Adabas Analytics on Linux, UNIX and Windows

Version 2.2

October 2017

This document applies to Adabas Analytics on Linux, UNIX and Windows Version 2.2 and 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: EAL-DOC-22-20171018

Table of Contents

Preface

This documentation describes the product Adabas Analytics for Linux, Unix and Windows platforms.

1 Concepts

Typically, an Adabas database is used in a commercial environment, and the data contained in the database are usually of a sensitive and confidential nature. Seen in this context, it is important to be able to answer the following questions (sometimes called the 5 W questions):

- Who has accessed the data?
- What has been accessed? This includes the database ID, the file number, the type of access (create, read, update, delete), the field names, etc.
- When was the data accessed?
- Where was the data accessed from?
- What has changed in the internal state of the database?

These 5 questions are of vital importance for the following reasons:

Fraud prevention
Identify security incidents in operational databases; who is accessing sensitive data?

Auditing
Keep track of and analyse compliance-relevant results; who did what, from where and when?

Performance monitoring
Central diagnosis of database performance and efficiency; how well is Adabas running?

Adabas Analytics adresses these requirement by enabling you to create an event each time there is a change of state in the Adabas nucleus.

A change of state can be triggered by:

- An Adabas call;
- A security event (authorization succeeded or failed, authentication succeeded or failed, etc.);
- A change in performance status (threshold reached, disk space exhausted, etc.).

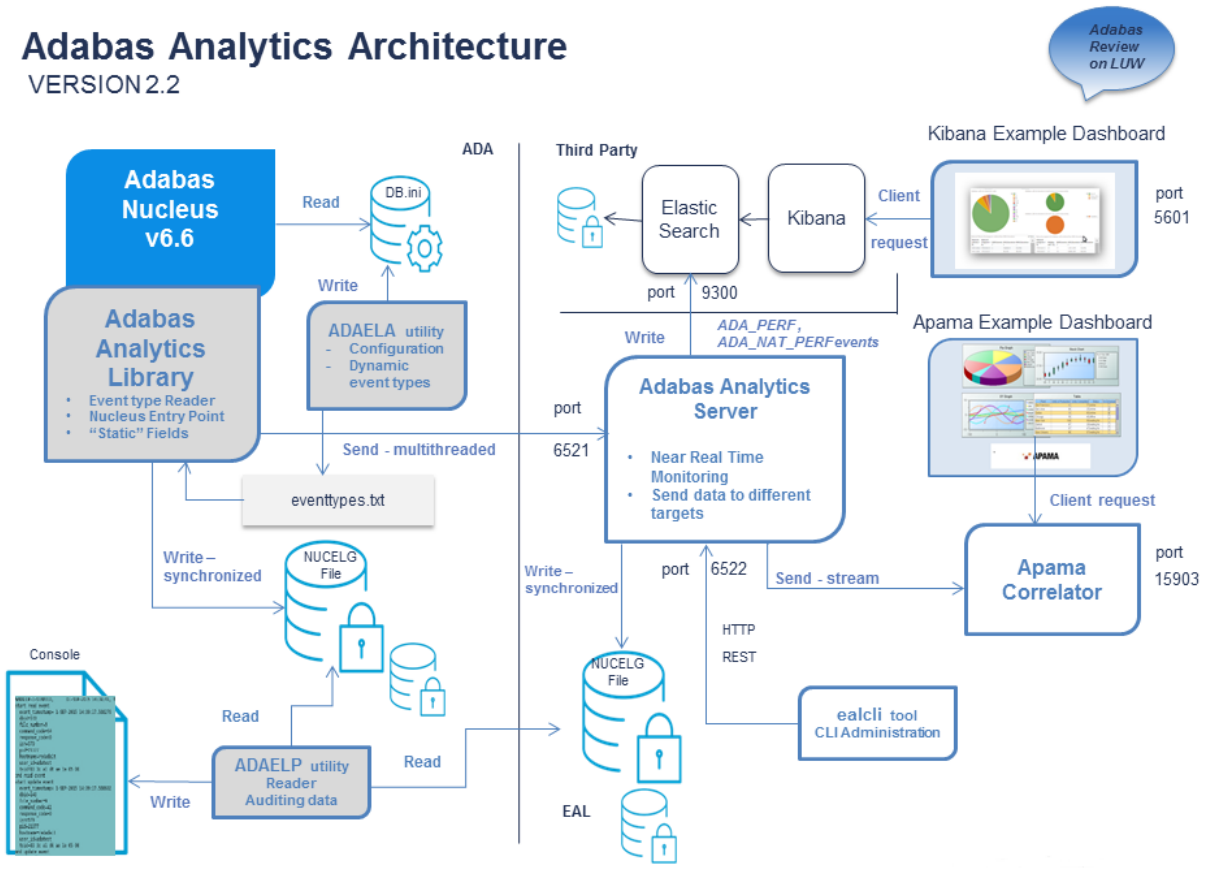
An event consists of a set of fields for data that is available at the time when it is generated.

Adabas Analytics currently supports 14 types of events related to Adabas calls. For further information about the event types, see the section *Adabas Analytics Event Types*. More event types relating to security and performance will be supported in later versions.

If yout only need to use Adabas Analytics sporadically (depending on your use case), it is possible to activate/deactivate the event anayltics component. Also, because only certain event types might be of interest, you can easily filter events by file number and event type.

The events generated by Adabas are either sent to the Adabas Analytics Server or written to a local log file called NUCELG; you can display the contents of this file with the new Adabas utility ADAELP (for further information, see the section *ADAELP (Event Log Report)*.

The following graphic shows the architecture of Adabas Analytics Version 2.2:



Adabas Analytics and Dynamic Event Types

Starting with Adabas Analytics Version 2.2, it is possible to create user-defined event types.

An event type consists of :

- An event type name;
- The area in the Adabas nucleus where the event is triggered;
- A number of event type fields.

The event type name must be unique within all event types.

The area must be one of the pre-defined nucleus areas.

The event type fields must come from a list of pre-defined event type fields; all event type fields must be compatible with the area of the event type.

The utility ADAELA, which provides commands to add and delete event types, is used to define an event type. For details, please refer to the section ***ADAELA (Event Analytics Administration)***.

The defined event types are stored in the file *eventtypes.txt*. The ADANUC process reads the contents of *eventtypes.txt* during startup.

When the Adabas nucleus executes one of the defined areas, it triggers all of the event types defined for that particular area.

Adabas Analytics and Elasticsearch/Kibana

Adabas Analytics uses Elasticsearch and its visualization component Kibana to store and visualize Adabas performance data. The combination of the Adabas nucleus, Adabas Analytics, Elasticsearch and Kibana lets you analyze Adabas performance data in near-realtime. The Kibana visualization of the data is in a browser interface.

The installation kit includes an example Kibana dashboard, which you can use to display Adabas performance data.

Adabas Analytics and Apama

The Adabas nucleus creates the Adabas Event Logfiles (`NUCELG.xxxx`) if the Adabas Eventing functionality is enabled.

The Adabas Analytics File Converter reads a single Adabas Event Logfile and sends it to the Apama Correlator.

In the Apama Correlator, the received events can be processed like any Apama event: use them in an Apama monitor or an Apama Correlator dashboard.

The Adabas Analytics File Converter and the Apama Correlator can be running on the same node or on distributed nodes.

2 Installing Adabas Analytics

The Adabas Analytics is installed using the Software AG Installer. Please refer to *Using the Software AG Installer* for detailled information about how to use the installer.

Prerequisites

The following prerequisites must be met for this version of Adabas Analytics:

- Java Version 1.8 or higher; an appropriate Java runtime is provided during the installation.
- Adabas Version 6.6.0 or higher.
- Natural Version 8.4.1 or higher.

Note: Only required if new event types like ADA_NAT_PERF are to be monitored.

Supported Operating System Platforms (Linux/Unix)

Adabas Analytics supports the following operating system platforms:

- AIX 7.1 (Power 64 bit)
- AIX 7.2 (Power 64 bit)
- HP-UX 11.i v3 (Itanium 64bit)
- Red Hat Enterprise Linux Server 6 (IBM System z 64bit)
- Red Hat Enterprise Linux Server 7 (IBM System z 64bit)
- Red Hat Enterprise Linux Server 6 (x86-64)
- Red Hat Enterprise Linux Server 7 (x86-64)
- Oracle Solaris 11 (SPARC 64bit)
- SUSE Linux Enterprise Server 11 (IBM System z 64bit)
- SUSE Linux Enterprise Server 11 (x86-64)
- SUSE Linux Enterprise Server 12 (x86-64)

Supported Operating Platforms (Windows)

Adabas Analytics supports the following operating system platforms:

- Windows Server 2008 R2 (Standard and Enterprise Edition, x86-64)
- Windows Server 2012 (Standard and Datacenter Edition, x86-64)
- Windows Server 2012 R2 (Standard and Datacenter Edition, x86-64)
- Windows 7 (Professional, Ultimate and Enterprise Edition, x86-64)
- Windows 8 (Pro and Enterprise Edition, x86-64)
- Windows 10 (Pro and Enterprise Edition, x86-64)

Home Editions of Microsoft Windows are not supported.

Notes:

1. We suggest that you install all of the recommended and important Microsoft Windows updates before you start the installation.
2. On Windows 8.1 and Windows Server 2012 R2 the installation will fail if the Microsoft update KB2919355 is missing.

