SAP Event Stream Processor: An Overview of Integration Options

Updated for SAP ESP 5.1 SP08
# TABLE OF CONTENTS

**EXECUTIVE SUMMARY** ................................................................................................................................... 3
**OVERVIEW OF KEY CONCEPTS** .................................................................................................................... 3
Cluster, Workspace, Server and Project ................................................................................................................ 4
The ESP Studio .................................................................................................................................................. 4
ESP Projects: Continuous Queries, Streams, Windows ..................................................................................... 4
Events and Event Streams ............................................................................................................................... 4

**ADAPTERS** ....................................................................................................................................................... 6
**STANDARD ESP ADAPTERS** .......................................................................................................................... 6
Standard Database Adapters ............................................................................................................................ 7
Specialized Adapters ...................................................................................................................................... 7

**THE ESP WEB SERVICE PROVIDER** ............................................................................................................. 8
**M2M, SCADA AND SAP PLANT CONNECTIVITY** .......................................................................................... 8
**OTHER PRODUCTS THAT SUPPORT ESP** .................................................................................................... 8

**CUSTOM ADAPTERS: THE ESP ADAPTER TOOLKIT** ..................................................................................... 9
**THE ESP SOFTWARE DEVELOPERS KIT (SDK)** .......................................................................................... 9
**CUSTOM INTERNAL ADAPTERS: THE ESP SHARED UTILITY LIBRARY** .................................................... 9

**THE ESP ADAPTER INTEGRATION FRAMEWORK** .......................................................................................... 10
**OTHER INTEGRATION OPTIONS** ................................................................................................................ 10
The CCL REFERENCE element ....................................................................................................................... 10
The getData() function ................................................................................................................................. 10
ESP Utilities ................................................................................................................................................. 10

**CONCLUSION** ................................................................................................................................................. 10

**FOR MORE INFORMATION** ...................................................................................................................... 10
This technical white paper provides an overview of the options available to integrate the SAP Event Stream Processor (ESP) with data sources and data consumers.

This document applies to SAP ESP 5.1 SP08 and above.

EXECUTIVE SUMMARY

SAP Event Stream Processor is a stream processing engine for analyzing and acting upon event streams in real-time. SAP Event Stream Processor (ESP) provides a number of different tools and interfaces to make it easy to integrate ESP with a wide variety of systems and applications that produce data, consume data, or both. This paper provides an introduction to some of those integration mechanisms.

One of the powerful aspects of ESP derives from the decoupled nature of event driven software: integration of ESP into an existing environment is often non-intrusive. ESP can be installed as an overlay to existing systems without needing to replace or re-engineer those systems.

Because ESP is used in so many different environments, with a need to integrate with many different types of data producers and data consumers, there is not just a single integration method but rather a range of options. The overriding goal of providing this range of options is to offer tools that are both versatile and simple. To this end, ESP integration options consist of:

- A set of standard adapters included "out-of-the-box"
- Optional specialized adapters available for purchase
- An adapter toolkit that permits assembly of pre-existing out-of-the-box modules, as well as easy implementation of custom modules, into fit-for-purpose adapters
- A “software developers kit” (SDK) composed of libraries that can be directly integrated with other applications to give them native ESP connectivity
- A web service provider interface that lets applications publish data to ESP via web services

OVERVIEW OF KEY CONCEPTS

SAP ESP is a distributed system designed to be very scalable and flexible. A single ESP system can receive data from any number of sources, regardless of whether the data is streaming or static, can process that data through any number of different “projects”, and continuously push results to any number of downstream systems. The following diagram provides a high level overview of the various inter-connected components that form an ESP system:
Cluster, Workspace, Server and Project

An ESP Project is the executable unit of event processing logic that runs in a Workspace in an ESP Cluster.

An ESP cluster is a virtual ESP server. It's called a cluster because, while it can run on a single physical server, it can scale across any number of physical servers. Each cluster has at least one Cluster Manager, and will typically have two or more cluster managers running to eliminate any single point of failure. The cluster managers assign projects to nodes in the cluster, resolve logical names to physical locations, and provide high availability and recovery services for the cluster.

