Apama Connectivity with Device Integration Platform

Version 9.12

October2016

APAMA

This document applies to Apama Cumulocity Connectivity Plug-in Transport Version 9.12 and to all subsequent releases.

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

Document ID: PAU-912-20161018

Table of Contents

About this Guide

This guide describes how to configure the Apama Connectivity with Device Integration Platform.

Documentation roadmap

Apama Connectivity with Device Integration Platform provides documentation in the following formats:

- HTML (viewable in a web browser)
- PDF (available from the documentation website)

Online Information

Software AG Documentation Website

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

Software AG Empower Product Support Website

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

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

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

Software AG TECHcommunity

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

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

Contacting customer support

If you have an account, you may open Apama Support Incidents online via the eService section of Empower at https://empower.softwareag.com/. If you do not yet have an account, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at https://empower.softwareag.com/public_directory.asp and give us a call.

1 Release Notes

Release Notes describes the changes introduced with the current Apama Connectivity with Device Integration Platform release.

What's new in 9.12 Release

This is the first release of Apama Connectivity with Device Integration Platform.

2 Introduction

The Apama Connectivity with Device Integration Platform allows you to receive information from devices, measurements, events and alarms from Device Integration Platform and send operations to Device Integration Platform (and thus to devices).

3 Installing Apama Connectivity with Device Integration Platform

Installation

- Download the `ApamaConnectivityWithDeviceIntegrationPlatform_version_ALL.zip` from Software AG Empower Product Support Website. See "Online Information" on page 5.
- Extract the Apama Connectivity with Device Integration Platform zip file to ${APAMA_HOME}/connectivity
- All the monitors, libraries, documents, bundles, samples are placed in the respective folder
  - `bundles` folder includes
    - `DeviceIntegrationPlatform` folder which includes `DeviceIntegrationPlatform.yaml` and `DeviceIntegrationPlatform-credentials.properties`
    - `DIP Event Definitions.bnd`
    - `DIP Transport.bnd`
    - `DIP Transformers Bridge.bnd`
  - `doc` folder includes
    - `DeviceIntegrationPlatform` folder which includes ApamaDoc
    - `ChangeLog-ApamaConnectivityWithDeviceIntegrationPlatform.txt`
  - `lib` folder includes
    - `DIPTransport.jar`
    - `DeviceIntegrationPlatform` folder which includes third party libraries
  - `monitors` folder includes
    - `DIP_DisplayOperation.mon`
    - `DIP_DisplayOperationService.mon`
    - `DIP_EventDefinitions.mon`
    - `DIP_IoTTransformersBridge.mon`
    - `DIP_ServiceMonitor.mon`
  - `Samples` folder includes `DeviceIntegrationPlatform` folder which includes sample files
  - `ApamaConnectivityWithDeviceIntegrationPlatform-version.txt`

Configuration Parameters

Provide the following parameters in a yaml configuration file. Refer to `DeviceIntegrationPlatform.yaml`.

- `requestAllDevices`. Requests for all assets at startup. Default value is true
- `subscribeToAllMeasurements`. Subscribes for measurements of all devices during startup. Default value is true
- `subscribeToDevices`. Subscribes for all devices during startup. Default value is true
- `tenant`. Unique name given to the application. Mandatory if the URL is not provided.
- `username`. Name of the user. Required for login
- `password`. Password of the user. Required for login
- `url`. URL of the Device Integration Platform.

Monitor Injection Order

To support the Apama Connectivity with Device Integration Platform, you must inject the following monitor files into the correlator before injecting your application monitors:

- `connectivity/monitors/DIP_EventDefinitions.mon`
- `connectivity/monitors/DIP_ServiceMonitor.mon`

As with other Connectivity plugins, the application should call `com.softwareag.connectivity.ConnectivityPlugins.onApplicationInitialized()` once it is injected. Use `samples/DeviceIntegrationPlatform/monitors/AppReady.mon` in installation as a reference.

4 Working with Apama Connectivity with Device Integration Platform

Prerequisite

Create an account with Device Integration platform. For more information, see Device Integration Platform User's Guide.

Creating an Application Key

To create an application key

1. Log in to Device Integration Platform account, and go to the Administration
2. Select **Applications > Own Applications**. Click **Create Application**.
3. Provide a valid tenant name, user name, password and application key.
4. Select application type as **External application**.
5. Click **Save**.