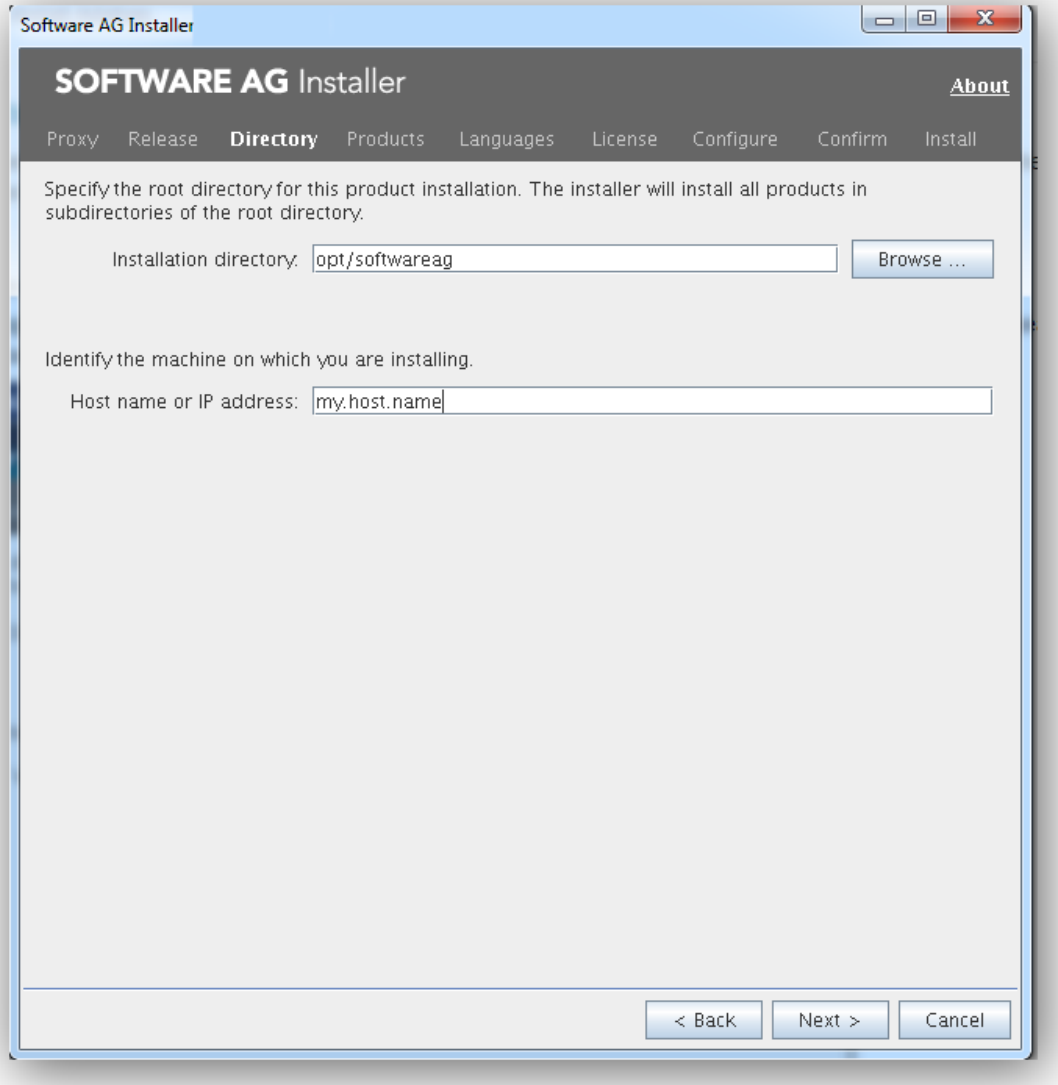
Installing on Linux/Unix

This installation documentation provides just a brief description on how to install Adabas Analytics directly on the target machine using the Software AG Installer GUI. For detailed information on the Software AG Installer, see *Using the Software AG Installer*.

To install Adabas Analytics

1 Start the Software AG Installer GUI as described in *Using the Software AG Installer*.

2 When the first page of the Software AG Installer GUI (the so-called Welcome panel) is shown, press the **Next** button repeatedly (and specify all required information on the shown panels as described in *Using the Software AG Installer*) until the panel containing the installation directory appears.



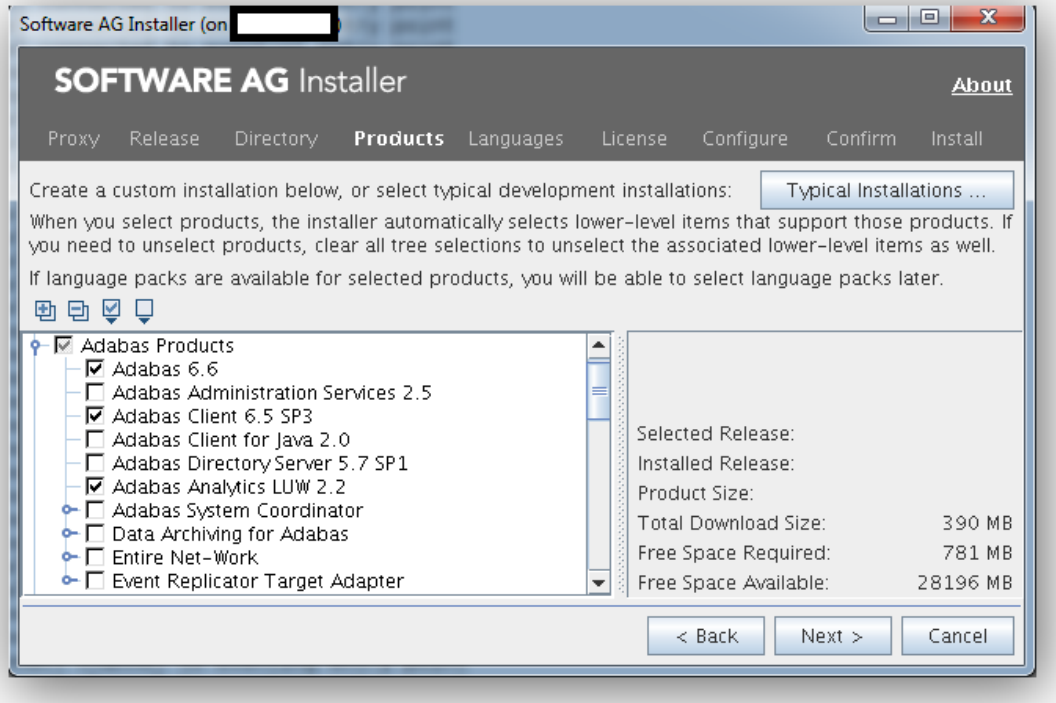
3 Specify the root directory and host name or IP address (optional).

Note: It is strongly recommended not to use the proposed default directory */opt/softwareag* but a sub-directory, for example */opt/softwareag/adabasanalytics22* to install the release version of Adabas Analytics 2.2. With this approach you can install several versions of Adabas Analytics in parallel directories.

4 Press the **Next** button.

Note: The panel shown below is an example of a possible product selection.

The panel containing the product selection tree appears. This tree lists the products for which you have valid credentials and which can be installed on the operating system of the machine on which you are installing.



Note: Products or product versions which are already installed in the selected installation directory are shown as disabled.

5 If you want to install Adabas and pre-selected product components, select the **Adabas Products** node.

Or:

If you want to customize the list of selected product components, expand the **Adabas Products** node, deselect Adabas Products and select the product components that you want to install.

6 If you want to install Adabas Client, select **Adabas Client** in the product selection tree. The Adabas Client is always installed together with Adabas, but can also be installed separately.

7 Press the **Next** button.

8 Read the license agreement, select the check box to agree to the terms of the license agreement, and press the **Next** button.

9 Specify whether to use sudo or not.

Some parts of the installation require root permissions. On the following sudo panel you must either select **Use sudo, with password** suppling a valid sudo password or you can skip these installation steps by selecting **Do not use sudo or sudo is not available**.

You will then have to execute those steps as described on the panel shown below. Both alternatives are equivalent.

Note: Using sudo without specifying a password is not possible.

10 On the last panel, review the list of products and items you have selected for installation. If the list is correct, press the **Next** button to start the installation process.

When the Software AG Installer has completed the first-time installation, additional configuration steps are required. See *Configuring Adabas Anlytics* for further details.

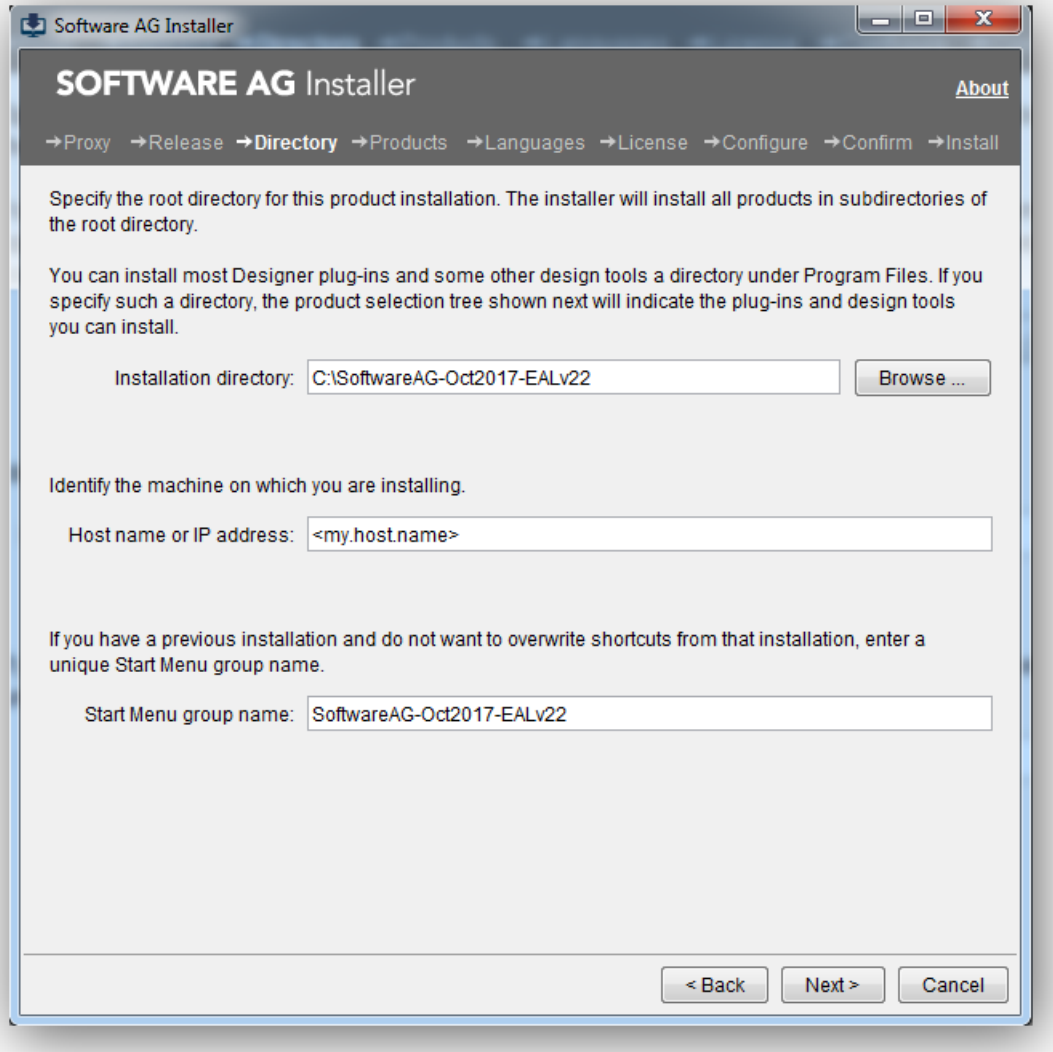
Installing on Windows

This installation documentation provides just a brief description on how to install Adabas Analytics directly on the target machine using the Software AG Installer GUI. For detailed information on the Software AG Installer, see *Using the Software AG Installer*.

