USER GUIDE | PUBLIC
Document Version: 1911a – 2019-11-11

Rule-based IoT Data Processing

Managing activities and events in response to timeseries data

THE BEST RUN SAP

Content

1 Rules: Overview

Overview

Speaking of the Internet of Things (IoT) is closely related to Big Data, that is, the handling of very large amounts of data that is generated by devices that are equipped with all kinds of sensors. However, while the large amounts of data can be used to open new perspectives on things and to gain completely new insights, they can, at the same time, also be seen as a problem. This is because the sheer amount of data can make it difficult to find the needle in the haystack (for example, the one or two outliers in a huge pile of data that indicate a technical problem that has either already occurred or is likely to happen in the near future).

At this point, rules for processing time series data come in handy. With rules, you can leave it up to the system to permanently keep an eye on each and every sensor data point coming in. You start with a virtual thing that you want to monitor, pick the properties to be observed, and associate both with a rule. In the Rule Modeler, you define the values for each property and what shall happen if that particular value is observed in the stream of incoming data.

> **i** Note
>
> Defining a rule-based system reaction to a particular value that has been observed is a two-stage process:
>
> - In the Rule Modeler, you set up the rules to be applied to incoming values. If a value matches the rule condition, the system triggers an event.
> - The event triggered by the rule is then used as input for the Action Modeler where you define what the system reaction to the observed value shall look like (e.g., send an HTTP request, send a notification mail, or trigger another action).

With these preparations done, you can rest assured that none of the critical values that you might think of goes unnoticed, thus making sure that all necessary action can be taken without delay.

Data Protection and Privacy

Rule modeling in SAP Leonardo IoT makes internal use of SAP Cloud Platform Business Rules. This means that creating and maintaining objects with SAP Leonardo IoT also leads to the creation and modification of objects that are handled by SAP Cloud Platform Business Rules. Depending on the nature of the data that you process in rules, questions of data protection and privacy may arise. For more information on how pesonal data may occur in rules and on how the underlying rule engine handles such cases, see Data Protection and Privacy.

Related Information

SAP Cloud Platform Business Rules

1.1 Naming Conventions for Rule Processing

Names and strings that may not be used in names of objects used in rules.

Reserved Names for Data Objects and Thing Properties

In SAP Leonardo IoT, there is a set of names or strings that you may **not** use in names for objects that are referenced by rules or rule contexts. See the following table for details.

> **i** Note
>
> These naming conventions are only valid for use in the context of rule processing. E.g., you are free to define a thing property named `SAP_RoomTemperature` in the Properties Catalog and use it for analyzing time series data with an application built in SAP Web IDE. However, if you try to use that property in a rule context, the system detects the name and raises an error message because property names starting with "SAP_" are **not** allowed in the context of rule processing.
>
> The reserved names are **not** case-sensitive. I.e., a property name fails the validation, regardless whether the name is "`system`", "`System`", or "`SYSTEM`".

Reserved Names

Object Type	Reserved Name
Data Object	*_aggregate
Data Object	*_dimension
Data Object	*_dimensions
Data Object	SAP
Data Object	SAP_*
Data Object	System
Data Object	System_*
Data Object	Result_Stream
Data Object	Result_Batch
Property	*_UpperUpper
Property	*_Upper
Property	*_Lower

Object Type	Reserved Name
Property	*_LowLower
Property	SAP
Property	SAP_*
Property	System
Property	System_*
Property	IoT_Rule_ID
Property	IoT_Event_Name
Property	IoT_Event_Severity

Naming Conventions for Rules and Rule Contexts

- For the names of rules and rule contexts, the following naming conventions are in effect:
  Allowed characters: `[A..Z]`, `[a..z]`, `[0..9_]`. Leading or trailing whitespace is **not** allowed. However, you may use whitespace inside of a name.
- For the names of data objects referenced by rule contexts, the same applies as for rules and rule contexts. However, for data objects, whitespace is generally **not** allowed, regardless of its position within the name string.

  > **i** Note
  >
  > The names of data objects are currently automatically derived from the name of the underlying property set. However, that may change in a future release.

1.2 Referential Integrity for Objects Related to Rules

How the system ensures a consistent database status after changes to rule-related objects.

Introduction

Whenever an object is changed (i.e., modified or deleted) that is used by a rule, the rule processing framework evaluates the consequences of that change for the affected rules. If necessary, the system automatically takes care of all the rule adjustments required to keep the database in a consistent state.

During rule modeling, quite a number of different types of objects are involved, such as rules, rule contexts, property sets, thing types, and things. In addition, there is a hidden layer of even more object types that belong to an internal rule processing engine that is used by SAP Leonardo IoT. Modeling rules basically means putting together instances of all these different object types according to the use case that you have in mind. While you

put together all the objects you are interested in, the system takes care of placing all the objects involved into a mesh of relationships behind the scenes.

However, when it comes to subsequent changes to objects referred to by a rule that has already been set up, chances are that a human expert would not be able to keep track of all those interrelated (and - from a UI perspective - invisible) objects. Therefore, it is essential that the system adapts the underlying database objects on all layers exactly to the changes that a user performs on the surface (such as modifying or deleting rule-related objects).

Impacts of Changes in Detail

In the following table, we have listed in detail what it means for the system when you add, change, or delete objects related to a rule.

Impact of Changes to Rule-Related Objects

Object Type	Change	Impact
Property Set Type	Deletion	• All usages in all rule contexts are deleted. • All usages in all rule inputs are deleted. • All usages of batch aggregates based on the property set type are deleted.
Reference Property Set Type	Deletion	• All usages in all rule contexts are deleted. • All usages in all rule inputs are deleted.
Thing Type	Deletion	• All rule contexts using the thing type are deleted. • All rules using a rule context based on the thing type are deleted. • All explicit rule assignments to the thing type are removed.
Property Set	Removal from Thing Type	• All usages in all rule contexts are deleted. • All usages in all rule inputs are deleted. • All usages of batch aggregates based on the property set type are deleted. • All thresholds that belong to the property set are removed.

Object Type	Change	Impact
Property	Addition to Property Set Type	• Property is propagated to underlying rule engine. • Aggregates are created. • All changes take immediate effect in underlying rule engine.
Threshold Property	Addition	• Property is propagated to underlying rule engine. • All changes take immediate effect in underlying rule engine.
Property	Deletion from Property Set Type	• Property is removed from underlying rule engine and cannot be used anymore. • Aggregates are removed and cannot be used anymore. • All thresholds referring to the property are deleted.
Sensitivity Flag of Property Set Type	Change to `True`	• Sensitive property sets cannot be used in rules. • All rules referring to the property set type are deactivated.
Property Set Type of Property Set within Thing Type	Change	• Change is propagated to underlying rule engine. • Aggregates are updated, created, or deleted as required.
Property Reference	Change (e.g. different data type)	• All usages in all rule contexts are deleted. • All usages in all rule inputs are deleted. • All usages of batch aggregates based on the property set type are deleted.

Related Information

Reference Property Set Type
Property Set Type
Data Protection and Privacy
Thing Type

1.3 Error Codes for Rule Execution

Overview of all types of errors that can occur during batch rule execution

In this section, you can inform yourself about the different types of errors that can occur when the system processes scheduled rules. Do not confuse these error codes with errors that can occur during the execution of actions that a rule may trigger. For an exhaustive overview of errors related to action execution, see Action-Related Error Codes [page 44].

Error Codes for Rule Execution

Error Code	Category	Root Cause	Solution
810000000	Runtime Exception	Unexpected error occurred during scheduled rule execution. Contact SAP support for further information.	Create a support ticket.
810000001	Runtime Exception	Error making remote service call.	Create a support ticket.
810000002	Rule Modeling Exception	Error while parsing message for rule activation payload.	Create a support ticket.
810000003	Runtime Exception	Rule not found in runtime environment.	Make sure that the rule has been activated.
810000004	Rule Modeling Exception	Invalid rule condition. At least one aggregate property required.	Provide aggregate property in rule condition.
810000005	Rule Modeling Exception	Incorrect time frame format.	Create a support ticket.
810000006	Rule Modeling Exception	Incorrect windowing format.	Create a support ticket.
810000007	Runtime Exception	Rule result conversion error.	Create a support ticket.
810000008	Runtime Exception	Date conversion error.	Create a support ticket.
810000009	Runtime Exception	Failed to get rule engine instance.	Create a support ticket.

2 Rule Context: Overview

Introduction

When you start setting up rules for sensor data generated by a device, you normally have a particular sensor property in mind and the values that this property might have. At that stage, before actually setting up a rule, you have to define an entity that mirrors the properties that you want to monitor by a rule. This entity is referred to as rule context (sometimes also called a "vocabulary"). It can be seen as an interface between the properties of a thing and the rule that is triggered in response to the sensor data related to these properties.

In the rule context, all the properties of a particular property set or property set type are reflected. After assigning a rule context to a rule, you can use any of these properties for monitoring by the rule.

> ℹ Note
>
> Rule contexts are either based on a property set of a particular thing type or on a self-contained property set type (that is, independent of a thing type).
>
> After a connection between a rule context and a property set has been established, the rule context automatically reflects any changes made to the property set, such as adding, changing, or deleting properties. However, a similar kind of synchronization between a rule context and a rule that uses the rule context is technically **not** possible. This may lead to a situation where a rule has been set up based on a property that is no longer available or has changed significantly. In such cases, the system sends a respective error message.

Property Sets and Property Set Types

As already mentioned, rules monitor and respond to measured values captured by the sensors of a thing. In the course of mapping a physical device to a virtual thing, the values measured by a sensor are reflected by a property in the thing model, such as temperature, pressure, or speed. A property, however, can exist in two different flavors in the thing model of SAP Leonardo IoT:

- Properties as part of a self-contained property set type: These are the properties that you maintain in the Thing Properties Catalog app of SAP Leonardo IoT. Properties of this kind are independent of a particular thing type and can be assigned to whatever thing inside the same package.
- Properties within a property set as sub-elements of a thing type: These are properties that are defined in the property set type which is assigned to a thing type. All the things based on such a thing type inherit exactly the same set of properties as defined for the thing type.

Using properties based on named property set types is the recommended approach for thing modeling in scenarios where a number of different thing types share one or more sets of properties that all thing types have in common (for example, 10 different types of coffee machines that all share the same water tank, or the same brewing unit). Keeping these common properties in a named property set type allows you to monitor the

property values of all things derived from the thing types in question with the help of one single rule. As opposed to that, using properties that are defined directly in the context of a thing type (that is, **without** a named property set type) would require to set up one rule for each of the thing types.

Related Information

Thing Properties Catalog: Overview

2.1 Create a Rule Context

Process steps for creating a rule context

Context

You define a rule context based on a property set or a property set type that is assigned to a thing type. The properties reflected by the rule context can then be used by a rule that is triggered depending on the values for a particular property.

The settings maintained for a rule context fall into two categories:

- *Basic Data*: Basic data comprise administrative data like name, ID, different description texts. These data are typically stable over time.
- *Data Object*: Here, the rule context is associated with an entity that serves as the data source for a rule. Currently, you can only assign property sets (or property set types, respectively) of a thing type that represent time series data (measured values).

Procedure

1. From the launchpad, start the *Rule Contexts* app.

   The system presents the list of rule contexts available in the current package.
2. Choose *New*.

