportunity Detect," on page 9

Fields on the queue connector panel

If you are unsure of how to complete the fields on the queue connector panel, use the information provided here.

Table 17. Fields on the queue connector panel

Field	Description
Name	A descriptive name for the queue connector.
Queue type	Select the type of queue server you are using: **Active MQ** or **Websphere MQ**.
Queue Name	Name of the queue as configured on the queue server.
Connection URL	An example value appropriate for the queue type you selected is provided.
Connection Factory	An example value appropriate for the queue type you selected is provided.

Table 17. Fields on the queue connector panel (continued)

Field	Description
Connection Factory Identifier	An example value appropriate for the queue type you selected is provided.
User Id	The user name for the account that the system uses to access the queue server.
Password	The password for the account that the system uses to access the queue server.
Confirm Password	The password for the account that the system uses to access the queue server. Active MQ only.
Description	A description for this queue connector.
Sharable	Select this checkbox if you want to be able to use this queue connector on more than one server group.

Mapping queue connectors to data sources

Follow this procedure to map a queue connector to a data source in a deployment configuration.

Before you begin

- You must have transaction data sources defined on the **Settings > Detect Settings > Data Sources** page.
- You must have queue connectors defined on the Data Source Connectors tab on the **Settings > Detect Settings > Server Groups** page.

About this task

This procedure assumes that you are mapping the transaction and Outcome data sources to queue connectors in a deployment configuration, overriding the default server group settings.

Procedure

1. Create a deployment configuration as described elsewhere in this guide.
2. On the Data Source Mapping tab, select the transaction data source you want to change.
3. Clear the **Server Group Default** checkbox.
4. In the Data Source Connector Mapping window, do the following.
   a. Select **Queue** as the **Connector Type**
   b. Select a queue connector.
5. Repeat steps 2-4, selecting any additional transaction and Outcome data sources you want to map.

Related concepts:

Chapter 3, "Data source setup in Opportunity Detect," on page 9

Chapter 6. Web Service connectors for input and output in Opportunity Detect

You can use a Web Service Connector for incoming transaction data, for Outcome data, or for both.

To use the Web Service Connector for either input or Outcome data, your organization must develop Java classes.

- When you specify the Web Service Connector for incoming transaction data, you develop a Java class that sends the data to an API over HTTP.
- When you specify the Web Service Connector for Outcome data, you develop a Java class that implements an interface provided by a servlet. You can write the Outcome data to a file, to a database, or you can develop your plugin to act like a web service.

This section provides the details required for this development effort.

Example use case

Here is an example of how a telephone company might use the Web Service connector for both input and Outcome data.

- A customer makes a call on her cell phone while on vacation.
- A call data record is sent to the Web Service connector.
- Opportunity Detect processes the transaction and detects an unusual event, in this case a call from outside the customer's data plan area.
- Opportunity Detect sends the Outcome data to two Web Service plugins.
  - One plugin updates profile variables used by Interact.

    This data improves the customer profile that Interact uses to evaluate effective offers when the customer visits the company's web site or calls customer service.
  - Another plugin alerts a company system, which sends an SMS offer to the customer's phone.

    For example, the message might offer to upgrade her data plan.