To install Adabas

1 Start the Software AG Installer GUI as described in *Using the Software AG Installer*.

2 When the first page of the Software AG Installer GUI (the so-called Welcome panel) is shown, press the **Next** button repeatedly (and specify all required information on the shown panels as described in *Using the Software AG Installer*) until the panel containing the installation directory appears.



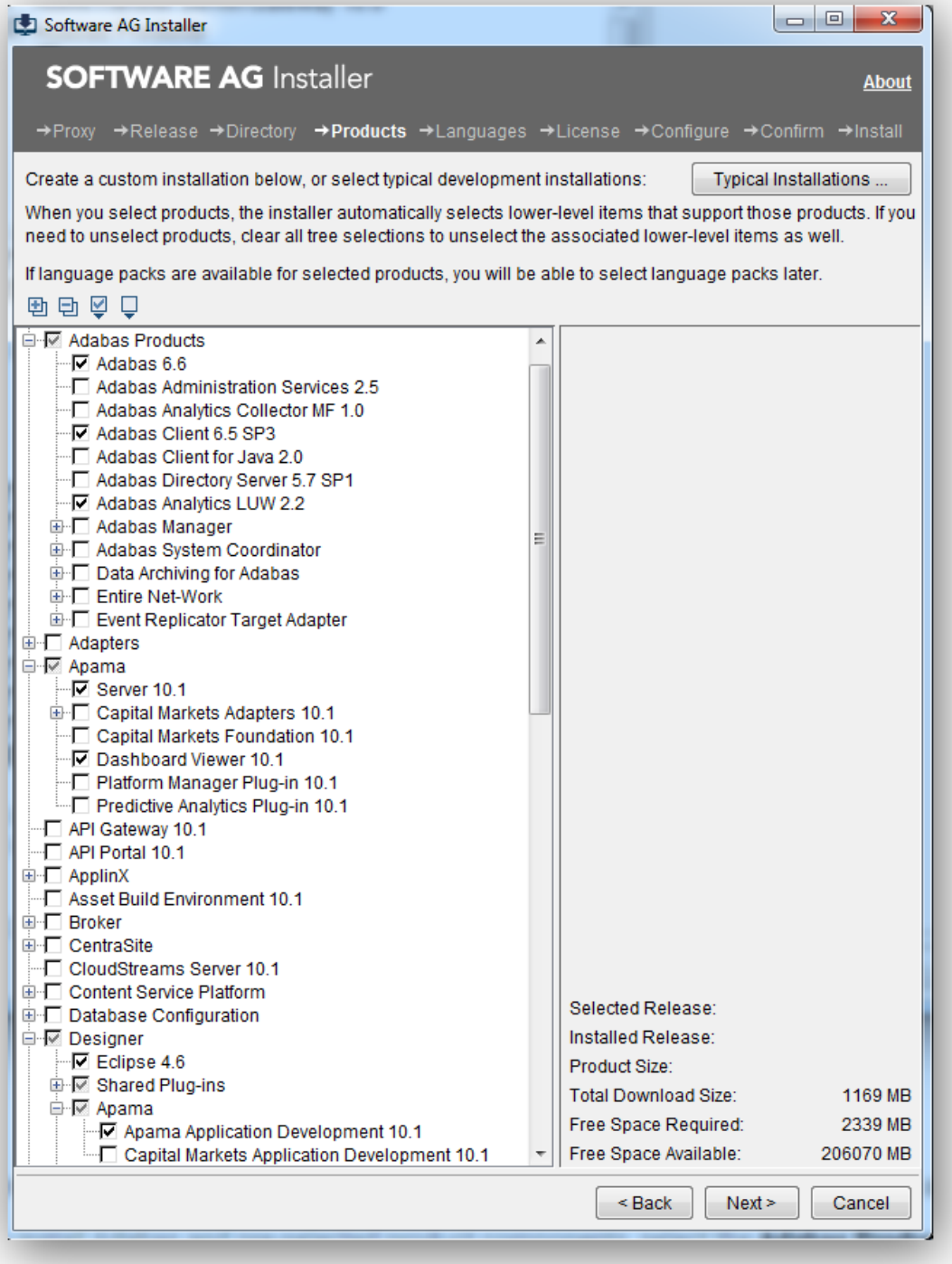
3 Specify the installation directory, host name or IP address (optional) and the Start Menu group name.

Note: It is strongly recommended not to use the proposed installation directory *C:\SoftwareAG* but a sub-directory, for example *C:\SoftwareAG\AdabasAnalytics22* and to change the Start Menu group name, for example to **Software AG Adabas Analytics 2.2** to install the release version of Adabas Analytics 2.2. With this approach you can install several versions of Adabas Analytics in parallel directories.

4 Press the **Next** button.

Note: The panel shown below is an example of a possible product selection.

The panel containing the product selection tree appears. This tree lists the products for which you have valid credentials and which can be installed on the operating system of the machine on which you are installing.



Note: Products or product versions which are already installed in the selected installation directory are shown as disabled.

5 If you want to install Adabas and pre-selected product components, select the **Adabas Products** node.

Or:

If you want to customize the list of selected product components, expand the **Adabas Products** node, deselect Adabas Products and select the product components that you want to install.

6 Press the **Next** button.

7 Read the license agreement, select the check box to agree to the terms of the license agreement, and press the **Next** button.

8 On the last panel, review the list of products and items you have selected for installation. If the list is correct, press the **Next** button to start the installation process.

When the Software AG Installer has completed the first-time installation, additional configuration steps are required. See *Configuring Adabas Analytics* for further details.

3 Getting Started

Once you have successfully installed Adabas Analytics, there are some steps that have to be performed before you can start to collect and display Adabas events:

1. Use the configuration utility ADAELA to make the entries in the *DBnnn.INI* file that are required to enable collecting Adabas events. Pleaser refer to ***ADAELA (Event Analytics Administration)*** for further information.
2. Use the event file converter to read and convert the contents of an existing Adabas event file, prior to displaying them in an Apama dashboard. Please refer to *The Event File Converter* for further information.
3. Install the third-party products Elasticsearch and Kibana if you want to visualize Adabas data in Kibana. Start the Adabas Analytics server to allow near real-time monitoring of your Adabas/Natural application. Please refer to *Getting Started with Elasticsearch/Kibana* for further information.
4. **Define own event types**.

4 Release Notes

This chapter gives an overview of the features of Adabas Analytics Version 2.2 that have been introduced or modified since the previous release (Version 2.1).

The chapter contains the following sections:

General Information

This section provides information which you should be aware of before you install and use Adabas Analytics Version 2.2.

Supported Operating Systems

Adabas Analytics supports the same operating systems and platforms as Adabas Version 6.6 for Linux, UNIX and Windows.

Software AG Installer

Adabas Analytics is installed using the Software AG installer. Please refer to the relevant installation documentation for further information.

Prerequisite Adabas and Natural LUW Versions

Adabas Version 6.6 is required because of extensions that have been added to support Adabas Analytics.

Natural Version 8.4.1 is required to support the 'Adabas Review on LUW' configuration using Natural parameter ADAPRM. Natural session data are sent to Adabas, and if Adabas is configured to catch *ADA_NAT_PERF* events (distributed via the Adabas Analytics Server), all data can be captured and visualized using the 3rd party product Elasticsearch/Kibana.

New, Modified and Dropped Features in Version 2.2

Adabas Extensions for Adabas Analytics

The configuration tool 'ealconfig' has been replaced by the new Adabas utility *ADAELA*, which is used to configure Adabas Analytics, as well as to define custom event types for a given number of fields.

Adabas Analytics Server

The Adabas Analytics Command Line Interface has been added to display configuration details and status information, as well as to enable/disable the use of *processors* and *sinks*.

License files (without time restrictions) are required to use one of the supported *sink* types:

- Apama Correlator
- Elasticsearch
- Raw File (compatible with Adabas *NUCELG* file)

New, Modified and Dropped Features in Version 2.1

Force Switch of NUCELG File (New)

You can now force a switch of the NUCELG file using the new ADAOPR function FEOF=ELOG.

Adabas Analytics Server (New)

The Adabas Analytics server, which is new with Version 2.1, helps to avoid bottlenecks when writing events to the NUCELG file, and is a first step towards near real-time monitoring. The Adabas Analytics server can be configured to support different targets - the Apama Correlator, or the new, third-party product Elasticsearch.

Note: It is still possible to send NUCELG files to Apama using the Adabas Analytics File Converter.

Elasticsearch/Kibana (New)

This version is delivered with the third-party products Elasticsearch and Kibana, which can be used to visualise data from the Adabas Analytics server in near real-time. An example Kibana dashboard is also provided.