Developing with Software AG Designer

To add Apama Connectivity with Device Integration Platform to your project

1. Add **Apama Connectivity with Device Integration Platform** bundle.
2. In the **Project Explorer** view, select the project and expand **Connectivity and Adapters -> Apama Connectivity with Device Integration Platform**.
3. Edit `DeviceIntegrationPlatform-credentials.properties` file. You must replace the ${DIP_...} parts with the username, password, tenant, and application key.
4. Start the application. See "Application Startup" on page 20.

Manual Deployment

1. Customize the `DIP_*` values in the `DeviceIntegrationPlatform-credentials.properties` file with the correct tenant.
   - DIP_USERNAME=<username>
   - DIP_TENANT=<tenant>
   - DIP_PASSWORD=<password>
   - DIP_APPKEY=<application key created above>

2. Add `--config DeviceIntegrationPlatform-credentials.properties --config DeviceIntegrationPlatform.yaml` to the correlator command line.
3. Inject the following monitors in the specified order:
   - `connectivity/monitors/DIP_EventDefinitions.mon`
   - `connectivity/monitors/DIP_ServiceMonitor.mon`
4. Start the application. See "Application Startup" on page 20.

5 Event Protocols

Refer to the included ApamaDoc installed at Apama/connectivity/doc/DeviceIntegrationPlatform.

Application Startup

As with all connectivity plug-ins, the application must declare that the application is ready to start receiving the events. The monitor file `AppReady.mon` is included for sample purpose in the connectivity/sample directory. When the `AppReady.mon` file in injected, the monitor file declares that the application is ready to start receiving the events.

Device Events

`com.apama.dip.Device` events are sent to the default channel when `onApplicationInitialized()` is called (unless `requestAllDevices` is set to false in the `DeviceIntegrationPlatform.yaml` file). After all assets are sent, the `com.apama.dip.RequestAllDevicesComplete(-1)` event is sent.

Example of a device event:

```
com.apama.dip.Device("12346084","RaspPi BCM2709 0000000071d36506
   Display ofV","c8y_TinkerForge_Display","",["c8y_Message","c8y_Relay"],
   [],[],{"attrs.c8y_Hardware":"c8y.Hardware@fc2f1534",
   "attrs.c8y_Message":"c8y.Message@e6ffa71a","attrs.c8y_Relay":"Relay{relayState=OPEN}",
   "attrs.c8y_SupportedOperations":"[c8y_Message,
   c8y_Relay]","owner":"device_0000000071d36506",
   "supportedOperations":"[\"c8y_Message\",\"c8y_Relay\"]"})
```

Any device added after `applicationInitialized` is called is sent to the default channel (unless `subscribeToDevices` is set to false in the `DeviceIntegrationPlatform.yaml` file).

Measurement Events

`com.apama.dip.Measurement` events are sent to the channel of their name after `applicationInitialized` (unless `subscribeToAllMeasures` is set to false in the `DeviceIntegrationPlatform.yaml` file). For example, `c8y_LightMeasurement` or `c8y_DistanceMeasurement`. These events may be sent before all assets are sent. Measurement events contain the assetID of the source of the measurement, the type of measurement, timestamp, and a dictionary of values which contain the numeric value, units and optional type, quantity and state.

Examples of measurement events:

```
com.apama.dip.Measurement("c8y_LightMeasurement","12346081",1464359004.89,
   {"e":com.apama.dip.MeasurementValue(108.1,"lux","","","",{})},{})
   com.apama.dip.Measurement("c8y_DistanceMeasurement","12346082",1464359005.396,
   {"distance":com.apama.dip.MeasurementValue(344,"mm","","","",{})},{})
```

Events and Alarms

The `com.apama.dip.Event` event is sent on any event or alarm from a device. This contains the assetID of the source, a timestamp (same form as `currentTime`), message text, and optional parameters.

Examples of events and alarms:

```
com.apama.dip.Event("c8y_EntranceEvent","12346082",1464364483.396,
   "Entrance event triggered.",{"distance":"317"})
   com.apama.dip.Event("c8y_UnavailabilityAlarm","12669915",1464365565.661,
   "No communication with device since
   2016-05-27T18:11:23.886+02:00",{"count":"1","severity":"MAJOR"})
```

Note: See the Apamadoc included with Apama Connectivity with Device Integration Platform.