Each ESP cluster can run any number of projects simultaneously, subject to the hardware capacity of the cluster. This holds regardless of how many physical machines are in the cluster. Thus a Cluster could be composed of a single Project on a single physical server, could have multiple projects running on a single server, or could consist of multiple projects running on multiple servers.

Each ESP project runs in a single Workspace on the cluster. The workspace simply provides a protected namespace on the cluster. Within a cluster, the combination of a Workspace name and Project name is unique; however the Project name may be used within another Workspace in the same Cluster.

The ESP Studio

The ESP Studio is an Eclipse® based integrated development environment for developing ESP Projects. It can run as a plug-in to the SAP HANA Studio or as a stand-alone application. The studio is a design-time tool and not a run-time tool. Once projects are built using the ESP Studio and deployed to an ESP Cluster, the Studio does not need to be running – it has no role to play in a run-time environment.

The ESP Cockpit

The ESP Cockpit is a browser-based tool for monitoring and managing an ESP Cluster. This provides an operations console that can be used to manage a production system. From the cockpit, and administrator can start and stop projects and adapter, change the cluster configuration, and monitor a variety of live operational statistics pertaining to all running components.

ESP Projects: Continuous Queries, Streams, Windows

An ESP project contains the definition of one or more input Streams or Windows and (typically) one or more output Streams or Windows. The project also contains the Continuous Queries that will transform the inputs into the desired outputs. Input streams and windows in an ESP project receive incoming events from external data sources. Output streams/windows publish events to downstream systems whenever there is new information, based on the continuous query logic defined in the project.

Events and Event Streams

ESP is designed to process Events in real-time as fast as they arrive. Each incoming event is a message that contains a set of typed fields and, optionally, an operation code (opCode).

The schema for a simple event stream that reports temperature readings from a sensor might look like this:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SensorID</td>
<td>string</td>
</tr>
<tr>
<td>Temperature</td>
<td>float</td>
</tr>
</tbody>
</table>

Or a schema for an event stream that receives information on security trades could look like this:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol</td>
<td>string</td>
</tr>
<tr>
<td>BuyOrSell</td>
<td>string</td>
</tr>
<tr>
<td>Price</td>
<td>money(2)</td>
</tr>
<tr>
<td>Shares</td>
<td>integer</td>
</tr>
</tbody>
</table>

Each input stream or window has a fixed schema and all events arriving on that stream (or published by that stream) must conform to the schema for that stream. Thus, incoming events with different schema will be received on different input streams.
Note that the incoming events don’t have to contain all the fields defined in the schema; missing fields will just be null. Thus it is possible to receive events containing different sets of fields on the same input stream as long as the schema of the input stream represents a superset of all fields that may be contained in the incoming events.

Events can optionally include an opCode that indicates if the event is an insert, update, upsert or delete. This is useful when the incoming events represent changes to state, like database operations where data is applied to tables in a database. When events have opCodes, those events can be directly applied to an input Window in an ESP project to update the contents of that window. When events do not have an opCode, they are treated as insert events. Note that input streams in an ESP project can only receive insert events, since they are stateless; data sources that produce update and/or delete events, in addition to inserts, should be connected to an ESP input window.

Notes:

- An upsert first tries to apply the event as an update to a window, but if the key value is not found in the window then it is applied as an insert to the window

- It says above that input streams can only receive insert events. Note, however, that the addition of support for keyed streams in SP08 changes this rule. Keyed streams will be able to receive events with any opcode and the opcode will be preserved as it passes through the input stream. As the input stream is stateless, however, the opcode will be ignored by the stream itself and will just be passed through unaltered.
ADAPTERS

Adapters are the most common way of integrating ESP with data sources and data destinations. An adapter connects to a source or destination using an interface provided by that system, connects to an ESP project, and converts between the data format of the external system and the ESP event format.

A single adapter instance may both publish and subscribe to a project. For example, an input adapter may subscribe to an ESP project to know what types of events are of interest, and then publish events of those types to the same project as they are received.

