LISTEN.
THINK.
SOLVE.®

PharmaSuite®

DCS PHASES

RELEASE 8.4

USER MANUAL

PUBLICATION PSDC-UM001D-EN-E-DECEMBER-2017
Supersedes publication PSDC-UM001C-EN-E

Allen-Bradley · Rockwell Software

Rockwell Automation

Contents

Figures

Chapter 1

DCS Phases

The DCS phases of PharmaSuite represent a collection of phases for communicating with a Distributed Control System (DCS). They provide functions for creating a batch on a DCS, for retrieving batch values and alarms from the DCS, and for configuring alarms to be retrieved at regular intervals.

The following phases are available:

- Create DCS Batch (page 13)
- Get DCS Batch Values (page 29)
- Get DCS Alarms (page 45)
- DCS Alarm-based Trigger (page 55)

Typographical Conventions

This documentation uses typographical conventions to enhance the readability of the information it presents. The following kinds of formatting indicate specific information:

Bold typeface	Designates user interface texts, such as ■ window and dialog titles ■ menu functions ■ panel, tab, and button names ■ box labels ■ object properties and their values (e.g. status).
Italic typeface	Designates technical background information, such as ■ path, folder, and file names ■ methods ■ classes.
CAPITALS	Designate keyboard-related information, such as ■ key names ■ keyboard shortcuts.
`Monospaced typeface`	Designates code examples.

TIP

Instructions in this manual are based on Windows 7. Select the appropriate commands if you are using a different operating system.

Structural Context

Whether or not a recipe's or workflow's unit procedure is suitable for holding DCS trigger phases does not only depend on its graph structure, but is also controlled by the capabilities assigned to the operations of the unit procedure:

- Capability prerequisites:
  - Only operations that hold both the **Event-triggered** capability and the **Trigger-enabled** capability can interpret the trigger events sent by a trigger phase.
  - Trigger phases become active and complete automatically and do not require user interaction. For this reason, they have no user interface and need to be located in an operation that runs on a server, invisible to operators who process orders with PharmaSuite for Production Execution. Thus an operation that holds trigger phases needs to have the **Server-run** capability.
- Graph structure prerequisites:
  - A trigger phase can only send trigger events to an **Event-triggered** operation if the operation is active to process the triggers. Consequently, the **Event-triggered** operation must run at the same time as the phase from which it receives trigger events. This means that an operation holding trigger phases must be located after the same simultaneous branch on a parallel track (page 3) to the **Event-triggered** operation whose runs it controls.

Trigger Phases

The placement of **DCS alarm-based trigger** phases in a recipe or workflow is determined by the capabilities their operations need to have and by the structural requirement to ensure their simultaneous activity with the **Event-triggered** operation to which they send trigger events.

TIP

Please note that

- an **Event-triggered** operation can reference several trigger phases for receiving trigger events from them.
- a trigger phase can be referenced by several **Event-triggered** operations for sending them trigger events.

The following rules apply with respect to the start and completion of trigger processing of each of the two phases:

- A trigger phase becomes active automatically as soon as processing reaches its operation, but trigger processing does not start until at least one **Event-triggered** operation that references the phase has become active as well. During execution this means that only when the template of an **Event-triggered** operation becomes visible in the Cockpit of PharmaSuite for Production Execution, does the trigger phase start processing and can send the trigger events, which create the runs of the operation.
- If no **Event-triggered** operation becomes active, the trigger phase completes automatically after its defined timeout period has elapsed.
- If trigger processing has started along with one of its **Event-triggered** operations, the phase continues to send trigger events until the last of its **Event-triggered** operations is completed, which happens when an operator removes the template of the **Event-triggered** operation from the Cockpit of PharmaSuite for Production Execution. Without a target to which it can send its trigger events, the trigger phase completes automatically.

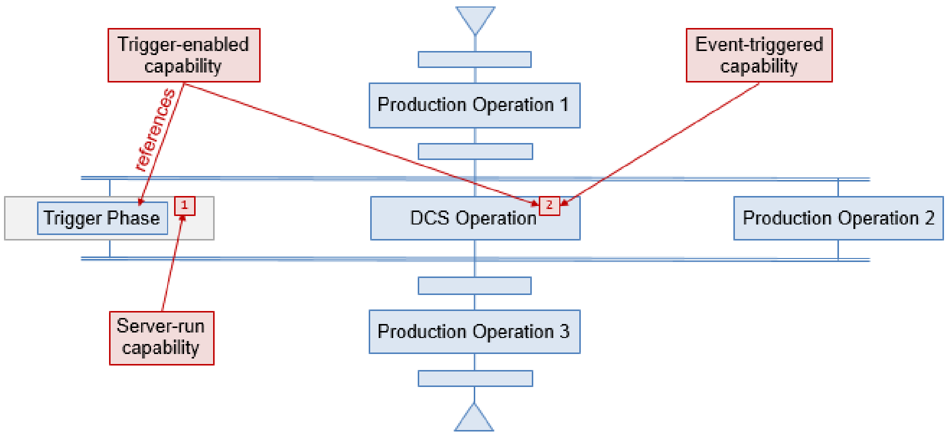


Figure 1: Unit procedure with trigger phase and Event-triggered operation

IMPORTANT

The general rules for building recipe structures as SFC graphs also govern unit procedures that hold trigger phases and **Event-triggered** operations. Thus it is possible to build recipes that are valid from an SFC graph perspective, but do not meet the functional requirements made by processing use cases such as event-triggered DCS alarm retrieval. A recipe with functionally invalid structures is bound to cause serious issues during execution.

Issue: Operations Not Strictly Parallel

Having operations precede either the **Server-run** operation with the trigger phases or the **Event-triggered** operation may lead to timing issues during execution.

- If the **Server-run** operation has a direct predecessor operation within the simultaneous branch, the system shows the following behavior during execution:
  - The template of the **Event-triggered** operation is visible in the Cockpit, but its trigger phase is not yet active to provide it with trigger events.
  - The **Server-run** operation holding the trigger phase can only become active when its preceding operation has been completed, thus causing an indeterminate delay of trigger processing.

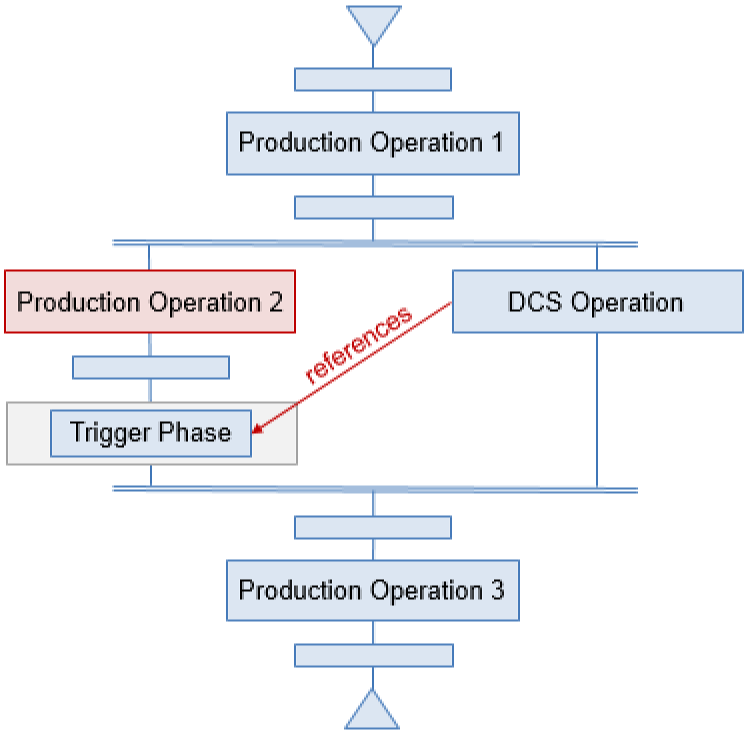


Figure 2: Functionally invalid - Server-run operation with preceding operation

- If the **Event-triggered** operation has a direct predecessor operation within the simultaneous branch, the system shows the following behavior during execution:
  - The trigger phase becomes active, but trigger processing does not start unless the operation preceding the **Event-triggered** operation has been completed and its template has become active.
  - The active trigger phase waits for its targeted **Event-triggered** operation to become active, but only until its defined timeout period has elapsed.
  - After the timeout period has elapsed, the trigger phase is completed automatically.
  - Thus the indeterminate delay of the targeted **Event-triggered** operation can cause its entire trigger schedule to fail, since its referenced trigger phase has timed out before the **Event-triggered** operation has become active at all.

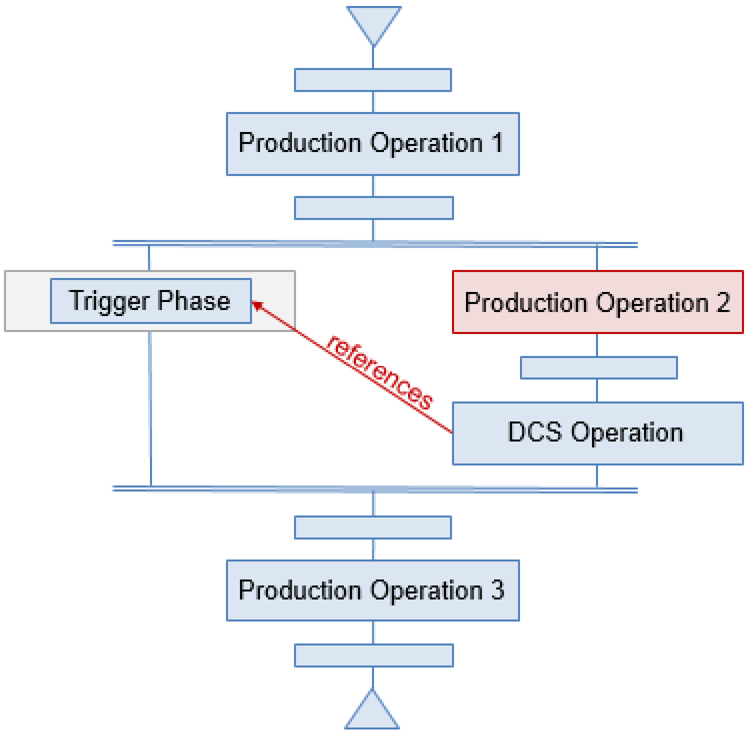


Figure 3: Functionally invalid - Event-triggered operation with preceding operation

TIP

To avoid issues of this type, place the predecessor operation before the simultaneous branch to establish a simultaneous activation of the **Server-run** operation holding the trigger phases and its targeted **Event-triggered** operation.

