



TIBCO ActiveMatrix® Service Grid

Hawk ActiveMatrix® Plug-in

Software Release 3.4.0

April 2019

Document Updated: September 2019

Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

ANY SOFTWARE ITEM IDENTIFIED AS THIRD PARTY LIBRARY IS AVAILABLE UNDER SEPARATE SOFTWARE LICENSE TERMS AND IS NOT PART OF A TIBCO PRODUCT. AS SUCH, THESE SOFTWARE ITEMS ARE NOT COVERED BY THE TERMS OF YOUR AGREEMENT WITH TIBCO, INCLUDING ANY TERMS CONCERNING SUPPORT, MAINTENANCE, WARRANTIES, AND INDEMNITIES. DOWNLOAD AND USE OF THESE ITEMS IS SOLELY AT YOUR OWN DISCRETION AND SUBJECT TO THE LICENSE TERMS APPLICABLE TO THEM. BY PROCEEDING TO DOWNLOAD, INSTALL OR USE ANY OF THESE ITEMS, YOU ACKNOWLEDGE THE FOREGOING DISTINCTIONS BETWEEN THESE ITEMS AND TIBCO PRODUCTS.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, the TIBCO logo, Two-Second Advantage, TIB, Information Bus, ActiveMatrix, Business Studio, Enterprise Message Service, Hawk, and Rendezvous are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2010-2019. TIBCO Software Inc. All Rights Reserved.

Contents

Figures	6
TIBCO Documentation and Support Services	7
TIBCO Hawk ActiveMatrix Plug-in Overview	9
ActiveMatrix Host Microagent and Service Microagent	10
Monitoring Architecture	10
ActiveMatrix Monitoring Service Application	11
Configurations	13
Configuration Prerequisites	13
Configuring Sample Rulebases	13
Configuring for Hawk 5.x	14
Configuring for Hawk 6.1	14
Configuring for TIBCO Enterprise Message Service	15
Configuring to Manage and Monitor Multiple TIBCO Host Instances in the Same Hawk Agent	16
Configuring Service Rulebase Samples	16
Rulebase Details	16
ActiveMatrix Host Microagent	18
isHostRunning	18
getHostInfo	19
getNodeInfo	21
onHostEvent	23
onNodeEvent	25
startNode	26
stopNode	26
Common Logging Event Publisher Microagent	28
startPublisher	28
stopPublisher	28
startAllPublishers	29
stopAllPublishers	29
getEventPublisherInfo	29
getEventDataSourceDetails	30
reloadEventDefinitions	31
Sample Event Definitions Configuration to CLEvents from Hawk	34
ActiveMatrix Service Microagent	38
getConfig	38
changeMonitoringConfig	39
getComponentInfo	39

getBindingInfo	41
onComponentEvent	42
onBindingEvent	44
getBindingOperationStats	46
getComponentOperationStats	47
onBindingOperationStats	49
onComponentOperationStats	50
Service Execution Stats	51
Configuration of TIBCO ActiveMatrix Hawk Rulebase Samples	53
Scenarios	53
Scenario A: When Tibco Host is Abnormally Shutdown	53
Scenario B: When Administrator Node is Abnormally Shutdown	54
Scenario C: Runtime Nodes Managed by connected AMX Host	54
Deploying and Testing Sample Application	56
Testing the Sample Application	56

Figures

Overview of ActiveMatrix Host and Service Microagents **9**

Monitoring Points for Available Execution Statistics **10**

Monitoring Architecture of ActiveMatrix Host and Service Microagents 11

5 Minute Rolling Window (10 Buckets) 12

TIBCO Documentation and Support Services

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit <https://docs.tibco.com>.

Product-Specific Documentation

Documentation for TIBCO ActiveMatrix® Service Grid is available on the <https://docs.tibco.com/products/tibco-activematrix-service-grid> page.

Use of the following features, installation profiles and development tools requires a TIBCO ActiveMatrix Service Grid license:



- TIBCO ActiveMatrix Policy Director Governance, TIBCO ActiveMatrix SPM Dashboard, and TIBCO ActiveMatrix SPM Runtime Server profiles; and
- TIBCO ActiveMatrix Service Grid development tools for Java, Webapp and Spring components.

Customers with only a TIBCO ActiveMatrix Service Bus license are not licensed to use these features, tools or profiles.

The following documents form the documentation set:

- *TIBCO ActiveMatrix Service Grid Concepts*: Read this manual before reading any other manual in the documentation set. This manual describes terminology and concepts of the platform. The other manuals in the documentation set assume you are familiar with the information in this manual.
- *TIBCO ActiveMatrix Service Grid Development Tutorials*: Read this manual for a step-by-step introduction to the process of creating, packaging, and running composites in TIBCO Business Studio.
- *TIBCO ActiveMatrix Service Grid Composite Development*: Read this manual to learn how to develop and package composites.
- *TIBCO ActiveMatrix Service Grid Java Component Development*: Read this manual to learn how to configure and implement Java components.
- *TIBCO ActiveMatrix Service Grid Mediation Component Development*: Read this manual to learn how to configure and implement Mediation components.
- *TIBCO ActiveMatrix Service Grid Mediation API Reference*: Read this manual to learn how to develop custom Mediation tasks.
- *TIBCO ActiveMatrix Service Grid Spring Component Development*: Read this manual to learn how to configure and implement Spring components.
- *TIBCO ActiveMatrix Service Grid WebApp Component Development*: Read this manual to learn how to configure and implement Web Application components.
- *TIBCO ActiveMatrix Service Grid REST Binding Development*: Read this manual to learn how to configure and implement REST components.
- *TIBCO ActiveMatrix Service Grid Administration Tutorials*: Read this manual for a step-by-step introduction to the process of creating and starting the runtime version of the product, starting TIBCO ActiveMatrix servers, and deploying applications to the runtime.
- *TIBCO ActiveMatrix Service Grid Administration*: Read this manual to learn how to manage the runtime and deploy and manage applications.

- *TIBCO ActiveMatrix Service Grid Hawk ActiveMatrix Plug-in*: Read this manual to learn about the Hawk plug-in and its optional configurations.
- *TIBCO ActiveMatrix Service Grid Policy Director Governance Custom Actions*: Read this manual to learn how you can configure and enforce policies for ActiveMatrix and external services hosted in third party containers, using TIBCO ActiveMatrix Policy Director Governance.
- *TIBCO ActiveMatrix Service Grid Service Performance Manager API Reference*: Read this manual to learn how to use the SPM APIs.
- *TIBCO ActiveMatrix Service Grid Error Codes*: Read this manual to know more about the error messages and how you could use them to troubleshoot a problem.
- *TIBCO ActiveMatrix Service Grid Installation and Configuration*: Read this manual to learn how to install and configure the software.
- *TIBCO ActiveMatrix Service Grid Security Guidelines*: Read this manual to learn more about security guidelines and recommendations for TIBCO ActiveMatrix Service Grid.
- *TIBCO ActiveMatrix Service Grid Release Notes*: Read this manual for a list of new and changed features, steps for migrating from a previous release, and lists of known issues and closed issues for the release.

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit <http://www.tibco.com/services/support>.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at <https://support.tibco.com>.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to <https://support.tibco.com>. If you do not have a user name, you can request one by clicking Register on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](https://community.tibco.com). For a free registration, go to <https://community.tibco.com>.

TIBCO Hawk ActiveMatrix Plug-in Overview

TIBCO Hawk ActiveMatrix Plug-in provides features to monitor and manage TIBCO ActiveMatrix host and node processes, as well as TIBCO ActiveMatrix services and components.

These features can be realized by several Hawk microagents:

- The ActiveMatrix Host Microagent
- The ActiveMatrix Services Microagent
- Common Logging Event Publisher Microagent

The Host Microagent enables simple monitoring and management of the TIBCO Host process, by providing basic status information about the host and nodes managed by the host.

The Service Microagent allows for monitoring of status and performance statistics of service components and bindings running on a node. The performance statistics provided are of short duration (in the last five minutes) and include metrics such as hits, faults, and average response time.

The Common Logging Event Publisher Microagent enables Hawk microagents to connect to the ActiveMatrix Common Logging Service, to provide events for logging purposes.

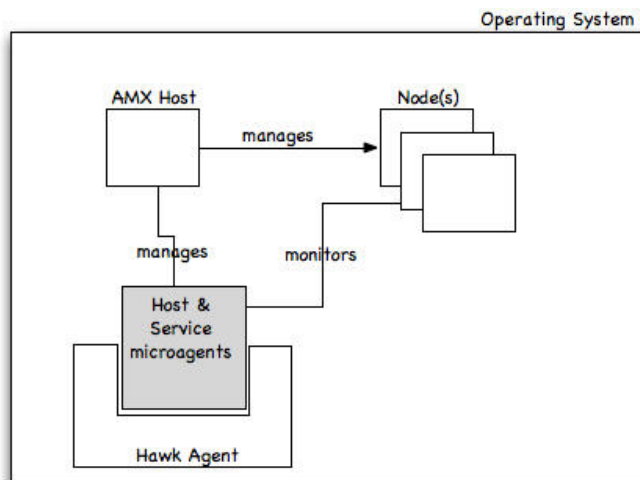
[Overview of ActiveMatrix Host and Service Microagents](#), illustrates how the Host and Service Microagents interact with the ActiveMatrix host and nodes.

The Host and Service Microagents are TIBCO Hawk Plug-in microagents and they are started as part of the Hawk Agent process.



These microagents require the Hawk HMA to be running to function correctly.