EALCONFIG (Modified)

Compared to the version provided with the Adabas Version 6.5 package, the new version of EALCONFIG has the following major differences:

- New subtopics TARGET_EAL_SERVER and TARGET_NUCELG;
- The items SWITCH_AFTER_EVENTS and SWITCH_AFTER_TIME are moved to TARGET_NUCELG;
- The item TARGET_NUCELG is ignored if TARGET_EAL_SERVER exists;
- The new event types NAT_INSERT, NAT_READ, NAT_UPDATE, NAT_DELETE, NAT_COMMIT, NAT_ROLLBACK, ADA_PERF and ADA_NAT_PERF are supported.

New, Modified and Dropped Features in Version 2.0

Adabas Extensions for Adabas Analytics

You can configure the database INI files for use with Adabas Analytics with the tool EALCONFIG. You can use the utility ADAELP to print events from an event log created by Adabas. Both of these components are part of the Adabas kit.

Adabas Analytics Version 2.0 no longer uses the Adabas replication exit; all of the functions required to trigger collection of event data to the NUCELG file are now part of the Adabas kernel Version 6.5 and above.

Event File Converter

The Event File Converter is a program that sends a NUCELG file as a stream to an Apama correlator. Other targets are currently not supported.

Apama Dashboard

This version provides an example Apama dashboard, which can be imported into the Eclipse-based Software AG Designer.

Documentation Updates and Changes

The most recent product documentation, hotfixes and other useful information can be found in Empower.

5 Adabas Analytics Command Line Interface

The Adabas Analytics command line interface is used to get status information about the Adabas Analytics Server, as well as dynamically enable/disable the use of sinks and processors.

Note: The command line interface does not require an Adabas Analytics environment, which means that it is independent of any settings in *ealenv* and/or *sagenv.new.*

Using the Command Line Interface

Starting the Command Line Interface on Windows

Open a command prompt from the Windows start menu and change to the directory where the executable jar file is located (default: *bin* directory of the EAL package).

In the command prompt, enter the command:

```
java -jar ealcli-2.2.0.0-all.jar
```

Alternatively, you can use the script:

```
ealcli
```

Starting the Command Line Interface on Windows

Change to the directory where the executable jar file is located (default: *bin* directory of the EAL package).

Enter the command:

```
java -jar ealcli-2.2.0.0-all.jar
```

Alternatively, you can use the script:

```
ealcli.sh
```

Usage of the Command Line Interface

The command line interface supports the following commands and options:

```
usage: [ {-p | --pretty} ] [ {-q | --quiet} ] [ {-v | --verbose} ]
       [ {-t | --target} <host[:<port>]>
       <command> [ <args> ]
```

where:

-p, --pretty
Returns formatted output - so-called pretty print.

-q, --quiet
Only error levels will be returned.

-v, --verbose
Returns HTTP requests and response information.

-t, target <host[:port]>
Sends the command to the Adabas Analytics Server; the default HTTP port number is 6522.

The following commands are supported:

Command	Use
status	Displays the status of the Adabas Analytics Server.
list	Lists a set of processors or sinks.
disable	Disables a set of processors or sinks.
enable	Enables a set of processors or sinks.
shutdown	Initiates a shutdown of the Adabas Analytics Server.

<args>
The arguments of a command. Use *help<command>* to get more information about a specific command, for example:

```
    ... \AdabasAnalytics\bin>ealcli help disable
NAME
        ealcli disable - - disables a set of processors or sinks
SYNOPSIS
        ealcli [ {-v | --verbose} ] [ {-t | --target} <host[:<port>]> ]
                [ {-q | --quiet} ] [ {-p | --pretty} ] disable { processors | ↩
sinks }
                [--] <cmd-args>
        Where command-specific arguments <cmd-args> are:
            processors: [ <processor_1> <processor_2> <...> <processor_n>... ]
            sinks: [ <sink_1> <sink_2> <...> <sink_n>... ]
       See 'ealcli help disable <command>' for more information on a specific ↩
command.
```

```
OPTIONS
        -p, --pretty
            Pretty prints Adabas Analytics Server responses
        -q, --quiet
            Will suppress all outputs to standard output and standard error
        -t <host[:<port>]>, --target <host[:<port>]>
            Host and port of the target machine. It must have the form of
            <host>[:<port>]
        -v, --verbose
            Will show all HTTP requests/responses sent to/received from the
            Adabas Analytics Server
```

6 The Event File Converter

The Adabas nucleus creates the Adabas event log files NUCELG.xxxx if the Adabas Eventing functionality is enabled. The Adabas Analytics event file converter reads an Adabas event log file and sends it to the Apama Correlator, where the events can be processed like any other Apama event, and displayed in an Apama monitor or in an Apama Correlator dashboard.

Note: The Adabas Analytics file converter and the Apama Correlator can be running on the same node or on distributed nodes.

Using the Event File Converter

Starting the Event File Converter on Windows

The Windows start menu contains an entry for a command prompt, from where you can call the event file converter. In the command prompt window, issue the command:

```
AdabasAnalyticsFileConverter.bat ↩
```

Starting the Event File Converter on UNIX

Before you can start the event file converter, you must first source the environment file *ealenv*. Alternatively, you can source the top-level environment file *sagenv.new*, which in turn sources *ealenv*. Then issue the command

Usage of the Event File Converter

The event file converter is a command line utility with this syntax:

```
AdabasAnalyticsFileConverter -f <name> -t <host[:port]> [-h]
```

where:

-f, eventfile <name>
The name of the event file to be read.

-t, target <host[:port]>
Send the events from the event file to the Apama Correlator; the default port number is 15903.

7 Apama Example Dashboard

You can use the example Apama dashboard provided with the installation to display the events contained in an event log file.

The following files and folders are provided with the installation:

File/Folder	Description
AdabasAuditingEvents.mon	The event definitions file for the Adabas events of type Auditing.
AdabasInternalEvents.mon	The event definitions file for the Adabas events of type Internal.
AdabasMonitoringEvents.mon	The event definitions file for the Adabas events of type Monitoring
ApamaExampleDashboard	Adabas Analytics example Apama dashboard application, this has to be imported into the Eclipse-based Software AG Designer.

Note: the dashboard provided is just an example of how to catch Auditing events. Please refer to the Apama documentation for details about how to build your own dashboard.

Using the Software AG Designer

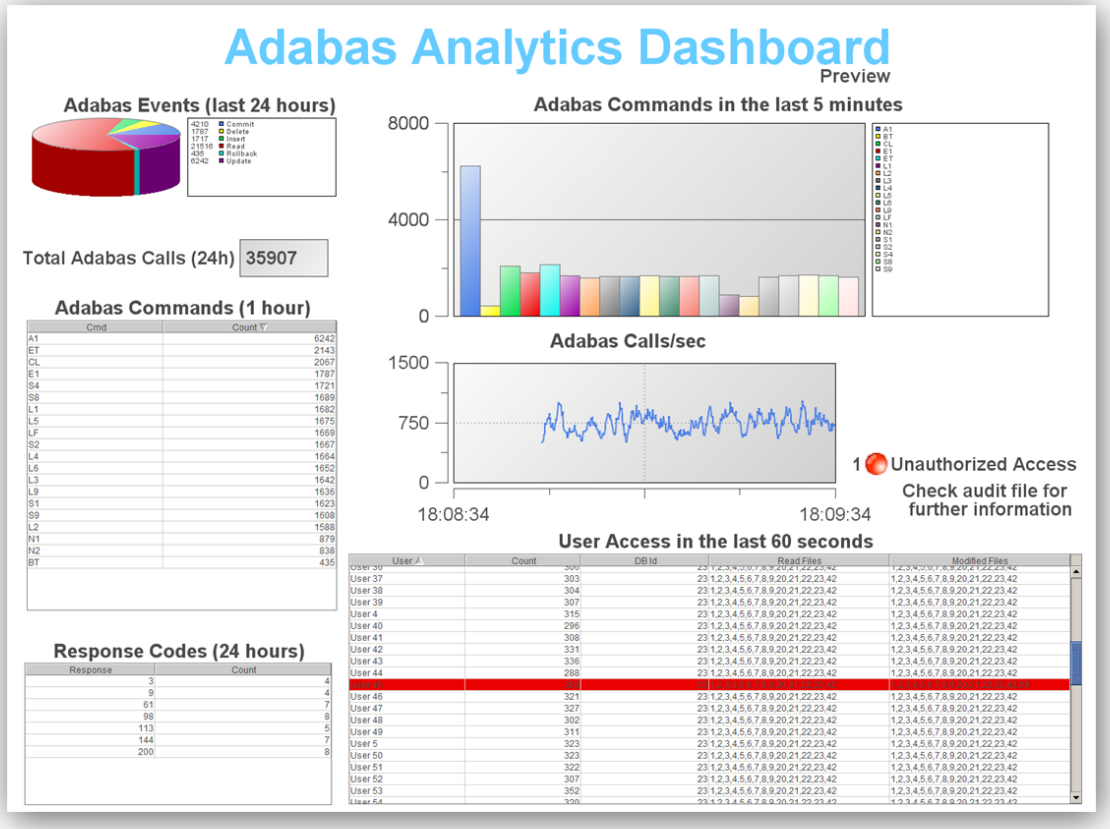
Before you can use the example provided, or develop your own applications in Apama, the event definitions for Adabas events have to be imported into an Apama project. In the Software AG designer, import the *Adabas<type>Events.mon* file into subdirectory *eventdefinitions*.

The files *AdabasEvents.mon*, *AdabasInternalEvents.mon* and *AdabasMonitoringEvents.mon* are also a part of the Apama dashboard example (*ApamaExampleDashboard* folder), which can be imported using the Software AG Import wizard.

To run the example as an Apama project in the Software AG Designer

1. Open the Software AG Designer.
2. Select **Import** from the **File** menu.
3. In the Import wizard, select and expand the *General* node, then select **Existing Projects into Workspace**.
4. Click the **Next** button, and the click on the **Browse** button in the Import Project step.
5. Navigate to *<installation directory>\AdabasAnalytics\apama\ApamaExampleDashboard* and select that folder.
6. In the **Options** panel of the **Import Projects** dialog, check the **Copy projects into workspace** check box.
7. To run the example, right-click the project and select **Run As -> Apama Application** from the **Apama Developer Perspective**. Then click on the **Start** button in the **Launch Control Panel** of the Apama Workbench Perspective.