Block diagram

The following diagram illustrates the Web Service connector.
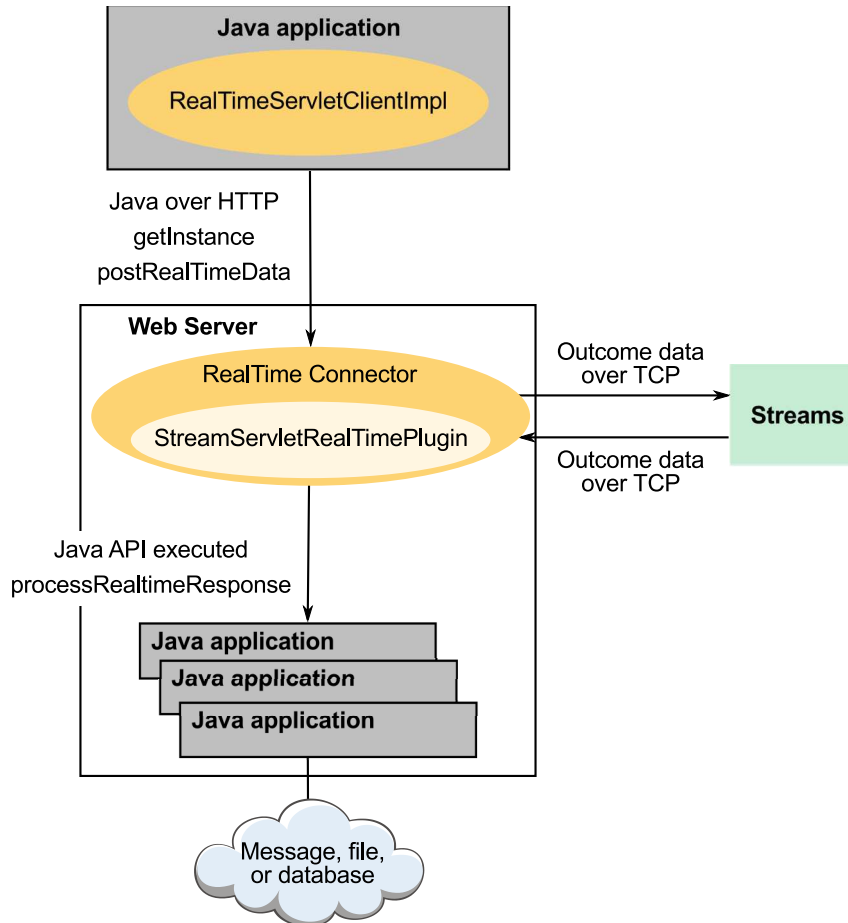
Location of the input jar files and the output class and plugin

All of the locations described in this section are under the run time installation of
Opportunity Detect.

- For input, your Java application must use the `RealTimeClient.jar` file, located
  here:

  `RealTimeConnector/wlp/usr/servers/RealTimeConnector/dropins/`
  `RealTimeConnector/WEB-INF/client`

  The following dependent .jar files must be included in the Java class path of
  your application. They are in the same location as the `RealTimeClient.jar` file.
  - `JSON4J.jar`
  - `log4j.jar`
- For Outcomes, your Java application must use the `StreamServletRealTimePlugin`
  class, located here:

  `RealTimeConnector/wlp/usr/servers/RealTimeConnector/dropins/`
  `RealTimeConnector/WEB-INF/classes/com/ibm/unica/detect/interact/servlet`

  Place the plugin you develop here:

  `RealTimeConnector/wlp/usr/servers/RealTimeConnector/dropins/`
  `RealTimeConnector/WEB-INF/lib`

Timestamp format

Timestamps in input data must be expressed in milliseconds.

Data types

No objects are used for either input or Outcomes using the Web Service connector. Only primitive data types are used.

Locale

Locale is handled by the Opportunity Detect engine.

Monitoring

Because Opportunity Detect is built on Streams, you can use the built-in monitoring capabilities of the Streams server to monitor processing. See the Streams documentation for details.

Related concepts:

"About data source connectors" on page 22

Class for input: RealTimeClient

Reference for the `RealTimeClient` class, used for sending input data to the Web Service connector.

You must compile your implementation against `RealTimeClient.jar`.