   The system presents an empty detail screen where you can enter the settings for the new rule context.
3. Enter a *Name* for the rule context. Optionally, you can also enter a *Short Description* and a *Description*. These three fields serve the following different purposes:

Field	Purpose
Name	Mandatory, **not** translatable. A name indicating the purpose of the rule context. You may use this field for entering

Field	Purpose
	some abbreviated or coded information that is common to your company (e.g., the ID of a building or a department). This field is also used as sorting criterion in the value helps of the Rules Modeler, so it is advisable to follow a systematic naming convention.
Short Description	Optional, translatable. A description in natural language indicating the purpose of the rule context.
Description	Optional, translatable. Use this field for a few sentences that help other users understand what the rule context contains, where it is used, and so on.

4. To select a property set, choose *Add* to the right of the *Thing Data - Property Sets* table.

   The system displays the value help window for selecting a property set.

5. In the property set value help window, select first a *Package* and then a *Thing Type* to narrow down the list of property sets.

   The system displays all property sets that are assigned to the thing type.

6. Choose one or more property sets that you want to use as the basis for the new rule context.

   The system immediately associates the selected property sets with the rule context and navigates you back to the rule context detail screen.

   > **i Note**
   >
   > For reasons of data protection and privacy, it is **not** allowed to assign a property set to a rule context that has been classified as holding personal data, or critical personal data.
   >
   > Also, if an initially non-classified property set is assigned to a rule context that is used by rules, any later classification of the property set as personal, or critical personal data leads to the deactivation of all affected rules. For more information, see Create a Property Set.

7. Click *OK* to confirm your selection.

   The system assigns the selected property sets to the rule context. Once this is done, you can modify the property set names within in the context of the rule context. This is useful, e.g., if the property set names are too technical to adequately reflect their purpose in a business scenario.

8. If desired, repeat the above steps accordingly in the context of the *Thing Data - Property Set Types* table.

   > **i Note**
   >
   > Unlike property sets, property set types are self-contained objects that exist independent from a thing type. Therefore, after choosing a package, the system immediately presents the list of available property set types without prior selection of a thing type.
   >
   > After adding a property set or a property set type to the rule context, you can modify the name of the assigned object. This is helpful, for example, to resolve conflicts with the naming conventions that are in effect for rule processing (for more information, see Naming Conventions for Rule Processing [page 4]).

9. Check the settings. If you are satisfied, choose *Save*.

Results

The new rule context is now available and ready to be assigned to a rule.

2.2 Maintain a Rule Context

Process steps for maintaining a rule context

Context

As an administrator, you want to modify an existing rule context.

> **i** Note
>
> You should keep in mind the following aspects:
>
> - As long as a rule context has **not** been assigned to a rule, you can modify its settings in both sections, *Basic Data* as well as *Data Object*.
> - If a rule context has already been assigned to a rule, modifying is only possible for the fields in the *Basic Data* section. In that case, the assigned *Data Object* cannot be changed. If you want to change the data object because its properties have been changed and are now causing an error when used in a rule, create a new rule context instead and reassign it to the rule.

> **i** Note
>
> For more detailed information on the various settings of a rule context, see Create a Rule Context [page 10].

Procedure

1. From the launchpad, start the *Rule Contexts* app.

   The system presents the list of rule contexts available in the system.
2. In the list of rule contexts, click the name of the rule context you want to change. If you don't see the rule context that you want to change, use the *Search* or the filter fields to narrow down the number of rule contexts presented in the list.

   After clicking the rule context name, the system presents the detail screen where you can enter the settings for the rule context.
3. Modify the field values according to your requirements.
4. Choose *Save*.

2.3 Delete a Rule Context

Process steps for deleting a rule context from the system

Context

As an administrator, you want to delete one or more rule contexts from the system. Deleting a rule context can be useful in different situations, for example:

- The rule context is not referenced by any rule. In fact, that is a prerequisite for deleting a rule context. If you realize there are many unused rule contexts, you may want to perform a mass deletion.
- The rule context is already used by a rule. However, the associated property set has been changed. As a consequence, the rule can no longer be processed. Since the property set referenced by a rule context cannot be changed, the only solution is to delete the rule context and create a new one with a reference to a different property set.

Procedure

1. From the launchpad, start the *Rule Contexts* app.

   The system presents the list of rule contexts available in the system.

Deleting a Single Rule Context

2. In the list of rule contexts, click the name of the rule context you want to delete. If you don't see the rule context that you want to delete, use the *Search* or the filter fields to narrow down the number of rule contexts presented in the list.

   After clicking the rule context name, the system presents the detail screen for the rule context.
3. In the detail screen, make sure if the rule context displayed really is the one you want to delete.
4. In the menu bar, choose *Delete*.

   The rule context is deleted, and the system navigates you back to the entry screen.

Deleting Multiple Rule Contexts

> **i Note**
>
> If you are planning a cleanup action and you are totally clear about which rule contexts to delete and which not, you can skip the navigation to the detail screen for each rule context. Instead, proceed as follows:

5. In the list of rule contexts, select the rule contexts you want to delete by ticking the checkbox in the first table column.
6. Choose *Delete*.

   The system presents a confirmation dialog where you have to confirm your decision to delete the selected rule contexts.
7. Confirm your decision.

Results

The rule context is deleted from the database.

3 Rule Modeling: Overview

General information on defining rules

Introduction

Collecting time series data in an IoT scenario is not a purpose in itself. Rather, in most cases, you want to monitor and analyze the incoming data. Even more important is to watch out for all kinds of irregularities (for example, an unwanted drift of sensor data, or a sudden increase of outliers) indicating that something is going wrong in the area where the sensors are installed. In such cases, it is important to keep the responsible experts informed as fast as possible so that they can react in an appropriate way.

With rules for IoT time series data, you can define a widely automated process to handle such critical situations. You decide which sensor data you want to monitor, which value ranges reflect a normal situation, and which not, and you define what the system is supposed to do in case the values indicate an error situation.

Rules can be active or inactive. Only active rules are executed by the system. Leaving rules in inactive status is a way for you to prepare either comprehensive changes to one rule or changes to a bigger set of rules that you want to make effective at a later point in time. The system supports you in handling bigger sets of rules by offering mass activation, or deactivation, of rules. You can accomplish this by selecting all relevant rules in the list offered on the entry screen of the app and choose *Activate*, or *Deactivate*.

Each rule contains a condition that is tested by the rule and an event that the rule triggers if the condition is fulfilled. Once you have specified a rule, you can define an action in the Action Modeler that the system performs when the rule condition is fulfilled.

Rule Types

In SAP Leonardo IoT, you can use the following types of rules:

- Streaming Rules
- Streaming Rules (hybrid)
- Scheduled Rules

You specify the type of rules upon rule creation. Once a new rule has been saved, you cannot change its type anymore.

Streaming Rules

You use streaming rules for monitoring the measured values that are sent from the sensors of a device in real time, or near real time, respectively. For the rules themselves, you can define expressions for describing conditions that the rule shall check. For example, a rule might check if a certain peak temperature is exceeded. For these expressions, a small set of comparison operators is available that helps you describe the condition check.

> **i Note**
>
> The maximum frequency for the system to trigger an event for a given thing monitored by a streaming rule is one event per 10 seconds. In other words, no more than six events per minute can be triggered.
>
> However, should your use case require a higher rate of events, we encourage you to get in touch with SAP to see if the threshold can be adjusted to your individual needs.

When working with streaming rules, a problem can arise when rule execution leads to a high number of events triggered by the rule (e.g., by sending dozens of more or less identical emails to an administrator) where one single event would already be sufficient to raise the attention of a person in charge. To tackle this challenge, you can decide to run a rule in sleep mode or toggle mode. For more information, see Reducing Event Creation with Sleep Mode and Toggle Mode [page 25].

> **i Note**
>
> Streaming rules (both hybrid or not) can only operate on time series data that has been ingested via SAP Cloud Platform Internet of Things service. That is, time series data that has been ingested by directly accessing a thing API as well as data that has been collected in a cloud-to-cloud interoperability landscape **cannot** be processed by streaming rules.

Streaming Rules (hybrid)

Hybrid streaming rules basically serve the same purpose as streaming rules. They differ from "normal" streaming rules in the following respects:

- As opposed to standard streaming rules, sensor data doesn't need to be transferred to the cloud for processing. Instead, hybrid rules support edge processing, i.e., sensor data can be processed immediately in the local surrounding of the connected physical device. In such an environment, immediate responses to critical sensor data are easier to realize. Also, in environments with poor data transfer capacity, edge processing may not only decrease the load on the available communication channels; moreover, it may be the only possible way of processing sensor data at all in near-real time mode.
- However, the advantages of edge processing come at a price: Compared to standard streaming rules, hybrid rules support a reduced feature scope. The generally reduced processing capacity at the edge sets the limit for hybrid rules, regardless whether they are activated for use at the edge, in the cloud, or for both target environments. For details of the supported feature scope of hybrid rules, see the closing section of Create a Rule [page 17].
- After having defined the rule and after its activation for use at the edge, an additional configuration task is required. For more information, see Hybrid Streaming Rule [page 21].

Scheduled Rules

Very soon after new measured values have been ingested (and maybe already processed by a streaming rule), the system stores the values in a dedicated storage section of the underlying database infrastructure. Once the values have been stored, you can access and analyze them at any later point in time. Also, while streaming rules take each single value into account, scheduled rules operate on aggregated chunks of values. For this kind of a deferred analysis of aggregated values, you use scheduled rules.

A scheduled rule is very similar to a streaming rule. However, for scheduled rules, you have to make additional settings:

- Windowing: You define the time frame for the values to be analyzed. Only values with a timestamp within the specified time frame are taken into account for the analysis.

- Scheduling: You specify whether a rule shall be executed only once at a particular point in time or if it shall be executed on a regular basis.

Another difference between streaming rules and scheduled rules is that for scheduled rules, the system supports a set of aggregate functions that you can use to analyze the values in the specified sample.

> **Example**
>
> If the average temperature of a cooler aggregate in the past two hours is higher than ten centigrade, send an alert to the person responsible for that aggregate.

Related Information

Reducing Event Creation with Sleep Mode and Toggle Mode [page 25]

3.1 Create a Rule

Process steps for creating a rule

Context

You define a rule by choosing one of the existing rule contexts and defining an error condition as well as an event that is triggered in response to that condition. A rule can either apply to streaming data, or it can operate on aggregated, stored data following a fixed schedule. You define the rule type at creation time. This setting cannot be changed afterwards.

Rules can be active or inactive. Only active rules are executed by the system. Leaving rules in inactive status is a way for you to prepare a bigger set of rules that you want to make effective at a later point in time.

> **Note**
>
> Instead of creating a rule from scratch, you can also use an already existing rule as a template for a new one. To accomplish this, simply select one of the existing rules. In the details screen of the rule, choose *Copy*. The system carries over all settings of the existing rule to a new one (except for the rule condition, which is left empty in the new rule), which you can then modify as desired and finally save under a new name.

> **Note**
>
> You can only create a rule if you have already prepared rule contexts to which the new rule can refer. A rule can only be saved with a valid reference to a rule context. For more information, see Rule Context: Overview [page 9].