The following shows an example of how the dashboard might look:



8 Elasticsearch/Kibana

Adabas Analytics can be used to analyze the performance of an Adabas database; it is possible to examine the performance data of each Adabas call executed in the database.

The current version of Adabas Analytics is delivered with the third-party product **Elasticsearch** (and its visualization component **Kibana**) to store and visualize the Adabas performance data. The combination of the Adabas nucleus, the Adabas Analytics server, Elasticsearch and Kibana lets you analyze Adabas performance data in near-realtime.

Getting Started with Elasticsearch/Kibana

Prerequisites

Adabas Version 6.5.1 and Adabas Analytics Version 2.1.0 must already be installed before you install and use Elasticsearch/Kibana.

The following default TCP ports are used:

- 6521, 6522 - Adabas Analytics server
- 9200, 9300 - Elasticsearch
- 5601 - Kibana

Please ensure that these ports are not already in use. Depending on your configuration and/or firewall settings, open this ports for inbound communications.

Installing, Configuring and Starting Elasticsearch

To install, configure and start Elasticsearch

1 Install the third-party software Elasticsearch from *$EALPROGDIR/third-party* by extracting the file *elasticsearch-5.1.2.zip* to a directory <*ELASTICSEARCH_INSTALL_DIR*> of your choice (Windows), or by extracting the file *elasticsearch-5.1.2.tar* to a directory <*ELASTICSEARCH_INSTALL_DIR*> of your choice (Unix).

2 Edit the Elasticsearch configuration file: the configuration file *elasticsearch.yml* is located in the *config* subdirectory of <*ELASTICSEARCH_INSTALL_DIR*>.

Change the line

```
#cluster.name: my-application
```

to

```
cluster.name: AdabasAnalyticsData
```

If the Adabas Analytics server and Elasticsearch are running on different hosts, change the line:

```
#network.host: 192.168.0.1
```

to

```
network.host: 0.0.0.0
```

Otherwise Elasticsearch will not be able to connect with the remote Adabas Analytics server.

You can find detailed information about configuring Elasticsearch here: **https://www.elastic.co/guide/en/elasticsearch/reference/current/settings.html**.

3 Start Elasticsearch by executing *ELASTICSEARCH_INSTALL_DIR\bin\elasticsearch.bat* (Windows), or *ELASTICSEARCH_INSTALL_DIR/bin/elasticsearch* (UNIX).

Elasticsearch should now be up and running on localhost:9200.

4 If Elasticsearch is running on a different host to the Adabas Analytics server, edit the Adabas Analytics configuration file *EALPROGDIR/configuration/config.xml*.

Change the line

```
<Location host="localhost" port="9300" />
```

to

```
<Location host="<ELASTICSEARCH_HOST>" port="9300" />
```

where <ELASTICSEARCH_HOST> is the name of the host on which Elasticsearch is running.

Installing, Configuring and Starting Kibana

To install, configure and start Kibana

1 Install the third-party software Kibana from *$EALPROGDIR/third-party* by extracting the file *kibana-5.1.2-windows-x86.zip* a to a directory <*KIBANA_INSTALL_DIR*> of your choice (Windows), or by extracting the file *kibana-5.1.2-linux-x86_64.tar.gz* to a directory <*KIBANA_INSTALL_DIR*> of your choice (UNIX).

2 Edit the Kibana configuration file: the configuration file *kibana.yml* is located in the *config* subdirectory of <*KIBANA_INSTALL_DIR*>.

Change the line

```
#elasticsearch.ssl.verify: true
```

to

```
elasticsearch.ssl.verify: false
```

If Kibana is installed on a different host to Elasticsearch, change the following lines (leave them unchanged if Kibana is installed on the same host as Elasticsearch):

Change

```
#server.host: "localhost"
```

to

```
server.host: <KIBANA_HOSTNAME>
```

where <KIBANA_HOSTNAME> is the name of the host on which Kibana is installed.

Change

```
#elasticsearch.url: "http://localhost:9200"
```

to

```
elasticsearch.url: "http://<HOSTNAME>:9200"
```

where <HOSTNAME> is the name of the host on which Elasticsearch is installed.

Change

```
#server.name: "your-hostname"
```

to

```
server.name: "your name"
```

You can find detailed information about configuring Kibana here: **https://www.elastic.co/guide/en/kibana/current/settings.html**.

3 Start Kibana by executing *KIBANA_INSTALL_DIR\bin\kibana.bat* (Windows), or *KIBANA_INSTALL_DIR/bin/kibana* (UNIX).

Kibana should now be up and running on localhost:5601.

Visualizing Adabas Analytics Data using Elasticsearch/Kibana

You can use the example Kibana dashboard provided with the installation to display Adabas/Natural performance data in near-realtime. The following section describes how to import and use the dashboard.

To visualize Adabas Analytics data using Elasticsearch/Kibana

1 Start the Adabas Analytics server.

On Windows:

```
EALPROGDIR/bin/AdabasAnalyticsServer.bat start
```

On UNIX:

```
EALPROGDIR/bin/AdabasAnalyticsServer.sh start
```

2 Create a demo Adabas database with the *crdemodb* utility.

```
crdemodb <dbid>
```

3 Run the configuration utility *ADAELA* and add an event filter to catch only ADA_NAT_PERF events.

For example:

```
adaela dbid=<nnn> ADD=FILTER,EVENT_FILTER=ADA_NAT_PERF
```

For further information about the *ADAELA* utility, see the section ***ADAELA (Event Analytics Administration)***.

4 Start the demo Adabas database that you created earlier.

```
adastart <dbid>
```

5 Generate some activity on the demo Adabas database. For example, run *getdbinfo <dbid>*, or run the *c_example.exe* on your database (*c_example.exe* is located in the */bin/examples* subdirectory of *AdabasClient*)

This step sends event data to Elasticsearch, and allows Elasticsearch to build the initial index structure.

6 Access Kibana from a browser (for example Firefox or Chrome) with the following URL:

```
http://<KIBANA_HOSTNAME>:5601
```

7 Create an index pattern for your event data:

1. In Kibana, go to **Management**->**Index Patterns**->**Add new**.
2. Keep the checkbox **Index contains time-based events** checked.
3. Enter the index name *adabas_analytics-**.
4. Wait for a few seconds and leave the time field selection at **event_timestamp**.
5. Click on the **Create** button.

Note: You can validate this step by going to **DevTools**->**Console** and entering the command `GET _cat/indices`. This should produce output of the following form:

```
...
... adabas_analytics-<CURRENT_DATE> ...  <CURRENT_SIZE>
... .kibana                         ...  <CURRENT_SIZE>
... ↩
```

8 You can now display your data in Kibana by going to **Discover**, and then selecting the index**adabas_analytics-*** from the drop-down box.

Note: If you don't see event data, try setting a longer period under review. In the top right corner of your Kibana browser window, click on **last 15 minutes** and select the timeframe for which you want to review the data.

9 Import the predefined Adabas/Natural dashboard for Kibana, as well as the visualization objects located in the directory *AdabasAnalytics/third-party/KibanaExampleDashboard*. In Kibana, go to **Management**->**Saved Objects**->**Import** and select the delivered json files (one at a time).

Note: The visualization objects can only be imported if index data already exists in ElasticSearch. A corresponding message is displayed if problems occur.

Troubleshooting

If an error occurs, check the contents of the *ealserver.log* file in the *log* subdirectory of your Adabas Analytics installation. Look for the entry "Elastic Search Sink "ElasticSearch" sending events to cluster <YOUR_CLUSTER_NAME>" where *YOUR_CLUSTER_NAME* matches the *cluster.name* entry in the *elasticsearch.yml* configuration file. If the log file contains the above entry and if the problem still persists, check your firewall settings.

If Elasticsearch doesn't start on your system, check the log files in the *logs* directory under your *<ELASTICSEARCH_INSTALL_DIR>* for configuration errors. If any configuration error is fixed

but Elasticsearch still refuses to start, the Elasticsearch documentation recommends that you deactivate the system call filters at your own risk. This is done by setting *bootstrap.system_call_filter* to false in the *elasticsearch.yml* config file.

Frequently Asked Questions

How can I share dashboards without Kibana administration features?

1. Click on **Dashboard** in the side navigation.
2. Open the dashboard you want to share.
3. Add *&embed=true* at the end of the address line of your Kibana dashboard URL.
4. Click on **Share**.
5. Copy the link you want to share. We recommend that you use use shortened snapshot URL.

How can I protect the Kibana index located in ElasticSearch?

1. Click on **Dev Tools** in the side navigation.
2. Execute the console command *GET _cat/indices* to get a list of existing indices.
3. Execute the console command *PUT /<yourKibanaIndexName>/_settings {"index.blocks.read_only": true}* to disable any modifications to your Kibana index.
4. Execute the console command *GET /<yourKibanaIndexName>/_settings* to check your current settings.

Note: In case of protection, a fatal error message will be returned

```
"Request to Elasticsearch failed: ...
                ... [FORBIDDEN/5/index read-only (api)];"
```

How can I backup and restore an individual ElasticSearch index?

1. Refer to the section *Elasticsearch Reference [5.1]* | *Modules* | *Snapshot And Restore* in the online documentaton **https://www.elastic.co/guide/en/elasticsearch/reference/5.1/modules-snapshots.html**.

How can I delete a daily index generated by receiving events from the Adabas Analytics Server?

1. Click on **Dev Tools** in the side navigation.
2. Execute the console command `GET _cat/indices/adabas_analytics-*` to get a list of existing daily Adabas Analytics indices.
3. Execute the console command `DELETE /adabas_analytics-<yourSelectedDailyIndex>`.
4. Execute the console command `GET _cat/indices/adabas_analytics-*` again to validate the result.