Control Events Sent to the Transport

The Apama Connectivity with Device Integration Platform listens on the "DeviceIntegrationPlatform" channel for most requests. For all events sent to the transport, a `CHANNEL` constant is defined on the event type with the correct channel to send the event to.

See the `com.apama.dip.RequestAllDevices` event for refreshing the list of devices. Note that the `com.apama.dip.RequestAllDevices` event is sent to the channel defined by the CHANNEL constant on `RequestAllDevices` event. The device events are sent to the channel specified in the `RequestAllDevices` event and followed by a `com.apama.dip.RequestAllDevicesComplete` event with the `requestId` sent in the `RequestAllDevices` event.

The `com.apama.dip.SubscribeMeasurements` and `com.apama.dip.UnsubscribeMeasurements` events control subscriptions to measurements from assets. Specify the assetId as either an assetId from a device event or as `"*"` for a wildcard (use the `WILDCARD` constant). Subscriptions are reference counted by the transport, so send as many Unsubscribe events as Subscribe events have been sent to completely unsubscribe. Unless the `subscribeToMeasurements` configuration property is false, the transport automatically subscribes as if `SubscribeMeasurements("*")` had been sent at `applicationInitialized` time. Subscriptions to measurements also subscribe to events and alarms.

Send a `com.apama.dip.DeviceOperation` or `com.apama.dip.DeviceOperationStringArray` event to send an operation to a device. For example, to set a device's display and modify a set of relays, send:

```
com.apama.dip.DeviceOperation("12346084","c8y_Message",{"text":"800.0mm"})
   com.apama.dip.DeviceOperationStringArray("12350781","c8y_RelayArray",["OPEN","CLOSED"])
```

6 Integrating with IoT Analytics Kit

Integrating with IoT Analytics Kit

If you are using the IoT Analytics kit, add the **Apama Connectivity with Device Integration Platform to Transformer Bridge** bundle as well. IoT Analytics kit can be downloaded from Apama Community Edition.

If you are using the IoT Analytics kit, add the IoT Analytics kit to Software AG Designer monitor script build path. Be sure to use the variable name *IOT_ANALYTICS_HOME* for defining the installation directory of the IoT Analytics kit.

Configuring Software AG Designer Using IoT Analytics Kit

To configure Software AG Designer

1. In Software AG Designer, click **Windows > Preferences**.
2. In the Preferences dialog, expand **Software AG > Apama**.
3. Select **MonitorScript Path Variables** and click **New**.
4. In the **Name** field, enter `IOT_ANALYTICS_HOME`.
5. In the **Path** field, click **Folder** and navigate to the bundle install directory. Click **OK**.
6. In the Preferences dialog, select **Apama > Catalogs**.
7. In the **Bundle catalogs** tab, click **Add Variable**.
8. In the New Variable Dependency Entry dialog, select the *IOT_ANALYTICS_HOME* variable. Click **Extend**.
9. Select the bundles subdirectory. Click **OK**.
10. Restart Software AG Designer.
11. Add **Apama Connectivity with Device Integration Platform to Transformer Bridge** bundle.

    In the **Project Explorer** view, select the project and expand **Connectivity and Adapter > Apama Connectivity with Device Integration Platform to Transformer Bridge**. Double-click **Connectivity** to edit the YAML file. You must replace the `${DIP_...}` parts with the username, password, tenant, and application key.

Manual Deployment Using IoT Analytics Kit

To manually deploy using IoT Analytics kit

1. Set `IOT_ANALYTICS_HOME` to the path where you downloaded the IoT Analytics Kit.

2. Customize the `DIP_*` values in the `DeviceIntegrationPlatform-credentials.properties` file with the correct tenant.
   - DIP_USERNAME=<username>
   - DIP_TENANT=<tenant>
   - DIP_PASSWORD=<password>
   - DIP_APPKEY=<application key created above>
3. Add `--config DeviceIntegrationPlatform-credentials.properties --config DeviceIntegrationPlatform.yaml` to the correlator command line.
4. Inject the monitors in the specified order:
   - `${APAMA_HOME}/monitors/ScenarioService.mon`
   - `${APAMA_HOME}/monitors/data_storage/MemoryStore.mon`
   - `APAMA_HOME/monitors/data_storage/MemoryStoreScenarioImpl.mon`
   - `connectivity/monitors/DIP_EventDefinitions.mon`
   - `connectivity/monitors/DIP_ServiceMonitor.mon`
   - `connectivity/monitors/DIP_DisplayOperation.mon`
   - `connectivity/monitors/DIP_DisplayOperationService.mon`
   - `connectivity/monitors/DIP_IoTTransformersBridge.mon`
5. Start the application. See "Application Startup" on page 20.

