

LISTEN.
THINK.
SOLVE.®

PharmaSuite®



IPC PHASES

RELEASE 8.4

FUNCTIONAL REQUIREMENT SPECIFICATION

PUBLICATION PSFRSIP-RM003E-EN-E-DECEMBER-2017

Supersedes publication PSFDIP-RM003D-EN-E



Allen-Bradley • Rockwell Software

**Rockwell
Automation**

Contact Rockwell See contact information provided in your maintenance contract.

Copyright Notice © 2017 Rockwell Automation Technologies, Inc. All rights reserved.
This document and any accompanying Rockwell Software products are copyrighted by Rockwell Automation Technologies, Inc. Any reproduction and/or distribution without prior written consent from Rockwell Automation Technologies, Inc. is strictly prohibited. Please refer to the license agreement for details.

Trademark Notices FactoryTalk, PharmaSuite, Rockwell Automation, Rockwell Software, and the Rockwell Software logo are registered trademarks of Rockwell Automation, Inc.

The following logos and products are trademarks of Rockwell Automation, Inc.:

FactoryTalk Shop Operations Server, FactoryTalk ProductionCentre, FactoryTalk Administration Console, FactoryTalk Automation Platform, and FactoryTalk Security.
Operational Data Store, ODS, Plant Operations, Process Designer, Shop Operations, Rockwell Software CPGSuite, and Rockwell Software AutoSuite.

Other Trademarks ActiveX, Microsoft, Microsoft Access, SQL Server, Visual Basic, Visual C++, Visual SourceSafe, Windows, Windows 7 Professional, Windows Server 2008, Windows Server 2012, and Windows Server 2016 are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Adobe, Acrobat, and Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

ControlNet is a registered trademark of ControlNet International.

DeviceNet is a trademark of the Open DeviceNet Vendor Association, Inc. (ODVA).

Ethernet is a registered trademark of Digital Equipment Corporation, Intel, and Xerox Corporation.

OLE for Process Control (OPC) is a registered trademark of the OPC Foundation.

Oracle, SQL*Net, and SQL*Plus are registered trademarks of Oracle Corporation.

All other trademarks are the property of their respective holders and are hereby acknowledged.

Warranty This product is warranted in accordance with the product license. The product's performance may be affected by system configuration, the application being performed, operator control, maintenance, and other related factors. Rockwell Automation is not responsible for these intervening factors. The instructions in this document do not cover all the details or variations in the equipment, procedure, or process described, nor do they provide directions for meeting every possible contingency during installation, operation, or maintenance. This product's implementation may vary among users.

This document is current as of the time of release of the product; however, the accompanying software may have changed since the release. Rockwell Automation, Inc. reserves the right to change any information contained in this document or the software at any time without prior notice. It is your responsibility to obtain the most current information available from Rockwell when installing or using this product.

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Chapter 1	Introduction	1
	Typographical Conventions	1
Chapter 2	IPC-related Phases and Operations	3
	Trigger Phases	4
	How to Model Recipes with ETOs and Trigger Phases	5
	Data Collection and Data Representation Phases	12
	Combination of the Data Collection and Data Representation Phases	12
	Scope of Data Collection	13
Chapter 3	Time-based Trigger Phase (SR0400+)	15
	Layout	15
	Representation during Execution	15
	Representation in Navigator	15
	Representation in Sub-report (SR0400.5+)	15
	Representation in Sub-record	16
	Business Logic (SR0400.2+).....	16
	Process Parameters (SR0400.8+)	19
	Exceptions (SR0400.3+).....	20
	System-triggered Exceptions (SR0400.3.2+).....	20
	Output Variables.....	21
Chapter 4	Counter-based Trigger Phase (SR0405+)	23
	Layout	23
	Representation during Execution	23
	Representation in Navigator	23

	Representation in Sub-report (SR0405.5+)	23
	Representation in Sub-record	24
	Business Logic (SR0405.2+).....	24
	Process Parameters (SR0405.8+)	29
	Exceptions (SR0405.3+).....	32
	System-triggered Exceptions (SR0405.3.2+).....	32
	Output Variables.....	34
Chapter 5	Get Values Phase (SR0440+)	37
	Layout	38
	Representation during Execution (SR0440.1+)	38
	Representation in Navigator (SR0440.4+).....	40
	Representation in Sub-report (SR0440.5+)	41
	Business Logic (SR0440.2+).....	41
	Measured Value Bundle	42
	Option Value Bundle	43
	Boolean Value Bundle.....	44
	Process Parameters (SR0440.8+)	45
	Measured Value Bundle	46
	Option Value Bundle	49
	Boolean Value Bundle.....	53
	Exceptions (SR0440.3+).....	56
	System-triggered Exceptions (SR0440.3.2+).....	56
	User-triggered Exceptions (SR0440.3.1+)	59
	Post-completion Exceptions (SR0440.3.3+).....	62
	Information Messages	67
	Questions	67
	Decisions	67
	Error Messages (SR0440.3.6+)	67
	Output Variables (SR0440.9+)	67
	Measured Value Bundle	68
	Option Value Bundle	69

	Boolean Value Bundle	69
Chapter 6	Show Values Phase (SR0450+)	71
	Layout	72
	Representation during Execution (SR0450.1+)	72
	Representation in Navigator (SR0450.4+)	74
	Representation in Sub-report (SR0450.5+)	74
	Business Logic (SR0450.2+)	75
	Process Parameters (SR0450.8+)	77
	Statistics Bundle	77
	Exceptions (SR0450.3+)	78
	System-triggered Exceptions	78
	User-triggered Exceptions	78
	Post-completion Exceptions	78
	Information Messages (SR0450.3.4+)	79
	Questions	79
	Decisions	79
	Error Messages	79
	Output Variables (SR0450.9+)	79
	Statistics Bundle	80
Chapter 7	Reference Documents	83
Chapter 8	Document Information	85
	Approval	85
	Version Information	85
	Revision History	85
Index	89

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Figure 1: Event-triggered operation within a recipe	4
Figure 2: Issue 1: Trigger phase is activated with delay.....	5
Figure 3: Issue 2: ETO template is activated with delay.....	6
Figure 4: Solution for issue 1, 2: Start ETOs simultaneously	7
Figure 5: Issue 3: Subsequent ETO templates reference the same trigger phase.....	8
Figure 6: Solution for issue 3: Provide sequential trigger phases for sequential ETO templates	9
Figure 7: Issue 4: Loop within a trigger/ETO-related parallel branch	10
Figure 8: Solution for issue 4: Loop includes the trigger/ETO-related parallel branch.....	11
Figure 9: Data collection across operation runs.....	13
Figure 10: Data collection within an operation run	13
Figure 11: Get values during execution	38
Figure 12: Show values during execution.....	72

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Introduction

This document details the requirements of the functions implemented by the phases specific to in-process control (IPC). The phases are either related to an event-triggered operation and executed in the Production Execution Client or to a server-run operation and executed on the Operation Execution server (OES) of PharmaSuite.

Each requirement is composed of a name and a unique identifier (e.g. Instruction (SR0450.8.1)). If a requirement's meaning is for requirement grouping only, the identifier is appended by a plus sign (e.g. Process parameters (SR0450.8+)).

For requirements with **Framework capability** as identifier, see "Functional Requirement Specification Execution Framework" for their unique identifier, [A1] (page 83).

The revision history (page 85) lists the changes made to the document with PharmaSuite 8.3 as the comparison baseline. Changes related to a requirement are marked as "Editorial", "Update", "New", or "Deleted", changes to the additional context information are marked as "Context information-related".

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 <ul style="list-style-type: none"> ■ window and dialog titles ■ menu functions ■ panel, tab, and button names ■ box labels ■ object properties and their values (e.g. status).
Monospaced typeface	Designates code examples.

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

IPC-related Phases and Operations

PharmaSuite for Production Execution uses event-triggered operations (ETOs) as templates to create specific runs (ETO instances), which then are executed by the operator.

The creation of the runs is triggered either manually by an operator or automatically by a trigger phase. This system behavior specifically supports In-process control (IPC)-related use cases, but it can also be applied to other use cases.

The typical structure of a recipe with an event-triggered operation can be modeled with the following characteristics:

- An operation with the **Event-triggered** capability represents an ETO. The runs can be created automatically by a trigger phase if the operation also holds the **Trigger-enabled** capability.
- For automatic triggers, the trigger phases are located in an operation that holds the **Server-run** capability. Thus the operation and its phases are not visible in the Production Execution Client.
- Both the **ETO template** and the **Server-run operation with trigger phase** operations are located on parallel branches, which means both operations become active during execution at the same time.
- The **Trigger-enabled** capability allows to reference specific trigger phases that typically run on a server (within a server-run operation).
One ETO can reference multiple trigger phases.
One trigger phase can be referenced from multiple ETOs.