Note: Refer to the section *Elasticsearch Reference [5.1]* | *Indices APIs* | *Delete Index* in the online documentation.

How can I develop my own dashboards?

We recommend that you refer to Kibana's "Getting Started" Tutorial - see **https://www.elastic.co/guide/en/kibana/current/getting-started.html**

9 Adabas Extensions for Adabas Analytics

The current version of Adabas includes the following extensions, which enable you to work more easily with Adabas Analytics:

Extention	Purpose
ADAELP	Event log report. Used to print events from the Adabas Anlytics event log.
ADAELA	Event Analytics configuration tool. Used to help you set up the Adabas Analytics component.

This chapteralso includes a section that contains administration information.

ADAELP (Event Log Report)

This section describes the utility "ADAELP".

Functional Overview

The ADAELP utility prints events from an event log created by Adabas Analytics.

Note: Event logging must be enabled and the replication user exit must be loaded in order to write event logs. For further information see the section *Concepts*.

The ADAELP parameters USER_ID, HOSTNAME and EVENT_TIMESTAMP select a subset of the events in the event log.

In the interactive mode, ADAELP displays the selected events when the keyword LIST is entered. If ADAELP is called with parameters, the selected events are displayed immediately.

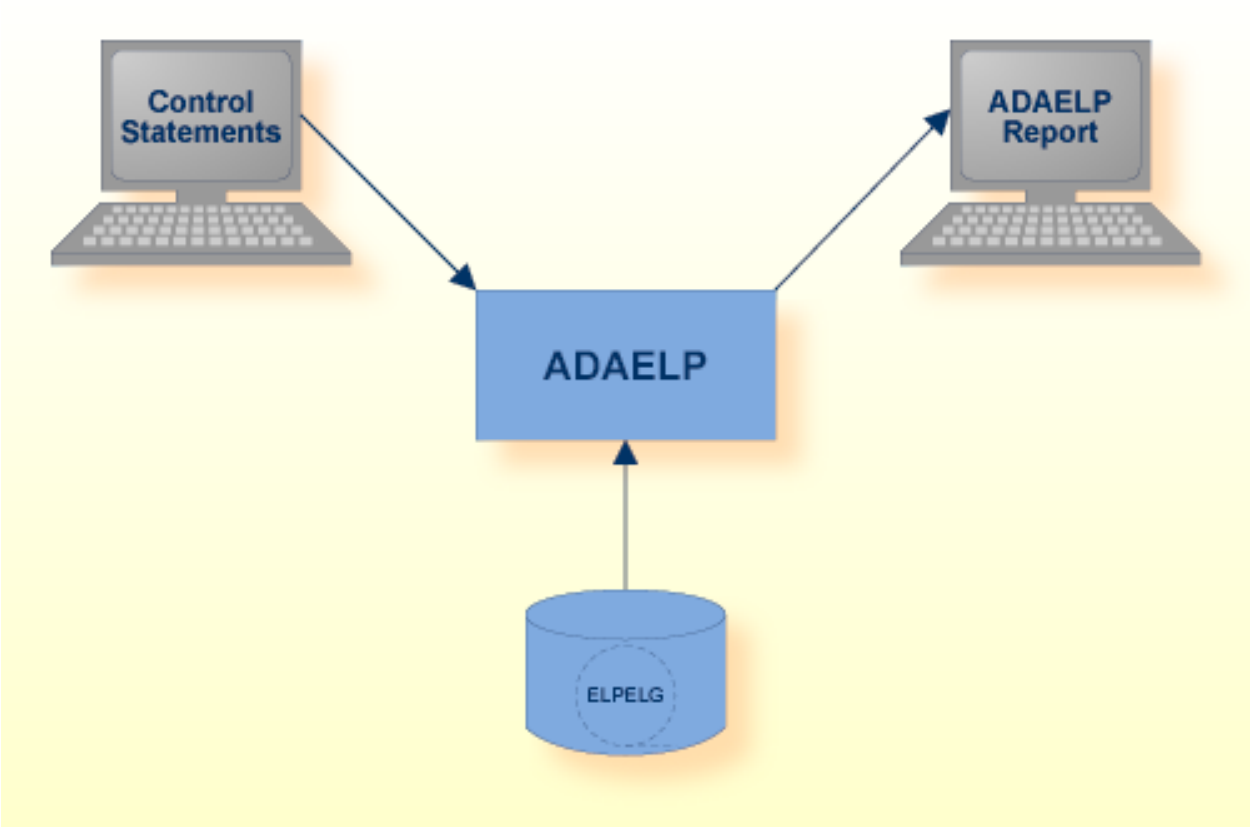
Events are displayed as follows: a first line with the event type is followed by lines that contain the field data of the event in question. The display of an event is concluded with the event type being repeated on the last line.

Example output

```
start read event
  event_timestamp=16-JUL-2015 11:51:01.977020
  dbid=163
  file_number=1
  command_code=L5
  response_code=0
  isn=993
  pid=6772
  hostname=PCST01
  user_id=st
  tsid=68 ba 56 02 fb 1a 05 00
end read event
```

This utility is a single-function utility.

Procedure Flow



Data Set	Environment Variable/ Logical Name	Storage Medium	Additional Information
Event log	ELPELG	Disk	
Control statements	stdin		see section Control Parameters
ADAELP report	stdout		

Checkpoints

The utility writes no checkpoints.

Control Parameters

The following control parameters are available:

```
D    DBID = number

    EVENT_TIMESTAMP = ([absolute-date][,[absolute-date]])

    HOSTNAME = string

    LIST

    USER_ID = string
```

DBID

```
DBID = number
```

This parameter specifies the database ID of the database for which the event log was written.

EVENT_TIMESTAMP

```
EVENT_TIMESTAMP = ([absolute-date][,[absolute-date]])
```

This parameter selects the log records in the range specified by the optional date strings. The date strings must correspond to the following absolute date and time format:

```
dd-mmm-yyyy[:hh:mm:ss[.mmmmmm]]
```

Leading zeroes in the date and time specification may be omitted. Any numbers not specified are set to 0, for example 28-jul-2015 is equivalent to 28-jul-2015:00:00:00.000000.

By default, all log records are selected.

Examples:

```
adaelp: event_timestamp = 8-aug-2015
```

The event with event_timestamp 8-AUG-2015 00:00:00 is selected.

```
adaelp: event_timestamp = (8-aug-2015:12,)
```

All events with time_stamp from 8-AUG-2015 12:00:00 onwards are selected.

```
adaelp: event_timestamp = (,8-aug-2012:12:34)
```

All events with time_stamp before 8-AUG-2015 12:34:00 are selected.

```
adaelp: event_timestamp = (16-JUL-2015 11:51:01.977020, 16-JUL-2015 11:51:02.177000)
```

All events with event_timestamp from 16-JUL-2015 11:51:01.977020 to 16-JUL-2015 11:51:02.177000 are selected.

HOSTNAME

```
HOSTNAME = string
```

This parameter selects all events with the hostname specified by 'string'. The length of the parameter value is limited to 8 characters.

LIST

```
LIST
```

This parameter lists the events selected with the parameters DBID, EVENT_TIMESTAMP, HOSTNAME and USER_ID.

USER_ID

```
USER_ID = string
```

This parameter selects all events with the user ID specified by 'string'. The length of the parameter value is limited to 8 characters.

Specifying Multiple Selection Criteria

If multiple selection criteria are specified, they are combined by a logical AND, e.g.

```
event_timestamp=(8-aug-2015:12:34,), user_id = guest, hostname = machine3
```

This selects all events after 8-aug-2015:12:34 with user_id = guest and hostname = machine3.

ADAELA (Event Analytics Administration)

This section describes the utility "ADAELA".

Functional Overview

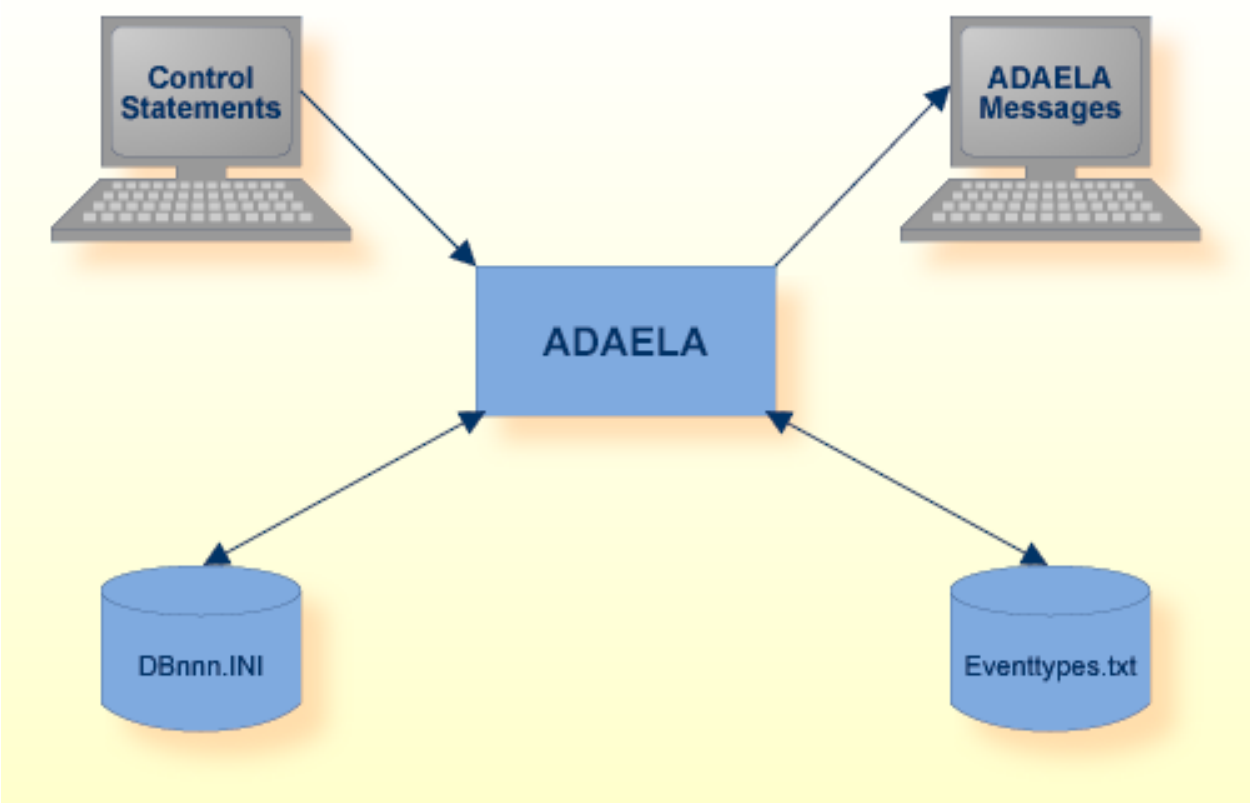
The administration utility ADAELA configures the Event Analytics add-on. It does not require the Adabas nucleus to be active, but if changes are made to the configuration, the nucleus must be restarted in order to make them active.