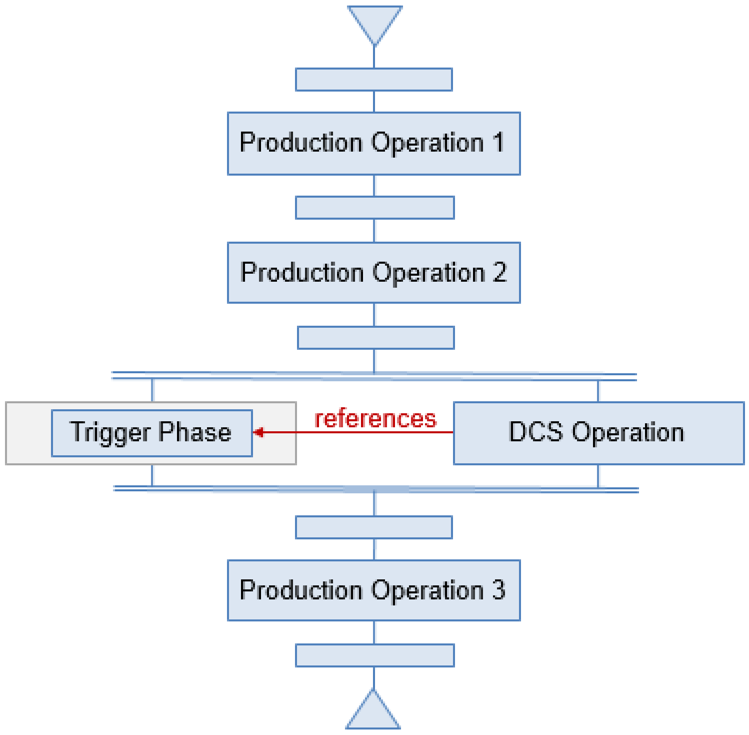


Figure 4: Functionally valid - strictly parallel Server-run and Event-triggered operations

Issue: Sequential Event-triggered Operations with Identical Phase Reference

If your unit procedure has two **Event-triggered** operations that are located on the same track after the simultaneous branch and that both reference the same trigger phase, the system shows the following behavior during execution:

- The **Server-run** operation holding the trigger phase becomes active together with the first **Event-triggered** operation and the trigger phase begins to send trigger events as scheduled.
- Once the first **Event-triggered** operation has completed, the trigger phase is informed of this fact and completes automatically.
- As a consequence, when the second **Event-triggered** operation becomes active, its trigger phase is not available and its trigger schedule is bound to fail.

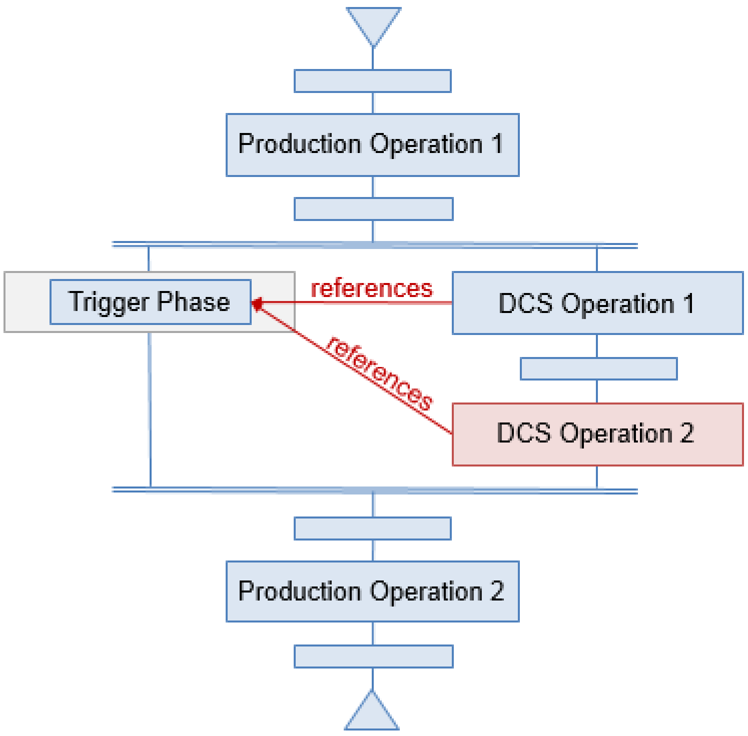


Figure 5: Functionally invalid - sequential Event-triggered operations with identical phase reference

TIP

To avoid issues of this type, place sequential trigger phases into the **Server-run** operation, one for each **Event-triggered** operation.

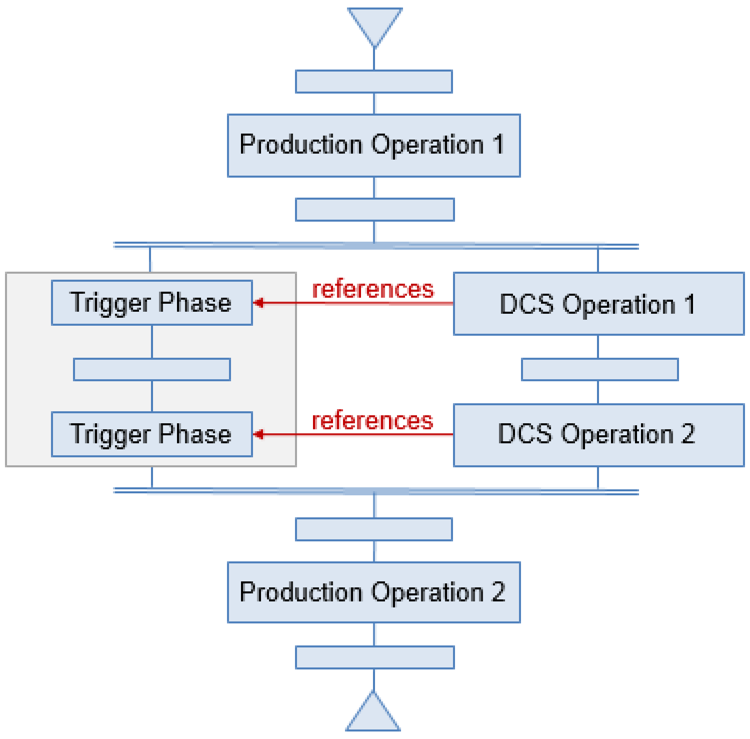


Figure 6: Functionally valid - sequential Event-triggered operations with sequential phases to reference

Issue: Looped Event-triggered Operation after Simultaneous Branch

If your unit procedure has a loop only around its single **Event-triggered** operation, the system shows the following behavior during execution:

- The **Server-run** operation holding the trigger phase becomes active together with the **Event-triggered** operation and the trigger phase begins to send trigger events as scheduled.
- Once the **Event-triggered** operation has completed for the first time and reaches the loop's decision transitions, the trigger phase is informed of this fact and completes automatically.
- As a consequence, if the transition decides that the loop has to be passed through and the **Event-triggered** operation needs to be run again, its trigger phase is not available and its trigger schedule is bound to fail.

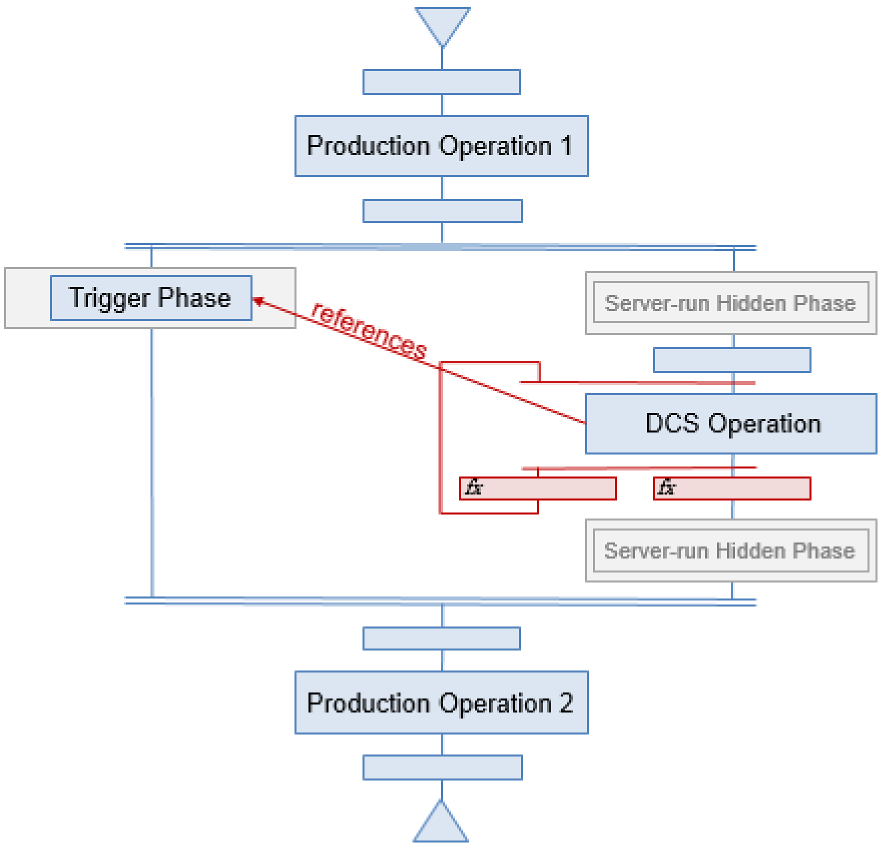


Figure 7: Functionally invalid - looped Event-triggered operation after simultaneous branch

TIP

To avoid issues of this type, enclose the entire simultaneous branch in the loop.

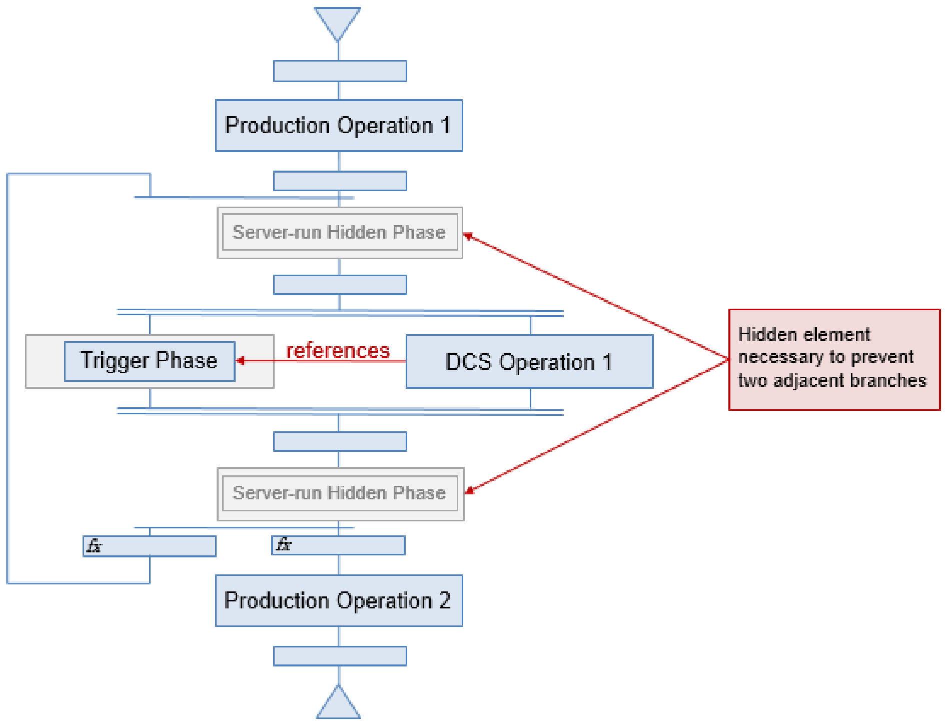


Figure 8: Functionally valid - loop around Server-run and Event-triggered operations

Create DCS Batch

The **Create DCS batch** phase allows an operator to request the creation of a batch on a DCS.