The `RealTimeClient` class has two methods:

- `getInstance`
- `postRealTimeData`

Code example

Here is a code example. The code opens an HTTP connection to the Interact Connector servlet and sends transaction data.

```
private static
 String
  urlString ="http://detectvm2.emmlabs.ibm.com:8282/servlets/StreamServlet";
RealTimeServletClient
  streamServletRealTimeClient = RealTimeServletClientImpl.getInstance(urlString);
String
 deploymentConfigurationId = "e0692797-70fc-4a34-9232-9f7df3e9bb68";
int version = 5;
String audienceId = "Acc09";
String inputDataSourceName = "RealTimeDataSource";
Map inputDataSourceFields = new Hashtable();
 // The names of the keys below are the names of datasource fields.
 // Start of transaction
 inputDataSourceFields.put("Boolean", true);
 inputDataSourceFields.put("Timestamp", new Date().getTime());
 inputDataSourceFields.put("Double", 99.9);
 inputDataSourceFields.put("Integer", 56);
 inputDataSourceFields.put("String", "Hello");
 inputDataSourceFields.put("AudienceId", audienceId);
 // End of transaction
```

```
streamServletRealTimeClient.postRealTimeData
 (deploymentConfigurationId, version, audienceId,
  inputDataSourceName, inputDataSourceFields);
```

In the example, the fields in the transaction hash table are defined on the **Detect Settings > Data Sources** page as shown below.

Fields

Name	Display Name	Description	App Type	Data Type	Name Value List
AudienceId	AudienceId		Audience Id : Cust(	String	
Timestamp	Timestamp		Transaction Date	Date	
Integer	Integer		Transaction Attrib	Integer	
Double	Double		Transaction Attrib	Double	
String	String		Transaction Attrib	String	
Boolean	Boolean		Transaction Attrib	Boolean	

Additional details are provided in the remainder of this section.

getInstance

```
getInstance (urlString, connectTimeout, readTimeout)
```

urlString

URL of the Interact connector. The value is the same as used for the **Opportunity Detection | System | Real Time Connector | ServerURL** configuration property, shown on the **Settings > Configuration** page.

postRealTimeData

```
postRealTimeData (deploymentConfigurationId, version, audienceId,
inputDataSourceName, inputDataSourceFields)
```

deploymentConfigurationId

The deployment configuration ID. For example `e0692797-70fc-4a34-9232-9f7df3e9bb68`.

To obtain the version number of a deployment configuration, select the Deployment tab of a workspace, select the relevant deployment configuration, and select the History tab in the panel that opens. Use the configuration ID listed in the **Message** column.

audienceId

The value in the audience ID field in your customer data source.

This field must be defined as an audience field on the Data Sources page in Opportunity Detect.

version

The version of the deployment.

To obtain the version number of the deployment, select the Deployment tab of the workspace, select the relevant deployment configuration, and select the History tab in the panel that opens. Use the most recent version listed in the Version column.

If you re-deploy the deployment configuration that you are using, you must update this version parameter in your Java program.

inputDataSourceName

The name of the data source that has been created on the Data Sources page in Opportunity Detect for real time transactions.

inputDataSourceFields

A map of fields in the data source as defined on the Data Sources page in Opportunity Detect.

Class for Outcomes: StreamServletRealTimePlugin

Reference for the `StreamServletRealTimePlugin` class, used for sending Outcome data to the Web Service connector.

Editing the Streams servlet web.xml

To make a plugin executable, you must edit the Streams servlet `web.xml` file.

The Streams servlet `web.xml` file is located under your Opportunity Detect run time installation in the following path:

```
RealTimeConnector/wlp/usr/servers/RealTimeConnector/dropins/
RealTimeConnector/WEB-INF.
```

Add the StreamServletRealTimePlugin class name to the following section of the Streams servlet `web.xml` file, as follows. Note that line breaks have been added to fit the example on the page.

```
<init-param>
  <param-name>PluginClassNames</param-name>
    <param-value>com.ibm.unica.detect.interact.servlet.
    StreamServletRealTimePluginImpl.com.company.ClassName</param-value>
  </init-param>
```

Code examples

Your plugin must implement the `StreamServletRealTimePlugin` interface, which has only one function: `processRealtimeResponse`.

Note: In the following examples, line breaks have been added to fit the examples on the page.