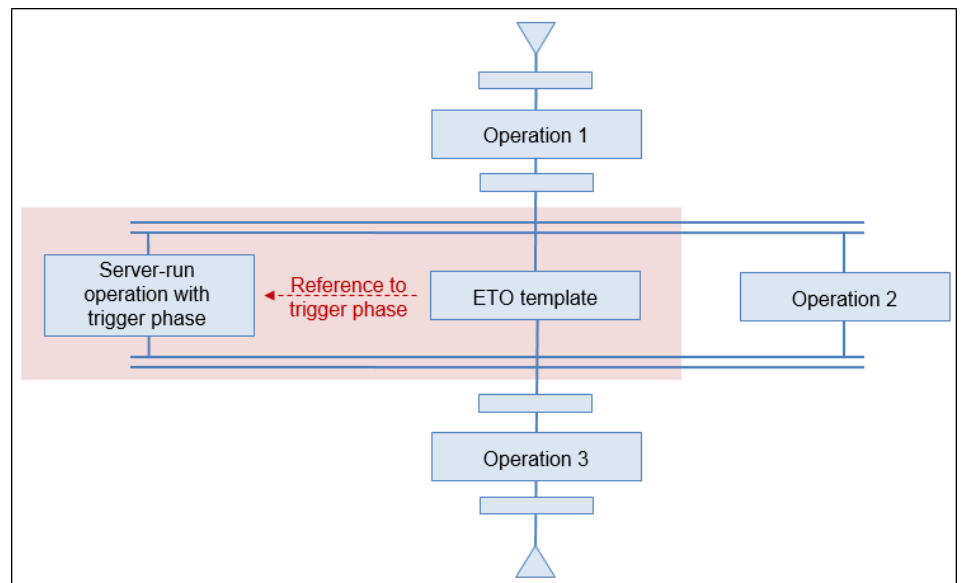


Figure 1: Event-triggered operation within a recipe

Trigger Phases

The following trigger phases are available:

- Time-based trigger (page 15)

The **Time-based trigger** phase allows to automatically create runs of an event-triggered operation (ETO) based on a time-related cycle.

The phase is designed for being run on a server without user interaction.

- Counter-based trigger (page 23)

The **Counter-based trigger** phase allows to automatically create runs of an event-triggered operation (ETO) based on a counter-based cycle.

The phase is designed for being run on a server without user interaction.

The following rules apply with respect to start and completion of trigger processing of a trigger phase. For details, see **Business Logic (SR0400.2+)** of the Time-based trigger phase (page 16) and **Business Logic (SR0405.2+)** of the Counter-base trigger phase (page 24).

- A trigger phase becomes active automatically according to SFC, but trigger processing does not start until at least one related ETO template has become active. That means that the template is visible in the Cockpit of all running Production Execution Clients, according to their station-level dispatching.
- If none of the related ETO templates becomes active, the trigger phase is completed automatically after its timeout period has elapsed.
- If trigger processing has started due to active ETO templates, the trigger phase is completed automatically as soon as there is no related ETO template active any

more. That means that the active ETO templates have been removed from the Cockpit by the operator.

- In case a unit procedure is paused by the operator, also the trigger processing of the trigger phases is paused. When the pause period of the unit procedure is ended by the operator, the trigger processing continues based on a new re-calculated trigger schedule.

How to Model Recipes with ETOs and Trigger Phases

This sections illustrates potential issues that may occur during recipe execution and how to avoid them during recipe design.

ISSUE 1: TRIGGER PHASE IS ACTIVATED WITH DELAY

The delayed activation of a trigger phase is probably caused by the position of **Operation 2** within the trigger/ETO-related parallel branch.

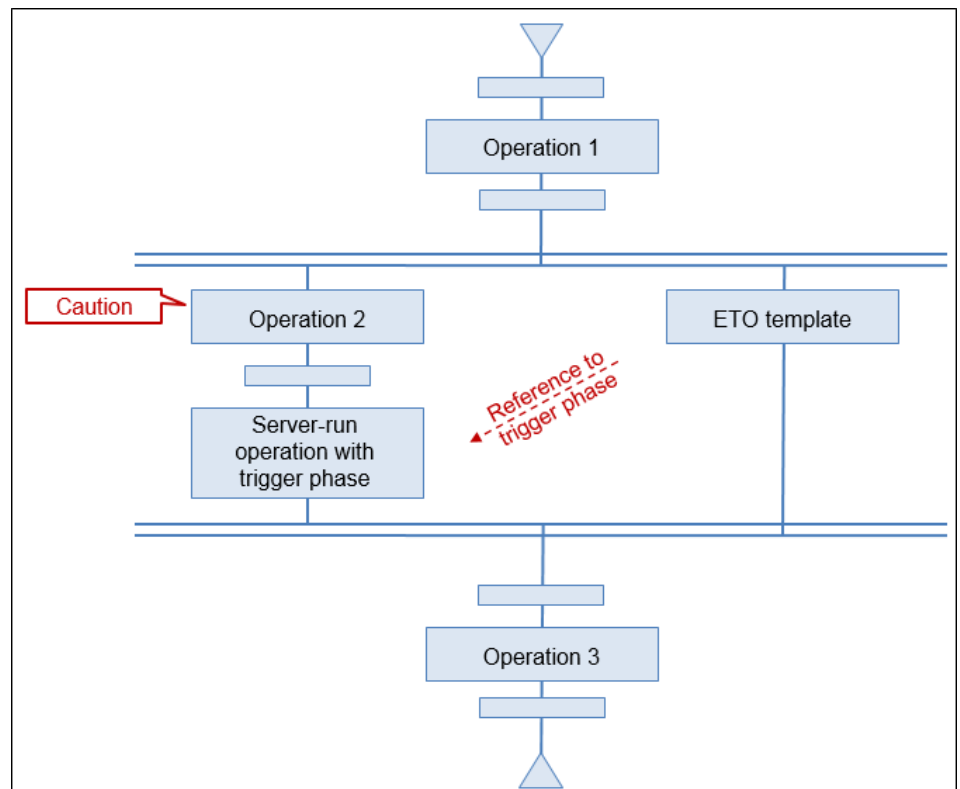


Figure 2: Issue 1: Trigger phase is activated with delay

The recipe design results in the following system behavior:

- The **ETO template** already becomes visible within the Cockpit, while automatic trigger processing is not started yet.
- The trigger phase and its trigger processing are not started unless **Operation 2** has been completed.

ISSUE 2: ETO TEMPLATE IS ACTIVATED WITH DELAY

The delayed activation of an ETO template is probably caused by the position of **Operation 2** within the trigger/ETO-related parallel branch.

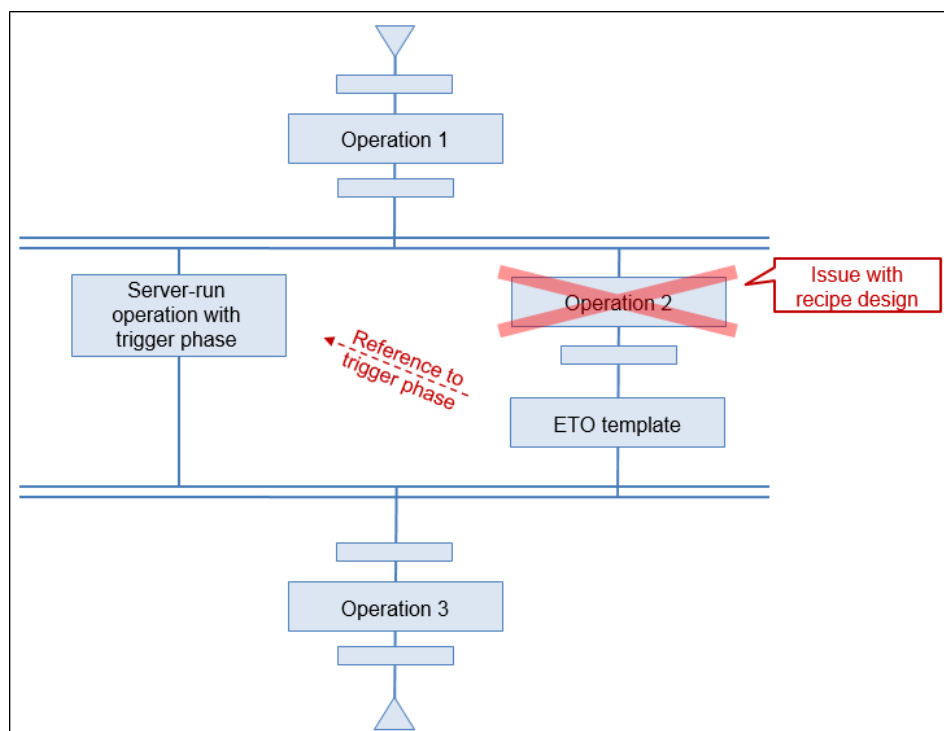


Figure 3: Issue 2: ETO template is activated with delay

The recipe design results in the following system behavior:

- The trigger phase becomes active, but trigger processing does not start unless **Operation 2** has been completed and **ETO template** has been activated.
- Trigger processing only waits for the **ETO template** to be activated within its defined timeout period.
- After the timeout period has elapsed, the trigger phase is completed automatically.
- If the **ETO template** becomes active after the trigger phase has been completed, automatic trigger events are not available for creation of new runs of the **ETO template**.