Procedure

1. From the launchpad, start the *Rules* app.

   The system presents the list of rules available in the system.
2. Choose *New Rule*.
3. From the menu, choose the type of the rule you want to create (*Scheduled*, *Streaming*, or *Streaming Hybrid*).

   The system presents an empty detail screen where you can enter the settings for the new rule on the *General Information* page.
4. Enter a *Name* for the rule. Optionally, you can also enter a *Short Text* and a *Description*. These three fields serve the following different purposes:

Field	Purpose
Name	Mandatory, **not** translatable. A name indicating the purpose of the rule. You may use this field for entering some abbreviated or coded information that is common to your company (e.g., the ID of a building or a department). This field is also used as sorting criterion in the value helps of the Rules Modeler, so it is advisable to follow a systematic naming convention.
Short Text	Optional, translatable. A description in natural language indicating the purpose of the rule.
Description	Optional, translatable. Use this field for a few sentences that help other users understand what the rule contains, where it is used, and so on.

5. Optionally, you may define a set of freely defined *Tags* that you can attach to a rule. You can use tags as filter criteria for the list of rules displayed on the entry screen of the app.
6. Once you are done with the *General Information*, move on to the *Definition* page.
7. From the list of *Rule Contexts*, select the one you want to use in the rule.

   If the selected rule context has multiple property sets or property set types assigned, you must specify the one that shall be used for the new rule. In other words, you have to decide which of the available property sets (or property set types, respectively) shall be associated with the new rule. Simultaneous usage of more than one property set in one rule is currently **not** supported.

   The system informs you that the selected combination of rule context and property set (type) cannot be changed anymore once it has been assigned to a rule. *Confirm* the information to assign the rule context. The system automatically displays the *Thing Type* to which the rule context belongs.
8. In the *Rule Editor* section, define the *If* condition that the rule shall test.

   The system expects the If condition in a simple, predefined syntax, which slightly differs depending on the rule type:

   - Streaming rule:
     `<property name> of <property set name> <operator> <value>`

     > **Example**
     >
     > `WaterTemperature of Engine5 is greater than 95`

- Scheduled rule:
  `<property name>_<aggregate_function> of <property set name>_aggregate <operator> <value>`

  > **Example**
  >
  > `WaterTemperature_AVG of Engine5_aggregate is greater than 95`

The following aggregate functions can be used for analyzing time series data with scheduled rules:

Aggregate Functions for Scheduled Rules

Function	Description
COUNT	Returns the number of records in the sample. Supported for all data types.
FIRST LAST	Returns the first or last record in the sample. Supported for all data types.
TFIRST TLAST	Returns the timestamp of the first or last record in the sample. Supported for all data types.
MIN MAX	Returns the lowest or highest measured value. Supported for numeric data types.
TMIN TMAX	Returns the timestamp of the lowest or highest measured value in the sample. Supported for numeric data types.
SUM AVG STDDEV	Returns the sum, average, or standard deviation of the measured values in the sample. Supported for numeric data types.

> **Note**
>
> You cannot type into the entry fields of the rule editor. Instead, the system offers you a list of all field names and operators that can be used for setting up the rule definition. To expand the list of possible entries, type a space character. Then, pick the desired object with the mouse or the cursor keys.

9. Define which event shall be triggered if the rule condition is fulfilled. For this, you must enter the *Event Name* and the *Event Severity*.

The following steps depend on the rule type:

Rule Type-Specific Settings

Rule Type	Steps
Scheduled Rule	1. In the *Windowing* section, define a time window for aggregated time series data to be monitored by the rule. To accomplish this, specify a *Time Base* (e.g., days, hours, or minutes) and a numeric *Value* that specifies the timespan backwards from the moment of rule execution. 2. In the *Scheduling* section, specify either the *Recurrence* pattern for a rule that shall be executed on a regular basis, or the *Start Time* and *End Time* for a rule that shall be executed only once. **i Note** ○ The specified time window must lie within a range from 2 minutes through 30 days. ○ You cannot define a combination of different *Time Base* values, e.g., "2 days 8 hours". Instead, adjust the numeric value in combination with one single time base, e.g., "56 hours". ○ If you choose **`Minute`** as the *Time Base*, the *Value* must be a multiple of 2.
Streaming Rule	1. In the *Settings* section, choose the desired *Execution Mode* for the rule. You have the following options: ○ *Default*: For each monitored value that fulfills the rule condition, the rule triggers an event. ○ *Sleeping*: The rule stops triggering events for a specified duration even if the rule condition is fulfilled. ○ *Toggle*: The rule only triggers an event if the rule condition is fulfilled **and** the condition was **not** fulfilled for previously tested values. Sleep mode and toggle mode are useful to avoid event flooding in certain scenarios. For more information, see Reducing Event Creation with Sleep Mode and Toggle Mode [page 25]. 2. Depending on the chosen execution mode, proceed as follows: ○ *Default*: No further settings required. ○ *Sleeping*: Specify the desired *Sleeping Period*. This value must lie within a range from 10 seconds to 1 hour. ○ *Toggle*: No further settings required.
Streaming Hybrid	*Streaming Hybrid* rules can be activated for use both in the cloud as well as at the edge (i.e., for immediate processing in the direct local surrounding of the physical device, without the need for transferring data to the cloud first). However, in edge mode, fewer features are supported. This limited feature set is also in effect for hybrid rules used in the cloud as well (in the sense of a least common denominator). Here is the list of features supported by hybrid rules: ○ Rule context must refer to a property set that contains time series data. ○ The rule condition can handle only one property, which must be numeric. ○ The rule condition supports the following operators: `greater than`, `less than`, `equals`. ○ Execution modes: Default and sleep mode (i.e., toggle mode is **not** supported).

10. Choose *Save*.

Results

The new rule is ready for use. However, for the rule to take effect, you must *Activate* it first.

> **i Note**
>
> In case of a hybrid rule, activating requires an additional decision whether the rule shall be used in the cloud, at the edge, or in both environments. Once you have made your decision, the rule is activated for the selected target environment.

Related Information

Rule Context: Overview [page 9]

3.1.1 Hybrid Streaming Rule

Rules for execution in the cloud or at the edge.

Overview

SAP Leonardo IoT supports the definition of rules and actions, that can be executed either in the cloud, via the SAP Leonardo IoT gateways, or in both environments.

To set up such a hybrid rule, you use the option to create a rule of type *Streaming Hybrid*. Once you have defined the rule settings, you decide upon activation time in which target environment the rule shall be used. If you decide to activate the rule for execution on the edge, additional steps are required for specifying the exact edge gateways as well as for deploying the rule to that gateway.

To accomplish these additional steps, SAP Leonardo IoT comes with an additional app named Edge Rules Configuration. This configuration app complements the general Rules app that has always been a part of the SAP Leonardo IoT solution. The purpose of this complementary app is the following:

- Selection of a set of hybrid streaming rules.
- Specification of an action to be executed if the rule condition is met.
- Compilation of an edge configuration that can be sent to the Edge Policy Service.

The following picture illustrates the required steps as well as the involved tools:

Constraints

Streaming Hybrid rules can be activated for use both in the cloud as well as at the edge (i.e., for immediate processing in the direct local surrounding of the physical device, without the need for transferring data to the

cloud first). However, in edge mode, fewer features are supported. This limited feature set is also in effect for hybrid rules used in the cloud as well (in the sense of a least common denominator). Here is the list of features supported by hybrid rules:

- Rule context must refer to a property set that contains time series data.
- The rule condition can handle only one property, which must be numeric.
- The rule condition supports the following operators: `greater than`, `less than`, `equals`.
- Execution modes: Default and sleep mode (i.e., toggle mode is **not** supported).

Related Information

Create Edge Configuration for Hybrid Rules [page 22]

3.1.1.1 Create Edge Configuration for Hybrid Rules

Process Steps for creating an edge configuration for hybrid rules.

Context

For executing hybrid streaming rules on the edge rather than in the cloud, the rule as well as the action triggered by the rule must be advertised to the respective edge gateway. To accomplish this, you set up a data record that contains a list of rules/action pairs. This data record is named configuration and must be published to the Edge Policy Server so that rule execution on the edge can take place. After publishing, a configuration has to be assigned to the relevant gateway (or to a group of gateways).

> **i** Note
>
> For executing the following process steps, you must be subscribed not only to SAP Leonardo IoT, but also to SAP Edge Services. For more information, see the SAP Cloud Platform product page of SAP Edge Services. Depending on your SAP Cloud Platform subscription status, you may have to take different preparations:
>
> - If you are already subscribed to SAP Leonardo IoT before you subscribe to SAP Edge Services, create a support ticket (component **`IOT-BSV-OPS`**) and request an update subscription. Make sure to use the term "update subscription" in the ticket so that SAP can respond accordingly to your request.
> - If you were first subscribed to SAP Edge Services and afterwards subscribed to SAP Leonardo IoT, the integration between SAP Edge Services and SAP Leonardo IoT works immediately without the need for further adjustments.

Procedure

1. From the launchpad, start the *Edge Rules Configuration* app.
   The system presents the list of configurations available in the system.
2. Choose *New*.
3. Enter the *Edge Configuration Name*.
4. In the *Rule* field, open the value help and choose from the rules offered in the list.
5. From the list of *Actions*, choose one or more to be assigned to the rule.
   If you want to bundle several rules in the configuration, repeat the previous steps.
6. As soon as you have added all rules that you want to reference via the configuration, choose *Publish*.
   The system automatically saves the configuration and deploys it to the edge.

> **i** Note
>
> The *Edge Rules Configuration* app also supports mass maintenance for the elements that you add to a configuration: To accomplish this, you can start by first adding all desired rules to the configuration. After that, choose *Assign Same Actions to all Rules*. Once you have selected the desired actions, all selected actions are assigned to all rules in the configuration.

Results

You are now finished with the preparations for executing rules on the edge. As a final step, you need to assign the published configuration to the relevant gateway (or a group of gateways).

3.2 Maintain a Rule

Process steps for maintaining a rule

Context

As an administrator, you want to modify an existing rule for monitoring timeseries data.

> **i** Note
>
> You should keep in mind the following aspects:
>
> - You cannot switch the type of a rule (streaming rules versus scheduled rules). This setting is defined upon rule creation and cannot be modified at a later point in time.
> - You cannot change the rule context that is assigned to a rule.
> - If a rule has already been assigned to an action, it is advisable to check the action as well.

> **Note**
>
> For more detailed information on the various settings of a rule, see Create a Rule [page 17].

Procedure

1. From the launchpad, start the *Rules* app.

   The system presents the list of rules available in the system.
2. In the list of rules, click the name of the rule you want to change. If you don't see the rule that you want to change, use the Search or the filter fields to narrow down the number of rules presented in the list.

   After clicking the rule name, the system presents the detail screen where you can maintain the settings for the rule.
3. Modify the field values according to your requirements.

   On the *Definition* tab, the system presents the current settings in read-only mode and copies them into editable fields. This helps you gaining an overview of the before/after situation for the rule.
4. Choose *Save*.