Adapters use the ESP SDK to connect to an ESP project and publish data to the project and/or receive data from the project. The SDK connects to an ESP project, running on an ESP cluster, via a socket. Thus adapters can run on the same machine as an ESP project or on a different machine. In fact an adapter can even be remote, connecting to an ESP project over a WAN, VPN or the Internet.

Adapters under ESP project control or cluster control

Adapters can operate under the control of an ESP project or can operate independently of an ESP Project.

If an ESP Project includes one or more ATTACH ADAPTER statements, when the project starts, it starts an adapter instance for each adapter referenced. When the project stops, it shuts down all the adapters that it started.

Alternatively, an adapter can be started independently of projects, either via scripts or command line control. In this case the lifecycle of the adapter is completely independent of any ESP project.

A new feature introduced in SP08 introduces another option: adapters that are managed by the cluster, but independently of a project. This only applies to adapters built from the ESP Adapter Toolkit, but this allows adapters to be started, stopped and monitored directly from the ESP Cockpit and/or cluster admin tools.

STANDARD ESP ADAPTERS

The ESP product ships with the most common adapters included as standard. These simply need to be configured to connect to the relevant source or destination to be used with ESP Projects. We regularly add new adapters, so please consult your ESP representative or check the ESP Developer Center if an adapter that you need is not listed here. The current set of standard adapters includes (but is not limited to) the following:

- **Message buses**: connect to a message bus to receive incoming events and/or publish events. Supported message buses include JMS, TIBCO®, and IBM Websphere MQ®
- **Files**: read input events from a file; write output events to a file. With CSV, XML, JSON and FIX parsing/formatting.
- **Hadoop**: the file input and output adapters can be configured to read from or write to the Hadoop File System (hdfs).
- **Socket**: receive events on a socket; write events to a socket. With CSV, XML, JSON and FIX parsing/formatting.
• **Database change capture**: Sybase Replication Server monitors database transaction logs, sending transactions along to ESP via the adapter for Replication Server. This adapter turns the transactions into real-time input streams for ESP.

• **Web Service Provider**: while technically not implemented as an adapter, the ESP Web Service Provider is a web service that exposes ESP input streams/windows as either REST or SOAP services, allowing clients to post events to ESP projects. See Web Service Provider section below for more information. As of SP08, the ESP Web Service Provider also supports WebSockets.

• **Web Service Client adapter**: on the input side, the ESP adapter will request data from a web service at a regular interval, parse the results and stream them into an ESP project. On the output side, the adapter will format events published by an ESP project and post them to a web service in real-time.

• **SAP RFC**: as an input adapter, calls a pre-configured RFC in the designated SAP System regularly at a fixed interval to retrieve data, parses the data, and streams it into ESP. As an output adapter, it calls an SFC to deliver data in real-time, whenever data is published by ESP.

• **Email (SMTP) output**: ESP output events can be delivered as email

• **Microsoft Excel**: more than just an adapter, this plug-in for Excel supports both real-time subscriptions of ESP data into Excel as well as publishing from Excel to ESP

• **Atom Reader**: polls a pre-configured URL for an ATOM data source to retrieve events

• **FTP**: both input and output

• **Log File** input: reads Apache, Tomcat and IIS log files; each line becomes an event

**Standard Database Adapters**

ESP also includes database adapters that use ODBC, JDBC as well as native APIs for connectivity to external databases. The ESP database adapters include support for:

• SAP HANA
• SAP Sybase IQ
• SAP Sybase Adaptive Server® Enterprise (ASE)
• SAP Sybase SQL Anywhere®
• Microsoft SQL Server
• IBM DB2
• Oracle
• Oracle TimesTen
• MySQL 5.x
• PostgreSQL

The database adapter features include the ability to pull data from a database into ESP as well as to deliver the output from ESP to a database. On the input side, there are several options for how data is retrieved from the database:

- data can be pulled from the database once at startup, e.g. for startup or seed data;
- the database can be polled at a pre-defined interval, e.g. for reference data;
- individual event-driven lookups can be performed against the database, e.g. to enrich events;
- Event history, previously-persisted to a database, can be replayed during development to help identify new high-value analytics or alerts, and for testing purposes.