Using the Transformers Bridge

If the **Apama Connectivity with Device Integration Platform to Transformer Bridge** bundle is added, then the device event is automatically translated to `com.industry.iot.Asset` event and measurement event is automatically translated to `com.industry.iot.Measure` event. The control events in the Device Integration Platform package can be used to control subscriptions or request measures.

For measurements which contain multiple values, the value with the lowest key name is used as the top-level value, and further values are available in the parameters dictionary. For events and alarms, the `dValue` is 0 and the `sValue` of the measure contains the message text. Note that for measure events, the events are only sent to the channel of the name (the first field in the event), which is the type of the measurement. Applications must subscribe to that channel to consume the events.

Example events:

```
com.industry.iot.Asset("12557881","c8y_Linux","RaspPi BCM2709 00000000e2cbada2",
"",["12475995","12560654","12560655","12599092","12599527","12475994"],
{"attrs.c8y_ActiveAlarmsStatus":"{major=2}","attrs.c8y_Availability":
    "c8y.Availability@355cadd7",
"attrs.c8y_Configuration":"c8y.Configuration@2bd6f193",
```

```
"attrs.c8y_Firmware":"c8y.Firmware@14750fc4","attrs.c8y_Hardware":
    "c8y.Hardware@de0caeca",
"attrs.c8y_IsDevice":"c8y.IsDevice@7ab33eef",
"attrs.c8y_RequiredAvailability":"c8y.RequiredAvailability@3",
"attrs.c8y_Software":"{pi4j-device=pi4j-device-0.0.5.jar,
 rest-representation=rest-representation-5.19.0.jar,
 jersey-multipart=jersey-multipart-1.8-5.19.0.jar, jsr311-api=jsr311-api-1.1.1.jar,
 device-capability-model=device-capability-model-5.19.0.jar,
 pi4j-gpio-extension=pi4j-gpio-extension-0.0.5.jar,
    common-notification=common-notification-5.19.0.jar,
 commons-io=commons-io-2.4.jar, pi4j-example=pi4j-example-0.0.5.jar,
    svenson=svenson-1.3.8-5.19.0.jar,
 slf4j-api=slf4j-api-1.7.0.jar, jcl-over-slf4j=jcl-over-slf4j-1.6.1.jar,
    commons-codec=commons-codec-1.4.jar,
 jetty-client=jetty-client-7.6.14.v20131031.jar,
    jersey-apache-client=jersey-apache-client-1.8-5.19.0.jar,
 core-model=core-model-5.19.0.jar, jersey-core=jersey-core-1.8-5.19.0.jar,
    tinkerforge=tinkerforge-2.1.2.jar,
 bayeux-api=bayeux-api-2.8.0.jar, cometd-java-common=cometd-java-common-2.8.0.jar,
 rpi-driver=rpi-driver-5.20.0.jar, logback-classic=logback-classic-1.0.13.jar,
 commons-httpclient=commons-httpclient-3.1-5.19.0.jar, joda-time=joda-time-1.6.2.jar,
 pi4j-core=pi4j-core-0.0.5.jar, jssc=jssc-2.6.0.jar,
    cometd-java-client=cometd-java-client-2.8.0.jar,
 jv-agent=jv-agent-5.20.0.jar, commons-beanutils=commons-beanutils-1.8.0.jar,
 jetty-io=jetty-io-7.6.14.v20131031.jar, mimepull=mimepull-1.4-5.19.0.jar,
 jersey-client=jersey-client-1.8-5.19.0.jar,
    tinkerforge-driver=tinkerforge-driver-5.20.0.jar,
 common-rest=common-rest-5.19.0.jar, logback-core=logback-core-1.0.13.jar,
    java-client=java-client-5.19.0.jar,
 piface-driver=piface-driver-5.20.0.jar, jv-driver=jv-driver-5.20.0.jar,
 jetty-util=jetty-util-7.6.14.v20131031.jar,
    jetty-http=jetty-http-7.6.14.v20131031.jar}",
"attrs.c8y_SupportedOperations":"[c8y_Restart, c8y_Firmware,
    c8y_Configuration, c8y_Software]",
"attrs.com_cumulocity_model_Agent":"com.cumulocity.model.Agent@13a00e5c",
   "owner":"device_00000000e2cbada2",
"supportedOperations":"[\"c8y_Restart\",\"c8y_Firmware\",\"c8y_Configuration\",
   \"c8y_Software\"]"})
com.industry.iot.Asset("12475995","c8y_TinkerForge_DistanceIR",
"RaspPi BCM2709 00000000e2cbada2 DistanceIR p1c","",[],
{"attrs.c8y_DistanceSensor":"c8y.DistanceSensor@3bd8011",
"attrs.c8y_Hardware":"c8y.Hardware@721b36b0","owner":"device_00000000e2cbada2"})
com.industry.iot.Asset("12475994","c8y_TinkerForge_Display",
"RaspPi BCM2709 00000000e2cbada2 Display ofV","",[],
{"attrs.c8y_Hardware":"c8y.Hardware@fc2f1534","attrs.c8y_Message":"c8y.Message@e6ffa71a",
"attrs.c8y_Relay":"Relay{relayState=CLOSED}","attrs.c8y_SupportedOperations":
   "[c8y_Message, _Relay]",
"owner":"device_00000000e2cbada2","supportedOperations":"[\"c8y_Message\",\"c8y_Relay\"]"})
com.industry.iot.Measure("c8y_DistanceMeasurement","r","12346082",1464864899.311,142,"",
   {"unit":"mm"})
com.industry.iot.Measure("c8y_EntranceEvent","r","12346082",1464865646.145,0,
"Entrance event triggered.",{"distance":"95"})
com.industry.iot.Measure("c8y_UnavailabilityAlarm","r","12669915",
1464871160.286,0,"No communication with device since 2016-06-02T14:35:16.664+02:00",
{"count":"1","severity":"MAJOR"})
```