3.3 Delete a Rule

Process steps for deleting a rule from the system

Context

As an administrator, you want to delete one or more rules from the system. Deleting a rule can be useful in different situations, for example:

- The rule is not referenced by any action. In fact, that is a prerequisite for deleting a rule. If you realize there are many unused rules, you may consider performing a mass deletion.
- The rule has already been replaced by an alternative one that better suits your business requirements. This is especially true when you had to make use of a different rule context, which is only possible by setting up a new rule. In that case, deleting old and outdated rules helps to keep the system clean and understandable.

> **Note**
>
> You cannot delete an active rule. Before deleting, you have to deactivate the rule.

Procedure

1. From the launchpad, start the *Rules* app.

   The system presents the list of rules available in the system.

Deleting a Single Rule

2. In the list of rules, click the name of the rule you want to delete. If you don't see the rule that you want to delete, use the Search or the filter fields to narrow down the number of rule contexts presented in the list.

   After clicking the rule name, the system presents the detail screen for the rule.

3. In the detail screen, make sure if the rule displayed really is the one you want to delete.
4. In the menu bar, choose *Delete*.

   The rule is deleted, and the system navigates you back to the entry screen.

Deleting Multiple Rules

5. > **i** Note
   >
   > If you are planning a cleanup action and you are totally clear about which rules to delete and which not, you can skip the navigation to the detail screen for each rule. Instead, proceed as follows:

   In the list of rules, select the rules you want to delete by ticking the checkbox in the first table column.

6. Choose *Delete*.

   The system presents a confirmation dialog where you have to confirm your decision to delete the selected rules.

7. Confirm your decision.

Results

The rule is deleted from the database.

3.4 Reducing Event Creation with Sleep Mode and Toggle Mode

Two different approaches for reducing the number of alerts during rule execution

Overview

Processing data in the Internet of Things means almost always processing mass data. Depending on the use case, device sensors may collect and send huge amounts of measured values, which can impose a significant load on the software layers that are in charge of processing the incoming data. Moreover, there are many use

cases where it is important and required to collect large amounts of measured values, but, at the same time, it is perfectly fine **not** to react on each single value.

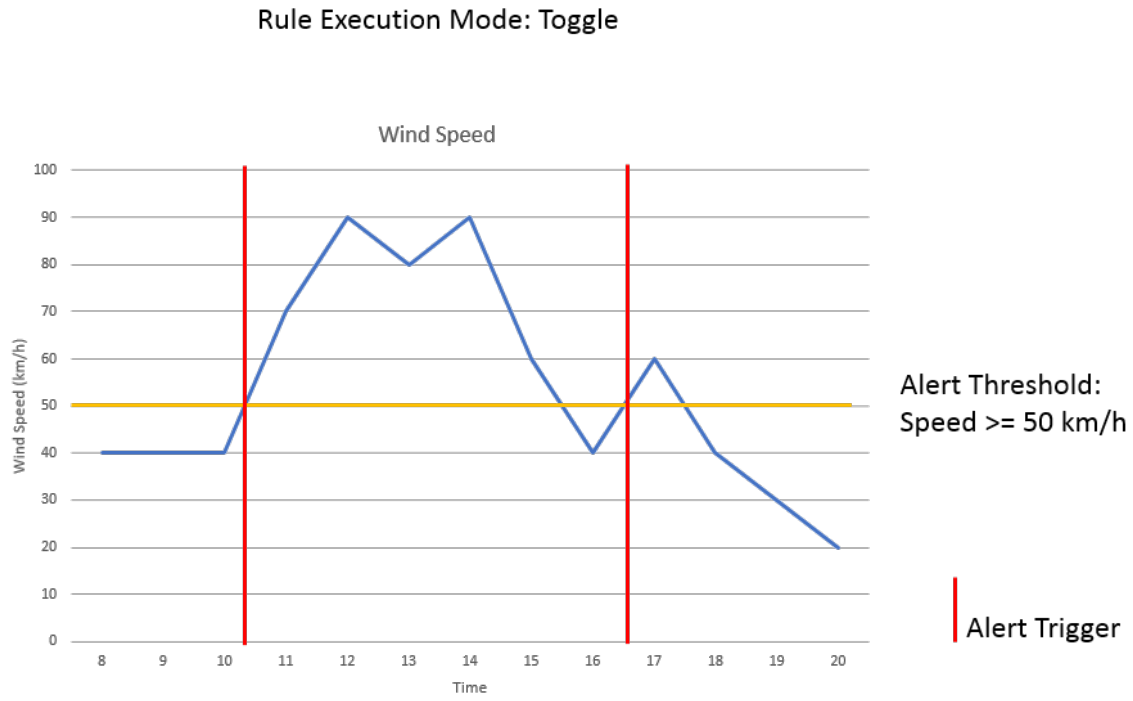
> ### Example
>
> In a building automation scenario, a number of sensors have been installed that measure the current wind speed. In addition, there are sensors used to detect whether a window is closed or not. A rule has been set up to alert a facility manager in case a certain wind speed is exceeded and there are still windows open that need to be shut manually.
>
> Assuming the wind speed threshold has been set to 50 km/h and the wind exceeds that threshold at 10:10 a.m.. The wind speed may even rise for the next 5 hours up to a maximum of 90 km/h, before the weather calms down and the wind speed decreases to 40 km/h at 4:00 p.m.. With the rule described above, this would mean that the system is flooding the facility manager with alert messages during a period of more than six hours (i.e., almost the entire workday), or at least up to the moment when all windows have been shut. However, for drawing the facility manager's attention to the potential threat caused by the combination of increasing wind speed and open windows, one single alert (instead of hundreds or thousands) would already be sufficient.

With the rule modeling feature of SAP Leonardo IoT, you can take advantage of two different approaches that help you reduce the number of alerts caused by a scenario as described in this example, sleep mode and toggle mode:

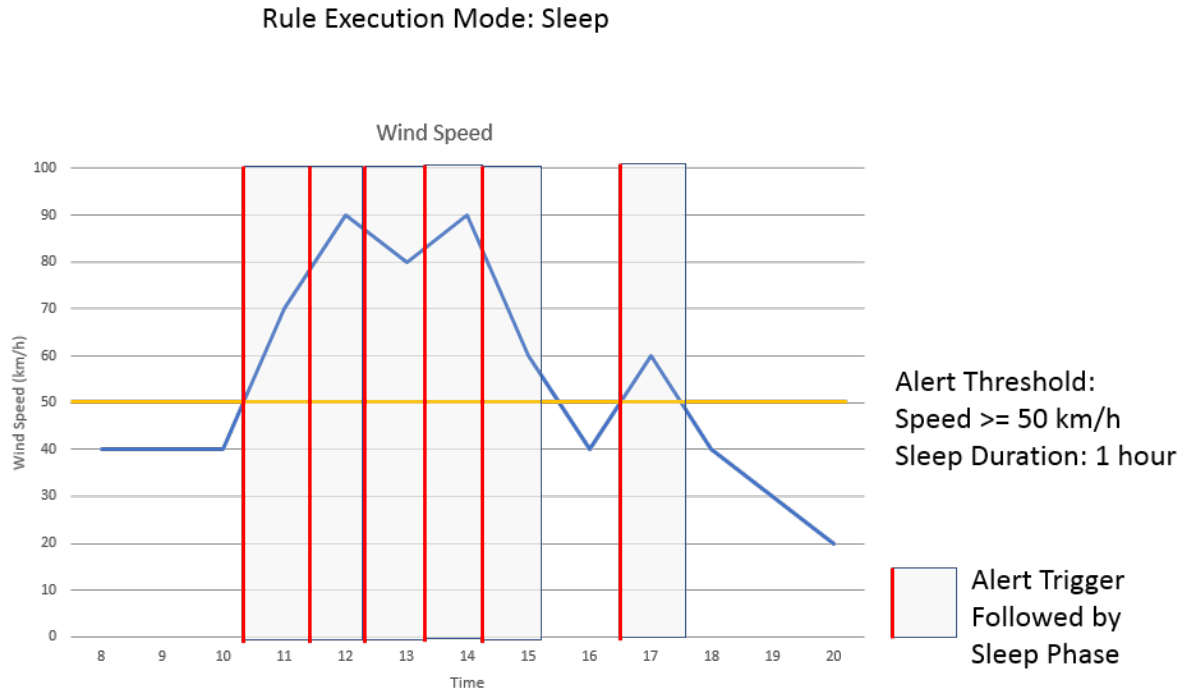
- Sleep Mode: A mode of rule execution in IoT scenarios where a rule stops triggering events for a specified duration even if the rule condition is fulfilled. This execution mode helps to prevent event flooding in case a specified threshold value is constantly exceeded but does not require an action for each individual occurrence.
- Toggle Mode: A mode of rule execution in IoT scenarios where a rule only triggers an event if the rule condition is fulfilled and the condition was not fulfilled for previously tested values. This execution mode helps to prevent event flooding in case a specified threshold value is constantly exceeded.

For the "high wind speed/open windows" scenario described in the example, toggle mode would be the desired setting: With that, the rule that is monitoring the sensor data would trigger exactly one alert after the wind speed exceeds the threshold value of 50 km/h and then remain silent for the entire rest of the day. Only if the

wind speed would rise again after having calmed down to less than 50 km/h, the rule might send a second alert. The following diagram illustrates the effect:



In contrast to toggle mode, see the effect of the same rule in sleep mode with the longest possible sleep phase of one hour after the rule condition was fulfilled:



As you can see, running the rule in toggle mode results in only two alerts, whereas in sleep mode, the rule triggers at least six alerts for the same set of time series data. However, even six alerts shouldn't be too annoying for the person in charge compared with approximately 400 alerts that the system would send based on a really moderate scan rate of just one measurement per minute.

Time stamps and Processing Sequence

Each single measured value comes along with an exact timestamp that indicates when the respective sensor has recorded that value.

> **i** Note
>
> For the event reduction modes, the following applies:
>
> - Sleep mode as well as toggle mode are supported for streaming rules, that is, for rules dealing with discrete, individual measured values. For scheduled rules dealing with aggregated data, only toggle mode is supported.
> - Sleep mode and toggle mode are mutually exclusive.
> - Due to delays and data point sequence reordering caused by the involved software layers as well as hardware behavior, SAP cannot guarantee that the ingested data is always processed in exactly the same order as the values have been originally measured by the sensors. This can lead to unexpected effects during rule processing. For further information, see the processing examples given for the different rule modes in the *Details* section.

Details

Processing Time versus Business Time

For rule processing, the system makes a differentiation between different time horizons, processing time and business time. These terms have the following meaning:

- Business time: The time at which a particular value has been measured by a device sensor. Each sensor takes care of providing a time stamp for each recorded measurement, which is a prerequisite for any subsequent time series analysis.
- Processing time: The time at which a rule is processing the sensor data. In SAP Leonardo IoT, rule processing of sensor data is never a real-time process due to various reasons (e.g., delays in the underlying ingestion pipeline, communication channel latencies, longer sensor synchronization intervals for power saving, etc.). The minimum delay that you can expect on average is approximately ten seconds. E.g., a temperature that has been measured at 09:10:25 can be processed by a rule at 09:10:35 or later.

> **i** Note
>
> For rule processing in sleep mode as well as in toggle mode, the processing time at which the rule is executed takes precedence over the business time at which a particular value has been measured. In case of potential delays during the data ingestion process, this can lead to unexpected results. For details, see the examples for the two different execution modes.