Overview of ActiveMatrix Host and Service Microagents



ActiveMatrix Host Microagent and Service Microagent

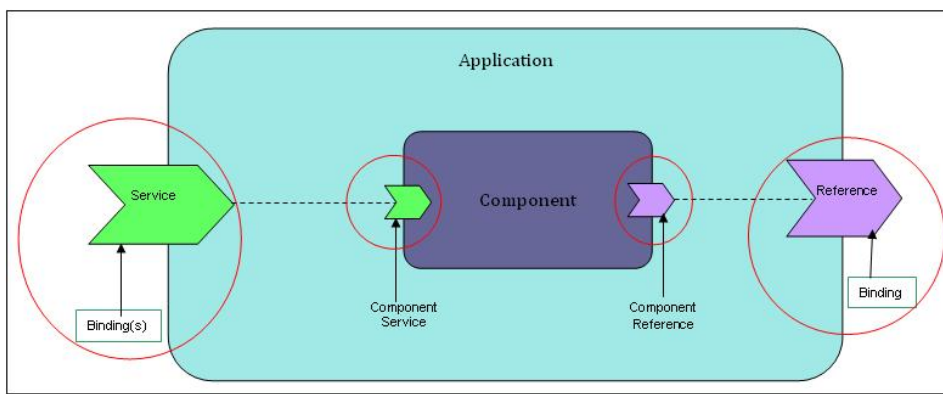
The ActiveMatrix Host Microagent enables the monitoring and management of TIBCO ActiveMatrix host and nodes running on each local machine. It provides methods to retrieve node status and host status, as well as control methods to start or stop nodes.

The ActiveMatrix Service Microagent provides the following monitoring capabilities:

- Monitor status of Component and Binding instances deployed in a node.
- Monitor execution statistics of service binding(s), reference binding, component services and component references of a deployed ActiveMatrix 3 Application.

[Monitoring Points for Available Execution Statistics](#), describes the various monitoring points for which execution statistics are available.

Monitoring Points for Available Execution Statistics



This diagram is representative of a typical ActiveMatrix 3.x Application (an Open SOA SCA Assembly). The indicated monitoring points are the parts of the deployed application for which the ActiveMatrix Service Microagent will provide statistics data.

For details on the metrics provided at each monitoring point, refer to [Service Execution Stats](#).

Monitoring Architecture

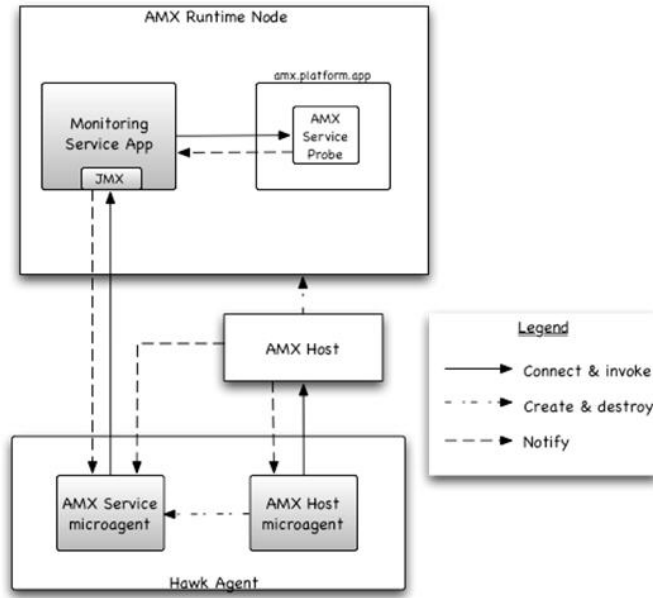
The ActiveMatrix Service microagent is designed to monitor services deployed in the ActiveMatrix node and it is integrated with ActiveMatrix Host microagent.

The ActiveMatrix Host microagent creates instances of the Service Microagent for each node (on node start) associated with ActiveMatrix host, and destroys these instances on Node stop.

In order to collect service operation statistics on each node, the Service microagent requires the deployment of an ActiveMatrix Monitoring Application 3.2 and above on each node that this microagent is required to monitor.

The architecture of the Host and Service microagent and the interaction with the partnering Monitoring Application is shown in [Monitoring Architecture of ActiveMatrix Host and Service Microagents](#).

Monitoring Architecture of ActiveMatrix Host and Service Microagents



The Service microagent connects to a node process using Java Attach API. Statistics data can then be retrieved using either synchronous or asynchronous method invocations.

In either approach, the Service microagent connects to the Monitoring Application's JMX MBean to retrieve the service operation statistics data.

ActiveMatrix Monitoring Service Application

The ActiveMatrix Monitoring Service application provides the Service Microagent with access to, or notification of service execution statistics.

The Monitoring Service application provides some configuration capabilities, which can be set in either the TIBCO ActiveMatrix Administrator, or the ActiveMatrix Service Microagent.

Monitoring Configuration

The monitoring configuration offered by the Monitoring Service application enables the configuration of a rolling window for statistics collection.

The maximum length of a rolling window is five minutes, but this is configurable.

The configuration properties are as follows:

Rolling Window Length

Length of rolling window (in seconds, up to 5 minutes)

Statistics Gathering Interval

Length of a bucket (in seconds)

Notification Interval

Rate at which statistics data is published (via notification)

Is Partial Window

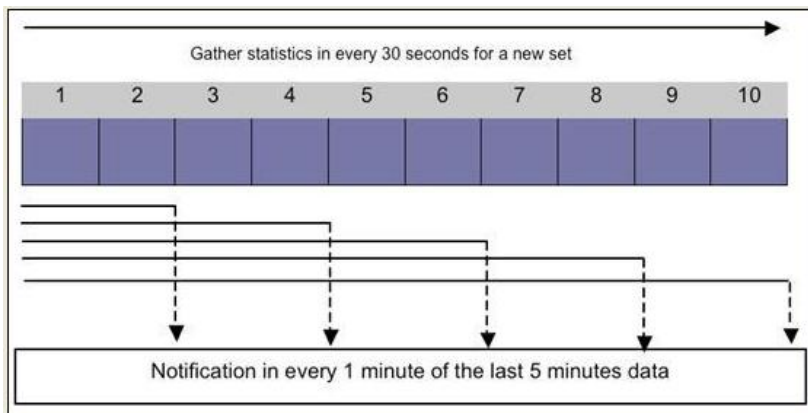
Whether to include the current statistics data (incomplete bucket) in notifications.

For example, the configuration settings to produce the following diagram would be:

- Rolling Window Length = 300
- Statistics Gathering Interval = 30

- Notification Interval = 60
- Is Partial Window = false

5 Minute Rolling Window (10 Buckets)



Common Logging Event Publisher Microagent

This microagent publishes events from TIBCO Hawk using the TIBCO Common Logging in the Base Event Format (BEF)/Common Base Event (CBE).

Configurations

This section covers optional configurations of two microagents and service rulebase samples provided with this product. This configuration provides the monitoring and management of ActiveMatrix Hosts, nodes, components, and services by using Hawk.

The rulebases and scripts provided in this sample demonstrate the use of the ActiveMatrix Host microagent, to monitor and manage ActiveMatrix 3.4 hosts and nodes. For more details on rules, refer to [Configuration of TIBCO ActiveMatrix 3.x Hawk Rulebase Samples](#).

Configuration Prerequisites

While configuring ActiveMatrix using TIBCO Configuration Tool, one or both of the following microagents get installed in CONFIG_HOME depending on your choice:

- ActiveMatrix Host Microagent
- ActiveMatrix Service Microagent

The two microagents are installed with TIBCO ActiveMatrix on every machine where ActiveMatrix nodes and hosts are running. With the following configuration procedure, the microagents can monitor multiple nodes and even multiple hosts on each machine.

Before configuring ActiveMatrix Host microagent and ActiveMatrix Service microagent:

- On a Windows machine, ensure that Windows PowerShell is installed
- On a Unix machine, ensure that /bin/bash or a symbolic link from /bash_Install_Dir/bash to /bin/bash is available
- On a Linux machine, ensure that the lsof package is installed

Configuring Sample Rulebases

To configure the sample rulebases to be used with Hawk:

1. Go to CONFIG_HOME/hkam/<host_instance_name>/scripts.



In the configure-hkam.properties file, use only forward slash ("/").

2. Edit configure-hkam.properties to specify the following variables:

- tibco.hawk.version - The version of Hawk.
- tibco.hawk.home - TIBCO_HOME used for Hawk installation.
- tibco.hawk.config.home - TIBCO configuration directory specified during Hawk installation. For example, c:/hk49data.



Set the following value to true if the communication between a Host and the Administrator Server is Secure:

```
amx.hpa.is.secure=true
```

3. Run the configure script from the command line:

- On Windows: configure-hkam.bat
- On UNIX: ./configure-hkam.sh



By default, the `admin_emsconfig_password` in the `tct-install.properties` file has empty value. Therefore configure scripts will generate empty password in `hawkems-<host_name>.hma` file. To pick up a non-empty EMS password, modify the value `admin_emsconfig_password` in the `tct-install.properties` file, and rerun the configure script to generate the `.hma` file. Alternatively, you can manually update the value of `-encryptedPassword` with the value of `${password}` in `hawkems-<host_name>.hma` file.

```
<arg>-encryptedPassword</arg>
<arg>${password}</arg>
```

4. Configure the CLEvent Publisher Microagent.

5. Modify the following properties:

- On Windows:

Launch `tibhawkconfig.exe`, and in the **Agent** tab, add or modify the following properties:

- Set the **Autoconfiguration Directory** as `CONFIG_HOME/hkam/<host_instance_name>/rulebases`
- Set the value of **Plugins** as `CONFIG_HOME/hkam/<host_instance_name>/plugin`
- Set the variables file as `CONFIG_HOME/hkam/<host_instance_name>/data/Hawk-amx_variables.properties`

- On UNIX:

Edit `HAWK_CONFIG/bin/hawkagent.cfg` to configure the following properties:

- `hma_plugin_dir "CONFIG_HOME/hkam/<host_instance_name>/plugin"`
- `variables "CONFIG_HOME/hkam/<host_instance_name>/data/Hawk-amx_variables.properties"`
- `auto_config_dir "CONFIG_HOME/hkam/<host_instance_name>/rulebases"`

Start Hawk HMA, Hawk Agent, Hawk Display and so on, as appropriate for your platform.