It can be used for processing requirements, such as:

- Creating a batch to be executed on an automation system
  Based on an existing master recipe and other product-specific parameters, the phase allows to create a DCS batch on a DCS.

Execution

In addition to the instruction text, the **Create DCS batch** phase displays the data required by the DCS system for creating a batch:

- the name of the DSC on which the batch is to be created,
- the definition data of the batch to be created:
  - Batch ID
  - Master recipe ID
  - Formula ID
  - Campaign ID
  - Scale to define the size of the batch
  - Description of the batch
- a table that indicates the unit binding and thus the unit on which the batch is to be processed,
- a table that lists all parameters and their values to be set on the DCS for batch processing.
- an information message that indicates the status of the batch creation, it includes the identifier if the created batch and its creation timestamp if the creation was successful.

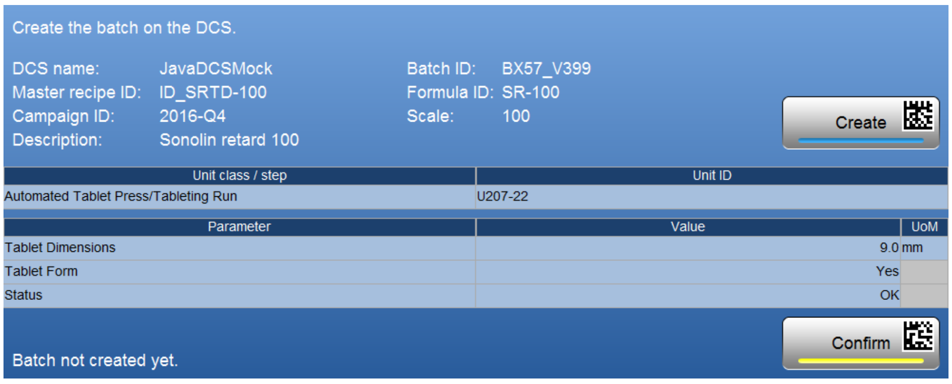
When the operator taps the **Create** button thus sending a batch creation request to the DCS, the phase accesses the DCS and creates a new batch with the defined data, unit binding, and parameter settings. The **Create** button is disabled after the first creation request has been sent. Any further requests or changes to the defined data are considered exceptions.
For this reason the phase provides user-triggered exceptions as long as it is active

- to create a batch manually when the first creation request has failed.
- to re-send the creation request and overwrite the previously created one. In this case the system creates a new batch with a new identifier on the DCS and discards the existing batch.
- to override the data defined with
  - the batch definition, such as batch ID or master recipe ID,
  - the unit binding,
  - the parameter values.

Different phase modes enable the usage in various situations that can occur during processing:

- In the **Manual completion** mode, the operator manually triggers the creation of the batch.
- In the **Automatic completion** mode, the phase creates the batch and is completed automatically without any operator interaction.

After completion the phase displays the created batch with its data in the Execution Window.
The Navigator displays the identifier of the created batch.


Create the batch on the DCS.

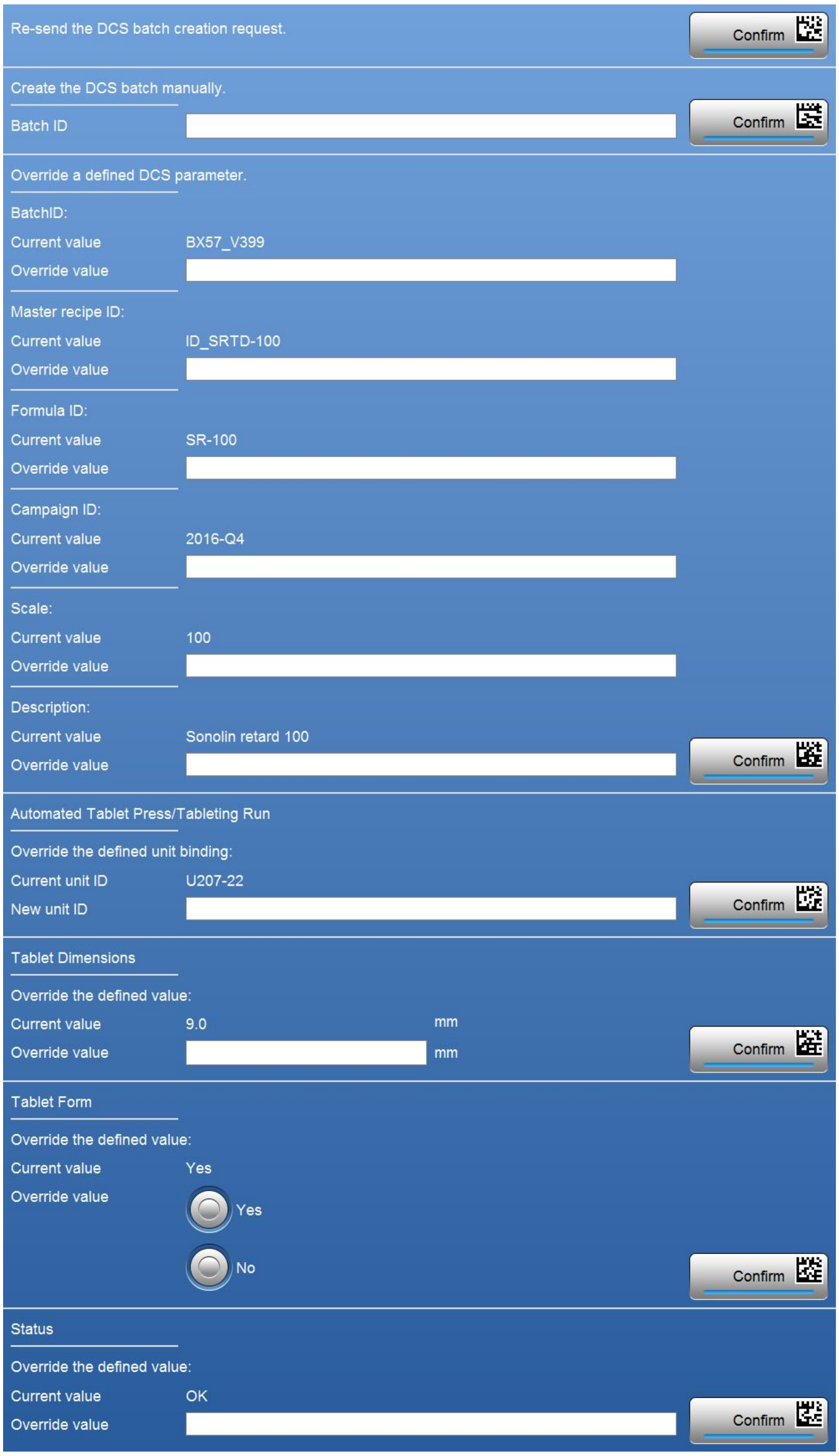
DCS name:	JavaDCSMock	Batch ID:	BX57_V399
Master recipe ID:	ID_SRTD-100	Formula ID:	SR-100
Campaign ID:	2016-Q4	Scale:	100
Description:	Sonolin retard 100		

Unit class / step	Unit ID
Automated Tablet Press/Tableting Run	U207-22

Parameter	Value	UoM
Tablet Dimensions	9.0	mm
Tablet Form	Yes	
Status	OK	

Batch not created yet.

Figure 9: Create DCS batch during execution


Re-send the DCS batch creation request. Confirm

Create the DCS batch manually.

Batch ID	

Confirm

Override a defined DCS parameter.

BatchID:

Current value	BX57_V399
Override value	

Master recipe ID:

Current value	ID_SRTD-100
Override value	

Formula ID:

Current value	SR-100
Override value	

Campaign ID:

Current value	2016-Q4
Override value	

Scale:

Current value	100
Override value	

Description:

Current value	Sonolin retard 100
Override value	

Confirm

Automated Tablet Press/Tableting Run

Override the defined unit binding:

Current unit ID	U207-22
New unit ID	

Confirm

Tablet Dimensions

Override the defined value:

Current value	9.0	mm
Override value		mm

Confirm

Tablet Form

Override the defined value:

Current value	Yes
Override value	Yes No

Confirm

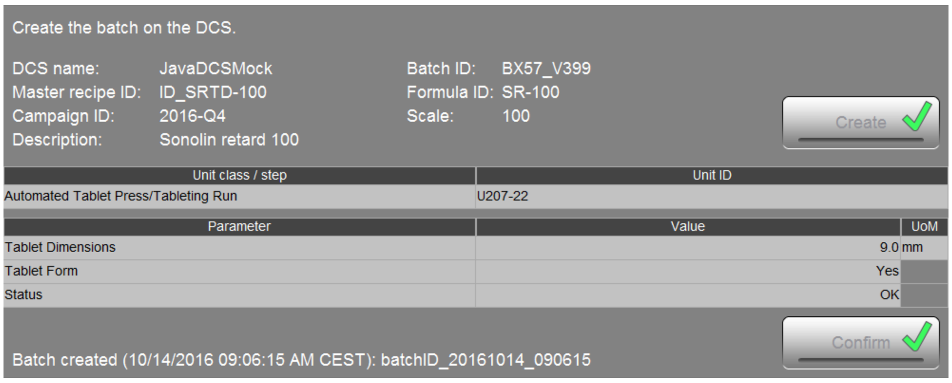
Status

Override the defined value:

Current value	OK
Override value	

Confirm

Figure 10: User-triggered exceptions of Create DCS batch



Create the batch on the DCS.