More than one event type can be added or removed during a single run of the utility. It is also possible to both add and remove event types during a single run. If more than one “add=event-type,…” and/or “remove=eventtype,…” statement is issued, each statement must be finished with the keyword `END_OF_EVENTTYPE`.

Note: If there is already a configuration present, it will be overwritten.

This utility is a multi-function utility.

Procedure Flow



Data Set	Environment Variable/ Logical Name	Storage Medium	Additional Information
Database configuration		DBnnn.ini	
Event types store		eventtypes.txt	
Control statements	stdin/SYS$INPUT		see section Control Parameters
ADAELA messages	stdout/SYS$OUTPUT		Messages and Codes

Checkpoints

The utility writes no checkpoints.

Control Parameters

The following control parameters are available:

```
ADD = SERVER | NUCELG | EVENTTYPE | FILTER
           [ ,server_keywords |
             ,nucelg_keywords |
             ,eventtype_keywords |
             ,filter_keywords]

M    DBID = number

    DISABLE

    DISPLAY = CONFIGURATION | EVENTTYPES

    ENABLE

    REMOVE = SERVER | NUCELG | EVENTTYPE, NAME=string | FILTER
```

ADD

ADD=SERVER

```
ADD = SERVER
D            [,HOST=string]
D            [,PORT=number]
D            [,RECONNECT_TIMEOUT=number]
D            [,RETRY=number]
D            [,ON_ERROR=keyword]
```

This parameter adds the Analytics Server configuration to the DBnnn.INI file. It is not possible to use the Analytics Server in parallel to writing events into a file (see NUCELG). If there is already a configuration for the NUCELG file present, it will be deleted.

HOST=string

The hostname where the Analytics Server is running. The default is *localhost*, which assumes that the server will run on the same host as the Adabas nucleus for the best performance.

PORT=number

The TCP/IP listen port number of the Analytics Server. The default value is 6521. The port number must match the port number configured in the Analytics Server.

RECONNECT_TIMEOUT=number

If the connection between the Adabas nucleus and the Analytics Server is lost, this parameter specifies the time (in seconds) that the Adabas nucleus waits between reconnect attempts to the Analytics Server. The default is 1.

Note: Reconnect attempts are time consuming and have a negative impact on the Adabas performance.

RETRY=number

If the connection between the Adabas nucleus and the Analytics Server is lost, this parameter specifies the number of reconnect attempts. The default is 0, which means "try continuously".

Note: If the maximum number of attempts is not 0, and if the maximum number of reconnect attempts is reached without reconnecting successfully, event logging will be deactivated.

ON_ERROR=keyword

This parameter defines the behavior of the Adabas nucleus in case an error occurs with the Analytics Server. The default is *IGNORE*. 'keyword' can take one of the following values:

Keyword	Meaning
ABORT	The Adabas nucleus will abort on any error condition with the Analytics Server.
IGNORE	The Adabas nucleus will attempt to reconnect to the Analytics Server.

ADD=NUCELG

```
ADD = NUCELG
D             [,TIME_SWITCH=number]
D             [,EVENT_SWITCH=number]
D             [,ELGFILE=path]
```

This parameter adds the configuration for writing events to a file. It is not possible to use the Analytics Server in parallel to writing events into a file. If there is already a configuration for the Analytics Server present, it will be deleted.

TIME_SWITCH=number

The time (in seconds) that elapses before starting a new log file. The default value is 0, which means that the log file is never switched after a given time.

EVENT_SWITCH=number

The number of events that have to occur before starting a new log file. The default value is 0, which means that the log file is never switched after a given number of events.

ELGFILE=path

The fully-qualified path name for the log file. The default is *$ADADATADIR/dbnnn/NUCELG* or *%ADADATADIR%\dbnnn\NUCELG*. A numbering suffix will be added automatically.

ADD=EVENTTYPE

```
ADD = EVENTTYPE ,NAME=string
                ,AREA=keyword
                ,FIELDS=(keyword [,keyword] …)
D              [,FILE=path]
D              [,END_OF_EVENTTYPE]
```

This parameter adds an event-type definition to the event types file.

NAME=string

The name of the event type.

FILE=PATH

The fully-qualified path name to the file which stores the event types.

AREA=keyword

The area where the event will be created. keyword can take one of the following values:

Keyword	Meaning
POST_COMMAND	The event is based on the Adabas Control Block. After a regular Adabas command has been processed, an event that depends on the command code will be sent. The area defines also which fields are available for the event.

FIELDS = (keyword [,keyword] …)

A list of the fields for the event type. The following fields are available:

Keyword	Area	Type, Length	Meaning
event_timestamp	POST_COMMAND	Integer, 8	Timestamp when the event was generated
Dbid	POST_COMMAND	Integer, 4	Database ID
file_number	POST_COMMAND	Integer, 4	Adabas file number
command_code	POST_COMMAND	String, 2	Adabas command code
response_code	POST_COMMAND	Integer,4	Nucleus response code
Pid	POST_COMMAND	Integer, 8	Process ID
Isn	POST_COMMAND	Integer, 8	ISN

Keyword	Area	Type, Length	Meaning
hostname	POST_COMMAND	String, 8	Host name
user_id	POST_COMMAND	String, 8	User ID
Tsid	POST_COMMAND	Binary, 8	Unique identifier
natapplication	POST_COMMAND	String, 8	Natural application name
natprogram	POST_COMMAND	String, 8	Natural program name
natlevel	POST_COMMAND	Integer, 4	Natural call level
natcount	POST_COMMAND	Integer, 8	Number of Adabas calls since last IO
natexec	POST_COMMAND	Integer, 8	Number of times a Natural object has been executed
natuser	POST_COMMAND	String, 8	Natural user ID
natstatement	POST_COMMAND	String, 4	Natural statement number
Natlib	POST_COMMAND	String, 8	Natural library name
natrpcclientuid	POST_COMMAND	String, 8	Natural RPC client user ID
natrpcid	POST_COMMAND	String, 16	Natural RPC ID
natrpcconvid	POST_COMMAND	String, 16	Natural RPC conversation ID
natsecgroup	POST_COMMAND	String, 8	Natural security group
additions1	POST_COMMAND	Binary, 8	Additions 1 field
duration	POST_COMMAND	Integer, 4	Command duration in microseconds
command_opt1	POST_COMMAND	Character, 2	Command option 1 field
command_opt2	POST_COMMAND	Character, 2	Command option 2 field

END_OF_EVENTTYPE

The keyword END_OF_EVENTTYPE must be specified if more than one event type is added, or if an event type is added and one or more event types are removed.

DBID

```
DBID = number
```

This parameter selects the database to be used.

DISABLE

```
DISABLE
```

This parameter disables the event logging feature.

DISPLAY

```
DISPLAY = CONFIGURATION | EVENTTYPES
```

This parameter display information about the event logging. The following keywords can be specified:

Keyword	Meaning
CONFIGURATION	Displays the configuration of the event logging. The information is taken from the *DBnnn.INI* file.
EVENTTYPES	Displays all configured event types.

ENABLE

```
ENABLE
```

This parameter enables the event logging feature.

REMOVE

```
REMOVE = SERVER | NUCELG | EVENTTYPE, NAME=string | FILTER
```

REMOVE = SERVER

This parameter removes the server configuration. The section *TARGET_SERVER* will be removed from the *DBnnn.INI* file.

REMOVE = NUCELG

This parameter removes the NUCELG configuration.

REMOVE = EVENTTYPE, NAME=string [,END_OF_EVENTTYPE]

This parameter removes an event type with the specified name from the event types file.

The keyword END_OF_EVENTTYPE must be specified if more than one event type is removed, or if an event type is removed and one or more event types are added.

Adminstration of Adabas Extensions for Adabas Analytics

The configuration of Adabas Analytics is stored in two files: the *DBnnn.INI* file and the *eventtypes.txt* file; both are located in the database directory (*$ADADATADIR/dbnnn*). Before you start to configure, ensure that the database and the DBnnn.INI file already exist.

Parameter Reference

Adabas Analytics uses a new topic [EVENT_ANALYTICS] in the configuration file DBnnn.INI. This section describes the keywords and subtopics contained in the [EVENT_ANALYTICS] topic. The corresponding command for the administration utility ADAELA is shown in square brackets below each parameter description.

Topic EVENT_ANALYTICS

This topic stores the configuration of Adabas Analytics, with the exception of the *NUCELG* parameter, which is stored under the topic *CONTAINERS*.

ACTION = YES/NO

This keyword activates/deactivates Adabas Analytics. If this keyword is omitted, Adabas Analytics is switched off. Valid keywords are YES and NO.

[adaela dbid=nnn enable|disable]

Subtopic TARGET_NUCELG

This target describes the configuration when writing the events to a file.

[adaela dbid=nnn add=nucelg]

SWITCH_AFTER_EVENTS = <count>

This keyword is used to switch to a new *NUCELG* file after `count` events have been written to the *NUCELG* file. The new file will be created with an increased suffix number, e.g. NUCELG.0002, NUCELG.0003... If this keyword is omitted, all events are logged to a single *NUCELG* file.