Configuring for Hawk 5.x

If Hawk 5.x and TIBCO ActiveMatrix are installed in the same `TIBCO_HOME` on a machine, do the following:

- On Windows, confirm that `attach.dll` resides in `TIBCO_HOME/tibcojre64/1.8.0/bin`
- On Linux, copy `libattach.so` to `TIBCO_HOME/tibcojre64/1.8.0/lib/amd64`

Configuring for Hawk 6.1

To make sure the ActiveMatrix Host Microagents of 3.4.0 work with Hawk 6.1:

1. Copy `tibcrypt.jar` from `<HAWK_HOME>\hawk\6.1\lib\compat` to `<HAWK_HOME>\hawk\6.1\lib\ext`.
2. To support Java 1.8_172 or higher, add the following entry to the `tibhawkagent.tra` file:

```
java.property.jceks.key.serialFilter=com.tibco.**;
java.lang.Enum;java.security.KeyRep;java.security.KeyRep$Type;
javax.crypto.spec.SecretKeySpec;!*
```

Troubleshooting Issues when Configuring for Hawk 6.1

Problem

When starting TIBCO Hawk Agent, the following error occurs:

```
java.lang.NoClassDefFoundError: org/w3c/dom/ElementTraversal
```

Workaround

Use one of the following workarounds to solve this problem:

Copy the `xml-apis.jar` to `<HAWK-HOME>\hawk\6.1\lib` folder.

OR

In the `configure-hkam-3.4.0.xml` file, change the

```
<include name="com.tibco.tpcl.javax.system.exports_5.0.500.*/xml-apis.jar"/> to
```

```
<include name="com.tibco.tpcl.javax.system.exports_5.1.0.*/xml-apis.jar"/>, before running the configure-hkam.bat file.
```

Configuring for TIBCO Enterprise Message Service

For all the operating systems, do the following configurations:

- Update `tibco.env.EMS_HOME` to point to `EMS_ROOT` in `tibhawkagent.tra`, if one of the following is true:

- Enterprise Message Service was installed after installing Hawk 5.x
- Enterprise Message Service and Hawk 5.x are installed in different `TIBCO_HOMES`

For example, `EMS_HOME/ems/7.0`

- If Enterprise Message Service and Hawk 5.x are installed on different machines, copy the following jar files from `EMS_HOME/lib` to `HAWK_ROOT/lib`:

- `jms.jar`
- `tibjmsadmin.jar`
- `tibjms.jar`

If you are using EMS 8.2 and above, use `tibjms-2.0.jar`.

- `tibcrypt.jar`

- If the ActiveMatrix Administrator server or TIBCO Host instance is configured to use SSL-enabled Enterprise Message Service, modify the `hawkems-<host_instance_name>.hma` file located at `CONFIG_HOME/hkam/<host_instance_name>/plugin` with the following parameters:

```
<arguments>
<arg>-user</arg>
<arg>admin</arg>
<arg>-encryptedPassword</arg>
<arg/>
<arg>-server</arg>
<arg><ems-server-url></arg>
<arg>-server_in_agent_name</arg>
<arg>-ssl_trace</arg>
<arg>-ssl_vendor</arg>
<arg><ssl-vendor-used-by-ssl-client></arg>
<arg>-ssl_trusted</arg>
<arg><path-to-ssl-trusted-certificate></arg>
<arg>-ssl_identity</arg>
<arg><path-to-ssl-identity-provider-certificate></arg>
<arg>-ssl_expected_hostname</arg>
<arg>server</arg>
<arg>-ssl_password</arg>
<arg><ssl-certificate-encrypted-password></arg>
</arguments>
```

Configuring to Manage and Monitor Multiple TIBCO Host Instances in the Same Hawk Agent

For generating rulebases and microagent configurations for multiple TIBCO Hosts on the same machine, and managing these through the same Hawk agent, do the following:

Procedure

1. Configure the rulebases for each TIBCO Host instance.
2. Copy all the files from the plugins and rulebases folders in the hkam directories of all TIBCO Host instances to any one of the common plugins and rulebases folder.
3. Copy all the properties from all the Hawk.variables files in the data folders to the common Hawk.variables file.
4. Configure the Hawk agent to use the common plugins and rulebases folders along with the common Hawk.variables file.

Configuring Service Rulebase Samples

On each node, you can create service rulebases that apply to all components or services. The parameters to the microagent methods can either be blank or use wildcards. Wildcards need to match the substrings in many or all available components and operations.

The sample rulebases, which are generated against the default node and default host, are present in `TIBCO_HOME/amx/<version>/samples/hawk/rulebases/servicerulebases`.

The default node and host name is `com.tibco.hawk.amx.SystemHost_DevNode_Service`.

To specify service rulebase sample

1. Change the name of the default node and host to reflect your own environment.
2. Load the rulebases through Hawk Display or copy into:

```
CONFIG_HOME/hkam/<host_instance_name>/rulebases
```



Some rulebases need an SMTP host to be configured.

Rulebase Details

The rulebase samples are provided with the installation to generate alerts from the services.

Rulebase	Description
Pending Responses	Alert if, more than five pending responses for an operation (may indicate hung service).
PowerCycle-RestartNodewithFaults	Restart node if, faults at 100% and more than five hits (assumes systemic failure).
ResponseTimeTrend	Alert if, the response time doubles over sampling the period.
Service Availability	Alert and Email notification if, the service binding has stopped unexpectedly.

Rulebase	Description
Service Faults	Alert if, more than 20% of hits are faults.
Service Hits Sudden Burst	Alert if, orderBook hits increases by more than 50% in the last five minutes.

ActiveMatrix Host Microagent

The ActiveMatrix Host Microagent (com.tibco.hawk.amx.AMXHostMicroAgent) allows monitoring and managing of TIBCO ActiveMatrix Host Instance running in the local machine.

AMXHost Microagent Configuration File

File name: AMXHostPluginConfig.xml

Content:

```
<?xml version="1.0" encoding="UTF-8"?>
<AMXHostConfiguration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="AMXHostPluginConfig.xsd">
  <HPAClientConfigPropFile>%TIBCO_HAWK_HOME_ESC%/plugin/amx/
hpaclientconfig.properties</HPAClientConfigPropFile>
  <MethodTimeout>10</MethodTimeout>
  <HostProcessPrefix>tibcohost*</HostProcessPrefix>
  <NodeProcessPrefix>tibamx_*</NodeProcessPrefix>
</AMXHostConfiguration>
```

XSD Configuration file: AMXHostPluginConfig.xsd

Content:

```
<?xml version = "1.0" encoding = "UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
  <xsd:element name="AMXHostConfiguration" type="AMXHostConfigurationType" />
  <xsd:complexType name="AMXHostConfigurationType">
    <xsd:sequence>
      <xsd:element ref="MicroAgentName" />
      <xsd:element ref="MicroAgentDisplayName" />
      <xsd:element ref="MicroAgentDesc" />
      <xsd:element ref="MethodTimeout" />
      <xsd:element ref="HPAClientConfigPropFile" />
      <xsd:element ref="HostProcessPrefix" />
      <xsd:element ref="NodeProcessPrefix" />
    </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="MicroAgentName" type="xsd:string" default="" />
  <xsd:element name="MicroAgentDisplayName" type="xsd:string" default="" />
  <xsd:element name="MicroAgentDesc" type="xsd:string" default="" />
  <xsd:element name="HPAClientConfigPropFile" type="xsd:string"
  default="hpaclientconfig.properties" />
  <xsd:element name="MethodTimeout" type="xsd:string" default="10" />
  <xsd:element name="HostProcessPrefix" type="xsd:string" default="tibcohost*" />
  <xsd:element name="NodeProcessPrefix" type="xsd:string" default="tibamx_*" />
</xsd:schema>
```

AMX Host Microagent Configuration

For configuration of AMX Host Microagent, refer to the `readme.txt` available in `plugin/amx`.

Methods

The methods in the microagent provides the information for the host instance running on the machine.

isHostRunning

The `isHostRunning` method pings the TIBCO ActiveMatrix Host and returns the value as `true` or `false`.

Type

Synchronous, `IMPACT_INFO`.

Arguments

None.

Returns

Name	Type	Description
Running: true/false	Boolean	Pings the TIBCO ActiveMatrix Host and returns true if host is running, or else false .

getHostInfo

The getHostInfo method returns the details of TIBCO ActiveMatrix Host instance running in the local machine.

Type

Synchronous, IMPACT_INFO.

Arguments

None.

Returns

Name	Type	Description
Process ID	String	The Process ID of the TIBCO Host Instance (OS Process ID). Note: ProcessID return "-1", if Host is not running.
Name	String	The name of the ActiveMatrix host (SystemHost).
Type	String	Type of the TIBCO ActiveMatrix host: (TIBCO Host).

Name	Type	Description
Runtime State	String	<p>The state of ActiveMatrix host.</p> <ul style="list-style-type: none"> UNKNOWN NOT_RUNNING INITIALIZING INITIALIZING_FAILED INITIALIZED STARTING - Starting of the host. The starting state should move to the running state BEFORE the nodes are started. In other words, Host state is independent of Node State. Nodes cannot start unless the Host starts successfully. STARTING_FAILED - The process of starting failed, the reason on the status will be set. STOPPING STOPPED RUNNING - The host is running successfully. UNBOUND - The host is not bound to the enterprise. LOOSING_CONTACT LOST_CONTACT - Lost contact with a host. Reasons being the host was killed unexpectedly, heartbeats are lost in a flood of traffic, or the network is no longer operational.
Version	String	Version of the TIBCO ActiveMatrix Host.
Time Stamp	String	The time stamp (date and time) of the state changed (For example, from the NOT_RUNNING state to the RUNNING State, the time stamp of the host started and running successfully). Time Stamp returns long value, which is available in Hawk in a readable format of date and time.
Binding Status	String	The binding status of TIBCO Host (for example, bound).
Host Platform Version	String	Platform version of the Host.
Internet Host Name	String	Name of the Internet Host.
OS Name	String	The Operating System name of the machine running the host.