Here is an example of code that implements the interface.

```
package com.ibm.unica.detect.interact.servlet;
 import java.util.Map;
  @SuppressWarnings("rawtypes")
    public interface StreamServletRealTimePlugin {
    public void processRealTimeResponse(
     final String audienceId,
     final String audienceLevel,
```

```
               final String componentId,
               final long timestamp,
               final Map response);
             }
```

Here is an example of a plugin implementation that prints all output parameters into the servlet log file.

```
package com.ibm.unica.detect.interact.servlet;
 import java.util.Map;
 import org.apache.log4j.Logger;
 @SuppressWarnings("rawtypes")
  public class StreamServletRealTimePluginImpl
   implements StreamServletRealTimePlugin {
  private static Logger logger =
   Logger.getLogger(StreamServletRealTimePluginImpl.class);
  public void processRealTimeResponse(
  final String audienceId,
  final String audienceLevel,
  final String componentId,
  final long timestamp,
  final Map response)
  {
  logger.info("audienceLevel: " + audienceLevel + ";
  audienceId: " + audienceId + ";
  componentId: " + componentId + ";
  timestamp: " + timestamp + ";
  processRealTimeResponse: " + response);
   }
   }
```

processRealtimeResponse

processRealtimeResponse (audienceId, audienceLevel, componentId, timestamp,
response)

audienceId

The value in the audience ID field in your customer data source.

This field must be defined as an audience field in Opportunity Detect.

audienceLevel

The audience level code assigned in Opportunity Detect to the audience level used
in the transaction that produced the Outcome.

componentId

The ID of the Action component that produced the Outcome.

timestamp

The timestamp when the Action component fired its Outcome.

response

A map of the Outcome data that was specified in the **Message** and **Additional
Information** fields of the Action component.

Appendix. Opportunity Detect configuration properties

This section describes the Opportunity Detect configuration properties on the Configuration page.

Detect

Properties in this category specify the default locale for Opportunity Detect and Interact Advanced Patterns.

currencyLocale

Description

The locale that governs the way currency is displayed and stored in Opportunity Detect.

Default value

en_US

supportedLocales

Description

A list of the locales supported for data storage in this version of Opportunity Detect. Changes to this value have no effect and are not recommended.

Default value

de,en,fr,ja,es,ko,pt,it,zh

defaultLocale

Description

The assumed locale for all Opportunity Detect users. Changes to this value are not recommended.

Default value

[default-locale]

Detect | Navigation

Properties in this category specify values that are used internally to navigate among IBM EMM products.

welcomePageURI

Description

The Uniform Resource Identifier of the IBM Opportunity Detect index page. This value is used internally by IBM EMM applications. Changes to this value are not recommended.

Default value

/index.jsp

seedName

Description

Used internally by IBM EMM applications. Changes to this value are not recommended.

Default value

`Detect`

type

Description

Used internally by IBM EMM applications. Changes to this value are not recommended.

Default value

`Detect`

httpPort

Description

The port number that is used by the application server for connections to the Opportunity Detect application.

Default value

`7001`

httpsPort

Description

The port number that is used by the application server for secure connections to the Opportunity Detect application.

Default value

`7001`

serverURL

Description

The URL of the Opportunity Detect installation. Accepts either the HTTP or HTTPS protocol. You must use the domain and the machine name (rather than the IP address) when IBM EMM products are installed on more than one machine.

Important: If IBM EMM products are installed in a distributed environment, you must use the machine name rather than an IP address in the navigation URL for all of the applications in the suite.

Default value

[server-url]

logoutURL

Description

Used internally. Changes to this value are not recommended.

IBM Marketing Platform uses this value to call the logout handler of each registered application if the user clicks the logout link in IBM EMM.

serverURLInternal

Description

Used internally. Changes to this value are not recommended.

displayName

Description

Used internally. Changes to this value are not recommended.

Default value

```
Opportunity Detect
```

Detect | System | Streams Remote Control Web Service

The property in this category specifies the URL for the IBM InfoSphere Streams remote control web service.