SOLUTION FOR ISSUE 1, 2: START ETOs SIMULTANEOUSLY

Placing **Operation 2** before the trigger/ETO-related parallel branch makes sure that the trigger phase and the **ETO template** are activated simultaneously and trigger processing can start as expected, per configuration of the trigger phase.

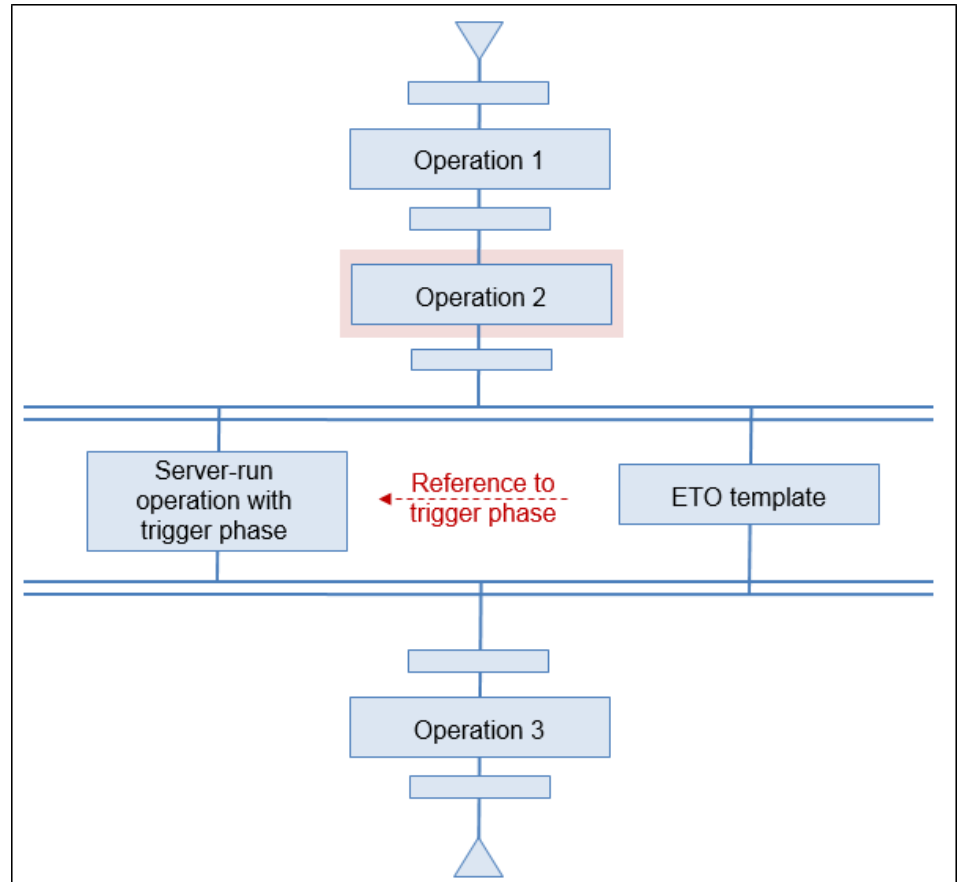


Figure 4: Solution for issue 1, 2: Start ETOs simultaneously

ISSUE 3: SUBSEQUENT ETO TEMPLATES REFERENCE THE SAME TRIGGER PHASE

Subsequent ETO templates must not reference the same trigger phase, assuming that both **ETO templates** should be triggered automatically by the phase.

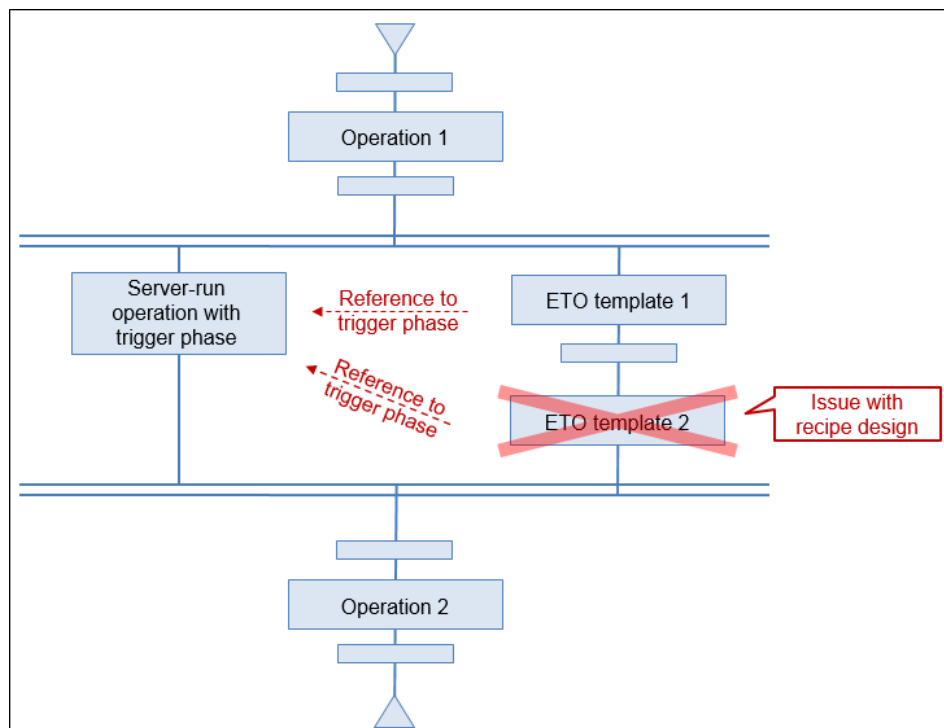


Figure 5: Issue 3: Subsequent ETO templates reference the same trigger phase

The recipe design results in the following system behavior:

- Trigger processing is completed automatically once no ETO template is active any more. In this specific scenario, this is usually the case after **ETO template 1** is completed.
- As a consequence, when **ETO template 2** becomes active, automatic trigger events are no longer available for creating new runs of **ETO template 2**.

SOLUTION FOR ISSUE 3: PROVIDE SEQUENTIAL TRIGGER PHASES FOR SEQUENTIAL ETO TEMPLATES

Place two sequential trigger phases into the server-run operation. This results in the following system behavior:

- **Phase 1** is completed automatically along with the completion of the **ETO template 1**.
- **Phase 1** is ready for trigger processing of **ETO template 2**.

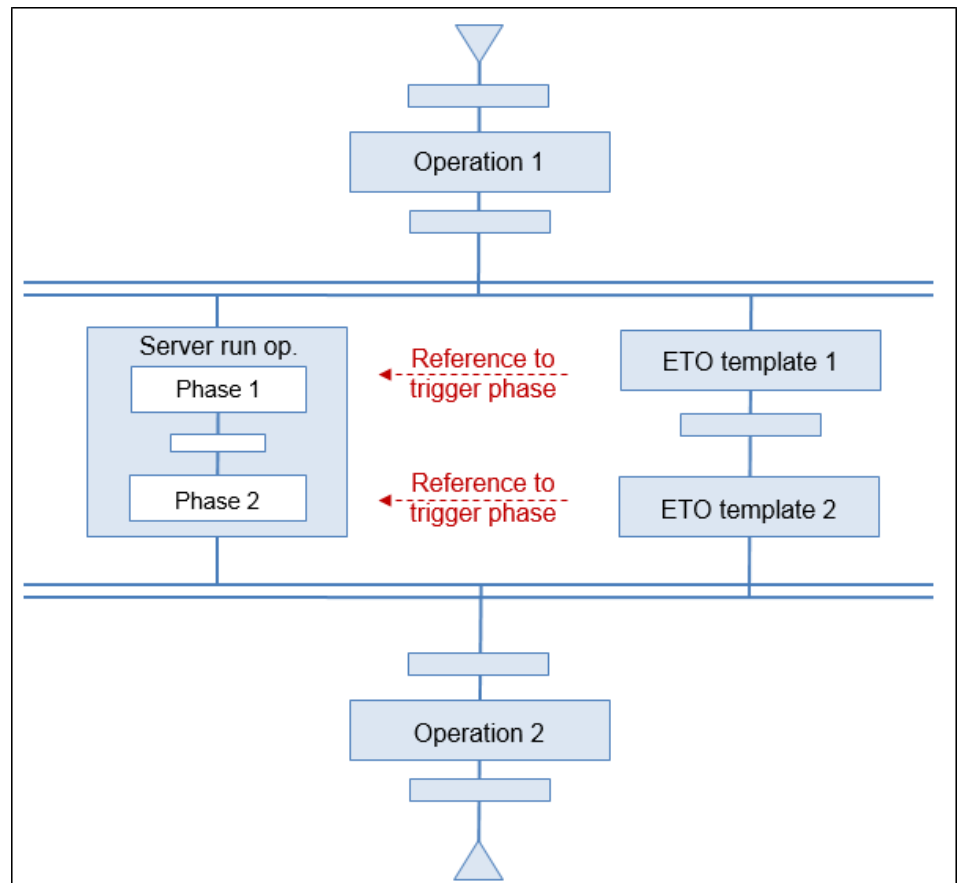


Figure 6: Solution for issue 3: Provide sequential trigger phases for sequential ETO templates

ISSUE 4: LOOP WITHIN A TRIGGER/ETO-RELATED PARALLEL BRANCH

Loops that include an ETO template must not be designed within a trigger/ETO-related parallel branch.