[adaela dbid=nnn add=nucelg, event_switch=number]

SWITCH_AFTER_TIME = <time in seconds>

This keyword is used to switch to a new *NUCELG* file after *time in seconds* has passed and a new event is to be generated. The new file will be created with an increased suffix number, e.g. NUCELG.0002, NUCELG.0003... If this keyword is omitted, all events are logged to a single *NUCELG* file.

[adaela dbid=nnn add=nucelg, time_switch=number]

NUCELG=path

The NUCELG parameter is the path name to the file for the events written by the nucleus. The default is *$ADADATADIR/dbnnn/NUCELG*. A numbering suffix will be added automatically, e.g. NUCELG.0001, NUCELG.0002...

[adaela dbid=nnn add=nucelg, elgfile=path]

Subtopic FILTER

The subtopic FILTER has two keywords: FILES and EVENT_TYPES. These keywords are used to determine which types of events are generated for which Adabas files.

[adaela dbid=nnn add=filter]

FILES = (list)

This keyword specifies the Adabas files for which events are generated. You can specify a list of files; the list can contain single files (<file1>, <file2>) or a range of files (<file3> - <file4>). If this keyword is omitted, events are generated for all Adabas files in the database.

[adaela dbid=nnn add=filter, file_filter=(n-m)]

EVENT_TYPES = (list)

This keyword specifies the types of events that are generated. The names and details of each event type are described in the section *Adabas Analytics Event Types*. If this keyword is omitted, all event types are generated for the files specified by the FILES keyword.

[adaela dbid=nnn add=filter, event_filter=(a-z)]

Subtopic TARGET_EAL_SERVER

The subtopic TARGET_EAL_SERVER contains the configuration parameters necessary to send the events to the Adabas Analytics Server.

[adaela dbid=nnn add=server]

HOST = <host name>

The host name or IP address where the Analytics Server is running. The default is *localhost*.

[adaela dbid=nnn add=server, host=name]

PORT = <number>

The port number on which the Analytics Server is listening.

[adaela dbid=nnn add=server,port=nnn]

MAX_RETRIES = <number>

If the connection between the Adabas nucleus and the Analytics Server is lost, this parameter specifies the number of reconnect attempts. The default is 0, which means 'try continuously'. A reconnect attempt is only made if there is an Adabas command active at this time.

Important: If the maximum number of attempts is not 0, and if the maximum number of reconnect attempts is reached without reconnecting successfully, the event logging will be deactivated.

[adaela dbid=nnn add=server, retry=nnn]

ON_ERROR = ABORT/IGNORE

Select ABORT if you want to abort the Adabas nucleus if a connection error between the Adabas nucleus and the Analytics Server occurs. Select IGNORE if you want to continue with the Adabas nucleus if a connection error between the Adabas nucleus and the Analytics Server; in this case, an automatic reconnect mechanism is enabled (see also MAX_RETRIES and RECONNECT_TIMEOUT).

[adaela dbid=nnn add=server, on_error=keyword]

RECONNECT_TIMEOUT = <seconds>

If the connection between the Adabas nucleus and the Analytics Server is lost, this parameter specifies the number of seconds the Adabas nucleus waits between retrying to connect to the Analytics Server. The default is 1. A reconnect attempt is only made if there is an Adabas command active at this time.

[adaela dbid=nnn add=server, reconnect_timeout=nnn]

Adabas Event Types Reference

The following event types are pre-defined and cannot be changed:

Event Type	Area	Field	Type, Length	Description
read	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
update	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
insert	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
delete	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number

Event Type	Area	Field	Type, Length	Description
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
commit	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
rollback	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
nat_read	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID

Event Type	Area	Field	Type, Length	Description
		tsid	Binary, 8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String,8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number
		natlib*	String, 8	The Natural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group
nat_update	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String, 8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number

Event Type	Area	Field	Type, Length	Description
		natlib*	String, 8	The Natural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group
nat_insert	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 3	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String, 8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number
		natlib*	String, 8	The Natural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group
nat_delete	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID

Event Type	Area	Field	Type, Length	Description
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 3	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String, 8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number
		natlib*	String, 8	The Natural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group
nat_commit	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 3	Client user ID
		tsid	Binary,8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String, 8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)

Event Type	Area	Field	Type, Length	Description
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number
		natlib*	String, 8	The Natural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group
nat_rollback	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID
		file_number	Integer, 4	Adabas file number
		command_code	String, 3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		hostname	String, 8	Client hostname
		user_id	String, 3	Client user ID
		tsid	Binary,8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String, 8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number
		natlib*	String, 8	The Natural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group
ada_perf	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 4	Database ID

Event Type	Area	Field	Type, Length	Description
		file_number	Integer, 4	Adabas file number
		command_code	Sting,3	Adabas command code
		response_code	Integer, 4	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		additions1	Binary, 8	Contains the Additions 1 field of the issued call
		duration	Integer, 4	Contains the time, in microseconds, that Adabas took to process the call
		copt1	Character, 2	Contains the value of the Command Options 1 field of the processed call
		copt2	Character, 2	Contains the value of the Command Options 2 field of the processed call
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
nat_perf	POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
		dbid	Integer, 8	Database ID
		file_number	Integer, 8	Adabas file number
		command_code	String,3	Adabas command code
		response_code	Integer, 8	Adabas response code
		pid	Integer, 8	Client process ID
		isn	Integer, 8	Adabas ISN
		additions1	Binary, 8	Contains the Additions 1 field of the issued call
		duration	Integer, 4	Contains the time, in microseconds, that Adabas took to process the call
		copt1	Character, 2	Contains the value of the Command Options 1 field of the processed call
		copt2	Character, 2	Contains the value of the Command Options 2 field of the processed call
		hostname	String, 8	Client hostname
		user_id	String, 8	Client user ID
		tsid	Binary, 8	Unique marker of the Adabas client
		natapplication*	String, 8	Name of the Natural application issuing the call
		natprogram*	String, 8	Name of the Natural program issuing the call
		natlevel*	Integer, 4	Number of Adabas calls since the last IO

Event Type	Area	Field	Type, Length	Description
		natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
		natexec*	Integer, 8	Natural call level of the program executed
		natuser*	String, 8	The Natural user issuing the call
		natstatement*	String, 4	Natural statement number
		natlib*	String, 8	The Nautural library issuing the call
		natrpcclientuid*	String, 8	Natural RPC client user ID
		natrpcid*	String, 16	Natural RPC ID
		natrpcconvid*	String, 16	Natural RPC conversation ID
		natsecgroup*	String, 8	Natural security group

Notes:

1. Fields marked with an asterisk (*) are only filled when using Natural version 8.4.1 or later, with the parameter `ADAPRM` set to *`ON`*.
2. The length for all fields of type string additionally have the string termination byte.

User-Defined Event Types

User-defined event types are stored in the file *eventtypes.txt*, which is located in the database directory. The event type consists of:

- A unique event type name
- The area in the Adabas nucleus where the event is triggered
- A list of fields. The available fields are dependent on the area; e.g. the POST_COMMAND area defines most of the fields of the Adabas control block.

You can define an event ytpe with the utility ADAELA:

```
adaela: dbid=nnn, add=event_type, name=<name>, area=<keyword>, fields=(field1, ↩
field2, ...) [file=<path>]
```

Parameter	Type, Length	Description
name	String, 255	Name of the event type
area	Keyword	Area in the nucleus where the event will be triggered. See *Event Types Area Reference* below for a list of available areas.
fields	List of keywords	List of field names. See *Event Types Fields Reference* below for a list of available fields for the appropriate area.

Event Types Area Reference

The following table shows the available areas:

Area Name	Description
POST_COMMAND	The event will be triggered after an Adabas command has been fully processed, and before the results are posted to the user.

Event Types Field Reference

The following table shows the fields available for each area:

Area	Field Name	Type, Length	Description
POST_COMMAND	event_timestamp	Integer, 8	Creation time of this event
	dbid	Integer, 4	Database ID
	file_number	Integer, 4	Adabas file number
	command_id	String, 2	Adabas command code
	response_code	Integer, 4	adabas response code
	pid	Integer, 8	Client process ID
	isn	integer, 8	Adabas ISN
	hostname	String, 8	Client hostname
	user_id	String, 8	Client user ID
	tsid	Binary, 8	Unique marker of the Adabas client
	natapplication*	String, 8	Name of the Natural application issuing the call
	natprogram*	String, 8	Name of the Natural program issuing the call
	natlevel*	Integer, 4	Number of Adabas calls since last IO
	natcount*	Integer, 8	Number of times a Natural object has been executed (internal)
	natexec*	Integer, 8	Natural call level of the program executed
	natuser*	String, 8	The Natural user issuing the call
	natstatement*	String, 4	Natural statement number
	natlib*	String, 8	The Natural library issuing the call
	natrpcclientuid*	String, 8	Natural RPC client ID
	natrpcid*	String, 16	Natural RPC ID
	natrpcconvid*	String, 16	Natural RPC conversation ID
	natsecgroup*	String, 8	Natural security group
	additions1	Binary, 8	Contains the Additions 1 field of the issued call
	duration	Integer, 4	Contains the time, in microseconds, that Adabas took to process the call

Area	Field Name	Type, Length	Description
	command_opt1	Character, 2	Contains the value of Command Option 1 of the processed call
	command_opt2	Character, 2	Contains the value of Command Option 2 of the processed call

Notes:

1. Fields marked with an asterisk (*) are only filled when using Natural version 8.4.1 or later, with the parameter `ADAPRM` set to *ON*.
2. The length for all fields of type string additionally have the string termination byte.

Event Types Filter

The filter mechanism gives you control over the events that are generated. You can filter the events by the following criteria:

- Adabas file number
- Event type

You can filter by Adabas file number only, by event type only, or you can combine both filters.

Important: It is not currently possible to filter the user-defined event types.

Adabas files can be specified as a comma-separated list of single file numbers, or ranges of file numbers, or a combination of both. The event types can be specified as a comma-separated list of single event types. For a list of event types, see ***Adabas Event Types Reference***.