ServerURL

Description

The URL for the IBM InfoSphere Streams remote control web service. For example, `http://IP_address:port/axis2/services/RemoteControl`.

Default value

```
http://[SRCSHost]:[SRCSPort]/axis2/services/RemoteControl
```

Detect | System | Real Time Connector

The property in this category specifies the URL for the web service used when Interact is integrated with Interact Advanced Patterns or when the Web Service connector is used for input data.

ServerURL

Description

The URL for the Interact web service. For example, `http://IP_address:port/servlets/StreamServlet`. This value is set to the value entered during installation. The default port number is 8282.

Default value

```
http://[RealTimeConnectorHost]:[RealTimeConnectorPort]/servlets/
StreamServlet
```

Interact Detect | System | Interact Design Service

The property in this category specifies the URL of the Interact web service.

ServerURL

Description

The URL for the Interact web service. For example, `http://IP_address:port/axis2/services/InteractDesignService`.

Detect | logging

The property in this category specifies the location of the Opportunity Detect log file.

log4jConfig

Description

> The location of the configuration file that Opportunity Detect uses for logging. This value is set automatically during installation, but if you change this path, you must restart the web application server to apply the change.

Default value

> `[absolute-path]/conf/detect_log4j.properties`

Before you contact IBM technical support

If you encounter a problem that you cannot resolve by consulting the documentation, your company's designated support contact can log a call with IBM technical support. Use these guidelines to ensure that your problem is resolved efficiently and successfully.

If you are not a designated support contact at your company, contact your IBM administrator for information.

Note: Technical Support does not write or create API scripts. For assistance in implementing our API offerings, contact IBM Professional Services.

Information to gather

Before you contact IBM technical support, gather the following information:
- A brief description of the nature of your issue.
- Detailed error messages that you see when the issue occurs.
- Detailed steps to reproduce the issue.
- Related log files, session files, configuration files, and data files.
- Information about your product and system environment, which you can obtain as described in "System information."

System information

When you call IBM technical support, you might be asked to provide information about your environment.

If your problem does not prevent you from logging in, much of this information is available on the About page, which provides information about your installed IBM applications.

You can access the About page by selecting **Help > About**. If the About page is not accessible, check for a version.txt file that is located under the installation directory for your application.

Contact information for IBM technical support

For ways to contact IBM technical support, see the IBM Product Technical Support website: (http://www.ibm.com/support/entry/portal/open_service_request).

Note: To enter a support request, you must log in with an IBM account. This account must be linked to your IBM customer number. To learn more about associating your account with your IBM customer number, see **Support Resources > Entitled Software Support** on the Support Portal.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
B1WA LKG1
550 King Street
Littleton, MA 01460-1250
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating

platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Privacy Policy and Terms of Use Considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. A cookie is a piece of data that a web site can send to your browser, which may then be stored on your computer as a tag that identifies your computer. In many cases, no personal information is collected by these cookies. If a Software Offering you are using enables you to collect personal information through cookies and similar technologies, we inform you about the specifics below.

Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user's user name, and other personal information for purposes of session management, enhanced user usability, or other usage tracking or functional purposes. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

Various jurisdictions regulate the collection of personal information through cookies and similar technologies. If the configurations deployed for this Software Offering provide you as customer the ability to collect personal information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for providing notice and consent where appropriate.

IBM requires that Clients (1) provide a clear and conspicuous link to Customer's website terms of use (e.g. privacy policy) which includes a link to IBM's and Client's data collection and use practices, (2) notify that cookies and clear gifs/web beacons are being placed on the visitor's computer by IBM on the Client's behalf along with an explanation of the purpose of such technology, and (3) to the extent required by law, obtain consent from website visitors prior to the placement of cookies and clear gifs/web beacons placed by Client or IBM on Client's behalf on website visitor's devices

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Online Privacy Statement at: http://www.ibm.com/privacy/details/us/en section entitled "Cookies, Web Beacons and Other Technologies."

IBM ®

Printed in USA