Persistence period

For streaming rules, a special mechanism is in place called "persistence period": This feature serves the following purpose:

For many use cases, it may be sufficient to pass on the recorded time series data to the rule processing unit in bigger chunks (e.g., once per day), rather than continuously listening to the current stream of data. For each measured value that fulfills the rule condition, the system keeps track of the end of the sleeping time according to the rule settings (e.g., measurement timestamp plus 45 minutes). This information on the expiry of the sleep time is stored in the system for the length of the persistence period. The persistence period is a hard-coded value of 36 hours. The value of 36 hours has been chosen to ensure that it is safe for customers to follow a 24 hours cycle for rule processing without running into situations where measurements with a timestamp near the edge of the 24 hours range could be lost.

Also, as already mentioned, it is not unusual for an IoT scenario that the original data sequence is disturbed or delayed for various reasons that are not always predictable. As a consequence, it could happen that data reaches the ingestion pipeline with massive delays so that it wouldn't make much sense for a rule to trigger any alerts based on the delayed data. To avoid such useless rule processing on outdated data, the system dismisses any data if the measurement time stamps are older than the predefined persistence period for data.

As an intentional and desired side effect, dismissing data older than the persistence period also serves the purpose of an implicit cleanup of the data ingestion queue. Otherwise, it could happen that over time, chunks of delayed data mess up the ingestion queue and impose unnecessary load on the storage system.

> **i** Note
>
> The fixed duration of the persistence period (currently 36 hours) may be subject to change in future releases.

Sleep Mode

A rule in sleep mode behaves in the following way:

As soon as the rule condition is fulfilled, the rule triggers an event as any other rule would do. However, after triggering the event, the rule stops responding to any further occurrences of fulfilled rule conditions for a specified time frame. That is, although the rule is still active, it ceases triggering events based on its settings. Once the specified time is over, the rule waits for the next fulfillment of its condition, triggers the event, and falls asleep again, and so on.

For rules in sleep mode, the following limitations apply:

- Sleep time: minimum 10 seconds, maximum one hour
- Persistence period: 36 hours

In the following table, you can find an example of how the system responds to measured values that the rule processes in a sequence that deviates from the measurement time stamps:

Rule Processing in Sleep Mode (sleep time: 1 minute)

Processing Time	Business Time	Condition fulfilled	Event Triggered
Mo 14:51:00	Mo 14:50:30	True	True
Mo 14:51:20	Mo 14:50:50	True	False (sleeping)
Mo 14:51:50	Mo 14:51:40	True	True

Processing Time	Business Time	Condition fulfilled	Event Triggered
Mo 14:51:55	**Mo 14:49:40**	True	False (data with newer business time 14:51:40 already processed)
Th 15:00:00	Mo 14:52:30	True	True (would be false if processing time would be before Wednesday, 02:51:40)

> **i** Note
>
> Time stamps deviating from the normal sequence are given in **emphasized** typeface.

Toggle Mode

A rule in toggle mode behaves in the following way:

As soon as the rule condition is fulfilled, the rule triggers an event as any other rule would do. However, after triggering the event, the rule stops responding to any further occurrences of values that fulfill the condition unless a value has been determined that does **not** fulfill the condition. After that, the next value that fulfills the condition triggers the event, and so on.

In the following table, you can find an example of how the system responds to measured values that the rule processes in a sequence that deviates from the measurement time stamps:

Rule Processing in Toggle Mode

Processing Time	Business Time	Condition fulfilled	Event Triggered	Comment
Mo 14:51:00	Mo 14:50:30	True	True	
Mo 14:51:20	Mo 14:50:50	True	False	no event creation for subsequent values matching condition
Mo 14:51:50	Mo 14:51:40	False	False	next matching value triggers event
Mo 14:51:55	**Mo 14:49:40**	True	False	value matches condition, but business time (Mo 14:49:40) is earlier than previous ingestion (Mo 14:51:40)
Mo 14:52:50	Mo 14:52:40	True	True	
Mo 14:52:55	Mo 14:52.42	True	False	
Mo 14:53:56	**Mo 14:52.41**	False	False	condition not fulfilled
Th 15:00:00	Mo 14:58.30	True	True	would be false if processing time would be before We 02:53:36

> **i** Note
>
> Time stamps deviating from the normal sequence are given in **emphasized** typeface.

Related Information

Rule Modeling: Overview [page 15]

4 Actions and Notifications: Overview

General information on setting up actions and notifications for rule-based thing monitoring

Introduction

In SAP Leonardo IoT, actions can be used for the following purposes:

- Action as receiver of an event triggered by a rule: In the definition of a rule, an action can be specified that shall be triggered if the rule condition is fulfilled.
- Action as a decision support option: In the definition of a recommendation to solve a particular business situation, one or more actions can be assigned as possible alternatives for the person in charge to solve the situation.

During rule definition, you take care of different system conditions and situations you want to react to. Rule modeling, thus, is dealing with setting up a formal model that reflects the occurrence of such system events and statuses.

However, a rule alone does not specify what shall happen in response to a rule condition that has been fulfilled. To accomplish this, you set up Actions and associate them with a rule that shall be used as a trigger for these actions.

Likewise, in a decision support scenario where the system cannot automatically determine which action to take, you specify various actions from which a user may choose the one that is likely to solve a given business situation best.

> **i** Note
>
> The maximum frequency for the system to trigger an event for a given thing monitored by a streaming rule is one event per 10 seconds. In other words, no more than six events per minute can be triggered.
>
> However, should your use case require a higher rate of events, we encourage you to get in touch with SAP to see if the threshold can be adjusted to your individual needs.

Related Information

Rules: Overview [page 3]
Decision Support: Overview

4.1 Create an Action

Process steps for creating an action

Context

An action can be triggered by a rule or an existing action. For example, you might create an action to generate a sales order or a service ticket when certain rule conditions are met. Likewise, an action can be triggered by an existing action. For example, you might get an e-mail notification informing about the newly generated sales order.

Actions can perform a variety of activities, like executing HTTP-based APIs, sending emails or creating notifications that can be displayed in Fiori Launchpads.

> ⚠ Caution
>
> When calling service methods with an action, you should avoid using the unsecure HTTP protocol. Instead, to ensure secure data transmission, SAP highly recommends using the HTTPS protocol for accessing the service methods by an action.

It depends on the selected Action type as to which activity should be performed.

Dynamic Content/Action Vocabulary

Actions support dynamic content. Most of the text input fields in the Action UI support placeholder tokens whose values are being replaced at run-time with actual values.

For example, an e-mail subject may contain the token **${thing.name}**, which at the action processing time, is being replaced with the actual name of the Thing that is in scope of this action.

Supported sources of values are:

- Thing Information
  - Thing ID, External ID, Name, Description, and so on
- Thing Basic Data
  - Properties that appear under the *Basic Data* tab in the Thing Modeler UI
- Thing Measured Values
  - Properties that appear under the *Measured Values* tab in the Thing Modeler UI
- Custom Master Data associated to the Thing
  - Data associated to a thing via Custom Master Data API
    See Custom Master Data for details on how to create and associate Custom Master Data to a thing type and thing.
- HTTPS API responses of other actions
  - Selected elements of HTTPS API responses from other actions can be made available to the current action.
- Other Information
  - Action name, Rule ID, current time, and so on

The availability of tokens in a particular action depends on the definition of the thing type associated with the Action.

A list of available tokens appears by entering **${** followed by any character in a dynamic content enabled text field. For example, typing **${t** opens a popup with the first 10 tokens containing a **t**.

> **i** Note
>
> Since there could be more than 10 tokens containing a small number of characters, it is advised to type as many characters as known.

Procedure

1. On the launchpad, select the *Actions* tile.
2. On the *Actions* page, choose *New* to create a new action.
3. Enter the name of the action in the *Name* field.
4. Enter the description of the action in the *Description* field.
5. In the *Trigger by* field, select `Action` or `Rule`.
6. Depending on your decision in the previous step, proceed as follows:

Trigger by	User Action
Event from Action	Select the desired action from the *Action* dropdown list.
Event from Rule	Select the desired rule from the *Rule* dropdown list.

Based on the selected *Action* or *Rule*, the system populates the *Thing Type* list.

7. Select the *Action Type*.
   a. *Action Type*: Service Integration

Field	Description
Action Type	To configure the destinations, choose *Service Integration*.
Destination	Choose the destination for the action. > **i** Note > The *Destination* field shows the *name* of the destinations that have been preconfigured in the Destination Service. For more information, see Service Destinations [page 39].
URL	The URL is displayed based on the destination selected.
Invocation Type	Choose the applicable option: ○ *Manual*: When the action is invoked, it remains in *Pending* status until it is executed manually. ○ *Auto*: When the action is invoked, it is executed immediately.

Field	Description
Method	Select the HTTP method: *GET*, *POST*, or *PUT*.
Request body	Enter the payload required by the service method. Payload templates are available for the SAP CPI iFlows that are available with SAP Leonardo IoT. **i Note** This field supports dynamic content. **i Note** You must customize the request body once you have inserted the payload template. **i Note** The maximum size allowed for the HTTP request body is 5000 characters.
Trigger subsequent event	Enable this checkbox if the current action should generate an event that could trigger other actions. With this option enabled, the system displays the *Response Payload Type* dropdown list.
Response Payload Type	Select the required response payload type: ○ *XML* ○ *JSON* This selection is needed in case the content of the service method response for the current action should be available as dynamic content in other actions.

Field	Description

Event Parameters	Define elements of the service method response as dynamic content that is available in other actions.
	
	Choose *New* to add an event parameter. Enter information as follows:
	○ *Name*: Enter a name to identify the parameter.
	○ *Response Payload Path*: Add the response payload path.
	
	In the following example, the event parameter with the response payload path `/RETURN/item/MESSAGE_V1` and name PREQ_NUMBER would make the number of the purchase requisition (0010000338) available as input parameter with the name PREQ_NUMBER of a subsequent action.
	
	`<RETURN>`
	`  <item>`
	`  <TYPE>I</TYPE>`
	`  <CODE>06402</CODE>`
	`  <MESSAGE>Purchase requisition number 0010000338 created</MESSAGE>`
	`  <LOG_NO/>`
	`  <LOG_MSG_NO>000000</LOG_MSG_NO>`
	`  <MESSAGE_V1>0010000338</MESSAGE_V1>`
	`  <MESSAGE_V2/>`
	`  <MESSAGE_V3/>`
	`  <MESSAGE_V4/>`
	`  </item>`
	`</RETURN>`
	
	Choose *Save*.

b. *Action Type*: Select *Email Notification*.

Field	Description
Action Type	Select *Email Notification* to create an action for which you can configure an e-mail.
Recipients	Enter the e-mail address of the recipient. You can enter multiple addresses, separated by semicolons. **i Note** This field supports dynamic content.
Subject	Enter the subject of the mail. **i Note** This field supports dynamic content.

Field	Description
Text	Enter the text of the mail. **i Note** This field supports dynamic content. **i Note** The maximum size allowed for the mail body is 5000 characters.

c. *Action Type*: Select *In-App Notification*.

Field	Description
Action Type	Select *In-App Notification* to create an action for which you can configure a notification.
Recipients	Enter the SAP Leonardo IoT login name of the recipient. You can enter multiple recipients, separated by semicolons. **i Note** This field supports dynamic content.
Text	Enter the text of the notification message. **i Note** This field supports dynamic content. **i Note** The maximum size allowed for the In-App body is 5000 characters.