DCS name:	JavaDCSMock	Batch ID:	BX57_V399
Master recipe ID:	ID_SRTD-100	Formula ID:	SR-100
Campaign ID:	2016-Q4	Scale:	100
Description:	Sonolin retard 100		

Unit class / step	Unit ID
Automated Tablet Press/Tableting Run	U207-22

Parameter	Value	UoM
Tablet Dimensions	9.0	mm
Tablet Form	Yes	
Status	OK	

Batch created (10/14/2016 09:06:15 AM CEST): batchID_20161014_090615

Figure 11: Create DCS batch after phase completion

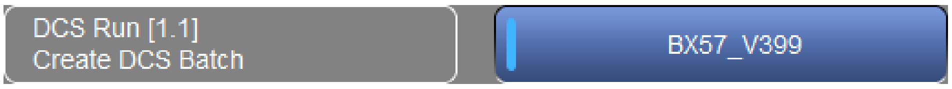


Figure 12: Create DCS batch in the Navigator

Phase Design

The characteristics of the **Create DCS batch** phase are defined via process parameters and their attributes.
Its user interface is designed in three columns that span several rows. When the phase is active, the merged columns of the first row provide space for textual instructions.
The following four rows of the left and center columns contain the defined batch data.
In the right column of these rows, the phase provides the **Create** button.
The next rows display two tabular views that span all columns. The first one contains the data of the unit binding and the second one shows the defined parameters with their values.
The merged left and center columns of the bottom row display the information message on the status of the batch creation.
The right column of the bottom row contains the **Confirm** button.

When the phase is completed it shows the same three-column, multi-row layout.
Exception handling during execution is controlled by a risk assessment classification and an exception message that are both defined by the recipe author in the exception's process parameter.

Process Parameters

The following process parameters are available to configure the phase's behavior during execution:

Instruction

Represents the instruction text that is visible on the preview, the active, and the completed view of the phase.

Attribute	Type	Comment
Text	HTML text	Instruction text to be displayed. Maximum length is 2000 characters (including HTML tags).

Mode

Defines if the phase expects operator interaction during execution.

Attribute	Type	Comment
Mode	Choice list	Defines the processing mode. **Manual completion** (default): Operator confirms the phase. **Automatic completion**: Phase is automatically completed after a batch has been created successfully on the DCS.

DCS

Defines the name of the DCS system to be accessed for alarms retrieval.

Attribute	Type	Comment
Name	String	Logical name of the DCS to be used. The available entries correspond to the entries in the **DCSNames** list. Default setting: First entry in the list.

Definition

Defines the basic data of the batch to be created.

Attribute	Type	Comment
Batch ID	String	Defines the identifier of the batch to be created on the DCS. This attribute is required by the DCS Adapter.
Master recipe ID	String	Defines the identifier of the master recipe to be used. This attribute is required by the DCS Adapter.
Formula ID	String	Defines the identifier of the formula to be used.
Description	String	Defines the description of the batch to be created.
Campaign ID	String	Defines the identifier of the campaign to be used.
Scale	BigDecimal	Defines the scale of the batch in percent.

Create batch manually

Represents a user-triggered exception that is accessible from the Exception Window.
The exception allows an operator to create the DCS batch manually by typing the internal batch identifier of the DCS batch. If the identifier is unknown at the point, the phase uses DEFAULT_BATCH_ID as batch identifier to create the DCS batch on the DCS.
If another batch with the defined batch data has been created before, it is discarded when the user confirms the exception and the new batch is created.
It covers incidents when batch creation failed due to a connection issue to the DCS and needs to be executed again.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.

Attribute	Type	Comment
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Override DCS parameter

Represents a user-triggered exception that is accessible from the Exception Window. The exception allows an operator to override the basic data given with the **Definition** parameter (page 18).
It covers incidents when the defined data is faulty and needs to be adjusted.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Override bundle parameter

Represents a user-triggered exception that is accessible from the Exception Window. The exception allows an operator to override the values defined with bundle parameters (page 21) of the **Boolean**, **Numeric**, or **String** data types.
It covers incidents when a defined value is faulty and needs to be adjusted.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.

Attribute	Type	Comment
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Override unit binding

Represents a user-triggered exception that is accessible from the Exception Window.
The exception allows an operator to override data defined with a bundle parameter (page 21) of the **Unit binding** type.
It covers incidents when batch execution has to be performed on a different unit than originally intended and the value given with the **Unit ID** attribute needs to be adapted for this reason.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Re-send creation request

Represents a user-triggered exception that is accessible from the Exception Window.
The exception allows an operator to send additional creation requests to the DCS after the first creation request has been processed and the **Create** button is no longer enabled.
If another batch with the defined batch data has been created before, it is discarded when the user confirms the exception and the new batch is created.
It covers incidents when a batch needs to be re-created with adjusted data, such as a different unit binding.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Parameter bundles

In addition to the permanent process parameters that are always present, the **Create DCS batch** phase provides parameter bundles as optional process parameters, which you can insert if required.

You can add process parameter bundles for up to 50 values of four different data types (**Boolean**, **Numeric**, **String**, **Unit binding**) to the **Create DCS batch** phase.

ADDING PARAMETER BUNDLES

1. Click the **Add parameter bundle** button.
   The system opens an option list that holds all data types available for the value.
2. Select the type.
   The system opens the **Add <Data Type>** dialog to define the value's identifier.
3. Type an identifier and click the **OK** button.
   The system adds all process parameters of the bundle to the list of parameters.

REMOVING PARAMETER BUNDLES

1. In the list of parameters, select header row that contains the identifier of the bundle you wish to remove.
2. Click the **Remove parameter** button.
   The system asks you to confirm the action and then removes the value bundle.

The following process parameters are available to configure the phase's behavior during execution:

Boolean value

Indicates the parameter with its value of the **Boolean** data type that is to be set on the DCS.

Attribute	Type	Comment
Parameter	String	Defines the identifier of the parameter to be used.
Value	Boolean	Defines the value of the parameter.

Numeric value

Indicates the parameter with its value of the **Numeric** data type that is to be set on the DCS.

Attribute	Type	Comment
Parameter	String	Defines the identifier of the parameter to be used.
Value	BigDecimal	Defines the value of the parameter.
UoM	Unit of measure	Must match a unit of measure available within PharmaSuite.

String Value

Indicates the parameter with its value of the **String** data type that is to be set on the DCS.

Attribute	Type	Comment
Parameter	String	Defines the identifier of the parameter to be used.
Value	String	Defines the value of the parameter.

Unit binding

Indicates the unit binding data that is to be set on the DCS.

Attribute	Type	Comment
Unit class/step	String	Defines the required unit class or step.
Unit ID	String	Defines the identifier of the unit to be used.

Output Variables

Instead of specifying a fixed value to be displayed or used during execution, you can also use an expression created in the Expression editor to draw the output of another phase or the calculated result of several phase outputs as value into a parameter attribute. When you reference phase outputs in this manner you need to be aware of the following restrictions:

- Only if a phase has been processed does it provide an output that can be fed into another phase as attribute value. For this reason, you can never reference an output of a phase that is a strict successor of the phase in which you try to use the output.
- Branches and loops, however, require special notice in this context, since they are only potentially passed through and/or completed during processing, so their outputs are not reliably available. Thus you can reference any such potentially available outputs, but need to be aware of the fact that the provided value may be **Undefined** so that the phase to which you are feeding the output must be able to deal with such an **Undefined** input value.

The **Create DCS batch** phase provides the following output variables:

Batch ID

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the DCS batch that was created on the DCS.

Campaign ID

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the campaign that was used to create the DCS batch.

Internal batch ID

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the internal batch that depends on the used DCS.

Master recipe ID

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the master recipe that was used to create the DCS batch.

Scale

- Data type: BigDecimal, floating point number that allows calculating with greater precision than Float.
- Usage: The output variable provides the scale that was used to create the DCS batch.

Boolean value

If you have added Boolean value bundles to the list of parameters, the system provides output variables for each of the bundles.

> TIP
> The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for Boolean value bundles:

Boolean value - Parameter

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the parameter that was used to create the DCS batch.

Boolean value - Value

- Data type: Boolean, with the values **true** and **false**
- Usage: The output variable provides the value of the parameter that was used to create the DCS batch.

Numeric value

If you have added Numeric value bundles to the list of parameters, the system provides output variables for each of the bundles.

TIP
The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for Numeric value bundles:

Numeric value - Parameter

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the parameter that was used to create the DCS batch.

Numeric value - Unit of measure

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**mm**" or "**ea**".
- Usage: The output variable provides the unit of measure of the parameter that was used to create the DCS batch.

Numeric Value - Value

- Data type: BigDecimal, floating point number that allows calculating with greater precision than Float.
- Usage: The output variable provides the value of the parameter that was used to create the DCS batch.

String value

If you have added String value bundles to the list of parameters, the system provides output variables for each of the bundles.

TIP
The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for string value bundles:

String value - Parameter

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the parameter that was used to create the DCS batch.

String value - Value

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the value of the parameter that was used to create the DCS batch.

Unit binding

If you have added unit binding bundles to the list of parameters, the system provides output variables for each of the bundles.

TIP
The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for unit binding bundles:

Unit binding - Unit class

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the unit class that was used to create the DCS batch.

Unit binding - Unit ID

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the identifier of the unit that was used to create the DCS batch.

Identifier

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Read Instruction**".
- Usage: The output variable provides the identifier of the phase.

Instance count

- Data type: Long, used for integral numbers:
  12345
- Usage: The output variable provides the count of the number of instances the phase has been processed, for example in a loop. The count is also increased when the phase is skipped from an operator's perspective, since the phase is still executed, but as a hidden phase.
  The count variable of a phase that has not been executed provides 0 as output value.

Start time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the start time of the phase.

Completion time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the completion time of the phase.

TIP

To calculate a duration from two timestamps and display it in a specific format, you need to use two conversion functions on the calculation:

- **Convert to Unitless Number** (**convertTo**) takes the calculated duration and converts it into the duration's value for one of its units (e.g. minutes or seconds).
- **Convert to String for Display** (**convertToDisplayString)** takes the converted value and displays it as string to which you can add the unit, also as string.

Example:

Sample Phase with Start time = 14-Nov-2014@10:15
Sample Phase with Completion time = 14-Nov-2014@11:47
The duration is to be displayed in minutes.

```
convertToDisplayString
   (convertTo
        ({Sample Phase}.{Completion time}-{Sample Phase}.{Start time},
"min")
   )
 + " min"
```

As result of the expression, the system displays "**92 min**".

Chapter 3

Get DCS Batch Values

The **Get DCS batch values** phase allows an operator to retrieve values from a batch processed on a DCS.

It can be used for processing requirements, such as:

- Adding data to the batch report from a batch executed on an automation system
  The phase can retrieve specific values of a batch executed on a DCS. The values are added to the phase-specific sub-report and thus available in the batch report.

Execution

In addition to the instruction text, the **Get DCS batch values** phase displays the following data:

- the name of the DSC from which the values will be retrieved,
- the identifier of the batch or the unit whose values will be retrieved,
- a table that lists all parameters and their values.