Name	Type	Description
OS Version	String	The Operating System version of the machine running the host.
System Arch	String	System Architecture of the machine running the host.
Stop Reason	String	The reason of the state host is in. It may be null during the normal operation.

getNodeInfo

The getNodeInfo method returns the Node information by the Node Name.

If the Node Name argument is blank, then all ActiveMatrix nodes are returned. On providing the Node Name argument, it serves as a regular expression used to filter the nodes returned.

Type

Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
Node Name	String	The node name.

Returns

Name	Type	Description
Process ID	String	The Process ID of the ActiveMatrix Node (OS Process ID.) Note: Process ID return "-1", if Node is not running.
Name	String	The name of the Node.
Host	String	The name of the host associated with the node.

Name	Type	Description
Runtime State	String	<p>The actual state of the node as reported by the host.</p> <ul style="list-style-type: none"> • NOT_RUNNING • INITIALIZING • INITIALIZED • INITIALIZATION_FAILED • STARTING - Starting of the node. The starting state should move to the running state BEFORE the nodes are started. • RUNNING - The node is running successfully. • START_FAILED • STOPPING • STOPPED • LOST_CONTACT - Lost contact with the node. Reasons being the node was killed unexpectedly, or the network is no longer operational. • UNKNOWN
Stop Mode	String	<p>In which mode the node is stopped. Node can be stopped in abort, immediate or normal mode through the ActiveMatrix Admin UI or end the process from the OS or using the API to the Stop Node.</p> <ul style="list-style-type: none"> • abort - The node is killed immediately. This mode is however not supported on all the HPAs. If you attempt to abort a node on a HPA that does not support node abort, the status block for that node shows an "operation-not-supported" error. • immediate - It implies that the node's infrastructure and applications may not have been able to shut down in a clean manner. This will potentially leave the node in an inconsistent state that will have to be recovered prior to the subsequent successful startup. • normal - Implies that all infrastructure and applications running on the node would have completed all or any task(s) normally associated with their shutdown

Name	Type	Description
Stop Reason	String	Stop reason text message. If the end process is from the OS, then reason returned is "OSGI Framework is stopping".
Stop Reason Code	String	Stop reason error code. Note: Most of the time it is Reason null and reason code -1 .
Time Stamp	String	The time stamp (date and time) of the state changed (For example, from the NOT_RUNNING state to the RUNNING State, the time stamp of the node started and running successfully). Time Stamp returns long value, which is available in Hawk in a readable format of date and time. Note: If an ActiveMatrix node was stopped while starting the Hawk agent, then getNodeInfo() would return 0 in the timestamp column for that particular node.



During the microagent startup, if the node state appears as `Starting` in ActiveMatrix Administrator UI, the `getNodeInfo()` method does not return the correct node state unless the node is restarted.

When the node is restarted (that is, when the microagent receives a Start notification from ActiveMatrix Administrator), the microagent returns the correct state of the node.



On Windows, when a node is stopped using "Terminate Node Process" option in the TIBCO ActiveMatrix Administrator, then the last state reported is `LOST_CONTACT` instead of `NOT_RUNNING`.

onHostEvent

The `onHostEvent` method provides notification of the changed run-time state of a TIBCO Host. This method supports only the subscribe mode.

Type

Asynchronous, `IMPACT_INFO`.

Arguments

None.

Returns

Name	Type	Description
Process ID	String	The Process ID of the TIBCO Host (OS Process Id)
Name	String	The name of the ActiveMatrix host (AMXAdminHost).

Name	Type	Description
Type	String	The type of the ActiveMatrix host (AMXAdminHost).
Runtime State	String	<p>The state of the ActiveMatrix host.</p> <ul style="list-style-type: none"> UNKNOWN NOT_RUNNING INITIALIZING INITIALIZING_FAILED INITIALIZED STARTING: The starting of the host. The starting state should move to the running state BEFORE starting the nodes. In other words, Host state is independent of Node State. Nodes can start unless the Host starts successfully. STARTING_FAILED: The starting process failed and the reason on the status will be set. STOPPING STOPPED RUNNING: The host is running normally. UNBOUND: The host is not bound to an enterprise. LOSING_CONTACT LOST_CONTACT: Lost contact with a host. Reason being the host was killed unexpectedly, heartbeats are lost in a flood of traffic, or the network is no longer operational.
Version	String	Version of the ActiveMatrix Host.
Time Stamp	String	The time stamp (date and time) of the state changed (For example, from the NOT_RUNNING state to the RUNNING State, the time stamp of the Host started and running successfully). Time Stamp returns long value, which is available in Hawk in a readable format of date and time.
Binding Status	String	Status of Host binding.
Host Platform Version	String	Version of Host Platform.
Internet Host Name	String	Internet Host name.
OS Name	String	Operating System name of the host.

Name	Type	Description
OS Version	String	Operating System version of the host.
System Arc	String	System Architecture of the Host.
Stop Reason	String	The reason of the state the host is in. It may be null during the normal operation.

onNodeEvent

The onNodeEvent method provides notification of the run-time state of the node when changed of a particular node or all nodes associated with the TIBCO Host.

If the Node Name argument is blank then it provides notification for all nodes associated with the TIBCO Host.

This method supports only subscribe mode.

Type

Asynchronous, IMPACT_INFO.

Arguments

Name	Type	Description
Node Name	String	The node name.

Returns

Name	Type	
Name	String	The name of the node.
Runtime State	String	The state of the node.
Host Name	String	The name of the host associated with the node.
Stop Mode	String	In which mode the node is stopped. Node can be stopped in abort , immediate or normal mode through the ActiveMatrix Admin UI or end the process from the OS or using the API to the Stop Node.
Stop Reason	String	Stop reason text message.
Stop Reason Code	String	Stop reason error code.

Name	Type	
Time Stamp	String	The time stamp (date and time) of the state changed (For example, from the NOT_RUNNING state to the RUNNING State, the time stamp of the node started and running successfully). Time Stamp returns long value, which is available in TIBCO Hawk in a readable format of date and time.

startNode

The startNode method starts the TIBCO ActiveMatrix node, if not running.

Type

Synchronous, IMPACT_ACTION.

Arguments

Name	Type	Description
Node Name	String	The node name.
Node start mode	String	Two types of mode can be specified for a node start-up. The Clean mode will explicitly clear the OSGi wiring during node start-up. Default value is normal.

Exceptions

Condition	Result
If node does not exist.	Throws a MicroAgentException with Message "TIBCO-AMX-HPA-012106: node TestNode does not exist".
If node is already started/running.	Throws a MicroAgentException with Message "Provided node name '<nodename> already started for host '<hostname>".

Returns

None.

stopNode

The stopNode method stops the TIBCO ActiveMatrix node.

Type

Synchronous, IMPACT_ACTION.

Arguments

Name	Type	Description
Node Name	String	The node name.
Reason	String	The stop reason text message.
Reason Code	String	The stop reason error code.

Exceptions

Condition	Result
If node does not exist.	Throws a <code>MicroAgentException</code> with Message "Provided node name '<nodename> not found ' for host '<hostname>'".
If node is already stopped.	Throws a <code>MicroAgentException</code> with Message "Provided node name '<nodename> already stopped ' for host '<hostname>'".

Returns

None.

Common Logging Event Publisher Microagent

The Common Logging Event Publisher Microagent (com.tibco.hawk.commonlogging.event.CLEventPublisher) publishes events from TIBCO Hawk using the TIBCO Common Logging in the Base Event Format (BEF) or Common Base Event (CBE).

Create the Event Definition configuration file with the following definitions:

- What events have to be published (Event Output).
- Event Data Source (from where to get and collect the events).
- Where to publish events (Common Logging Configuration)
- How to publish (Event Subscribe).

Sample event definition is available in the folder *TIBCO_HOME/amx/<version>/hkam/templates/hma/cl*.

CLEvent Publisher Microagent Configuration

For configuration of CLEventPublisher Microagent, refer to the *readme.txt* available in *TIBCO_HOME/amx/<version>/hkam/templates/hma/cl*.

Methods

The methods in the microagent allows to start or stop event publisher and returns event publisher's information.

startPublisher

The startPublisher method starts the event publisher to start publishing events from the TIBCO Hawk based on the Event Publisher Description defined in the Event Definitions file.

Type

Synchronous, *IMPACT_ACTION*.

Arguments

Name	Type	Description
Event Publisher Name	String	The Event Publisher Name.

Returns

None.

stopPublisher

The stopPublisher method stops the Event Publisher.

Type

Synchronous, *IMPACT_ACTION*.

Arguments

Name	Type	Description
Event PublisherName	String	The Event Publisher Name.

Returns

None.

startAllPublishers

The startAllPublishers method starts all valid and not started Event Publishers.

Type

Synchronous, IMPACT_ACTION.

Arguments

None.

Returns

None.

stopAllPublishers

The stopAllPublishers method stops all started Event Publishers.

Type

Synchronous, IMPACT_ACTION.

Arguments

None.

Returns

None.

getEventPublisherInfo

The getEventPublisherInfo method returns the Event Publisher information by Event Publisher name.

If the event publisher name argument is blank then all event publishers are returned. On specifying the event publisher name argument, it serves as a regular expression used to filter the event publisher returned.

Type

Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
Event Publisher Name	String	The Event Publisher Name.

Returns

Name	Type	Description
Name	String	Name of the Event Publisher.
Status	String	The current status of the event publisher (started/stop).
Auto Start	String	The flag to indicate event publisher is auto start configured.
Interval Time	String	The interval time to publish events.
Subscribe On Event Source Name	String	The name of the Event Source on the Event Publisher Subscribe.
CL Logger Name	String	The TIBCO Common Logging logger name.
CL Extended Model Name	String	The TIBCO Common Logging Extended Model name.
CL Extended Model Package Name	String	The TIBCO Common Logging Extended Model package name.

getEventDataSourceDetails

The getEventDataSourceDetails method returns the event data source details.