In-App Notifications can be configured to be actionable. When you click the In-App Notifications in the Fiori Launchpad, the browser will navigate to the target object and target action and pass on the target parameters defined in the Action Modeler UI.

Field	Description
Target Object	The navigation target object (example, Purchase Order Application)
Target Action	The action of the navigation target object (example, View)
Target Parameters	Define parameters that are passed along to the navigation target object and action (for example, the ID of the purchase order that should be viewed)

d. *Action Type*: Decision Support

Field	Description
Action Type	To configure the destinations, choose *Decision Support*.
Alias	To link the decision support information to the action, enter a name in the *Alias* field. **i Note** Make sure that the name in the *Alias* field matches with the name in the *Action* field.
Payload	Enter the payload required by the service method. Payload templates are available for the SAP CPI iFlows that are available with SAP Leonardo IoT. **i Note** This field supports dynamic content. **i Note** You must customize the request body once you have inserted the payload template. **i Note** The maximum size allowed for the HTTP request body is 5000 characters.

e. *Action Type*: Enterprise Messaging System

The Enterprise Messaging System (EMS) allows you to specify the message payload to be created and also lets you choose the message channel, that is, Queues or Topics. For more information on Enterprise Messaging System (ESM), see What is Enterprise Messaging and Initial Setup respectively.

For more information on Messaging Channels, see Messaging Concepts.

Field	Description
Action Type	To configure the destinations, choose *Enterprise Messaging System*.
Destination	Choose the destination for the action. **i Note** The *Destination* field shows the *name* of the destinations that have been preconfigured in the Destination Service. For more information, see Service Destinations [page 39].

Field	Description
Message Channel	Choose the Message Channel option: ○ *Topics*: When the action is invoked, the service enables a sending application to publish the message to a topic. ○ *Queues*: When the action is invoked, the service enables a sending application to publish the message to a specific queue.
Topic Name or *Queue Name*	Depending on your choice in the previous step, proceed as follows: **Trigger by** / **Action** *Topic Name* / Enter a name for the topic. *Queue Name* / Enter a name for the queue.
Message	Enter the message for the action. **i Note** This field supports dynamic content. **i Note** The maximum size allowed for the In-App body is 5000 characters.

8. Choose *Save*.

Results

The new action is created.

Related Information

Service Destinations [page 39]

4.1.1 Service Destinations

Actions of type Service Integration use the destinations that are defined in the SAP Cloud Platform Destination Service.

Those destinations define how to access the endpoints (for example, SAP systems) which are the targets of those actions. Hence, actions of type Service Integration must have one of these destinations assigned.

Creating Destinations

Destinations can be created in the following ways:

- Via the Destination Service UI of SAP Cloud Platform Cockpit: In the cloud platform cockpit, within the menu on the left, *Connectivity* → *Destinations* → *New Destination* (certain roles and permissions are needed to create a destination from the UI).
- Via the Destination Service API provided by SAP Cloud Platform.

Supported Destination Type and Authentication Mechanisms

Destinations have to be of type HTTP and support multiple, but not all, authentication types. Supported authentication types are:

- NoAuthentication
- BasicAuthentication
- OAuth2ClientCredentials (Client Credentials and Password grant types are supported.)

Supported Destination Proxy Types

Destinations can describe Cloud systems (reached via public internet) or On-Premise systems (reached via SAP Cloud Connector). Depending on the location of the destination, the proxy type has to be defined accordingly:

- Internet
- On-premise

Supported Destination URLs

Destinations of type HTTP need to have a URL. The URL can have tokens, which are replaced with actual values during the execution of an action. These tokens are optional.

Since destinations are defined outside of the Action UI, there is no value help available for those tokens. The token format is still the same though: **${token.name}**.

An example with the thing ID, name, and type tokens in a URL looks like this: `https://www.sap.com/do/${thing.type}/something?thingid=${thing.id}&thingname=${thing.name}`

Destinations with HTTP Headers

HTTP Headers can be defined using (additional) destination properties. In the destination service UI, a property can be added by clicking the *New Property* button; the left field being the property name, and the right being the property value. Properties can also be defined via the destination service API.

To be used as an HTTP Header, the property must have the prefix **sap.iot.header.** (including the last dot) followed by the name of the actual header variable.

Example: sap.iot.header.**X-Client-Version** or sap.iot.header.**X-Client-ID**

Destinations for Endpoints Which Require a CSRF Token

SAP systems often require a CSRF (cross-site request forgery) token.

This protection can often be disabled, but in case this is not appropriate or possible, the destination definition can be enhanced to indicate that a CSRF token needs to be retrieved and added to the request.

The two properties needed for that are:

- **sap.iot.fetchXcsrf** with the value **true**

- **sap.iot.XcsrfURL** with the value of the URL, which returns the CSRF token. This is often the same URL as the actual URL of the endpoint.

Destinations for Endpoints with a Self-signed Certificate

Although not considered secure, there might be a need to use the action framework to connect to systems that are using self-signed certificates (for example, demo or development systems).

The property **sap.iot.certificationEnabled** with the value **false** can be used in a destination to indicate that the certificate check does **not** apply for this destination.

4.2 Maintain an Action

Process steps for modifying an action

Context

This feature allows you to update or modify an existing action.

> **i** Note
>
> For more detailed information on the various settings of an action, see Create an Action [page 33].

Procedure

1. On the launchpad, select the *Actions* tile.
2. On the *Actions* page, click the link of the action to be modified, to navigate to the Action detail page.
3. Change the content of the field on the selected action page.
4. Select the required options from the *Trigger by* and *Action Type* fields.
5. Enter the updated details in the HTTP, Email, In-App Information, Decision Support Information, or Enterprise Messaging System Information on the selected *Action Type*.
6. Choose *Save*.

Results

The changes are applied to the selected action.

4.3 Delete an Action

Process steps for deleting an action

Context

This feature allows you to delete an existing action.

> **i** Note
>
> One or more actions can be selected and deleted.

Procedure

1. On the launchpad, select the *Actions* tile.
2. On the *Actions* page, enable the check box against the action to be deleted.
3. Choose *Delete*.

Results

The selected action is deleted.

4.4 Search for an Action

Process steps for searching an action

Context

This feature allows you to search for a specific action. In addition to obvious search criteria such as *Name* or *Type* of an action, you can also filter the list for actions that are associated with a particular *Thing Type* or that are *Triggered By* a particular action or rule.

Procedure

1. On the launchpad, select the *Actions* tile.
2. On the *Actions* page, enter the name of the Action you want to search for, or enter the applicable search string.
3. Select *Service Integration*, *Email Notification* or *In-App Notification* from the *Action Type* dropdown menu.
4. Select the applicable rule or action from the *Triggered By* dropdown menu.
5. Choose *Go*.

   The page displays all the actions that match the search criteria.
6. Choose *Clear* to clear all the search strings and search results.
   The system then repopulates the list of actions with all available actions, and you may start a new search.

4.5 Notification and Email Setup

This feature allows you to configure the mail server and to set up notifications and email alerts.

Before you can start sending email alerts, you first have to provide the necessary information about your email infrastructure (such as server names and ports to be used). After that, you can define actions of type *Email Notification*.

> **i** Note
>
> After having sent an email, the system waits for a response from the receiving mail server. If the receiving mail server does **not** send a response code indicating successful delivery, the notification service resends the mail. However, attempts to resend the mail are made only within the first three minutes after the first sending attempt. After that, the mail is considered as undeliverable.

4.5.1 Set up a Notification Service

Process steps for configuring a new mail server

Context

This feature allows you to set up a new mail server configuration.

> **i** Note
>
> The mail server has to be configured first, before an email can be sent.

Procedure

1. To set up a new mail server configuration, choose *Mail Server Configuration*.
2. Enter the *SMTP Server Host* address.
3. Enter the *SMTP Server Port* detail.
4. Enter the *SMTP Server Authentication User* name.
5. Enter the *SMTP Server Authentication Password*.
6. Enter the email address of the notification sender in the *Notification Sender Email Address* field.
7. Enter the default subject of the notification in the *Default Notification Address* field.
8. Choose *Save*.

4.6 Delete Personal Data

In order to delete personal data, you have to use the list page to locate the actions with type EMAIL or INAPP. Open those actions and remove the e-mail address (EMAIL Action) or the Cloud Foundry login name (INAPP) of the person whose data needs to be deleted from the list of recipients.

> **i** Note
>
> Since the input fields for both e-mail address and login name are validated by the system, you cannot simply delete existing personal data from these fields because the system wouldn't allow saving an action with invalid or missing data in a mandatory field. Instead, enter some phantasy data that matches the formal requirements for the field in question (e.g., you might enter `some.person@customer.com` as a dummy e-mail address and then save the action).

4.7 Action-Related Error Codes

Overview of all types of errors that can occur with action processing

In this section, you can inform yourself about the different types of errors that can occur when the system processes an action. Note that many of these error types can only occur when you are programming against the respective API services. As opposed to that, when you define actions via the Action Modeler apps delivered by SAP, these apps take care of the necessary proper format settings or the allowed values for certain properties of an action.

> **i** Note
>
> When looking at the list of error codes, don't get confused about any of the following observations:
>
> - The list of error codes is **not** a gapless sequence of numbers. This is with intent and **not** a mistake.
> - Several errors are very similar or even identical. However, the unique error code of each error helps our support experts to determine the source code location where the error occurred.

- For a number of error situations, there is no way for customers to correct the situation. For these cases, the suggested solution is "Create a support ticket." The proper component for all errors listed here is **IOT-BSV-ACT**.

Action-Related Error Codes

Error Code	Category	Root Cause	Solution
iot.action.error.1	Invocation Type Missing	An action requires an invocation type, which can either be "AUTO" or "MANUAL". An empty type is not allowed.	Specify the invocation type of the action.
iot.action.error.2	Wrong Invocation Type	An invocation type other than "AUTO" or "MANUAL" has been specified for an action.	Make sure that the invocation type is correctly specified.
iot.action.error.3	Action Processing Failed	n/a	Create a support ticket.
iot.action.error.4	Action Processing Failed	Trigger event for the action could not be processed.	Create a support ticket.
iot.action.error.6	Action Type Wrong or Missing	An unsupported action type has been specified, or no action type at all. This is not allowed.	Make sure that the action type is correctly specified as one of the following: **INAPP**, **EMAIL**, **HTTP**.
iot.action.error.7	Runtime Exception	Error occurred while processing an action.	Create a support ticket.
iot.action.error.8	Runtime Exception	Action or event has been deleted while the action was executed.	Create a support ticket.
iot.action.error.16	User Error	Recipient missing for e-mail action	Provide a valid e-mail address for the mail recipient.
iot.action.error.17	User Error	Subject missing for e-mail action	Provide a subject for the e-mail.
iot.action.error.18	User Error	Content missing for *Body* field of e-mail or in-app action.	Provide some content to help end users solve the situation.
iot.action.error.19	Runtime Exception/Notification Error	Exception occurred while sending a notification.	Create a support ticket.
iot.action.error.20	User Error	Request body of HTTP action is empty.	Provide a valid request body.
iot.action.error.21	Runtime Exception/Run ID Generation	Proper run ID could not be generated.	Create a support ticket.
iot.action.error.22	Runtime Exception/Run ID Generation	Error occurred while processing chained actions.	Create a support ticket.
iot.action.error.23	Runtime Exception/ Parameter Processing	Unable to find proper parameter sources for action processing.	Create a support ticket.
iot.action.error.24	Runtime Exception/ Parameter Processing	Unable to find proper parameter paths for action processing.	Create a support ticket.