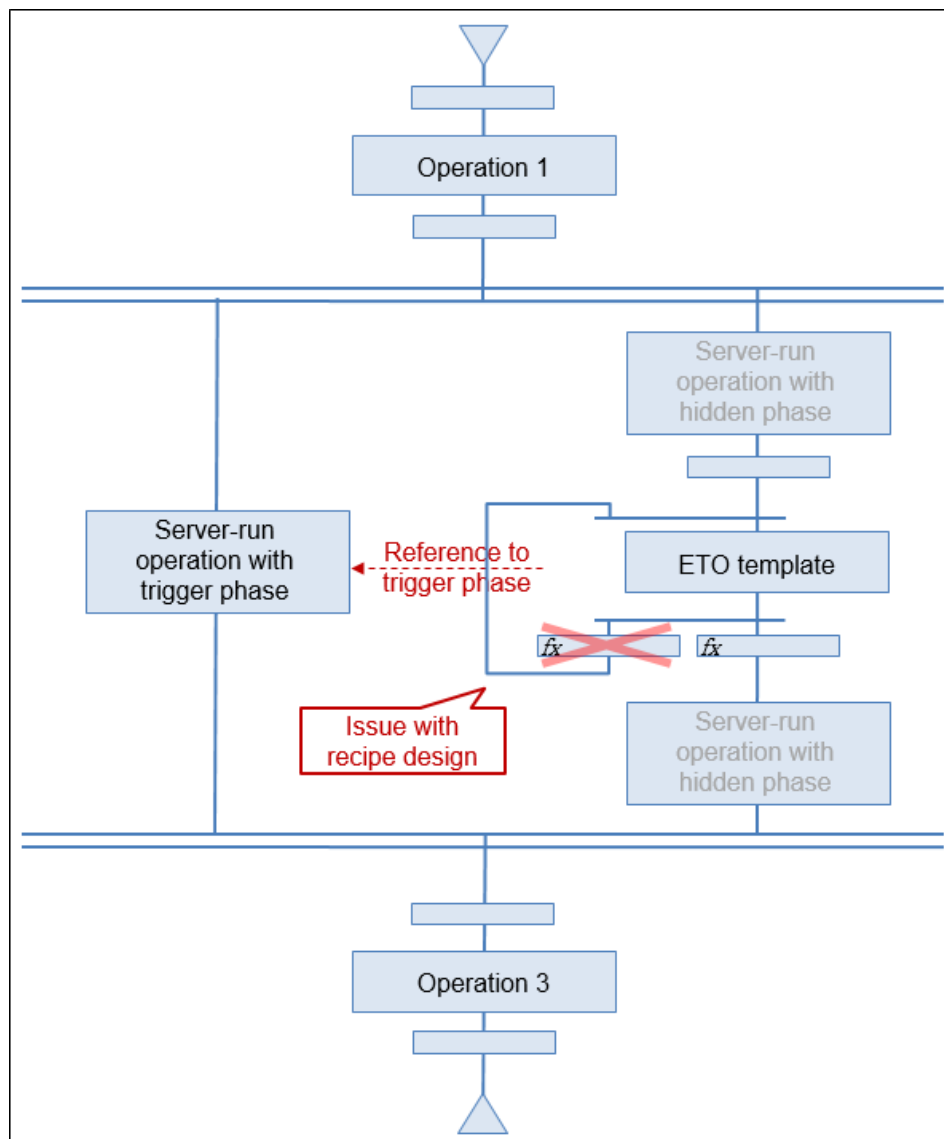


Figure 7: Issue 4: Loop within a trigger/ETO-related parallel branch

The recipe design results in the following system behavior:

- Trigger processing is completed automatically once no ETO template is active anymore, which means after the **ETO template** has been completed for the first time.
- As a consequence, when the **ETO template** is looped and becomes active again, automatic trigger events are no longer available for creating new runs of the ETO template.

SOLUTION FOR ISSUE 4: LOOP INCLUDES THE TRIGGER/ETO-RELATED PARALLEL BRANCH

Loops, if required, always need to enclose the entire trigger/ETO-related parallel branch.

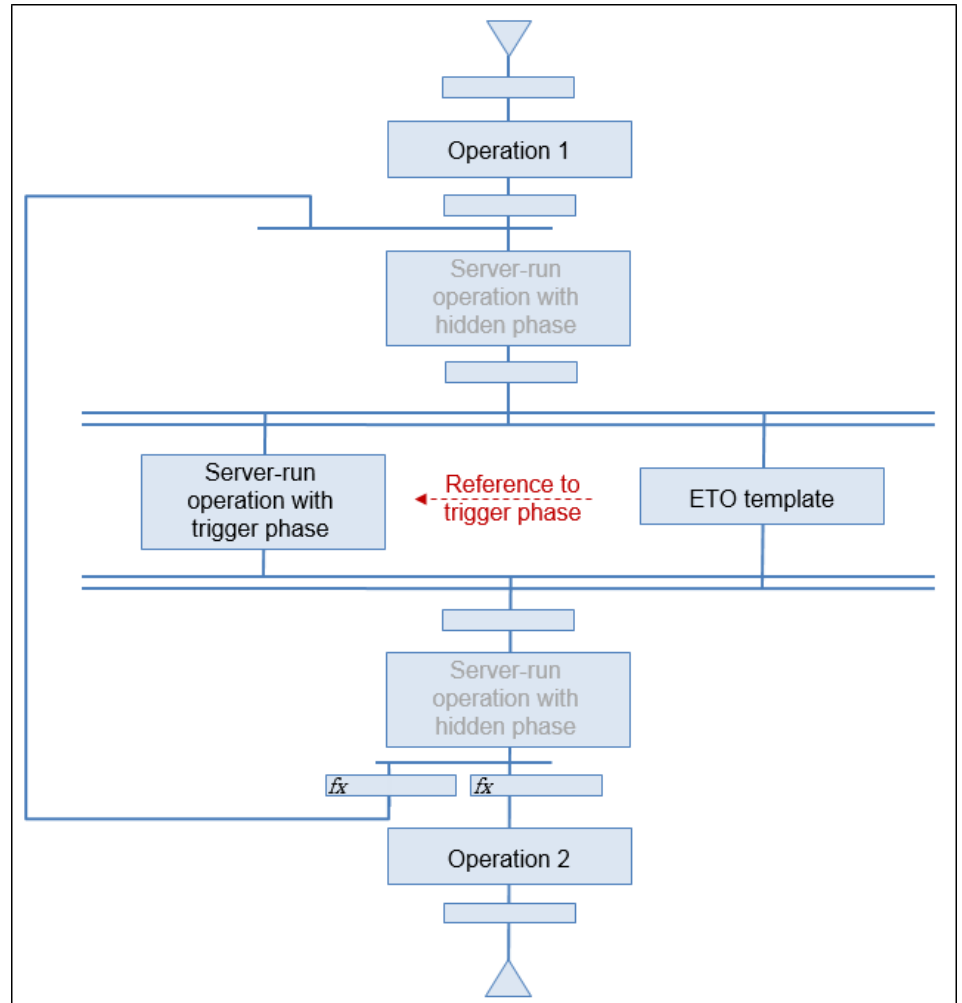


Figure 8: Solution for issue 4: Loop includes the trigger/ETO-related parallel branch

Data Collection and Data Representation Phases

The following data collection and data representation phases are available:

- **Get values** (page [37](#))
The **Get values** phase allows an operator to collect up to five values of the following data types: Measured Value, Option Value (choice list), and Boolean Value.
- **Show values** (page [71](#))
The **Show values** phase allows an operator to view the data that has been collected along with the **Get values** phase (page [37](#)) across multiple runs, which are based on loops or ETO templates. The data is represented in a table format and supports up to five columns with values of the following data types: Measured Value, Option Value (choice list), and Boolean Value.