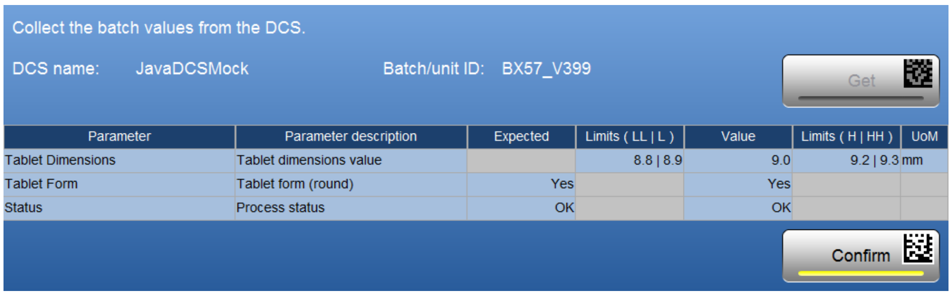
When the operator taps the **Get** button the phase accesses the DCS and retrieves the specified values. For Numeric values, it can be configured to perform limit checks on the retrieved values against two pre-defined sets of limits for low and high (**L/LL** and **H/HH**). For Boolean and String values, the system checks the retrieved values against an expected value. Which of the limits or expected values are actually available and enabled for checking is configured with the phase's process parameters.
The **Get** button is disabled after all values have been retrieved once. Any further changes to the retrieved values are considered exceptions.
During execution, if a value is affected by a technical issue, such as a communication failure, its cell background assumes a different color (red) and displays a marker symbol (X).

Different phase modes enable the usage in various situations that can occur during processing:

- In the **Manual completion** mode, the operator manually triggers retrieving the batch values.
- In the **Automatic completion** mode, the phase retrieves the batch values and is completed automatically without any operator interaction.

As long as the phase is active, it provides user-triggered exceptions to override the retrieved values.

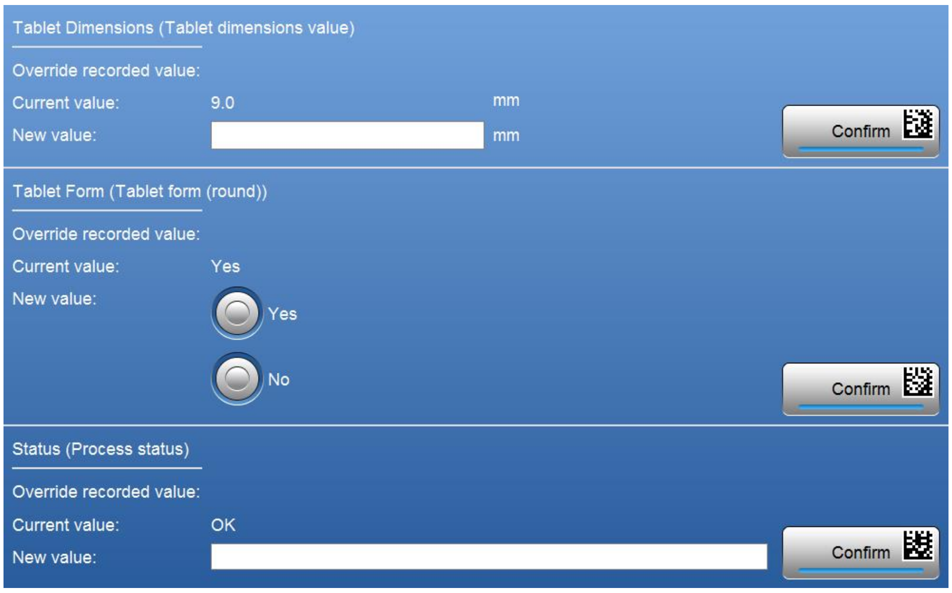
After completion the phase displays the affected parameters and their values in the Execution Window.
The Navigator displays the identifier of the batch whose values were retrieved.


Collect the batch values from the DCS.

DCS name: JavaDCSMock Batch/unit ID: BX57_V399

Parameter	Parameter description	Expected	Limits (LL | L)	Value	Limits (H | HH)	UoM
Tablet Dimensions	Tablet dimensions value		8.8 | 8.9	9.0	9.2 | 9.3	mm
Tablet Form	Tablet form (round)	Yes		Yes		
Status	Process status	OK		OK		

Figure 13: Get DCS batch values during execution


Tablet Dimensions (Tablet dimensions value)

Override recorded value:

Current value: 9.0 mm

New value: mm

Tablet Form (Tablet form (round))

Override recorded value:

Current value: Yes

New value: Yes No

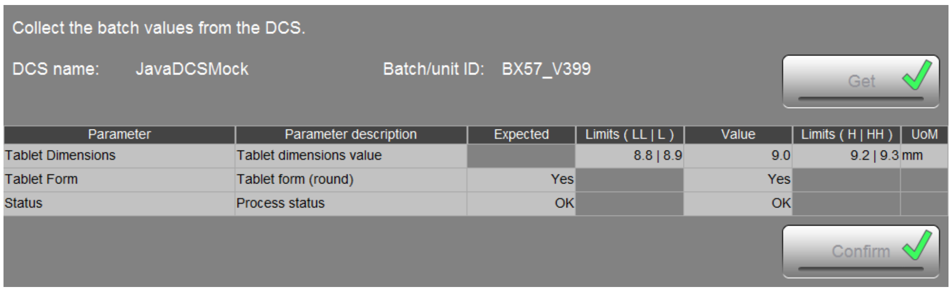
Status (Process status)

Override recorded value:

Current value: OK

New value:

Figure 14: User-triggered exceptions of Get DCS batch values



Collect the batch values from the DCS.

DCS name: JavaDCSMock Batch/unit ID: BX57_V399

Parameter	Parameter description	Expected	Limits (LL | L)	Value	Limits (H | HH)	UoM
Tablet Dimensions	Tablet dimensions value		8.8 | 8.9	9.0	9.2 | 9.3	mm
Tablet Form	Tablet form (round)	Yes		Yes		
Status	Process status	OK		OK		

Figure 15: Get DCS batch values after phase completion

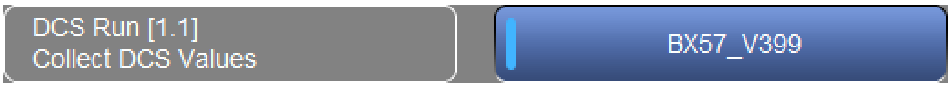


Figure 16: Get DCS batch values in the Navigator

Phase Design

The characteristics of the **Get DCS batch values** phase are defined via process parameters and their attributes.
Its user interface is designed in three columns that span several rows. When the phase is active, the merged columns of the first row provide space for textual instructions.
In the left and center column of the second row, the phase displays the name of the DCS and the identifier of the batch or unit to which it refers. The right column contains the **Get** button. The next rows display a tabular view of relevant data of each retrieved parameter and its values, spanning all columns. If there are values that have not been defined, such as limits for a Numeric value, the respective table cell displays N/A (if no limit is defined) or --- (if one of the two possible limits is not defined). Table cells that can never hold an entry, such as limits for Boolean or String values, show with a gray background.
When an operator adds an exception, the phase displays an exception marker at the affected value.
The right column of the bottom row contains the **Confirm** button.

When the phase is completed it shows the same three-column, multi-row layout.
Exception handling during execution is controlled by a risk assessment classification and an exception message that are both defined by the recipe author in the exception's process parameter.

Process Parameters

The following process parameters are available to configure the phase's behavior during execution:

Instruction

Represents the instruction text that is visible on the preview, the active, and the completed view of the phase.

Attribute	Type	Comment
Text	HTML text	Instruction text to be displayed. Maximum length is 2000 characters (including HTML tags).

DCS

Defines the name of the DCS system to be accessed for value retrieval.

Attribute	Type	Comment
Name	String	Logical name of the DCS to be used. The available entries correspond to the entries in the **DCSNames** list. Default setting: First entry in the list.

Definition

Defines the basic or unit whose values will be retrieved.

Attribute	Type	Comment
Batch/unit ID	String	Defines the identifier of the batch/unit to be used for data retrieval.

Mode

Defines if the phase expects operator interaction during execution.

Attribute	Type	Comment
Mode	Choice list	Defines the processing mode. **Manual completion** (default): Operator confirms the phase. **Automatic completion**: Phase is automatically completed after the batch values have been retrieved successfully from the DCS batch.

Override DCS batch value

Represents a user-triggered exception that is accessible from the Exception Window.
The exception allows an operator to override a value retrieved from the DCS.
It covers incidents when a retrieved value is faulty and needs to be adjusted.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Parameter bundles

In addition to the permanent process parameters that are always present, the **Get DCS batch values** phase provides parameter bundles as optional process parameters, which you can insert if required.

You can add process parameter bundles for up to 50 values of four different data types (**Boolean**, **Numeric**, **String**) to the **Get DCS batch values** phase.

ADDING PARAMETER BUNDLES

1. Click the **Add parameter bundle** button.
   The system opens an option list that holds all data types available for the value.
2. Select the type.
   The system opens the **Add <Data Type>** dialog to define the value's identifier.
3. Type an identifier and click the **OK** button.
   The system adds all process parameters of the bundle to the list of parameters.

REMOVING PARAMETER BUNDLES

1. In the list of parameters, select header row that contains the identifier of the bundle you wish to remove.
2. Click the **Remove parameter** button.
   The system asks you to confirm the action and then removes the value bundle.

The following process parameters are available to configure the phase's behavior during execution:

Boolean value - Master (bundle identifier)

Indicates the parameter with its description and path of the **Boolean** data type that is to be retrieved from the DCS.

Attribute	Type	Comment
Parameter	String	Defines the identifier of the parameter to be read.
Parameter description	String	Defines an alias for the parameter.
Defines the path of the parameter.	String	Defines the path of the parameter.

Expected value configuration

Represents a system-triggered exception that is displayed in the Exception Window to define if the value retrieved from the DCS must be checked against an expected value.

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the **Value** attribute of the **Expected value definition** process parameter (page 35) is set. If it is not set, the validation will fail. Default setting: **No**.
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Expected value definition

Defines the value (by its display text) required as expected value if the respective check is enabled.

Attribute	Type	Comment
Value	Text	Defines the expected value. Maximum length is 2000 characters.

Numeric value - Master (bundle identifier)

Indicates the parameter with its description, path, and unit of measure of the **Numeric** data type that is to be retrieved from the DCS.

Attribute	Type	Comment
Parameter	String	Defines the identifier of the parameter to be read.
Parameter description	String	Defines an alias for the parameter.

Attribute	Type	Comment
Parameter path	String	Defines the path of the parameter.
UoM	Unit of measure	Must match a unit of measure available within PharmaSuite.