Error Code	Category	Root Cause	Solution
`iot.action.error.25`	Runtime Exception/ Parameter Processing	Invalid combination of parameter path and source.	Create a support ticket.
`iot.action.error.26`	Runtime Exception/Notification Error	Sending of in-app notification failed.	Create a support ticket.
`iot.action.error.32`	Runtime Exception/PST Actions	Unable to find named property set for actions based on property set types (PST).	Create a support ticket.
`iot.action.error.33`	Runtime Exception/PST Actions	No thing ID found while processing the action.	Create a support ticket.
`iot.action.error.34`	Runtime Exception/PST Actions	Processing failed: Neither thing type nor property set type provided.	Create a support ticket.
`iot.action.error.35`	Runtime Exception/PST Actions	No proper parameters found for action based on property set type.	Create a support ticket.
`iot.action.error.36`	Runtime Exception/PST Actions	Action references both thing type and property set type. This is not allowed.	Create a support ticket.
`iot.action.error.48`	User Error/E-Mail Action	E-mail address missing.	Provide a valid e-mail address.
`iot.action.error.49`	User Error/E-Mail Action	Invalid e-mail address.	Provide a valid and well-formed e-mail address (e.g., **`name@company.com`**).
`iot.action.error.50`	Runtime Exception	Illegal access exception occurred while extracting token.	Create a support ticket.
`iot.action.error.53`	Runtime Exception/ Metadata	Retrieving metadata value for a thing failed.	Create a support ticket.
`iot.action.error.54`	Runtime Exception/Metadata	Metadata value returned for a thing was null.	Create a support ticket.
`iot.action.error.55`	Runtime Exception/ Parameter Processing	No parameter found for the given source to process the actions. Note: Parameters are the tokens that you type while creating an action while source refers to the parameter type (e.g., `TS` for time series).	Create a support ticket.
`iot.action.error.56`	User Error/Master Data	Error occurred while processing business partner data.	Provide proper business partner values. For details, see Modeling Custom Master Data Overview.
`iot.action.error.57`	User Error/Master Data	Unknown parameters for business partner.	Provide proper business partner values and parameters. For details, see Modeling Custom Master Data Overview.

Error Code	Category	Root Cause	Solution
`iot.action.error.59`	User Error/Master Data	The provided subtype is not supported for given master data type.	Provide proper subtype for given master data type. For details, see Modeling Custom Master Data Overview.
`iot.action.error.60`	User Error/Master Data	Custom master data property `Name` not found.	Check master data for correctness. If you can't find any error, create a support ticket.
`iot.action.error.61`	Runtime Exception/ Master Data	Runtime exception occurred while processing master data properties.	Create a support ticket.
`iot.action.error.62`	Runtime Exception/ Master Data	Invalid values returned while retrieving master data associations for a thing.	Create a support ticket.
`iot.action.error.63`	Runtime Exception/ Master Data	No value returned while retrieving master data associations for a thing.	Create a support ticket.
`iot.action.error.64`	User Error/Master Data	Business partner instance found without ID.	Provide proper master data values. For details, see Modeling Custom Master Data Overview.
`iot.action.error.65`	User Error/Master Data	No business partner instance found from master data associations for business partner.	Provide proper master data values. For details, see Modeling Custom Master Data Overview.
`iot.action.error.67`	Runtime Exception/ Master Data	Error parsing master data association values.	Create a support ticket.
`iot.action.error.68`	Runtime Exception/ Master Data	Problem detected in validating associations for master data.	Create a support ticket.
`iot.action.error.69`	Runtime Exception/ Master Data	Error retrieving the detailed master data values.	Check master data associations for correctness. If you can't find any error, create a support ticket.
`iot.action.error.70`	User Error/Master Data	Parameters not valid for master data source type.	Check that master data is properly defined.
`iot.action.error.71`	User Error/Master Data	Retrieved empty object instance association for master data.	Check that object type for master data is properly defined.
`iot.action.error.72`	Runtime Exception/ Master Data	No values returned by master data APIs	Create a support ticket.
`iot.action.error.73`	Runtime Exception/ Master Data	Empty object returned by master data APIs.	Create a support ticket.
`iot.action.error.74`	Runtime Exception/ Master Data	The metadata values for master data properties are empty.	Create a support ticket.
`iot.action.error.75`	Runtime Exception/ Master Data	Exception occurred while parsing the JSON response for master data properties.	Create a support ticket.

Error Code	Category	Root Cause	Solution
iot.action.error.76	User Error/Master Data Notification	No non-token recipient and no instance object found for master data.	Check that the master data instance objects are defined properly.
iot.action.error.77	User Error/Master Data Notification	No non-token recipient and master data recipients found.	Check that the master data recipients are defined properly.
iot.action.error.78	Runtime Exception/Database	Required fields to save action result not found.	Create a support ticket.
iot.action.error.79	Runtime Exception/Database	Action results not saved due to database error.	Create a support ticket.
iot.action.error.80	Runtime Exception/Action Processing	Message broker fails to send events for the action.	Create a support ticket.
iot.action.error.83	Runtime Exception/ Master Data	Error retrieving values from business partner URL.	Create a support ticket.
iot.action.error.84	User Error/Master Data	Business partner not defined.	Check that master data is properly defined.
iot.action.error.85	Runtime Exception/Action Processing	Error retrieving value from snapshot API.	Wait for retry. If the action still fails, create a support ticket.
iot.action.error.86	Runtime Exception/ Master Data	Error parsing JSON data for association.	Create a support ticket.
iot.action.error.87	Runtime Exception/ Master Data	Data association URL is empty.	Create a support ticket.
iot.action.error.100	User Error/Destination	Destination details missing or invalid.	Ask your system administrator.
iot.action.error.101	User Error/Destination	Destination URL missing.	Ask your system administrator.
iot.action.error.102	User Error/Destination	Destination authentication missing.	Ask your system administrator.
iot.action.error.103	User Error/Destination	Destination user missing.	Ask your system administrator.
iot.action.error.104	User Error/Destination	Destination password missing.	Ask your system administrator.
iot.action.error.105	User Error/Destination	Destination client ID missing.	Ask your system administrator.
iot.action.error.106	User Error/Destination	Destination client secret missing.	Ask your system administrator.
iot.action.error.107	User Error/Destination	Token service URL missing for destination.	Ask your system administrator.
iot.action.error.110	User Error/Destination	Token service user missing for destination.	Ask your system administrator.

Error Code	Category	Root Cause	Solution
`iot.action.error.111`	User Error/Destination	Token service password missing for destination.	Ask your system administrator.
`iot.action.error.112`	User Error/Destination	Unsupported authentication type for destination.	Ask your system administrator.
`iot.action.error.113`	User Error/Master Data	Retrieving master data association failed: Thing ID missing.	Provide the thing ID.
`iot.action.error.114`	User Error/Master Data	No proper composite master data URL found.	Provide the correct thing ID.
`iot.action.error.115`	User Error/Master Data	Business partner instance found without business partner ID.	Provide a correct business partner ID.
`iot.action.error.116`	User Error/Master Data	No basic data found for business partner.	Provide a correct business partner ID.
`iot.action.error.117`	User Error/Master Data	Business partner application name is null.	Provide a correct business partner ID.
`iot.action.error.118`	User Error/Master Data	Business partner ID is null or empty.	Provide a correct business partner ID.
`iot.action.error.120`	Runtime Exception/Acion Processing	Error retrieving value for time series parameters; service may be down.	Wait for retry. If the action still fails, create a support ticket.
`iot.action.error.121`	Runtime Exception/Acion Processing	Call to external endpoint failed.	Wait for retry. If the action still fails, create a support ticket.
`iot.action.error.122`	User Error/Action Processing	HTTP method is null.	Provide a proper HTTP method (e.g., `GET` or `POST`).
`iot.action.error.123`	User Error/Action Processing	Body of HTTP GET request contains data.	Check the request (payloads for GET requests cannot be processed).
`iot.action.error.125`	User Error/Action Processing	No response parameter found for HTTP call.	Define proper response parameter for chained action. See Create an Action [page 33].
`iot.action.error.126`	User Error/Action Processing	Response document either missing or of wrong type.	Make sure that the response type is either JSON or XML.
`iot.action.error.127`	Internal Error/Action Processing	Parameters not found for processing actions for a particular source.	Create a support ticket.
`iot.action.error.128`	User Error/Chained Action Processing	No saved result found for last action run.	Check the definition of the chained action.
`iot.action.error.129`	User Error/Chained Action Processing	Saved result for last action run is empty or null.	Check the definition of the chained action.
`iot.action.error.146`	Internal Error/Event Processing	Action-based event without action reference.	Create a support ticket.

Error Code	Category	Root Cause	Solution
`iot.action.error.147`	Internal Error/Event Processing	Unexpected event type.	Create a support ticket.
`iot.action.error.148`	Internal Error/Event Processing	Event element "data → thing ID" missing or invalid.	Create a support ticket.
`iot.action.error.149`	Internal Error/Event Processing	Property set value missing for action based on property set type.	Create a support ticket.
`iot.action.error.150`	Internal Error/Action Processing	Invalid action type.	Create a support ticket.
`iot.action.error.151`	User Error/Action Processing	Invalid X-CSRF or ETag.	Provide a proper URL for X-CSRF or ETag.
`iot.action.error.152`	User Error/Action Processing	Failed to retrieve X-CSRF token.	Provide a proper URL for X-CSRF,
`iot.action.error.153`	User Error/Action Processing	Failed to retrieve ETag token.	Provide a proper URL for ETag.
`iot.action.error.154`	User Error/Action Processing	Failed to perform HTTP request with URL syntax.	Provide a proper URL in actions.
`iot.action.error.155`	User Error/Action Processing	HTTP call failed.	Create a support ticket.
`iot.action.error.156`	Internal Error/Action Processing	Call to custom master data API failed.	Create a support ticket.

5 Backend Integration Overview

Backend integration helps in creating and updating business objects in the SAP backend system so as to trigger it from SAP Leonardo IoT.

As a key user, you should be able to create a rule with an action that automatically triggers the creation or update of a business object in the SAP business system. The rule action should pass all the relevant device and master data parameters that are required to create or update the remote business object. This enables you to react on the alert notification which enables the "Insight to Action" scenario.

> **Example**
>
> The temperature is >18 and the cooler is powered on for more than 12 hours --> Create a service request once in the CRM or Cloud for Service.

Predefined and configurable iFlows are made available in this release, for the following business objects:

- Creation of Service Tickets (SAP C4C, SAP CRM)
- Creation of Sales Order
- Creation of Purchase Requisition

> **Note**
>
> In addition to the above list, the execution of HTTP-based APIs, for example, SAP CPI iFlow endpoints or customer backend systems are allowed.