Combination of the Data Collection and Data Representation Phases

During recipe design, the following rules apply with respect to the combination of **Get values** and **Show values** phases:

- A **Show values** phase always needs to reference a **Get values** phase.
- Both phases must use the same bundle identifiers during configuration.
- A **Show values** phase can be placed within or outside the operation that contains its referenced **Get values** phase.

Scope of Data Collection

The **Show values** phase provides two different modes related to the scope of the data collection.

- Data collection **across** operation runs (default):
Data is collected across all runs of an operation and their phase instances, even across multiple unit procedures if a unit procedure was reactivated.

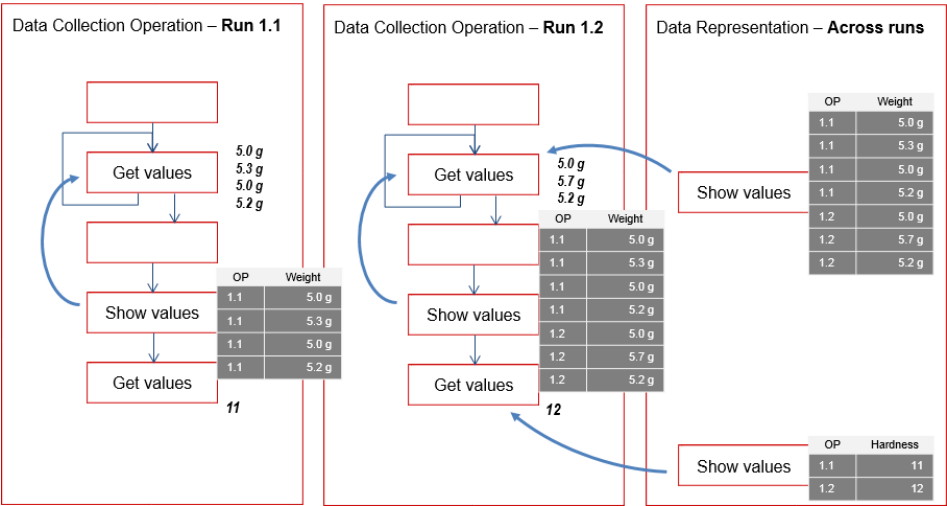


Figure 9: Data collection across operation runs

- Data collection **within** an operation run:
Data is collected only from the phase instances within the given operation.

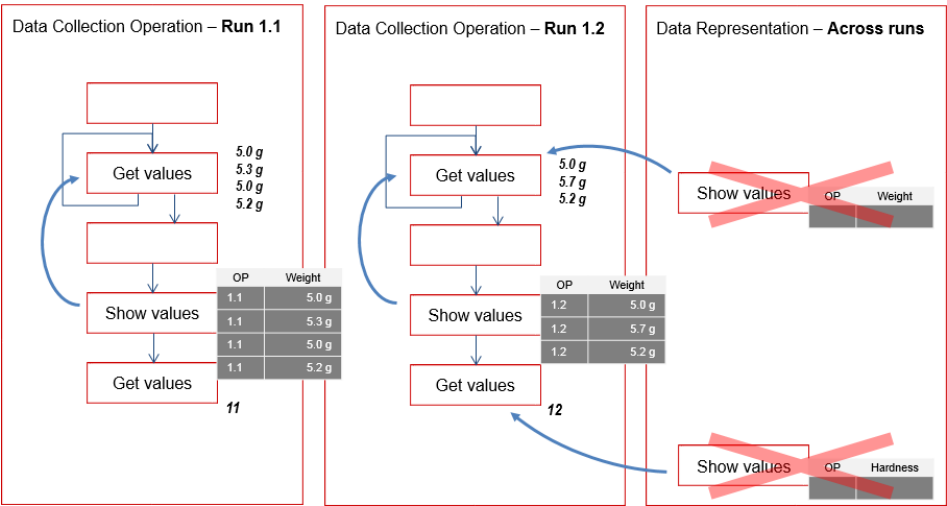


Figure 10: Data collection within an operation run

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Time-based Trigger Phase (SR0400+)

The **Time-based trigger** phase allows to automatically create runs of an event-triggered operation (ETO) based on a time-related cycle.

Example use cases are:

- In manufacturing, average tablet weight and hardness have to be recorded every 30 minutes.
- In packaging, visual checks of filled folding cartons have to be executed every hour.

The number of fired triggers and their time information is stored in the batch record, thereby becoming available for documentation purposes in the sub-report and batch report (page 15).

Anomalies that occur during processing are covered by the phase exception handling (page 20) (e.g. timeout of the phase).

Layout

The phase provides a layout for its representation in the sub-report (page 15).

Representation during Execution

As a server-run phase, the phase has no graphical representation (UI).

Representation in Navigator

As a server-run phase, the phase is not visible in the Navigator.

Representation in Sub-report (SR0400.5+)

The sub-report contains the following information:

Common sub-report elements (Framework capability)

- <Start time>
- <Completion time>
- <Unit procedure> / <operation> / <phase>
- <Work center> / <station> / <device> - <phase completion user>

For phases running on a server, the phase completion-user corresponds to the system.

Sub-report elements (SR0400.5.1)

- Delay time, cycle time
- Timeout
- Number of fired triggers
- List of pause events: Paused from <timestamp> until <timestamp>

Representation in Sub-record

The sub-record contains the following information:

Batch record-related elements (SR0400.5.2)

- Timestamp information of fired triggers

Business Logic (SR0400.2+)

The phase implements the following business logic.

Phase activation (SR0400.2.1)

- Function: Start the trigger processing
- Trigger: Phase becomes active
- Postcondition: Trigger processing is started

Step	#	Description
Phase activation	10	<p>Phase checks</p> <ul style="list-style-type: none">■ if the unit procedure is currently paused (check passes if the unit procedure is not paused),■ if at least one of the relevant ETO templates is already active (check passes if at least one ETO template is active), and■ if the timeout period has not elapsed yet (defined with the Timeout period (SR0400.8.3) process parameter (page 20)) (check passes if an ETO template has become active before the timeout period has elapsed). <p>If all checks have passed, continue with step 10.4.</p>
	10.1	<p>If the unit procedure is paused, the phase waits without any action (no trigger processing, no timeout clock is running) until the unit procedure is continued.</p>

Step	#	Description
	10.2	If the unit procedure is running, no relevant ETO template is active, and the timeout period has not elapsed, trigger processing is still waiting and timeout clock is running.
	10.3	If the timeout period has elapsed without any ETO becoming active, the phase is completed automatically according to the Phase completion (SR0400.2.5) function (page 19).
	10.4	If a relevant ETO template becomes active, trigger processing starts according to the Fire triggers (SR0400.2.2) function (page 17). At this point, the timeout period no longer applies.

Fire triggers (SR0400.2.2)

- Function: Fire the triggers
- Trigger: Trigger processing has started
- Postcondition: Triggers are fired

Step	#	Description
Start trigger processing	10	Phase fires the first trigger after the delay time has elapsed (defined with the Delay time (SR0400.8.1) process parameter (page 19)).
Delay time has elapsed	20	Phase repeatedly fires a trigger, each time the cycle time has elapsed (defined with the Cycle time (SR0400.8.2) process parameter (page 19)).

Pause trigger processing (SR0400.2.3)

- For recent changes, see revision history (page 85).
- Function: Pause/continue the trigger processing
- Precondition: Trigger processing is active
- Trigger: Unit procedure is paused/continued by the operator (for details, see **Pausing a Unit Procedure (SR1089.8.3)** in "Functional Requirement Specification Execution Framework" [A1] (page 83))
- Postcondition: Trigger processing is paused/continued

Step	#	Description
Operator pauses unit procedure	10	At the very moment when the unit procedure of the phase is paused, trigger processing stops. No further triggers are fired automatically.

Step	#	Description
Operator continues paused unit procedure	20.1	At the very moment when the paused unit procedure of the phase is continued, trigger processing continues where it was stopped and continues to fire triggers according to the Fire triggers (SR0400.2.2) function (page 17) and a new re-calculated trigger schedule. The new trigger schedule is based on the fact that the duration of the pause shall not count against the cycle time.
	20.2	In case no relevant ETO template was active during the pause period and the timeout period has not elapsed yet: trigger processing is still waiting and the timeout clock is running, however, the timeout clock is reset upon resume of the pause.