Type

Synchronous, IMPACT_INFO.

Arguments

None.

Returns

Name	Type	Description
EventSourceId	String	ID of the Event Source.
Microagent	String	The name of the TIBCO Hawk microagent.
Method	String	The name of the microagent method.

Name	Type	Description
Status	String	<p>The status of the event source.</p> <ul style="list-style-type: none"> Valid - Whether the microagent available method name is valid. Invalid - Whether the microagent is available and/or not available and/or method is not valid.
Description/Help	String	Description/Help of the microagent method or reason for its invalid status.

reloadEventDefinitions

The reloadEventDefinitions method stops all started event publishers, reloads event definitions from the configuration file and starts all valid event publishers with an auto start.

You can make changes in the configuration file and reflect changes without restarting the TIBCO Hawk Agent.

Type

Synchronous, `IMPACT_ACTION`.

Arguments

None.

Returns

None.

Event Definitions XSD Details

File: `<HAWK_HOME>/plugin/commonlogging/eventdefs.xsd`

Content:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://xsd.tns.tibco.com/hawk/cleventpublisher/
eventdef" elementFormDefault="qualified"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:eventdef="http://
xsd.tns.tibco.com/hawk/cleventpublisher/eventdef">
  <xsd:annotation>
    <xsd:documentation>Provide details of Event Publisher</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType name="EventSourceType">
    <xsd:attribute name="id" type="xsd:string" use="required"></xsd:attribute>
    <xsd:attribute name="methodName" type="xsd:string" use="required"></
xsd:attribute>
    <xsd:attribute name="microAgentName" type="xsd:string" use="required"></
xsd:attribute>
  </xsd:complexType>
  <xsd:complexType name="EventDataSourceType">
    <xsd:sequence>
      <xsd:element name="EventSource" type="eventdef:EventSourceType"
minOccurs="1" maxOccurs="unbounded"></xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="EventDefinitionsType">
    <xsd:sequence>
      <xsd:element name="EventPublishers" type="eventdef:EventPublishersType"
```

```

minOccurs="1" maxOccurs="1">
    </xsd:element>
    <xsd:element name="EventDataSource" type="eventdef:EventDataSourceType"
minOccurs="1" maxOccurs="1"></xsd:element>
    <xsd:element name="CommonLoggingConfigs"
type="eventdef:CommonLoggingConfigsType" minOccurs="1" maxOccurs="1">
    </xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="EventPublisherType">
    <xsd:sequence>
        <xsd:element name="CommonLoggingConfigRef"
            type="eventdef:CommonLoggingConfigRefType" minOccurs="1"
            maxOccurs="1">
        </xsd:element>
        <xsd:element name="EventDataSourceRef"
            type="eventdef:EventDataSourceRefType" minOccurs="1"
            maxOccurs="1">
        </xsd:element>
        <xsd:element name="EventOutput"
            type="eventdef:EventOutputType" minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="EventSubscribe"
            type="eventdef:EventSubscribeType" minOccurs="1" maxOccurs="1">
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required"></xsd:attribute>
    <xsd:attribute name="validateAtStartup" type="xsd:boolean" use="optional"
default="false"></xsd:attribute>
</xsd:complexType>
<xsd:complexType name="CommonLoggingConfigType">
    <xsd:annotation>
        <xsd:documentation>Common Logging details.</xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="ExtendedModel" type="eventdef:ExtendedModelType"
minOccurs="1" maxOccurs="1"></xsd:element>
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:string" use="required"></xsd:attribute>
    <xsd:attribute name="loggerName" type="xsd:string" use="required"></
xsd:attribute>
</xsd:complexType>
<xsd:element name="EventDefinitions" type="eventdef:EventDefinitionsType"></
xsd:element>
    <xsd:complexType name="EventDataSourceRefType">
        <xsd:sequence>
            <xsd:element name="EventSourceRef" type="eventdef:EventSourceRefType"
minOccurs="1" maxOccurs="unbounded"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="EventSourceRefType">
        <xsd:sequence>
            <xsd:element name="InputParameters" type="eventdef:InputParametersType"
minOccurs="0" maxOccurs="1"></xsd:element>
            <xsd:element name="EventCorrelation"
type="eventdef:EventCorrelationType" minOccurs="0" maxOccurs="1"></xsd:element>
        </xsd:sequence>
        <xsd:attribute name="ref" type="xsd:string" use="required"></xsd:attribute>
    </xsd:complexType>
    <xsd:complexType name="InputParametersType">
        <xsd:sequence>
            <xsd:element name="InputParameter" type="eventdef:InputParameterType"
minOccurs="1" maxOccurs="unbounded"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="InputParameterType">
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute name="name" type="xsd:string" use="required"></
xsd:attribute>
            </xsd:extension>
        </xsd:simpleContent>

```



```

</xsd:complexType>
<xsd:complexType name="EventOutputType">
  <xsd:sequence>
    <xsd:element name="EventElements" type="eventdef:EventElementsType"
minOccurs="1" maxOccurs="1"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="EventElementsType">
  <xsd:choice>
    <xsd:element name="EventElement"
      type="eventdef:EventElementType" minOccurs="0"
      maxOccurs="unbounded">
    </xsd:element>
    <xsd:element name="eventSourceRefs" type="xsd:string" minOccurs="1"
maxOccurs="1"></xsd:element>
  </xsd:choice>
</xsd:complexType>
<xsd:complexType name="EventElementType">
  <xsd:attribute name="source" type="xsd:string" use="required"></
xsd:attribute>
  <xsd:attribute name="output" type="xsd:string" use="optional"></
xsd:attribute>
</xsd:complexType>
<xsd:complexType name="EventSubscribeType">
  <xsd:sequence>
    <xsd:element name="IntervalTime" type="xsd:int"
      minOccurs="0" maxOccurs="1" nillable="false">
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="sourceEventId" type="xsd:string" use="required"></
xsd:attribute>
  <xsd:attribute name="autoStart" type="xsd:boolean" use="optional"
default="false"></xsd:attribute>
</xsd:complexType>
<xsd:complexType name="FilterType">
  <xsd:attribute name="identifier" type="xsd:string" use="required"></
xsd:attribute>
  <xsd:attribute name="operator" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="="></xsd:enumeration>
        <xsd:enumeration value=">"></xsd:enumeration>
        <xsd:enumeration value="<"></xsd:enumeration>
        <xsd:enumeration value="=>"></xsd:enumeration>
        <xsd:enumeration value="<="></xsd:enumeration>
        <xsd:enumeration value="!="></xsd:enumeration>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="LoggerType">
  <xsd:attribute name="name" type="xsd:string" use="required"></xsd:attribute>
  <xsd:attribute name="configFile" type="xsd:string" use="optional"></
xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ExtendedModelType">
  <xsd:sequence>
    <xsd:element name="Classpath" type="eventdef:ClasspathType"
minOccurs="0" maxOccurs="1"></xsd:element>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required"></xsd:attribute>
  <xsd:attribute name="packageName" type="xsd:string"
    use="required">
  </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="EventPublishersType">
  <xsd:sequence>
    <xsd:element name="EventPublisher" type="eventdef:EventPublisherType"
minOccurs="1" maxOccurs="unbounded"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CommonLoggingConfigsType">

```

```

        <xsd:sequence>
          <xsd:element name="CommonLoggingConfig"
type="eventdef:CommonLoggingConfigType" minOccurs="1" maxOccurs="unbounded">
            </xsd:element>
          </xsd:sequence>
          <xsd:attribute name="logConfigFile" type="xsd:string" use="required"></
xsd:attribute>
        </xsd:complexType>
        <xsd:complexType name="CommonLoggingConfigRefType">
          <xsd:attribute name="ref" type="xsd:string" use="required"></xsd:attribute>
        </xsd:complexType>
        <xsd:complexType name="EventCorrelationType">
          <xsd:annotation>
            <xsd:documentation>To correlate with other Event Data Source
</xsd:documentation>
          </xsd:annotation>
          <xsd:simpleContent>
            <xsd:extension base="xsd:string">
              <xsd:attribute name="returnElement" type="xsd:string"
use="required">
                <xsd:annotation>
                  <xsd:documentation>Return Element of Hawk MicroAgent Method
</xsd:documentation>
                </xsd:annotation></xsd:attribute>
              </xsd:extension>
            </xsd:simpleContent>
          </xsd:complexType>
          <xsd:complexType name="ClasspathType">
            <xsd:sequence>
              <xsd:element name="path" type="xsd:string" minOccurs="1"
maxOccurs="unbounded"></xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:schema>

```

Sample Event Definitions Configuration to CLEvents from Hawk

The JVMAndProcessInfoEventdefs.xml is the sample Event Definitions configuration to publish the JVM Info and Process Info event of the AMX 3 TIBCO Host instance, and all nodes associated with this Host Instance.

File: <HAWK_HOME>\plugin\commonlogging\JVMAndProcessInfoEventdefs.xml.

Content:

```

<?xml version="1.0" encoding="UTF-8"?>
<def:EventDefinitions xmlns:def="http://xsd.tns.tibco.com/hawk/cleventpublisher/
eventdef"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xsd.tns.tibco.com/hawk/cleventpublisher/eventdef
eventdefs.xsd">
  <!-- Event Publishers Definition for
    1. TIBCO Host JVM Process Info
    2. Process Events for TIBCO Host and AMX Nodes running in local Machine. --
>
  <!-- Event Publishers Details -->
  <def:EventPublishers>
    <!--Sample Event Publisher to Publish Process Info Event for TIBCO AMX Host
-->
    <def:EventPublisher name="AMXHostProcessInfoEventPublisher">
      <!-- Common Logging Configuration Details -->
      <def:CommonLoggingConfigRef ref="ProcessCLConfig" />
      <!--Event Data Source Ref -->
      <def:EventDataSourceRef>
        <def:EventSourceRef ref="AMXHost"/>
        <def:EventSourceRef ref="Process">
          <def:EventCorrelation returnElement="ID Process">${
AMXHost.Process Id}</def:EventCorrelation>
        </def:EventSourceRef>
      </def:EventDataSourceRef>
      <!-- Event Ouput -->

```