Specialized high speed output adapters for **SAP HANA** and **SAP Sybase IQ** allow data to be captured at extreme message rates.

**Specialized Adapters**

Specialized adapters are available from SAP as separately licensed add-ons, including adapters for Financial Market data (NYSE Technologies), and the F.I.X. (Financial Information eXchange) protocol.
THE ESP WEB SERVICE PROVIDER

ESP includes a Web Service Provider that can be used on the input side to receive incoming events from applications that are web service enabled, posting those events to ESP projects in real-time. Any ESP Project can be configured to accept events via the ESP Web Service Provider. REST, SOAP and WebSockets are supported.

The ESP Web Service Provider also provides the ability to subscribe to streaming ESP output via WebSockets.

M2M, SCADA AND SAP PLANT CONNECTIVITY

SAP Plant Connectivity is a software component that enables the exchange of data between an SAP system and various Supervisory Control and Data Acquisition (SCADA) systems. PCo supports connectivity to a variety of data sources using industrial automation standard interfaces such as OPC Data Access, OPC Historical Data Access, OPC Alarms & Events and OPC Unified Architecture. PCo also supports integration with software services called historians that accumulate time series of events and alarms from industrial control systems, such as GE Proficy Historian, OSIsoft PI, AspenTech IP21 and Citect. Additionally, PCo also provides integration with plant systems via databases, files and sockets.

With SAP Plant Connectivity (PCo), tags and events from connected source systems can be pushed automatically to connected SAP systems, including ESP. The automatic pushing of tags and events is considered a Notification process in PCo; PCo also supports a Query process that allows software like SAP MII to query PCo for specific tags and events.

ESP and PCo interoperability is based around the Notification process. PCo supports ESP as a destination for notifications. PCo agents are configured to monitor events and tags having specific identifiers from source systems, and then the PCo agents deliver the events and tags into ESP streams and windows. ESP can be used to correlate multiple events and tags including cross-stream correlation, perform time-based analysis and pattern matching on event and tag data, and generate alerts.

Alerts identified by ESP can be delivered back to SAP Manufacturing infrastructure using SAP MII (via HTTP Post, JMS, or web services). ESP can also be used to persist event and tag notifications, as well as alerts derived from these events, into SAP HANA.

For more information on SAP PCo, please see help.sap.com:

For more information on SAP MII, please see:
https://help.sap.com/mii

OTHER PRODUCTS THAT SUPPORT ESP

There are a number of other products from SAP and from SAP partners that have support for ESP. SAP products that support ESP include:

- SAP NetWeaver Process Orchestration: exchange events between ESP and Process Integration (PI) via Web Services, using the ESP web service client adapter and/or ESP Web Service Provider
- SAP Plant Connectivity version 2.3 SP04, part of the SAP Manufacturing Integration & Intelligence (MII) product line, now has ESP support (see previous section)
- SAP MII 14.0 can publish into ESP streams or windows using a set of custom MII actions that leverage ESP’s Java SDK and are available on the MII SCN. ESP can send events (alerts, notifications) to MII’s message listeners using the HTTP Post Output Adapter and XML-formatted messages.
- SAP Business Suite and BW: the ESP SAP RFC adapter provides connectivity
CUSTOM ADAPTERS: THE ESP ADAPTER TOOLKIT

The ESP Adapter Toolkit (released in ESP 5.1 SP03) makes it easier and faster than ever before to build custom adapters. The adapter toolkit uses out-of-the-box connector modules to connect to ESP projects, but provides an extensible framework that links “transport” modules and “formatter/parser” modules in any combination along with built in ESP connectors to form a complete adapter.

You define how connector, transporter and formatter modules are to be used together to comprise an adapter. Thus, rather than writing a custom adapter from scratch, all you need to do to add support for a new transport protocol, or support for a new data format, is to write a simple module in Java -- the toolkit takes care of the rest.