Resume trigger processing (SR0400.2.4)

- Function: Resume the trigger processing
(for the continuation of a paused unit procedure, see **Pause trigger processing (SR0400.2.3)** function (page 17))
- Precondition: Trigger processing is active
- Trigger: Phase is restarted, e.g. along with the restart of the operation that runs on the OES (for details, see **Resuming Server-run Operations (SR1200.1.3)** in "Functional Requirement Specification Execution Framework" [A1] (page 83))
- Postcondition: Trigger processing resumes where it was interrupted

Step	#	Description
Phase is restarted	10	As soon as the phase is restarted, it checks if a trigger was missed.
	10.1	If no trigger was missed, the phase resumes the trigger processing based on the assumption that the cycle time clock has never stopped running. This means, the triggers continues to fire triggers according to the Fire triggers (SR0400.2.2) function (page 17) and according to its original trigger schedule.
	10.2	If triggers were missed during the time the phase was not running, they are considered to be lost and will not be re-fired when the trigger processing is resumed. The phase fires a trigger as soon as reasonable, <ul style="list-style-type: none"> ■ if the unit procedure is not paused, immediately, ■ if the unit procedure is paused, as soon as it is continued, and continues to fire the subsequent triggers according to the Fire triggers (SR0400.2.2) function (page 17) and a re-calculated trigger schedule.

Phase completion (SR0400.2.5)

- Function: Completion of phase
- Trigger: Timeout period has elapsed or no ETO template is active any more
- Postcondition: Phase is completed

Step	#	Description
Timeout period has elapsed	10	If the timeout period has elapsed, the phase is completed automatically without having fired any triggers and creates a Timeout (SR0400.3.2.1) system-triggered exception (page 21).
None of the previously running related ETO templates is active any more	20	Phase stops trigger processing and is completed automatically.

Process Parameters (SR0400.8+)

The following process parameters define the behavior of the phase.

BASIC PARAMETERS**Delay time (SR0400.8.1)**

Attribute	Type	Comment
Duration	Duration	Defines the delay between phase start and its first trigger. Null is interpreted as 0 seconds during execution.

Cycle time (SR0400.8.2)

Attribute	Type	Comment
Duration	Duration	Defines the duration between two consecutive triggers. Minimum default cycle time is 30 seconds, i.e. null or a duration less than 30 seconds is interpreted as 30 seconds during execution.

Timeout period (SR0400.8.3)

Attribute	Type	Comment
Duration	Duration	Defines the duration that the phase waits for its event-triggered operation to become active, before the phase is automatically completed. Null is interpreted as 30 minutes (default timeout) during execution.

CONFIGURATION OF SYSTEM-TRIGGERED EXCEPTIONS

Timeout exception (SR0400.8.4)

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.

See also **Timeout (SR0400.3.2.1)** system-triggered exception (page [21](#)).

Exceptions (SR0400.3+)

The phase supports system-triggered exceptions (page [20](#)) and their configuration by means of process parameters.

System-triggered Exceptions (SR0400.3.2+)

A system-triggered exception of a server-run phase is automatically recorded in the batch report without any user interaction.

The following system-triggered exceptions are available.

Timeout (SR0400.3.2.1)

In case the timeout period has elapsed, the system automatically records a system-triggered exception:

Representation of the exception:

- <Exception text>
(taken from **Timeout exception (SR0400.8.4)** process parameter (page 20))
Phase finished automatically due to timeout after <timeout period>.
- Example:
Timeout occurred.
Phase finished automatically due to timeout after 30 minutes.

Timeout - Logic (SR0400.3.2.1.1)

- Trigger: Phase is completed automatically due to timeout
- Postcondition: Exception is recorded

Step	#	Description
Timeout occurs	10	Phase automatically records exception without user interaction.

Output Variables

The following output variables are available to reference the phase's output.

Instance count (Framework capability)

- Data type: Long
- 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 (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the start time of the phase.

Completion time (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the completion time of the phase.

Identifier (Framework capability)

- Data type: String
- Usage: The output variable provides the identifier of the phase.

Counter-based Trigger Phase (SR0405+)

The **Counter-based trigger** phase allows to automatically create runs of an event-triggered operation (ETO) based on a counter-based cycle.

Example use cases are:

- In manufacturing, average tablet weight and hardness have to be recorded every 5,000 tablets.
- In packaging, visual checks of filled folding cartons have to be executed every 200 cartons.

The number of fired triggers and their count and timestamp information is stored in the batch record, thereby becoming available for documentation purposes in the sub-report and batch report (page 23).

Anomalies that occur during processing are covered by the phase exception handling (page 20) (e.g. timeout of the phase).

Layout

The phase provides a layout for its representation in the sub-report (page 23).

Representation during Execution

As a server-run phase, the phase has no graphical representation (UI).

Representation in Navigator

As a server-run phase, the phase is not visible in the Navigator.

Representation in Sub-report (SR0405.5+)

The sub-report contains the following information:

Common sub-report elements (Framework capability)

- <Start time>
- <Completion time>
- <Unit procedure> / <operation> / <phase>
- <Work center> / <station> / <device> - <phase completion user>

For phases running on a server, the phase completion-user corresponds to the system.

Sub-report elements (SR0405.5.1)

- Delay count, cycle count
- Timeout
- Reading cycle
- Number of fired triggers
- List of pause events: Paused from <timestamp> until <timestamp>

Representation in Sub-record

The sub-record contains the following information:

Batch record-related elements (SR0405.5.2)

- Count and timestamp information of fired triggers

Business Logic (SR0405.2+)

The phase implements the following business logic.

Phase activation (SR0405.2.1)

- Function: Start the trigger processing
- Trigger: Phase becomes active
- Postcondition: Trigger processing is started

Step	#	Description
Phase activation	10	<p>Phase checks</p> <ul style="list-style-type: none"> ■ if the unit procedure is currently paused (check passes if the unit procedure is not paused), ■ if at least one of the relevant ETO templates is already active (check passes if at least one ETO template is active), and ■ if the timeout period has not elapsed yet (defined with the Timeout period (SR0405.8.3) process parameter (page 30)) (check passes if an ETO template has become active before the timeout period has elapsed). <p>If all checks have passed, continue with step 10.4.</p>

Step	#	Description
	10.1	If the unit procedure is paused, the phase waits without any action (no trigger processing, no timeout clock is running) until the unit procedure is continued.
	10.2	If the unit procedure is running, no relevant ETO template is active, and the timeout period has not elapsed, trigger processing is still waiting and timeout clock is running.
	10.3	If the timeout period has elapsed without any ETO template becoming active, phase is completed automatically according to the Phase completion (SR0405.2.5) function (page 29).
	10.4	If a relevant ETO template becomes active, trigger processing starts according to the Fire triggers (SR0405.2.2) function (page 25). At this point, the timeout period no longer applies.

Fire triggers (SR0405.2.2)

- Function: Fire the triggers
- Trigger: Trigger processing has started
- Postcondition: Triggers are fired

Step	#	Description
Start trigger processing	10	Phase reads the current count as a reference point and starts to repeatedly pull for the next counter reading according to the interval defined with the Duration attribute of the Reading cycle (SR0405.8.9) process parameter (page 30).
	20	Phase fires the first trigger after the delay count has elapsed (defined with the Delay count attribute of the Interval definition (SR0405.8.1) process parameter (page 30)).
After delay count has elapsed	30	Phase repeatedly fires a trigger, each time the cycle count has elapsed (defined with the Cycle count attribute of the Interval definition (SR0405.8.1) process parameter (page 30)).

Technical description:

- **currentCounter** value is read from the automation layer and compared to the **scheduledCounter** value.
- If **currentCounter** value \geq **scheduledCounter** value, a trigger is fired.
- Initial setting: **scheduledCounter** value = NULL
- As soon as trigger processing is started:
scheduledCounter(new) value = **currentCounter** value + **delayCount**

■ As soon as first trigger is fired (after delay count):

$$\text{scheduledCounter(new) value} = \text{scheduledCounter(old) value} + \text{cycleCount}$$

Pause trigger processing (SR0405.2.3)

➤ For recent changes, see revision history (page 85).

- Function: Pause/continue the trigger processing
- Precondition: Trigger processing is active
- Trigger: Unit procedure is paused/continued by the operator (for details, see **Pausing a Unit Procedure (SR1089.8.3)** in "Functional Requirement Specification Execution Framework" [A1] (page 83))
- Postcondition: Trigger processing is paused/continued

Step	#	Description
Operator pauses unit procedure	10	At the very moment when the unit procedure of the phase is paused, the phase reads the current count as a "Pause started" reference point and trigger processing stops. No further triggers are fired automatically.
Operator continues paused unit procedure	20	At the very moment when the paused unit procedure of the phase is continued, trigger processing continues where it was stopped. This means, the phase reads the current count as a "Pause finished" reference point and continues to fire triggers according to the Fire triggers (SR0405.2.2) function (page 25) and a new re-calculated trigger schedule. The new trigger schedule is based on the fact that any increase of the count that may have occurred between "Pause started" and "Pause finished" shall be ignored. In case the system detects a reset of the counter value, phase creates a Counter reset (SR0405.3.2.3) system-triggered exception (page 34) and re-calculates the trigger schedule according to the exception.
	20.1	In case either the "Pause started" or the "Pause finished" count cannot be read, phase continues / resumes the trigger processing based on the assumption that the unit procedure has not been paused at all. This means, the phase continues to fire triggers according to the Fire triggers (SR0405.2.2) function (page 25) and according to its original trigger schedule. With respect to the count that could not be read, the phase creates an Automation error (SR0405.3.2.2) system-triggered exception (page 33). As soon as the automation interface (AI) becomes available again, phase reads the current count and checks if a trigger was missed according to the Resume trigger processing (SR0405.2.4) function (page 28). The system adds a comment to the already recorded exception according to the Automation error - Resume (SR0405.3.2.2.2) function (page 33) of the Automation error (SR0405.3.2.2) system-triggered exception (page 33).