L-H configuration

Represents a system-triggered exception that is displayed in the Exception Window to define if the value retrieved from the DCS is checked against the limits defined with the **Limit definition** process parameter (page 38). If both checks are enabled, they are performed in the following order:

1. LL-HH (defined with the **LL-HH configuration** process parameter (page 37))
2. L-H

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the **L limit** or **H limit** attributes of the **Limit definition** process parameter (page 38) are set. If they are not set, the validation will fail. Default setting: **No**.
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

LL-HH configuration

Represents a system-triggered exception that is displayed in the Exception Window to define if the value retrieved from the DCS is checked against the limits defined with the **Limit definition** process parameter (page 38). If both checks are enabled, they are performed in the following order:

1. LL-HH
2. L-H (defined with the **L-H configuration** process parameter (page 36))

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the **LL limit** or **HH limit** attributes of the **Limit definition** process parameter (page 38) are set. If they are not set, the validation will fail. Default setting: **No**.
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Limit definition

Limits are defined as absolute values. Make sure that the limits are strictly sequential and do not overlap, so that

- LL limit < L limit < H limit < HH limit

Attribute	Type	Comment
LL limit	BigDecimal (Double, Float, Integer)	Defines the values of the lower limits (including the values themselves). Limit values with more than 7 digits are truncated at the end in the Phase Preview.
L limit	BigDecimal (Double, Float, Integer)	
H limit	BigDecimal (Double, Float, Integer)	Defines the value of the upper limit (including the values themselves). Limit values with more than 7 digits are truncated at the end in the Phase Preview.
HH limit	BigDecimal (Double, Float, Integer)	

String value - Master (bundle identifier)

Indicates the parameter with its description and path of the **String** data type that is to be retrieved from the DCS.

Attribute	Type	Comment
Parameter	String	Defines the identifier of the parameter to be read.
Parameter description	String	Defines an alias for the parameter.
Parameter path	String	Defines the path of the parameter.

Expected value configuration

Represents a system-triggered exception that is displayed in the Exception Window to define if the value retrieved from the DCS must be checked against an expected value.

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the **Value** attribute of the **Expected value definition** process parameter (page 39) is set. If it is not set, the validation will fail. Default setting: **No**.
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Expected value definition

Defines the value (by its display text) required as expected value if the respective check is enabled.

Attribute	Type	Comment
Value	Text	Defines the expected value. Maximum length is 2000 characters.

Output Variables

Instead of specifying a fixed value to be displayed or used during execution, you can also use an expression created in the Expression editor to draw the output of another phase or the calculated result of several phase outputs as value into a parameter attribute. When you reference phase outputs in this manner you need to be aware of the following restrictions:

- Only if a phase has been processed does it provide an output that can be fed into another phase as attribute value. For this reason, you can never reference an output of a phase that is a strict successor of the phase in which you try to use the output.
- Branches and loops, however, require special notice in this context, since they are only potentially passed through and/or completed during processing, so their outputs are not reliably available. Thus you can reference any such potentially available outputs, but need to be aware of the fact that the provided value may be **Undefined** so that the phase to which you are feeding the output must be able to deal with such an **Undefined** input value.

The **Get DCS batch values** phase provides the following output variables:

Retrieval successful

- Data type: Boolean, with the values **true** and **false**
- Usage: The output variable states if the get operation from the DCS batch was successful.
  - The value is `true` if all batch values have been read successfully.
  - The value is `false` if at least one of the batch values could not be read from the DCS batch or has been overridden by using the respective user-triggered exception.

Boolean value

If you have added Boolean value bundles to the list of parameters, the system provides output variables for each of the bundles.

TIP
The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for Boolean value bundles:

Boolean value - Retrieval successful

- Data type: Boolean, with the values **true** and **false**
- Usage: The output variable states if the get operation from the DCS batch was successful.
  - The value is `true` if the batch value of the boolean parameter has been read successfully.
  - The value is `false` if the batch value of the boolean parameter could not be read from the DCS batch or has been overridden by using the respective user-triggered exception.

Boolean value - Value

- Data type: Boolean, with the values **true** and **false**
- Usage: The output variable provides the value of the boolean parameter. The value is Null if N/A is the phase result.

Numeric value

If you have added Numeric value bundles to the list of parameters, the system provides output variables for each of the bundles.

> TIP
> The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for Numeric value bundles:

Numeric value - Retrieval successful

- Data type: Boolean, with the values **true** and **false**
- Usage: The output variable states if the get operation from the DCS batch was successful.
  - The value is `true` if the batch value of the numeric parameter has been read successfully.
  - The value is `false` if the batch value of the numeric parameter could not be read from the DCS batch or has been overridden by using the respective user-triggered exception.

Numeric value - Unit of measure

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**mm**" or "**ea**".
- Usage: The output variable provides the unit of measure of the numeric parameter. The value is Null if N/A is the phase result.

Numeric value - Value

- Data type: BigDecimal, floating point number that allows calculating with greater precision than Float.
- Usage: The output variable provides the actual value of the numeric parameter as a **BigDecimal** value. The value is Null if N/A is the phase result.

String value

If you have added string value bundles to the list of parameters, the system provides output variables for each of the bundles.

> TIP
> The output variables of a value bundle are prefixed with its bundle identifier.

The following output variables are available for string value bundles:

String value - Retrieval successful

- Data type: Boolean, with the values **true** and **false**
- Usage: The output variable states if the get operation from the DCS batch was successful.
    - The value is `true` if the batch value of the string parameter has been read successfully.
    - The value is `false` if the batch value of the string parameter could not be read from the DCS batch or has been overridden by using the respective user-triggered exception.

String value - Value

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Coater-A**" or "**In process**".
- Usage: The output variable provides the value of the string parameter. The value is Null if N/A is the phase result.

Identifier

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Read Instruction**".
- Usage: The output variable provides the identifier of the phase.

Instance count

- Data type: Long, used for integral numbers:
  12345
- Usage: The output variable provides the count of the number of instances the phase has been processed, for example in a loop. The count is also increased when the phase is skipped from an operator's perspective, since the phase is still executed, but as a hidden phase.
  The count variable of a phase that has not been executed provides 0 as output value.

Start time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the start time of the phase.

Completion time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the completion time of the phase.

TIP

To calculate a duration from two timestamps and display it in a specific format, you need to use two conversion functions on the calculation:

- **Convert to Unitless Number** (**convertTo**) takes the calculated duration and converts it into the duration's value for one of its units (e.g. minutes or seconds).
- **Convert to String for Display** (**convertToDisplayString)** takes the converted value and displays it as string to which you can add the unit, also as string.

Example:

Sample Phase with Start time = 14-Nov-2014@10:15
Sample Phase with Completion time = 14-Nov-2014@11:47
The duration is to be displayed in minutes.

```
convertToDisplayString
   (convertTo
        ({Sample Phase}.{Completion time}-{Sample Phase}.{Start time},
"min")
   )
 + " min"
```

As result of the expression, the system displays "**92 min**".

Chapter 4

Get DCS Alarms

The **Get DCS alarms** phase allows an operator to request alarm-specific data from a batch running on a DCS.

It can be used for processing requirements, such as:

- Concurrent retrieval of alarms for recording in the batch report
  Alarms on the DCS are retrieved every five minutes. In case an alarm has occurred, the alarm can be converted into an exception. Then it is documented in the batch report and included in the review and approval process.
- Retrieval of alarms after completion of a batch run for recording in the batch report
  After a batch run has been completed, all alarms that have occurred during the run are retrieved and converted into exceptions. They are documented in the batch report and included in the review and approval process.

Execution

In addition to the instruction text, the **Get DCS alarms** phase displays the following data:

- the filter criteria applied when the system retrieves alarms from the DCS.
  - DSC name
  - Start and end of the retrieval period
  - Batch ID
  - Unit ID
  - Modules
- a table that lists all alarms that have been retrieved and have not yet been converted into exceptions. For each retrieved alarm, the system displays the following information:
  - the timestamp when the alarm occurred,
  - the source of the alarm,
  - the alarm text as created by the DCS,
  - an alarm comment, if a comment has already been added in the DCS.
- an information message shows the status of the retrieval:

- **Time of last update unknown** indicates that no retrieval has been performed yet,
- **Update in progress** indicates that the operator has tapped the **Get** button and the retrieval process is ongoing,
- **Last update completed (<timestamp>)** indicates that the last retrieval was successful and when it took place,
- **Last update failed (<timestamp>)** indicates that the last retrieval encountered an issue and when the issue occurred.

When the operator taps the **Get** button, the phase accesses the DCS and retrieves all alarms that match the filter criteria. The operator can tap the button multiple times in order to retrieve any further matching alarms that have occurred since the last **Get** operation.
If the phase is set to retrieve alarms automatically (page 50), it accesses the DCS and queries for alarms as soon as it becomes active. Afterwards, it continues to query the DCS at the defined interval until the operator taps the **Confirm** button to complete the phase.
Alarms are sorted by their timestamps in descending order. This means that newly retrieved alarms always appear at the top of the list of alarms.

TIP

Please note that when the system retrieves and lists new alarms, it considers all unit procedures of the order or workflow and only displays those alarms that have not yet been processed by any other phase or phase run.

Once the phase has retrieved alarms from the DCS, it displays them in the table with one row per alarm. The operator can tap to select individual alarm rows or select all alarms at once by tapping the checkbox in the table header. Selecting an alarm enables the **Exception** button to convert the alarm into an exception (page 51). The exception indicates the number of converted alarms and shows the alarm details received from the DCS as comment, one comment per alarm. Alarms that have been converted are removed from the table.

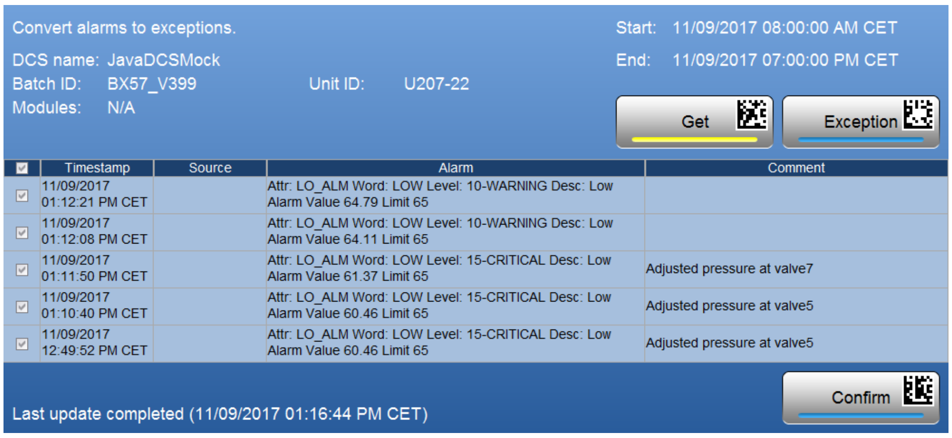
The phase expects all alarms to be converted into exceptions. If the operator tries to complete the phase when there are still alarms listed in the table, he can only do this by recording an exception for the unconverted alarms (page 51).

TIP

The system can also be configured to allow completing the phase without having converted the retrieved alarms into exceptions.

Tapping the Confirm **button** always triggers another alarms retrieval to ensure that the operator can only complete the phase if there are no unprocessed alarms.