THE ESP SOFTWARE DEVELOPERS KIT (SDK)

The ESP Software Developers Kit (SDK) provides an API that can be used to build adapters or can be embedded directly in an application that will connect to ESP. The ESP SDK is available for Java, C/C++, and .NET C#. The ESP SDK provides a simple way of:

- connecting to an ESP Cluster
- building an event with a schema corresponding to the Stream/Window that will receive it
- publishing an event to a specific input Stream or Window in a specific Project
- subscribing to events from an output Stream or Window in a Project
- parsing the event received from an output Stream/Window

For building custom adapters, we recommend using the ESP Adapter Toolkit, rather than using the ESP SDK directly. It’s much simpler to build a new plug-in module for the adapter toolkit, than it is to write a complete adapter using the SDK.

There are some use cases, however, to use the SDK directly. The most notable is if you must use C++ or .NET to connect to ESP, since the adapter toolkit is currently only available for Java. Another is if you wish to connect to ESP directly from within another external process, e.g. from within another application.

CUSTOM INTERNAL ADAPTERS: THE ESP SHARED UTILITY LIBRARY

It is possible to implement custom internal adapters using the ESP Adapter Shared Utility Library. This library exposes APIs with a C interface that allows you to implement life cycle and information management functions for your custom adapter. The C interface lets you implement your custom adapter in either C or C++ without compiler restrictions. Note, however, that with the release of the ESP Adapter Toolkit, this interface is now being deprecated and we encourage all new adapters to be built as external adapters using either the Adapter Toolkit or the SDK.
THE ESP ADAPTER INTEGRATION FRAMEWORK

Custom adapters can be fully integrated with the ESP Studio and with CCL to provide the same user experience as the built-in adapters. This is true for adapters built with the Adapter Toolkit as well as adapters that use the SDK directly. An adapter configuration file called a .cnxml file must be supplied for the custom adapter to appear within the Studio Adapter Palette and to be referenced in a CCL ATTACH ADAPTER statement in an ESP project. If the adapter supports schema discovery, external schema can be imported through the adapter using the visual editor in the ESP Studio. See the section “Adapter Integration Framework” in the “Building Custom Adapters” guide in the ESP product documentation.

OTHER INTEGRATION OPTIONS

The CCL REFERENCE element

The CCL REFERENCE statement allows you to directly reference tables and views in SAP HANA within an ESP project. Once you use the CREATE REFERENCE statement to give a local alias to a table or view in SAP HANA, you can then use the REFERENCE element in other CCL statements. The most common use is to join an ESP stream or window to a table in HANA.

The getData() function

The event processing language of ESP – CCL -- includes a function getData() that can be used to run event-driven queries against an external database. This function is typically called within a SPLASH script embedded in the project (SPLASH is the scripting language that lets you define custom operators in CCL). The getData() function takes a parameterized SQL statement, substitutes in values for the parameters, and stores the resulting rows in a vector of records for the SPLASH code to manipulate. A common use case would be to receive an event, retrieve some contextual reference data from a database, augment the event with the contextual data, and then send the contextualized event(s) out to downstream operators.

ESP Utilities

ESP includes two utilities that can, when used in concert, provide another way to integrate ESP with external data sources. Within a custom script, the esp_upload utility can be used along with the esp_convert utility to pipe external XML or CSV data into ESP as events.

The esp_subscribe utility can be used in a script to subscribe to one or more ESP Output Streams or Output Windows, writing the events to standard out in either CSV or XML format.

CONCLUSION

There are a wide-ranging set of options available to integrate ESP into your existing event and data architecture. We encourage further exploration of the topics introduced in this document using the links below. Should you have further thoughts, questions, requests for clarifications or additional integration features, please post your questions or suggestions to the discussion forum in the ESP Developer Center and we will continue the dialogue there.

FOR MORE INFORMATION

ESP Developer Center: http://scn.sap.com/community/developer-center/esp
ESP Product Documentation:
- Adapters Guide
- Building Custom Adapters
- Utilities Guide
- C SDK
- Java SDK
- .NET SDK