Step	#	Description
	20.2	In case no relevant ETO template was active during the pause period and the timeout period has not elapsed yet: trigger processing is still waiting and the timeout clock is running, however, the timeout clock is reset upon resume of the pause.

The pause-related system behavior is based on the assumption that, along with pausing a unit procedure by the operator, typically also the counter on automation level is paused.

However, it also covers use cases where, e.g. IPC needs to be paused due to problems with the quality of the produced boxes. So, while the unit procedure is paused in MES, you continue to run the machine to adjust the parameters, until produced boxes are back in spec. At the same time, the counter value has been increased, but the counts were all related to the adjustment of the parameters, i.e. to waste.

If the paused unit procedure now is continued by the operator, all the waste-related counts that occurred during the pause-related time window are ignored and a new trigger schedule is re-calculated by the system accordingly.

Technical description

- "Pause started" is reflected by the **pauseStartCounter** value.
- "Pause finished" is reflected by the **pauseAfterCounter** value.
- When UP is paused,
pauseStartCounter value = currentCounter value
- When UP is continued,
pauseAfterCounter value = currentCounter value and
scheduledCounter(new) value = scheduledCounter(old) value + pauseAfterCounter - pauseStartCounter
- If counter reset is detected (**currentCounter value < lastGoodCounter value**):
When UP is paused,
pauseStartCounter value = lastGoodCounter value and
scheduledCounter(new) value = lastGoodCounter value
When UP is continued, trigger is fired and
scheduledCounter(new) value = currentCounter value + cycle count
- If an AI exception occurred related to "Pause started" or "Pause finished":
When UP is continued, the pause basically is ignored and **scheduledCounter(new)** value is set dependent on if a trigger was missed.
No trigger missed: **scheduledCounter(new)** is not updated.
Trigger missed: Trigger is fired and
scheduledCounter(new) value = currentCounter value + cycle count

Resume trigger processing (SR0405.2.4)

- Function: Resume the trigger processing
(for the continuation of a paused unit procedure, see **Pause trigger processing (SR0405.2.3)** function (page 26))
- Precondition: Trigger processing is active
- Trigger: Phase is restarted, e.g. along with the restart of the operation that runs on the OES (for details, see **Resuming Server-run Operations (SR1200.1.3)** in "Functional Requirement Specification Execution Framework" [A1] (page 83)) or automation interface (AI) becomes available again
- Postcondition: Trigger processing resumes where it was interrupted

Step	#	Description
Phase is restarted or automation interface (AI) becomes available again	10	As soon as the phase is restarted or the AI becomes available again, it reads the current count and checks if a trigger was missed. In case the system detects a reset of the counter value, phase creates a Counter reset (SR0405.3.2.3) system-triggered exception (page 34) and re-calculates the trigger schedule according to the exception.
	10.1	If no trigger was missed, phase resumes the trigger processing based on the assumption that the cycle counter intervals have never stopped running. This means, the phase continues to fire triggers according to the Fire triggers (SR0405.2.2) function (page 25) and according to its original trigger schedule.
	10.2	If triggers were missed during the time the phase was not running or the AI was not available, they are considered to be lost and will not be re-fired when the trigger processing is resumed. The phase fires a trigger as soon as reasonable, <ul style="list-style-type: none"> ■ if the unit procedure is not paused and the AI is available, immediately, ■ if the unit procedure is paused or the AI is not available, as soon as the unit procedure is continued and the AI is available again, and continues to fire the subsequent triggers according to the Fire triggers (SR0405.2.2) function (page 25) and a re-calculated trigger schedule.

The resume-related system behavior is based on the assumption that the system does not know what kind of error happened (e.g. Was the machine down as well or did the machine continue to run and produce boxes while only the counter signal was down?). So, just to be on the safe side, as soon as the counter signal is back, the system checks if a trigger was missed.
Only if the system can determine that no trigger was missed, will it continue to fire triggers according to its original trigger schedule.
In all other cases (trigger was missed or no way to determine if a trigger was missed), the system fires a new trigger. This trigger now is used as the starting point to

re-calculate a new trigger schedule, based on the original and still valid cycle count interval between two consecutive triggers.

Phase completion (SR0405.2.5)

- Function: Completion of phase
- Trigger: Timeout period has elapsed or no ETO template is active any more
- Postcondition: Phase is completed

Step	#	Description
Timeout period has elapsed	10	If the timeout period has elapsed, the phase is completed automatically without having fired any triggers and creates a Timeout (SR0405.3.2.1) system-triggered exception (page 32).
None of the previously running related ETO templates is active any more	20	Phase stops trigger processing and is completed automatically.

Process Parameters (SR0405.8+)

The following process parameters define the behavior of the phase.

REFERENCE PARAMETERS

Identified equipment entity (SR0405.8.5)

Attribute	Type	Comment
Equipment object	Reference	Reference to the output of a preceding phase that provides an identified equipment entity.

Numeric property (SR0405.8.6)

Attribute	Type	Comment
Property	String	Equipment property to be read.

Property Selection Editor (SR0405.8.6.1)

The system provides a Property Selection editor for selecting an equipment property based on its data type (numeric, string, boolean).

BASIC PARAMETERS

Interval definition (SR0405.8.1)

Attribute	Type	Comment
Delay count	Long	Defines the number of counts between phase start and its first trigger. Null is interpreted as 0 during execution.
Cycle count	Long	Defines the interval between two consecutive triggers. Minimum default cycle count is 1, i.e. null or 0 is interpreted as 1 during execution.

Timeout period (SR0405.8.3)

Attribute	Type	Comment
Duration	Duration	Defines the duration that the phase waits for its event-triggered operation to become active, before the phase is automatically completed. Null is interpreted as 30 minutes (default timeout) during execution.

Reading cycle (SR0405.8.9)

Attribute	Type	Comment
Duration	Duration	Defines the interval in seconds between two consecutive reading actions. Null or any value less than 2 seconds is interpreted as 2 seconds during execution.

CONFIGURATION OF SYSTEM-TRIGGERED EXCEPTIONS

Timeout exception (SR0405.8.4)

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.

See also **Timeout (SR0405.3.2.1)** system-triggered exception (page 32).

Automation error exception (SR0405.8.7)

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.

See also **Automation error (SR0405.3.2.2)** system-triggered exception (page 33).

Counter reset exception (SR0405.8.8)

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.

See also **Counter reset (SR0405.3.2.3)** system-triggered exception (page 34).

Exceptions (SR0405.3+)

The phase supports system-triggered exceptions (page 32) and their configuration by means of process parameters (page 29).

System-triggered Exceptions (SR0405.3.2+)

A system-triggered exception of a server-run phase is automatically recorded in the batch report without any user interaction.

The following system-triggered exceptions are available.

Timeout (SR0405.3.2.1)

In case the timeout period has elapsed, the system automatically records a system-triggered exception:

Representation of the exception:

- <Exception text>
(taken from **Timeout exception (SR0405.8.4)** process parameter (page 31))
Phase finished automatically due to timeout after <timeout period>.
- Example:
Timeout occurred.
Phase finished automatically due to timeout after 30 minutes.

Timeout - Logic (SR0405.3.2.1.1)

- Trigger: Phase is completed automatically due to timeout
- Postcondition: Exception is recorded

Step	#	Description
Timeout occurs	10	Phase automatically records exception without user interaction.

Automation error (SR0405.3.2.2)

In case the counter-related property cannot be read by the automation interface, the system automatically records a system-triggered exception.

Representation of the exception:

- <Exception text>
(taken from **Automation error exception (SR0405.8.7)** process parameter (page 31))
Value of the <property identifier> property could not be read.
System errors: <automation-related message>.
- Example:
Blister machine-related automation error occurred.
Value of the Counter property could not be read.
System errors: <automation-related message>.