The link between the SAP Leonardo IoT and the backend systems for these scenarios can be established through CPI I-Flows (or other such mechanisms). You should be able to use this mechanism/framework to enable the creation of business objects (irrespective of customizations).

Here, the creation of these business objects can be achieved through an action framework. You should have an API/UI component to define the business object (CPI I-Flow URL, authentication, payload, and so on).

The following parameters (not limited to) need to be configurable:

- The SAP Leonardo IoT tenant through which you want to perform the integration
- The backend system where the business object shall be created
- Payload and other parameters to be passed for the business object creation (depending on the customization at different customers, and so on)
- Different types of data that can get used (master data, time series data, thing metadata, and so on)
- Protocol followed (HTTP, and so on)
- Type of Call (GET, POST, PUT)
- Authentication
- Notification template

The data about the business object should be persisted and the creation should be logged (business object type, timestamp, associated device, rule associated with creation, created by). The persisted data should be available for consumption in the other parts of the application (KPIs, visualization, reports, and so on). For example, the number of service requests on device X

Auto and manual mode of action (business object creation) execution: Auto mode is to send the action right away when the rule condition is met, manual mode is to give the user the option to reconsider/adjust the action prior to sending it out.

A notification is triggered every time a business object is triggered. The notification should have the details about the business object that was created (business object number, and so on).

5.1 CPI iFlows and Setup

5.1.1 Pre-delivered CPI iFlows

This section explains the pre-delivered CPI iFlow templates for creating a service ticket, purchase order, and sales order.

The following Cloud Platform Integration (CPI) iFlows are delivered as templates for integration with SAP Leonardo IoT and are intended for reference only. As a customer, you must create your own iFlows based on your own system setup.

Post Service Notifications to C4C

This iFlow makes an OData service call to the SAP Cloud for Customer (C4C) to create a service request.

The iFlow is triggered from the action of a rules framework. This service accepts a payload from the action of a rules framework and calls an OData service in the C4C. The payload should be in a valid Service Notification Creation format (for example, the `ServiceRequestCollection`, `ServiceRequestDescriptionCollection`, or `ServiceRequestItemCollection` entity sets). This is a passthrough iFlow and the response from the C4C is passed directly back to the SAP Leonardo IoT. SAP Leonardo IoT consumes this to handle the downstream logging and processes. This iFlow service is called from an action configured in SAP Leonardo IoT.

Partner Directory Config Parameters Used

PID	Parameter
`CNG`	ServicePost (Value should be `ServiceNotificationPost`)
`ServiceNotificationPost`	URL (C4C URL including entity set)
`ServiceNotificationPost`	Credential Name for ECC in HCI

Where Used	In Action configuration
Source Systems	Cloud for Customer (C4C)
Adapters	http sender/receiver

Post Sales Order to ECC

This iFlow makes an OData service call to the SAP ERP Central Component (SAP ECC) gateway system to create a sales order in a SAP ECC back-end system.

The iFlow is triggered from the action of a rules framework. This iFlow service accepts a payload from the action of a rules framework and calls an OData service in the SAP ECC gateway system. The incoming payload should be mapped to the OData service. The gateway OData service uses `BAPI_SALESORDER_CREATEFROMDAT2`.

This iFlow has the following prerequisites:

- Cloud Connector and RFC destination in SAP Cloud Platform must be configured.
- **The gateway service must be configured/developed in a gateway system connected to the back-end ECC system where sales orders are created.** This gateway service must map the inbound fields to the SAP ECC BAPI input structures and should map the outbound structures and return table.

This iFlow service is called from an action configured in SAP Leonardo IoT.

Partner Directory Config Parameters Used

PID	Parameter
`CNG`	SalesOrderPost
`SalesOrderPost`	URL (Gateway service URL)
`SalesOrderPost`	Query
`SalesOrderPost`	Credential Name for SAP ECC in HANA Cloud Integration (HCI)

Where Used	In Action configuration
Calls	None
Source Systems	ECC (through a gateway service)
Adapters	http sender/receiver
Prerequisite	An OData service using `BAPI_SALESORDER_CREATEFROMDAT2` must exist in the gateway server connected to the ECC

Post Consignment Sales Order to ECC

This iFlow makes an RFC call using the CPI RFC sender adapter to create a Sales Order (SO) in an ECC back-end system. This iFlow is triggered from the action of a rules framework. The service accepts a payload from the action of a rules framework and calls `BAPI_SALESORDER_CREATEFROMDAT2` in the back-end ECC gateway system.

This is not strictly a consignment order creation iFlow. Depending on the value in the *Document Type* (DocType) field in the payload, different types of orders can be created, such as **`TA`** for a standard sales order or **`KB`** for consignment orders. This mapping step in the iFlow maps the inbound fields in the payload to the ECC BAPI

input structures. Depending on the business scenario, applicable fields from the payload should be mapped to the relevant ECC BAPI input structures to ensure that the back-end order has all the required attributes.

The `BAPIRETURN` table is returned from the back end to the sender. This iFlow service is called from an action configured in SAP Leonardo IoT.

This iFlow has the following prerequisite:

- Cloud Connector and RFC destination in HCP must be configured.

Partner Directory Config Parameters Used

Externalized Parameter	Parameter
RFC Destination	{{Receiver_destination_0}}

Where Used	In Action configuration
Calls	None
Source Systems	ECC
Adapters	http sender/RFC receiver

Post Purchase Requisition to ECC

This iFlow makes an RFC call using the CPI RFC sender adapter to create a purchase requisition in an ECC back-end system. This iFlow is triggered from the action of a rules framework. The service accepts a payload from the action of a rules framework and calls `BAPI_REQUISITION_CREATE` in the back-end ECC gateway system.

This mapping step in the iFlow maps the inbound fields in the payload to the ECC BAPI input structures. Depending on the business scenario, the applicable fields from the payload can be mapped to the relevant ECC BAPI input structures to ensure that the back-end order has all the required attributes.

The `BAPIRETURN` table is returned from the back end to the sender. This iFlow service is called from an action configured in SAP Leonardo IoT.

This iFlow has the following prerequisites:

- Cloud Connector and RFC destination in HCP must be configured.

Partner Directory Config Parameters Used

Externalized Parameter	Parameter
RFC Destination	{{Receiver_destination_0}}

Where Used	In Action configuration
Calls	None
Source Systems	ECC
Adapters	http sender/RFC receiver

Post Purchase Order to ECC

This iFlow makes an RFC call using the CPI RFC sender adapter to create a purchase order in an ECC back-end system. This iFlow is triggered from the action of a rules framework. The service accepts a payload from the action of a rules framework and calls `BAPI_PO_CREATE1` in the back-end ECC gateway system.

This mapping step in the iFlow maps the inbound fields in the payload to the ECC BAPI input structures. Depending on the business scenario, the applicable fields from the payload can be mapped to the relevant ECC BAPI input structures to ensure that the back-end order has all the required attributes.

The `BAPIRETURN` table is returned from the back end to the sender. This iFlow service is called from an action configured in SAP Leonardo IoT.

This iFlow has the following prerequisites:

- Cloud Connector and RFC destination in HCP must be configured.

Partner Directory Config Parameters Used

Externalized Parameter	Parameter
RFC Destination	{{Receiver_destination_0}}
Where Used	In Action configuration
Calls	None
Source Systems	ECC
Adapters	http sender/RFC receiver

5.2 Workflow Integration

Backend integration can also help trigger both SAP Cloud Platform workflows and SAP S/4 workflows from SAP Leonardo IoT. Workflows can be triggered in a similar way as iFlows. As a customer, you need to define the destination through the Cloud Platform Destination service and provide the corresponding payloads.

To trigger a *SAP Cloud Platform workflow*, you need to define the destination as shown in the following sections.

Destination Type and Authentication Mechanisms

Authentication Mechanisms

Authentication	Description
Authentication	OAuth2ClientCredentials
Client ID	`<CLIENT ID>` of your SAP Cloud Platform Workflow Service (CF)

Authentication	Description
Client Secret	`<CLIENT SECRET>` of your SAP Cloud Platform Workflow Service (CF)
Token Service URL	`https://<SUB DOMAIN of your tenant>.authentication.<YOUR REGION>.hana.ondemand.com/oauth/token`
Token Service User	`<SAP Cloud Platform user>` to initiate the workflow (user needs to have the required permissions)
Token Service Password	`<SAP Cloud Platform password of above user>`

Destination Proxy Type: Internet

Destination URL: `https://api.workflow-sap.cfapps.<REGION>.hana.ondemand.com/workflow-service/rest/v1/workflow-instances`.

> ### Example
>
> A sample URL can be https://api.workflow-sap.cfapps.eu10.hana.ondemand.com/workflow-service/rest/v1/workflow-instances. For more information about SAP Cloud Platform Workflow, please refer to https://api.sap.com/api/SAP_CP_Workflow_CF/resource.

Sample Payload

Format: **JSON**

A sample payload (request body) has the following structure. Make sure that you replace the value of the `definitionId` with your own workflow `Id`, and remove the context variable placeholder.

> ### Code Syntax
>
```
{
  "definitionId": "<WORKFLOW ID>",
  "context": {
<YOUR CONTEXT VARIABLES IN JSON FORMAT>
                      }
}
```

To trigger a *S/4 workflow*, user needs to define the destination as below:

Destination Type and Authentication Mechanisms

Authentication Mechanisms

Authentication	Description
Authentication	BasicAuthentication
User	`<S/4 HANA user>` to initiate the workflow (You need to have the required permissions)
Password	`<S/4 HANA user password>`

Destination Proxy Type: OnPremise for On-premise S/4 HANA systems

> **i** Note
>
> This also requires the configuration of a Cloud Connector

Destination URL: `http://<HOST NAME>:<PORT>/SAP/BC/WORKFLOW_XML/?~protocol=01&~localkey=<WORKFLOW ID>&sap-client=<CLIENT NUMBER>`.

Sample Payload

Format: **WF-XML**

A sample payload (request body) has the following structure. Make sure that you replace the value of the `<KEY>` with your own S/4 system and workflow `Id`.

> Code Syntax

```
<?xml version="1.0" ?>
<WfMessage Version="SAP.1.0" xmlns="http://www.wfmc.org/standards/doc/WF-XML">
    <WfMessageHeader>
        <Request>
            <ResponseRequired>Yes</ResponseRequired>
        </Request>
        <Key>https://mys4hana.sap.com:44300/SAP/BC/WORKFLOW_XML/?
~protocol=01&~localkey=WS02000061</Key>
        <Operation>CreateProcessInstance</Operation>
    </WfMessageHeader>
    <WfMessageBody>
        <CreateProcessInstance>
            <Key>https://mys4hana.sap.com:44300/SAP/BC/WORKFLOW_XML/?
~protocol=01&~localkey=WS02000061</Key>
            <ContextData>
                <YOUR_CONTEXT_VARIABLE 1></YOUR_CONTEXT_VARIABLE 1>
                <YOUR_CONTEXT_VARIABLE 2></YOUR_CONTEXT_VARIABLE 2>
            </ContextData>
```

```
            <StartImmediately>Yes</StartImmediately>
        </CreateProcessInstance>
    </WfMessageBody>
</WfMessage>
```

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

www.sap.com/contactsap

THE BEST RUN SAP