The **Apama Connectivity with Device Integration Platform to Transformer Bridge** bundle also includes a `DisplayOperation` transform for sending measure values to a Device Integration Platform display. Configuration parameters specify either a single assetId or a deviceName. The transform will send operations to all devices which match the device name. A sample demonstration is included in the demo directory which:

- Sends data from all distance measurements to a dataview
- Calculates a smoothed value using the `MovingAverage` transform and a smoothed velocity using the 'Gradient' transform

The events to configure the transforms are given below:

```
com.industry.iot.Transform("Gradient", ["c8y_DistanceMeasurement"],
    ["SmoothedVelocity"], {"bucketTimeWindow":"0.0"})
 com.industry.iot.Transform("MovingAverage", ["c8y_DistanceMeasurement"],
    ["SmoothedDistance"], {"timeWindow":"5.0"})
 com.industry.iot.Transform("CumulocityDisplayOperation", ["SmoothedVelocity"],
    [], {"deviceName":"c8y_TinkerForge_Display"})
 com.industry.iot.Transform("DataViewer", ["SmoothedVelocity"], ["SmoothedVelocity"], {})
 com.industry.iot.Transform("DataViewer", ["c8y_DistanceMeasurement"], ["Distance"], {})
 com.industry.iot.Transform("DataViewer", ["SmoothedDistance"], ["SmoothedDistance"], {})
```

You can modify the above events if you have a different type of sensor input or display. You can also modify the `timeWindow` of the moving average or the sensor's polling interval to obtain suitable smoothing. Configuration can be modified using the Device Integration Platform application and changing the gateway's configuration).

Monitors and a dashboard are included in the Apama/connectivity/samples/DeviceIntegrationPlatform directory. Copy all the monitor files and dashboard into a project with the Apama Connectivity With Device Integration Platform and **Apama Connectivity with Device Integration Platform to Transformer Bridge** added. Mark the dashboard as the default dashboard in the **Run Project** configuration and run the project. The dashboard should display a table of all distance measurements.