```

        <def:EventOutput>
            <def:EventElements>
                <def:EventElement source="${Process.ID Process}"
output="processId" />
                <def:EventElement source="${Process.Process Name}"
output="processName" />
                <def:EventElement source="${Process.User Name}"
output="userName" />
                <def:EventElement source="${Process.CPU Time}"
output="processCpuTime" />
                <!-- On Windows platform Virtual KBytes,Stack KBytes,Heap
KBytes, % CPU and % Memory not available-->
                <def:EventElement source="${Process.Virtual KBytes}"
output="virtualMemory" />
                <def:EventElement source="${Process.Stack KBytes}"
output="stackSize" />
                <def:EventElement source="${Process.Heap KBytes}"
output="heapSize" />
                <def:EventElement source="${Process.% CPU}"
output="cpuUsageInPercent" />
                <def:EventElement source="${Process.% Memory}"
output="memoryUsageInPercent" />
                <def:EventElement source="${Process.Start time}"
output="startTime" />
                <def:EventElement source="${AMXHost.Name}"
output="physicalCompId.matrix.host" />
            </def:EventElements>
        </def:EventOutput>
        <!-- Event Subscription Details -->
        <def:EventSubscribe sourceEventId="AMXHost" autoStart="false">
            <def:IntervalTime>10000</def:IntervalTime>
        </def:EventSubscribe>
    </def:EventPublisher>
    <!--TIBCO Host JVM InfoEvent Publisher -->
    <def:EventPublisher name="AMXHostJVMInfoEventPublisher">
        <def:CommonLoggingConfigRef ref="JVMnfoEventCLConfig" />
        <def:EventDataSourceRef>
            <def:EventSourceRef ref="AMXHost"/>
            <def:EventSourceRef ref="JVMInfo">
                <def:EventCorrelation returnElement="Process Id">${
AMXHost.Process Id}</def:EventCorrelation>
            </def:EventSourceRef>
        </def:EventDataSourceRef>
        <def:EventOutput>
            <def:EventElements>
                <def:EventElement source="${AMXHost.Name}"
output="physicalCompId.matrix.host" />
                <def:EventElement source="${JVMInfo.Maximum Heap Size}"
output="maxMemory" />
                <def:EventElement source="${JVMInfo.Committed Memory}"
output="totalMemory" />
                <def:EventElement source="${JVMInfo.Free Heap Size}"
output="freeMemory" />
                <def:EventElement source="${JVMInfo.Non Heap Memory Used}"
output="nonHeapMemoryInUse" />
                <def:EventElement source="${JVMInfo.Start Time}"
output="startTime" />
                <def:EventElement source="${JVMInfo.Up Time}" output="uptime" />
                <def:EventElement source="${JVMInfo.Live Threads}"
output="threadCount" />
                <def:EventElement source="${JVMInfo.Daemon Threads}"
output="daemonThreadCount" />
                <def:EventElement source="${JVMInfo.Peak Threads}"
output="peakThreadCount" />
            </def:EventElements>
        </def:EventOutput>
        <def:EventSubscribe sourceEventId="AMXHost" autoStart="false">
            <def:IntervalTime>10000</def:IntervalTime>
        </def:EventSubscribe>
    </def:EventPublisher>
    <!--AMX Node(s) ProcessEventInfo Publisher -->
    <def:EventPublisher name="AMXNodeProcessInfoEventPublisher">

```

```

        <def:CommonLoggingConfigRef ref="ProcessCLConfig" />
        <def:EventDataSourceRef>
            <def:EventSourceRef ref="AMXNodeInfo">
                <def:InputParameters>
                    <def:InputParameter name="Node Name"></def:InputParameter>
                </def:InputParameters>
            </def:EventSourceRef>
            <def:EventSourceRef ref="Process">
                <def:EventCorrelation returnElement="ID Process">${
{AMXNodeInfo.Process Id}</def:EventCorrelation>
            </def:EventSourceRef>
        </def:EventDataSourceRef>
        <def:EventOutput>
            <def:EventElements>
                <def:EventElement source="${Process.ID Process}"
output="processId" />
                <def:EventElement source="${Process.Process Name}"
output="processName" />
                <def:EventElement source="${Process.User Name}"
output="userName" />
                <def:EventElement source="${Process.CPU Time}"
output="processCpuTime" />
                <!-- On Windows platform Virtual KBytes,Stack KBytes,Heap
KBytes, % CPU and % Memory not available-->
                <def:EventElement source="${Process.Virtual KBytes}"
output="virtualMemory" />
                <def:EventElement source="${Process.Stack KBytes}"
output="stackSize" />
                <def:EventElement source="${Process.Heap KBytes}"
output="heapSize" />
                <def:EventElement source="${Process.% CPU}"
output="cpuUsageInPercent" />
                <def:EventElement source="${Process.% Memory}"
output="memoryUsageInPercent" />
                <def:EventElement source="${Process.Start time}"
output="startTime" />
                <def:EventElement source="${AMXNodeInfo.Name}"
output="physicalCompId.matrix.node" />
            </def:EventElements>
        </def:EventOutput>
        <def:EventSubscribe sourceEventId="AMXNodeInfo" autoStart="false">
            <def:IntervalTime>10000</def:IntervalTime>
        </def:EventSubscribe>
    </def:EventPublisher>
</def:EventPublishers>
<!-- Event Data Source Details -->
<def:EventDataSource>
    <def:EventSource id="Process" methodName="getProcess"
microAgentName="COM.TIBCO.hawk.hma.Process" />
    <def:EventSource id="AMXHost" methodName="getHostInfo"
microAgentName="com.tibco.hawk.amx.AMXHost" />
    <def:EventSource id="AMXNodeInfo" methodName="getNodeInfo"
microAgentName="com.tibco.hawk.amx.AMXHost" />
    <def:EventSource id="JVMInfo" methodName="getVirtualMachineInfo"
microAgentName="com.tibco.hawk.jvm.JavaVirtualMachine" />
</def:EventDataSource>
<!-- Common Logging Configuration -->
<def:CommonLoggingConfigs logConfigFile="%TIBCO_HAWK_HOME_ESC%/plugin/
commonlogging/sample_log4j.xml">
    <def:CommonLoggingConfig id="ProcessCLConfig"
loggerName="hawk.clevent.logger">
        <def:ExtendedModel
packageName="com.tibco.governance.cl.extension.model.process.impl.ProcessInfoEventPa
ckageImpl"
name="com.tibco.governance.cl.extension.model.process.impl.ProcessInfoEventImpl" />
    </def:CommonLoggingConfig>
    <def:CommonLoggingConfig id="JVMnfoEventCLConfig"
loggerName="hawk.clevent.logger">
        <def:ExtendedModel
packageName="com.tibco.governance.jvminfo.model.jvminfoevent.impl.JvminfoeventPackag
eImpl"

```

```

name="com.tibco.governance.jvminfo.model.jvminfoevent.impl.JVMInfoEventImpl">
    <def:Classpath>
        <def:path>%TIBCO_COMPONENT_STORE_PLUGINS%/
com.tibco.governance.jvminfo.model_1.0.0.004.jar</def:path>
        <def:path>%TIBCO_COMPONENT_STORE_PLUGINS%/
com.tibco.governance.commonlogging.extension.jvminfo.client_1.0.0.004.jar</def:path>
    </def:Classpath>
</def:ExtendedModel>
</def:CommonLoggingConfig>
</def:CommonLoggingConfigs>
</def:EventDefinitions>

```

Sample Common Logging Configuration file

File: <HAWK_HOME>/plugin/commonlogging/sample_log4j.xml

Content:

```

<?xml version="1.0" encoding="UTF-8"?>
<main_configuration>
    <base_hierarchy>
        <log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
            <appender name="CommonLoggingJMSAppender"
class="com.tibco.commonlogging.appender.BEFJMSAppender">
                <param name="serverUrl" value="tcp://localhost:7222" />
                <param name="queueName" value="amx.governance.stats" />
                <param name="userName" value="admin" />
                <param name="password" value="" />
                <param name="type" value="direct" />
            </appender>
            <logger name="hawk.clevent.logger" additivity="false">
                <appender-ref ref="CommonLoggingJMSAppender"/>
            </logger>
            <logger name="root" additivity="false">
                <level value="INFO" />
                <appender-ref ref="CommonLoggingJMSAppender" />
            </logger>
        </log4j:configuration>
    </base_hierarchy>
</main_configuration>

```

Refer to the <HAWK_HOME>/plugin/commonlogging folder for configuration details.

ActiveMatrix Service Microagent

The ActiveMatrix Service Microagent (com.tibco.hawk.amx.AMXService) allows runtime monitoring of the ActiveMatrix components and services on each local node.

Methods

The methods in the microagent allows to change monitoring configurations and returns components and bindings information.

getConfig

The getConfig method provides monitoring configuration details.


Type

Synchronous, IMPACT_INFO.

Arguments

None.

Returns

Name	Type	Description
AMXHost	String	The name of the ActiveMatrix host associated with the node.
Environment	String	The name of the environment.
Node	String	The name of the node.
Monitoring Application State	String	Runtime state of Monitoring Application.  To get service execution statistics, monitoring application must be in RUNNING state.
Notification Enabled	Long	Returns true or false for statistics notification enable or disable
Notification Interval	Long	Notification Interval in milliseconds.
Rolling Window Length	Long	Rolling window length in milliseconds.
Number of Buckets	Long	Number of time intervals in rolling window.
Statistics Gathering Interval	Long	Statistics gathering interval time in milliseconds.
Is Partial Window	Boolean	Indicates whether the sliding window statistics includes the current working bucket.

changeMonitoringConfig

The changeMonitoringConfig method provides option to change monitoring configuration.

User can change rolling window length, statistics gathering interval (bucket size), notification interval and isPartialWindow. This modification overrides configuration details provided by monitoring application and reset after monitoring application restart.

Type

Synchronous, IMPACT_ACTION.

Arguments

Name	Type	Description
Rolling Window Length (in seconds)	Long	Select rolling window length in seconds. Possible values are: 60, 120, 180, 240, 300.
Statistics Gathering Interval (in seconds)	Long	Selects statistics gathering interval time in seconds. The content of list box are: 5, 10, 15, 20, 30, 40, 50, 60.
Notification Interval (in seconds)	Long	Selects notification interval in seconds. The content of list box are: 10, 15, 20, 30, 40, 50, 60.
Is Partial Window	Boolean	Select true/false for the sliding window statistics include the current working bucket? Possible values are: true, false.

Returns

None.

GetComponentInfo

The GetComponentInfo method returns Component information by Component and Application name. It serves as a regular expression used to filter the components returned.

Type

Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.

Name	Type	Description
Component Name	String	A substring that matches the component name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Runtime State	String	<p>Select Runtime</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • Blank: All • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED

Returns

Type: `COM.TIBCO.hawk.talon.TabularData`

Element Name	Type	Description
AMX Host	String	The name of the ActiveMatrix host associated with the node.
Node	String	The name of the node.
Environment	String	The name of the environment.
Application	String	The name of the application.
Component	String	The name of the component.
Runtime State	String	<p>Runtime state of the component.</p> <ul style="list-style-type: none"> • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED • START_FAILED • WAITING_FOR_DEPENDENCIES
Version	String	Version of the component.

Element Name	Type	Description
URI	String	URI of the component
Time Stamp	String	Time stamp of the changed runtime state of the component.



Application Key generated from Application Name, if application deployed only in one environment without any space in words of Application, then Application Key is same of Application Name. But if application contains spaces then spaces replaced with “_” underscore. If same application deployed across the environment with same name then Application name postfix with incremental of one.

getBindingInfo

The getBindingInfo method returns service and reference binding information by binding and application name. It serves as the regular expression used to filter the bindings returned.

Type

Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
BindingName	String	A substring that matches the name of the binding. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Runtime State	String	Select Runtime Possible values are: <ul style="list-style-type: none"> • Blank: All • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED

Returns

Type: `COM.TIBCO.hawk.talon.TabularData`

Element Name	Type	Description
AMX Host	String	ActiveMatrix host Name
Node	String	ActiveMatrix Node Name
Environment	String	Environment Name
Application	String	Application Name
Binding	String	Binding Name
Binding Type	String	Binding Type (Service or Reference)
Contract	String	Contract name (that is, Service or Reference name)
Runtime State	String	Runtime state of binding <ul style="list-style-type: none"> • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED • START_FAILED • WAITING_FOR_DEPENDENCIES
Binding URI	String	Binding URI
Time Stamp	String	Time stamp of runtime state change of binding.

onComponentEvent

The onComponentEvent method provides notification when the runtime state of component is changed.

Notification can be filtered by component and application name and it serves as the regular expression used to filter the components returned.

Type

Asynchronous, IMPACT_INFO.

Arguments

Name	Type	Description
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Component Name	String	A substring that matches the component name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Runtime State	String	Select runtime state of the component. Possible values : <ul style="list-style-type: none"> • Blank: All • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED

Return

Type: `COM.TIBCO.hawk.talon.CompositeData`

Element Name	Type	Description
AMX Host	String	The name of the ActiveMatrix host.
Node	String	Name of the node.
Environment	String	The name of the environment.
Application	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Component	String	A substring that matches the component name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding Type	String	Binding type (Service or Reference)

Element Name	Type	Description
Runtime State	String	Runtime state of the component. <ul style="list-style-type: none"> • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED • START_FAILED • WAITING_FOR_DEPENDENCIES
Component URI	String	Component URI.
Time Stamp	String	Time stamp of runtime state change of the binding.

onBindingEvent

The onBindingEvent method provides notification when the runtime state of service or reference binding is changed.

Notification can be filtered by binding and application name and it serves as the regular expression used to filter the bindings returned.

Type

Asynchronous, IMPACT_INFO

Arguments

Name	Type	Description
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding Name	String	A substring that matches the name of the service or reference binding. An empty string (the default) matches all names. The substring provided can also be a regular expression.

Name	Type	Description
Runtime State	String	<p>Select runtime state of the binding.</p> <p>Possible values:</p> <ul style="list-style-type: none"> • Blank : All • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED

Return

Type: `COM.TIBCO.hawk.talon.CompositeData`

Element Name	Type	Description
AMX Host	String	The name of the ActiveMatrix host.
Node	String	Name of the Node.
Environment	String	The name of the environment.
Application	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding	String	A substring that matches the name of the binding. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding Type	String	Binding type (Service or Reference)
Contract	String	Contract name (that is, Service or Reference name)

Element Name	Type	Description
Runtime State	String	Runtime state of binding <ul style="list-style-type: none"> • RUNNING • NOT_RUNNING • STOPPED • LOST_CONTACT • INSTALLED • UNINSTALLED • START_FAILED • WAITING_FOR_DEPENDENCIES
Binding URI	String	Binding URI.
Time Stamp	String	Time stamp of runtime state change of the binding.

getBindingOperationStats

The `getBindingOperationStats` method provides statistics of service and reference binding operation execution by operation name, service or reference binding name, application name and binding type. It serves as the regular expression used to filter the operations statistics returned.

Type

Synchronous, `IMPACT_INFO`.

Arguments

Name	Type	Description
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding Name	String	A substring that matches the name of the binding. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Operation Name	String	A substring that matches the operation name. An empty string (the default) matches all names. The substring provided can also be a regular expression.

Name	Type	Description
Binding Type	String	Select type. Possible values are : <ul style="list-style-type: none"> • <none: All service and reference bindings. • SERVICE: Only service binding. • REFERENCE: Only reference bindings.

Return

Type: `COM.TIBCO.hawk.talon.TabularData`

Element Name	Type	Description
AMX Host	String	ActiveMatrix host name.
Node	String	Node name.
Environment	String	Environment name.
Application	String	Application name.
Binding	String	Service or Reference binding name.
Binding Type	String	Binding type (that is, Component Service or Reference name).
Contract	String	Contract name.
Operation	String	Service operation name
Binding URI	String	Service or Reference binding URI.
Endpoint Direction	String	Endpoint direction (that is, Inbound or Outbound)
Service Execution Stats		Stats. Refer Service Execution Stats .

getComponentOperationStats

The `getComponentOperationStats` method provides statistics of component service and reference operation execution by operation name, component service or reference name, application name and type. It serves as the regular expression used to filter the operations statistics returned.

Type

Synchronous, `IMPACT_INFO`

Arguments

Name	Type	Description
Operation Name	String	A substring that matches the operation name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Contract Name	String	A substring that matches the component service or reference name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Component Name	String	A substring that matches the component name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Contract Type	String	<p>Select type.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • <none>: All service and reference. • SERVICE: Only Component Services. • REFERENCE: Only Component References.

Return

Type: `COM.TIBCO.hawk.talon.TabularData`

Element Name	Type	Description
AMX Host	String	ActiveMatrix host name.
Node	String	Node name.
Environment	String	Environment name.
Application	String	Application name.
Component	String	Component name.
Contract	String	Contract name.
Contract Type	String	Contract type (that is, Service or Reference)

Element Name	Type	Description
Operation	String	Operation name
URI	String	Component URI.
Service Execution Stats		Stats. Refer Service Execution Stats .

onBindingOperationStats

The onBindingOperationStats method provides notification in each interval period defined in monitoring configuration with statistics of service and reference binding operation for last <n> by operation name, binding name, application name and binding type. It serves as the regular expression used to filter the operations statistics returned.

Type

Asynchronous, IMPACT_INFO

Arguments

Name	Type	Description
Operation Name	String	A substring that matches the operation name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding Name	String	A substring that matches the name of the service or reference binding. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Binding Type	String	Select type. Possible values are: <ul style="list-style-type: none"> • <none>: All service and reference bindings. • SERVICE: All Service bindings. • REFERENCE: All Reference bindings.

Return

Type: COM.TIBCO.hawk.talon.TabularData

Element Name	Type	Description
AMX Host	String	ActiveMatrix host name.

Element Name	Type	Description
Node	String	Node name.
Environment	String	Environment name.
Application	String	Application name.
Binding	String	Service or Reference binding name.
Binding Type	String	Binding type (that is, Service or Reference
Contract	String	Contract Name.
Operation	String	Service operation name
Binding URI	String	Service or Reference binding URI.
Endpoint Direction	String	Endpoint direction (that is, Inbound or Outbound)
Service Execution Stats		Stats. Refer Service Execution Stats .

onComponentOperationStats

The onComponentOperationStats method provides notification in each interval period defined in monitoring configuration with statistics.

The statistics include component service and reference operation for the last <n> by operation name, component service or reference name, component name, application name and type and it serves as the regular expression used to filter the operations statistics returned.

Type

Asynchronous, IMPACT_INFO

Arguments

Name	Type	Description
Operation Name	String	A substring that matches the operation name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Contract Name	String	A substring that matches the component service or reference name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Component Name	String	A substring that matches the component name. An empty string (the default) matches all names. The substring provided can also be a regular expression.

Name	Type	Description
Application Name	String	A substring that matches the application name. An empty string (the default) matches all names. The substring provided can also be a regular expression.
Contract Type	String	Select type. Possible values are: <ul style="list-style-type: none"> • <none>: All Component Services and References. • SERVICE: All Component Services. • REFERENCE: All Component References.

Return

Type: `COM.TIBCO.hawk.talon.TabularData`

Element Name	Type	Description
AMX Host	String	ActiveMatrix host name.
Node	String	Node name
Environment	String	Environment name.
Application	String	Application name
Component	String	Component name
Contract	String	Contract name
Contract Type	String	Contract type (that is, Service or Reference)
Operation Name	String	Service operation name
URI	String	Component URI.
Service Execution Stats		Stats. Refer to Service Execution Stats .

Service Execution Stats

The statistics-fetching method of ActiveMatrix Service Microagent returns different statistics for the service executed.

Column Name	Type	Description
Start Time	String	The start time of the collection period ISO-8601 format.

Column Name	Type	Description
End Time	String	The end time of the collection period ISO-8601 format.
Total Time	Long	Collection period in milliseconds.
Hits	Double	Number of execution responses for last period.
Faults	Double	Number of execution faults for last period.
Pending Responses	Double	Difference between count of requests and responses at end of the window.
Avg. Response Time	Double	Average response time for collection period, in milliseconds.
Min Response Time	Double	Minimum response time over collection period, in milliseconds.
Max Response Time	Double	Maximum response time over collection period, in milliseconds.
Hits Per Minute	Double	Average number of executions per minute over collection period.
Faults Per Minute	Double	Average number of faults per minute over collection period.
Fault Percent	Double	Ratio of faults to total hits between 0.0 and 100.0
Standard Deviation	Double	Standard Deviation in response time over collection period.

Configuration of TIBCO ActiveMatrix Hawk Rulebase Samples

The samples cover different scenarios depending upon the state of various nodes and EMS server.

Samples are available in `TIBCO_HOME/amx/<version>/samples/hawk/rulebases/`. Refer to the `readme.txt` file in the folder.

The sample provides the functionality related to the AMX Host microagent and the TIBCO EMS microagent:

- Detect when the connected ActiveMatrix Host (TibcoHost) is abnormally shutdown and restart it. Restart is only attempted if, the connected EMS server is running.
- Detect when the Administrator Node (SystemNode) was abnormally shutdown and restart it. (Assumes that the SystemNode is also managed by the connected ActiveMatrix Host). Restart is only attempted when the Admin database (if external) is running, and the connected EMS server is running.

A sample DBping microagent is included to support this scenario. Note that the rulebases to support this scenario B are only setup when the ActiveMatrix Administrator instance is created via TCT.

- Auto-detect runtime nodes managed by the connected ActiveMatrix Host (TibcoHost), and provide or generate rulebases for each node that restart the node when abnormally shutdown. Restart is only attempted when the connected EMS server is running.

The rulebases and scripts are auto-configured from the output of a TIBCO Configuration Tool (TCT) run.

When TCT is run on a machine to setup either an ActiveMatrix Administrator instance or a TIBCO Host instance, its output directory can be used to configure the Hawk rulebases and scripts to manage the installed ActiveMatrix Administrator and/or TIBCO Host and nodes.

Scenarios

The rulebase samples are configured for different scenarios related to the AMX Host microagent and TIBCO EMS microagent.

Scenario A: When Tibco Host is Abnormally Shutdown

Detect when connected TIBCO Host is abnormally shutdown, and restart only if EMS server is running:

Rulebase Name:

```
SystemNodeRB-${tibcohost.instance.name}
```

Microagents Used:

- `com.tibco.hawk.amx.AMXHost`
- `com.tibco.hawk.tibjms.HawkListener`
- `COM.TIBCO.hawk.hma.Process`

Procedure

1. Use `AMXHost.isHostRunning()` to determine if the tibcohost is running
if Running is `False`
then post 'th_not_running' condition and send Medium level alert: "TibcoHost is not running".
2. Use `AMXHost.getHostInfo` to determine runtime state of the tibcohost.

- if Runtime State = 'LOST_CONTACT' or Runtime State='NOT_RUNNING'
- then post 'th_lost_contact' condition and send Medium alert: "Lost Contact with Tibco Host, Host is in a NOT_RUNNING state".
3. Use `HawkListener<tibcohost.ems.url>.isRunning()` to determine the state of the EMS server connected to tibcohost.

if running is True

then post `ems.server.<tibcohost.ems.url>.isrunning` condition

if running is False

then send High alert: "Tibcohost Qin EMS Server is down: Rules will NOT restart TH".
 4. Use `Process.getInstanceCount(ProcessName='tibcohost')` to determine whether there is a running OS process for the tibcohost.

if Process Count < 1 AND the following posted conditions exist:

`ems.server.<tibcohost.ems.url>.isrunning` AND `th_not_running` then execute `start_tibcohost` script, and send High alert: "ActiveMatrix Host is not running, going to restart it."

Scenario B: When Administrator Node is Abnormally Shutdown

Detect when the Administrator Node (SystemNode) was abnormally shutdown and restart it, only if the external database (if used) is alive, and if the connected EMS server is running.

Rulebase Name:

- For external database - `SystemNodeRB-extdb-${tibcohost.instance.name}`
- For embedded database - `SystemNodeRB-${tibcohost.instance.name}`

Microagents Used:

- `com.tibco.hawk.amx.AMXHost`
- `com.tibco.hawk.tibjms.HawkListener`
- `COM.TIBCO.hawk.hma.Process`
- `com.tibco.hawk.samples.DBPingMicroAgent`

During auto-configuration of the rulebases, determine whether an external database is used.

Scenario C: Runtime Nodes Managed by connected AMX Host

Auto-detect runtime nodes managed by the connected AMX Host (TibcoHost), and provide or generate rulebases for each node that restart the node when abnormally shutdown. Restart is only attempted when the connected EMS server is running.

Rulebase Name:

`AMXNodeManagerRB`

Microagents Used:

- `com.tibco.hawk.amx.AMXHost`
- `COM.TIBCO.hawk.microagent.RuleBaseEngine`

Use `AMXHost.getNodeInfo` to determine the names of all nodes (every 30 seconds).

If Node Name != 'SystemNode' then execute the ANT project: 'scripts/anrunner generate-noderb.xml' with argument amx.node.name= Node Name. After it, then execute the method RuleBaseEngine.loadRuleBaseFromFile(\${Node Name}RB-\${tibcohost.instance.name}.hrb).



The ANT project generate-noderb.xml will generate a rulebase file in the Hawk autoconfig folder called '\${Node Name}RB-\${tibcohost.instance.name}.hrb'.

This rulebase is based on a template that has the following behavior.

Rulebase Template Name:

AMXNodeRB

Microagents Used:

- com.tibco.hawk.amx.AMXHost
- com.tibco.hawk.tibjms.HawkListener
- COM.TIBCO.hawk.hma.Process

Procedure

1. Use AMXHost.onNodeEvent to determine runtime state of the AMXNode.
if Runtime State = 'LOST_CONTACT' or Runtime State='NOT_RUNNING'
then post th_lost_contact_node condition and send High Alert: "Lost Contact with \${Node Name}".
2. Use HawkListener<tibcohost.ems.url>.isRunning() to determine the state of the EMS server connected to tibcohost.
if running is True
then post ems.server.<tibcohost.ems.url>.isrunning condition.
3. Use Process.getInstanceCount(ProcessName=\${Node Name}) to determine whether there is a running OS process for the AMXNode.

Result

if Process Count < 1 AND the following posted conditions exist:

ems.server.<tibcohost.ems.url>.isrunning AND th_lost_contact_node

then execute AMXHost.startNode('\${Node Name}').

Deploying and Testing Sample Application

The sample ActiveMatrix application (`BookOrderTutorial.app`) can be deployed on ActiveMatrix 3.x installations for use in testing the TIBCO Hawk ActiveMatrix Plug-in.

This application is built using two Java components and two Binding Services.

The Java components used are:

- `BookOrderComponent`
- `BookSearchComponent`

The binding services used are:

- `BookOrderService`
- `BookSearchService`

Each has one operation that is, `orderBook` and `searchBook` (respectively).

The `BookOrderComponent` invokes the `BookSearchComponent` with a Component reference.

The summary of monitorable binding services and references, component services and references is given (this detail is important to understand the monitoring results from the ActiveMatrix Service microagent).

Binding and Component Details

ApplicationName	<code>BookOrderTutorial.app</code>
Bindings	<code>BookOrderServiceBinding</code> <code>BookSearchServiceBinding</code>
BookOrderComponent	Services: <code>BookOrderService</code> (<code>Contract=BookOrderPT</code>) References: <code>BookOrderReference</code> (<code>Contract=BookSearchPT</code>)
BookSearchComponent	<code>ServicesBookSearchService</code> (<code>Contract=BookSearchPT</code>)

Testing the Sample Application

A load test is provided which exercises the `BookOrderTutorial` application previously deployed. You need to run this load test.

Prerequisites

- Running installation of ActiveMatrix
 1. Go to `TIBCO_HOME/amx/<version>/samples/hawk/amxproject`.
 2. Edit the `build.properties` to set the `TIBCO_HOME` (root directory for ActiveMatrix installation).
 3. Edit the `remote_props.properties` to configure the `adminURL` and authentication settings for your Admin Server.

4. Run the ANT project with the following command (replace \$TIBCO_HOME with your installation path):
 - `java -cp $TIBCO_HOME/tools/lib/antpackage.jar`
 - `org.apache.tools.ant.launch.Launcher deploy.bookorder`
- soapUI (see soapui.org)

Procedure

1. Import the provided soapUI project : `TIBCO_HOME/amx/<version>/samples/hawk/amxproject/BookOrderTutorial/soapui/AMX3BookOrderTutorial-soapui-project.xml`.
A load test is provided which exercises the BookOrderTutorial application previously deployed. This load test is called "orderBookLoadTest".
2. Run this load test.
You can now verify the service metrics using the ActiveMatrix Service microagent.