After completion the phase displays the number of retrieved alarms in the Execution Window.
The Navigator displays the number of retrieved alarms.


Convert alarms to exceptions.
DCS name: JavaDCSMock
Batch ID: BX57_V399 Unit ID: U207-22
Modules: N/A
Start: 11/09/2017 08:00:00 AM CET
End: 11/09/2017 07:00:00 PM CET

✓	Timestamp	Source	Alarm	Comment
✓	11/09/2017 01:12:21 PM CET		Attr: LO_ALM Word: LOW Level: 10-WARNING Desc: Low Alarm Value 64.79 Limit 65	
✓	11/09/2017 01:12:08 PM CET		Attr: LO_ALM Word: LOW Level: 10-WARNING Desc: Low Alarm Value 64.11 Limit 65	
✓	11/09/2017 01:11:50 PM CET		Attr: LO_ALM Word: LOW Level: 15-CRITICAL Desc: Low Alarm Value 61.37 Limit 65	Adjusted pressure at valve7
✓	11/09/2017 01:10:40 PM CET		Attr: LO_ALM Word: LOW Level: 15-CRITICAL Desc: Low Alarm Value 60.46 Limit 65	Adjusted pressure at valve5
✓	11/09/2017 12:49:52 PM CET		Attr: LO_ALM Word: LOW Level: 15-CRITICAL Desc: Low Alarm Value 60.46 Limit 65	Adjusted pressure at valve5

Last update completed (11/09/2017 01:16:44 PM CET)

Figure 17: Get DCS alarms during execution

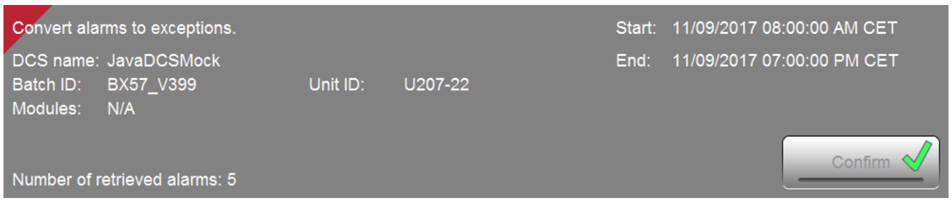

Convert alarms to exceptions.
DCS name: JavaDCSMock
Batch ID: BX57_V399 Unit ID: U207-22
Modules: N/A
Start: 11/09/2017 08:00:00 AM CET
End: 11/09/2017 07:00:00 PM CET
Number of retrieved alarms: 5

Figure 18: Get DCS alarms after phase completion

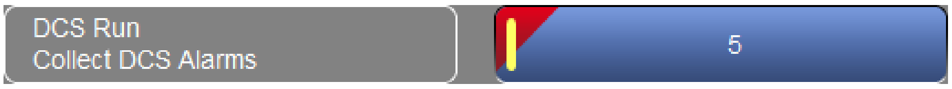


Figure 19: Get DCS alarms in the Navigator

Phase Design

The characteristics of the **Get DCS alarms** phase are defined via process parameters and their attributes.
Its user interface is designed in three columns that span several rows. When the phase is active, the merged columns of the first row provide space for textual instructions. The right column of the first row shows the timestamp of the query start and below that, in the second row, the query end timestamp.
The second, third, and fourth rows of the left column contain the remaining filter criteria, as well as the third row of the center column.
In the fourth row of the right column, the phase provides the **Get** and **Exception** buttons.
The next rows display a tabular view of retrieved alarms and their data, spanning all columns.
The merged left and center columns of the bottom row display the information message on the status of the alarms retrieval.
The right column of the bottom row contains the **Confirm** button.

When the phase is completed it shows the same three-column, multi-row layout, however without the **Get** and **Exception** buttons and a text row with the number of retrieved alarms instead of the table.
Exception handling during execution is controlled by a risk assessment classification and an exception message that are both defined by the recipe author in the exception's process parameter.

Process Parameters

The following process parameters are available to configure the phase's behavior during execution:

Instruction

Represents the instruction text that is visible on the preview, the active, and the completed view of the phase.

Attribute	Type	Comment
Text	HTML text	Instruction text to be displayed. Maximum length is 2000 characters (including HTML tags).

DCS

Defines the name of the DCS system to be accessed for alarms retrieval.

Attribute	Type	Comment
Name	String	Logical name of the DCS to be used. The available entries correspond to the entries in the **DCSNames** list. Default setting: First entry in the list.

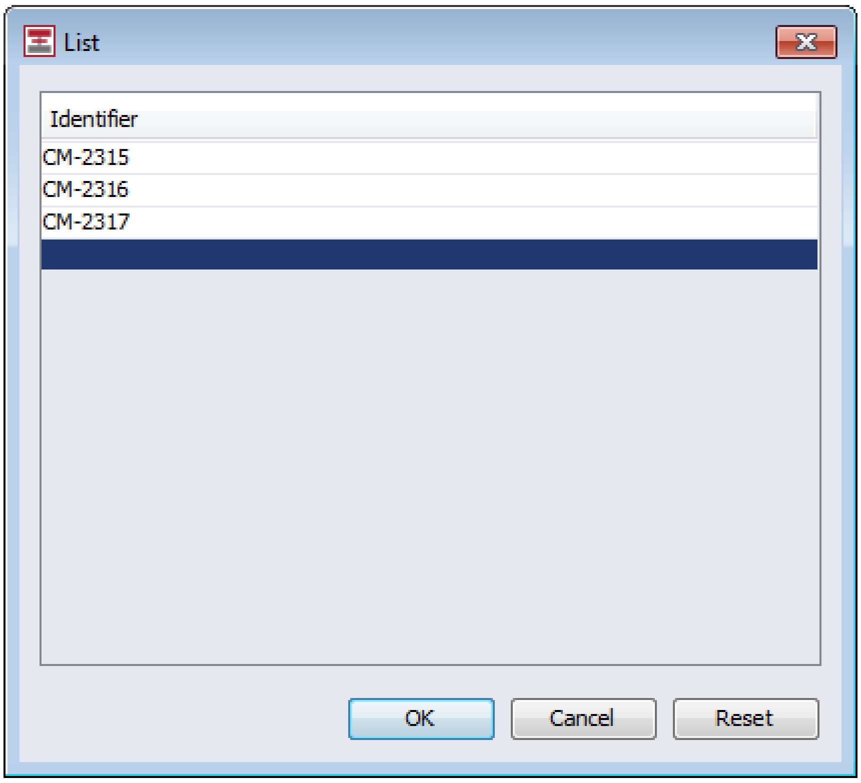
Filter criteria

Defines the filter that is applied to the alarms that occur on the selected DCS.

Attribute	Type	Comment
Batch ID	String	Optional parameter to define the identifier of the batch running on the DCS. If not defined, phase displays "N/A".
Unit ID	String	Optional parameter to define the identifier of a unit of the DCS. If not defined, phase displays "N/A".

Attribute	Type	Comment
Query start	Timestamp	Optional parameter to set the start timestamp from which on the alarms are queried. If not defined, phase displays "N/A".
Query end	Timestamp	Optional parameter to set the end timestamp up to which the alarms are queried. If not defined, phase displays "N/A".
Module IDs	Text (structured)	Optional parameter to define the list of equipment module IDs and control module IDs. If not defined, phase displays "N/A".

For compiling the list of module identifiers, the system provides a List editor.



Identifier
CM-2315
CM-2316
CM-2317

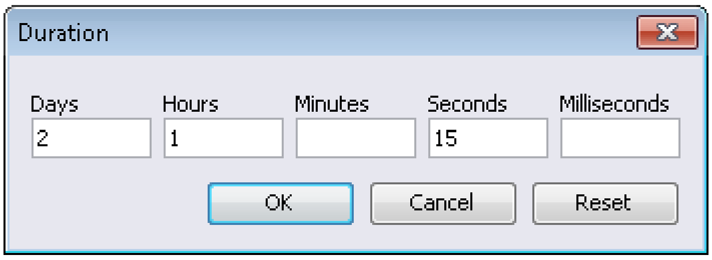
Figure 20: List editor

Automatic update

Defines if the phase accesses the DCS automatically at a given interval without further operator interaction.

Attribute	Type	Comment
Enabled	Boolean	Controls if the query is repeated automatically. If so, make sure to define the **Update interval** attribute.
Update interval	Duration	Defines the interval between re-querying. The minimum interval is set to 1 minute if the interval is not defined at all or configured to be less than that.

For defining the interval, the system provides a Duration editor.


Duration

Days	Hours	Minutes	Seconds	Milliseconds
2	1		15	

Figure 21: Duration editor

Retrieval exception

Represents a system-triggered exception that is displayed in the Exception Window to record incidents when the DCS cannot be accessed for retrieving alarms.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Unconverted alarms

Represents a system-triggered exception that is displayed in the Exception Window to record incidents when one or more alarms that were retrieved from the DCS are not converted into exceptions before the phase is completed.

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. Default setting: **Yes**
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Record alarm exception

Represents a user-triggered exception that is accessed via the **Exception** button above the table of alarms in the Execution Window.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Output Variables

Instead of specifying a fixed value to be displayed or used during execution, you can also use an expression created in the Expression editor to draw the output of another phase or the calculated result of several phase outputs as value into a parameter attribute. When you reference phase outputs in this manner you need to be aware of the following restrictions:

- Only if a phase has been processed does it provide an output that can be fed into another phase as attribute value. For this reason, you can never reference an output of a phase that is a strict successor of the phase in which you try to use the output.
- Branches and loops, however, require special notice in this context, since they are only potentially passed through and/or completed during processing, so their outputs are not reliably available. Thus you can reference any such potentially available outputs, but need to be aware of the fact that the provided value may be **Undefined** so that the phase to which you are feeding the output must be able to deal with such an **Undefined** input value.

The **Get DCS alarms** phase provides the following output variables:

Number of retrieved alarms

- Data type: Long, used for integral numbers:
  12345
- Usage: The output variable provides the number of alarms that were retrieved by the phase.

Number of converted alarms

- Data type: Long, used for integral numbers:
  12345
- Usage: The output variable provides the number of alarms that were converted into exceptions.

Identifier

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Read Instruction**".
- Usage: The output variable provides the identifier of the phase.

Instance count

- Data type: Long, used for integral numbers:
  12345
- Usage: The output variable provides the count of the number of instances the phase has been processed, for example in a loop. The count is also increased when the phase is skipped from an operator's perspective, since the phase is still executed, but as a hidden phase.
  The count variable of a phase that has not been executed provides 0 as output value.

Start time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the start time of the phase.

Completion time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the completion time of the phase.

TIP

To calculate a duration from two timestamps and display it in a specific format, you need to use two conversion functions on the calculation:

- **Convert to Unitless Number** (**convertTo**) takes the calculated duration and converts it into the duration's value for one of its units (e.g. minutes or seconds).
- **Convert to String for Display** (**convertToDisplayString)** takes the converted value and displays it as string to which you can add the unit, also as string.

Example:

Sample Phase with Start time = 14-Nov-2014@10:15
Sample Phase with Completion time = 14-Nov-2014@11:47
The duration is to be displayed in minutes.

```
convertToDisplayString
   (convertTo
        ({Sample Phase}.{Completion time}-{Sample Phase}.{Start time},
"min")
   )
 + " min"
```

As result of the expression, the system displays "**92 min**".

DCS Alarm-based Trigger

The **DCS alarm-based trigger** phase allows to automatically create runs of an event-triggered operation (ETO) based on DCS alarms whenever new DCS alarms are retrieved in the defined check cycle.

It can be used for processing requirements, such as:

- Several operators are responsible for a production area in which several orders are executed. The operators have to be notified when new alarms have been retrieved from the DCS. Subsequently, an operator can document the alarms as PharmaSuite exceptions.

Execution

The **DCS alarm-based trigger** phase is intended for being run on the Operation Execution (OE) server. It generates events to trigger the runs of event-triggered operations, such as operations that check for alarms that have occurred on a connected DCS.
As a server-run phase it starts automatically and becomes active when an operator starts an operation that references the phase as its trigger phase. Once active it generates trigger events according to its configured schedule.
The phase completes automatically when the last operation whose runs it triggers is completed by an operator.

Phase Design

The characteristics of the **DCS alarm-based trigger** phase are defined via process parameters and their attributes.
Since it runs on the OE server and does not require operator interaction, it is not visible to operators during processing.
When the phase is completed, the batch report shows the configuration of the alarm query, as well as the number of alarms it retrieved and how many triggers it sent while it was active.

Process Parameters

The following process parameters are available to configure the phase's behavior during execution:

DCS

Defines the name of the DCS to be accessed for alarms retrieval.

TIP

Make sure the DCS you select is identical with the DCS selected for the **Get DCS alarms** phase of the operation whose **Trigger-enabled** capability references the trigger phase.

Attribute	Type	Comment
Name	String	Logical name of the DCS to be used. The available entries correspond to the entries in the **DCSNames** list. Default setting: First entry in the list.

Filter criteria

Defines the filter that is applied to the alarms that occur on the selected DCS.

TIP

Make sure the filter criteria you define are identical with those defined for the **Get DCS alarms** phase of the operation whose **Trigger-enabled** capability references the trigger phase.

Attribute	Type	Comment
Batch ID	String	Optional parameter to define the identifier of the batch running on the DCS. If not defined, phase displays "N/A".
Unit ID	String	Optional parameter to define the identifier of a unit of the DCS. If not defined, phase displays "N/A".
Query start	Timestamp	Optional parameter to set the start timestamp from which on the alarms are queried. If not defined, phase displays "N/A".

Attribute	Type	Comment
Query end	Timestamp	Optional parameter to set the end timestamp up to which the alarms are queried. If not defined, phase displays "N/A".
Module IDs	Text (structured)	Optional parameter to define the list of equipment module IDs and control module IDs. If not defined, phase displays "N/A".

For compiling the list of module identifiers, the system provides a List editor.

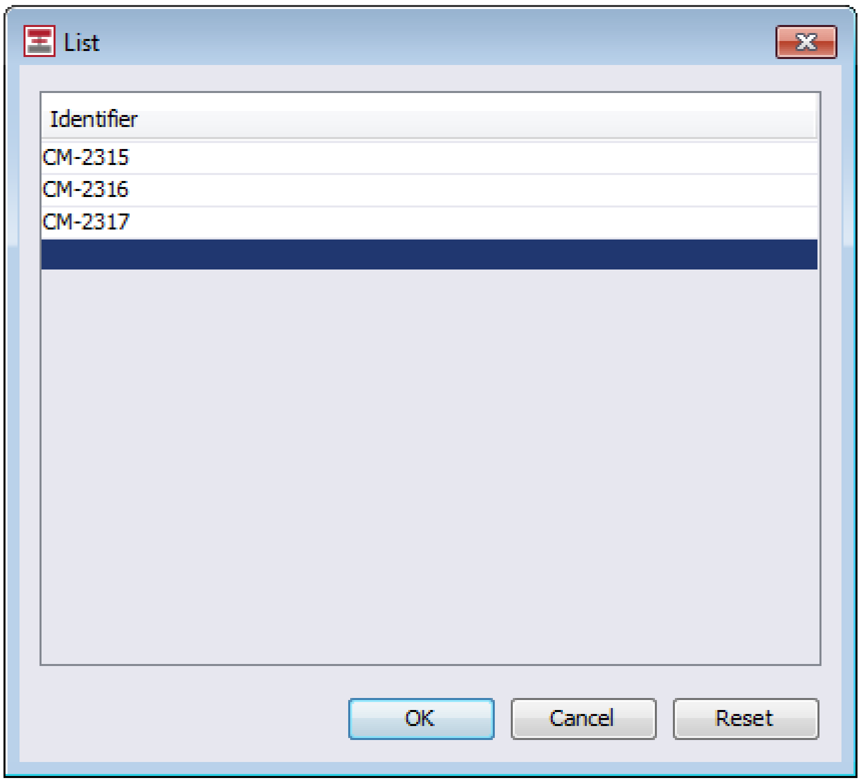


Figure 22: List editor

Retrieving cycle

Defines the interval between two consecutive reading actions.

Attribute	Type	Comment
Duration	Duration	The minimum interval is one minute. So if you set a value of less than one minute or leave the cell blank, the system interprets this as one minute.

For defining the interval, the system provides a Duration editor.

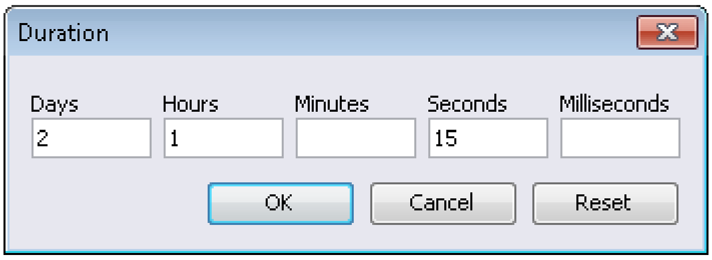


Figure 23: Duration editor

Timeout period

Defines how long the phase waits for its first event-triggered operation to be started before it completes automatically. This prevents deadlocks that might occur if an event-triggered operation fails to start entirely.

Attribute	Type	Comment
Duration	Duration	If left blank, the system interprets this as 30 minutes.

For defining the period, the system provides a Duration editor.

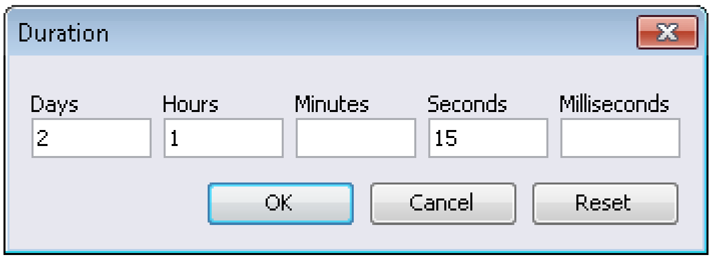


Figure 24: Duration editor

Timeout exception

Represents a system-triggered exception that is displayed in the overview Exception Window and in the batch report.
The system records a timeout incident as exception, since the quality of a product may be compromised if scheduled events, such as reading alarms, do not take place.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception and thus controls the related signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Retrieval exception

Represents a system-triggered exception that is displayed in the overview Exception Window and in the batch report.
The system records a failed retrieval incident as exception, since the quality of a product may be compromised if scheduled events, such as reading alarms, cannot be executed.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception. Since there is no operator interaction for the exception, it is not linked to a signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Post-ETO alarms exception

Represents a system-triggered exception that is displayed in the overview Exception Window and in the batch report.
The system records an exception when new alarms have occurred in the DCS after the ETO template was removed and before the corresponding **Get DCS alarms** phase was completed, since this may compromise product quality.

Attribute	Type	Comment
Risk assessment	Choice list	Defines the risk level of the exception. Since there is no operator interaction for the exception, it is not linked to a signature privilege. Available settings: **None**, **Low**, **Low (mandatory comment)**, **Medium**, **Medium (mandatory comment)**, **High**, **High (mandatory comment)**. Default setting: **High**.
Exception text	Text	Defines the exception description used during exception handling and within the batch record. Maximum length is 250 characters.

Output Variables

Instead of specifying a fixed value to be displayed or used during execution, you can also use an expression created in the Expression editor to draw the output of another phase or the calculated result of several phase outputs as value into a parameter attribute. When you reference phase outputs in this manner you need to be aware of the following restrictions:

- Only if a phase has been processed does it provide an output that can be fed into another phase as attribute value. For this reason, you can never reference an output of a phase that is a strict successor of the phase in which you try to use the output.
- Branches and loops, however, require special notice in this context, since they are only potentially passed through and/or completed during processing, so their outputs are not reliably available. Thus you can reference any such potentially available outputs, but need to be aware of the fact that the provided value may be **Undefined** so that the phase to which you are feeding the output must be able to deal with such an **Undefined** input value.

The **DCS alarm-based trigger** phase provides the following output variables:

Identifier

- Data type: String, used for displaying a pre-defined sequence of characters, such as "**Read Instruction**".
- Usage: The output variable provides the identifier of the phase.

Instance count

- Data type: Long, used for integral numbers:
  12345
- Usage: The output variable provides the count of the number of instances the phase has been processed, for example in a loop. The count is also increased when the phase is skipped from an operator's perspective, since the phase is still executed, but as a hidden phase.
  The count variable of a phase that has not been executed provides 0 as output value.

Start time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the start time of the phase.

Completion time

- Data type: Timestamp, used for displaying dates and times and for time-related calculations.
  To use a timestamp in a phase attribute, you have to make sure it has a matching data type, so to display it in an instruction text, you have to convert it into a string.
- Usage: The output variable provides the completion time of the phase.

TIP

To calculate a duration from two timestamps and display it in a specific format, you need to use two conversion functions on the calculation:

- **Convert to Unitless Number** (**convertTo**) takes the calculated duration and converts it into the duration's value for one of its units (e.g. minutes or seconds).
- **Convert to String for Display** (**convertToDisplayString)** takes the converted value and displays it as string to which you can add the unit, also as string.

Example:

Sample Phase with Start time = 14-Nov-2014@10:15
Sample Phase with Completion time = 14-Nov-2014@11:47
The duration is to be displayed in minutes.

```
convertToDisplayString
   (convertTo
        ({Sample Phase}.{Completion time}-{Sample Phase}.{Start time},
"min")
   )
 + " min"
```

As result of the expression, the system displays "**92 min**".

Index