Automation error - Logic (SR0405.3.2.2.1)

- Trigger: Counter-related property cannot be read by the automation interface.
- Postcondition: Exception is recorded

Step	#	Description
Automation error occurred	10	Phase automatically records exception without user interaction.

Automation error - Resume (SR0405.3.2.2.2)

- Trigger: Counter-related property can be read by the automation interface.
- Postcondition: Comment is added to the corresponding exception

Step	#	Description
Automation error is resolved	10	As soon as the phase is able to read the counter-related property again, the system adds a comment to the exception (Access to the <property identifier> property has been reestablished.).

Counter reset (SR0405.3.2.3)

In case the external counter is reset to a smaller value or a counter overflow occurs, the system automatically resets the actual count cycle interval and records a system-triggered exception.

Representation of the exception:

- <Exception text>
(taken from **Counter reset exception (SR0405.8.8)** process parameter (page 32))
A reset of the external counter occurred and caused a reset of the count cycle interval.
- Example:
Tableting machine-related counter issue.
A reset of the external counter occurred and caused a reset of the count cycle interval.

Counter reset- Logic (SR0405.3.2.3.1)

- Trigger: External counter is reset to a smaller value.
- Postcondition: Count cycle interval is reset and exception is recorded

Step	#	Description
External counter reset detected	10	Phase automatically records exception without user interaction.
	20	Phase fires a new trigger and resets the count cycle interval.

Output Variables

The following output variables are available to reference the phase's output.

Instance count (Framework capability)

- Data type: Long
- 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 (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the start time of the phase.



Completion time (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the completion time of the phase.

Identifier (Framework capability)

- Data type: String
- Usage: The output variable provides the identifier of the phase.

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Get Values Phase (SR0440+)

The **Get values** phase allows an operator to collect up to five values of the following data types:

- Measured Value (measured value): value and unit of measure,
- Option Value (choice list): choice from a pre-defined list of options, and
- Boolean Value: choice between Yes and No (`true` and `false`).

Example use cases are:

- Recording of tablet weights during IPC
The tablet weight must range between 5 g and 6 g. These boundary values can be defined as limits and corresponding limit violations can be tracked as exceptions.
- Recording of visual appearance during IPC
During the inspection of a liquid product sample, the visual appearance of the sample can be selected from a pre-defined list (e.g. Transparent, Cloudy, Dark).
- Recording of the execution of mandatory checks during IPC
The execution of a visual check needs to be recorded by confirming with **Yes**.

Each value can be entered manually during execution or can be populated as default value from a previous phase.

The values are checked against their limit configuration or expected value configuration.

Each recorded value is stored in the batch record, thereby becoming available for documentation purposes in the sub-report and batch report (page 41).

Anomalies that occur during processing are covered by the phase exception handling (page 56) (e.g. limit violation).

After completion the phase displays the recorded values in the Execution Window.

The Navigator displays the phase completion timestamp and provides access to the post-completion exceptions.

Figure 11: Get values during execution

Layout

The phase provides individual layouts for its representation during execution (page 38), in the Navigator (page 40), and in the sub-report (page 41).

Representation during Execution (SR0440.1+)

The representation during execution depends on the phase mode.

Preview mode (SR0440.1.1)

1. <Instruction text>
(taken from **Instruction (SR0440.8.1)** process parameter (page 46))
2. Any combination of up to five values:
 - **Measured Value Bundle:**
 - <Description>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
 - Box for <actual value, default value>
(UoM taken from **Unit of measure (SR0440.8.3)** process parameter (page 47) and default taken from **Limit definition (SR0440.8.5)** process parameter (page 48))
 - Configured limits
(taken from **Limit definition (SR0440.8.5)** process parameter (page 48))

■ **Option Value Bundle:**

- <Description>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
- List of options and default
(taken from **List of options (SR0440.8.9)** process parameter (page 50))
Default value will be ignored if its format is invalid.

■ **Boolean Value Bundle:**

- <Description>
(taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
- Options (Yes, No) and default
(taken from **Boolean options (SR0440.8.15)** process parameter (page 53))

3. **Confirm** button (disabled).

Active mode (SR0440.1.2)

1. <Instruction text>
(taken from **Instruction (SR0440.8.1)** process parameter (page 46))
2. Any combination of up to five values:

■ **Measured Value Bundle:**

- <Description>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
- Box for <actual value, default value><UoM>
(UoM taken from **Unit of measure (SR0440.8.3)** process parameter (page 47), editable status and default taken from **Limit definition (SR0440.8.5)** process parameter (page 48))
- Configured limits
(taken from **Limit definition (SR0440.8.5)** process parameter (page 48))

■ **Option Value Bundle:**

- <Description>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
- List of options and default
(taken from **List of options (SR0440.8.9)** process parameter (page 50),
editable status and default taken from the **Expected value definition**

(SR0440.8.11) process parameter (page 51))

Default value will be ignored if its format is invalid.

■ **Boolean Value Bundle:**

■ <Description>

(taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))

■ Options (Yes, No) and default

(taken from **Boolean options (SR0440.8.15)** process parameter (page 53), editable status and default taken from the **Expected value definition (SR0440.8.17)** process parameter (page 54))

3. **Confirm** button.

Completed mode (SR0440.1.3)

1. <Instruction text>

(taken from **Instruction (SR0440.8.1)** process parameter (page 46))

2. Table of up to five values that have been entered.

The header titles are populated from the **Short description** attributes of the related **Master (bundle identifier)** process parameter.

■ **Measured Value Bundle:**

Master (bundle identifier) (SR0440.8.2) process parameter (page 46)

■ **Option Value Bundle:**

Master (bundle identifier) (SR0440.8.8) process parameter (page 50)

■ **Boolean Value Bundle:**

Master (bundle identifier) (SR0440.8.14) process parameter (page 53)

3. **Confirm** button (completed).

Representation in Navigator (SR0440.4+)

The Navigator provides the following details:

Phase column (Framework capability)

■ <Phase name>

■ Example:

Processing values

Information column (SR0440.4.1)

■ <Phase completion timestamp>

■ Example: 03/03/2014 12:34:12 EDT

Action column (SR0440.4.2)

- Correct, provides exceptions to correct the recorded values.

Representation in Sub-report (SR0440.5+)

The sub-report contains the following information:

Common sub-report elements (Framework capability)

- <Start time>
- <Completion time>
- <Unit procedure> / <operation> / <phase>
- <Work center> / <station> / <device> - <phase completion user>

Sub-report elements (SR0440.5.1)

- Instruction text
- Table of up to five values that have been entered during execution.
The header titles are populated from the **Short description** attributes of the related **Master (bundle identifier)** process parameters (SR0440.8.2, SR0440.8.8, SR0440.8.14).

Business Logic (SR0440.2+)

The phase implements the following business logic.

Confirm phase (SR0440.2.5)

- Function: Completion of phase
- Trigger: Operator confirms phase
- Postcondition: Phase is completed

Step	#	Description
Operator confirms phase	10	Operator confirms all values.
Check of required operator inputs	15	If values or selections are still missing, phase displays the No value entered or selected (SR0440.3.6.1) error message (page 67). When the error message has been confirmed, phase returns to the Active mode (SR0440.1.2) layout (page 39). Otherwise continue with step 20.

Step	#	Description
Final validation of values	20	If the related checks are enabled, phase triggers the Validate Measured Value (SR0440.2.2) function (page 42), the Validate Option Value (SR0440.2.4) function (page 44), or the Validate Boolean Value (SR0440.2.7) function (page 45), again for each recorded value. (This check is required because values could have been changed by means of an override exception or not expected values could still be selected due to a canceled exception.)
	30	If none of the checks are violated or related exceptions have already been recorded, phase is completed and values are recorded.

Measured Value Bundle

Get Measured Value (SR0440.2.1)

- Function: Get a Measured Value
- Trigger: Phase becomes active
- Postcondition: Measured Value is documented

Step	#	Description
Phase activation	10	Phase displays its user interface according to the Active mode (SR0440.1.2) layout (page 39).
	20	<ul style="list-style-type: none"> ■ If no default value is set, operator enters value. (If the value is not editable, the operator can only enter a value by using the Override value (SR0440.3.1.1) user-triggered exception (page 59).) ■ If a default value is set and the value is editable, the operator accepts the default value or enters another value. ■ If a default value is set and the value is not editable, the operator can only accept the default value or override the default by using the Override value (SR0440.3.1.1) user-triggered exception (page 59).
Cursor leaves box	30	If the check is enabled according to the Limit configuration (SR0440.8.4) process parameter (page 47), phase triggers the Validate Measured Value (SR0440.2.2) function (page 42).

Validate Measured Value (SR0440.2.2)

- Function: Validate a Measured Value
- Trigger: Check is enabled and cursor leaves the box that holds the actual value or phase is confirmed
- Postcondition: Measured Value is validated

Step	#	Description
Validation	10	Phase checks the value against the settings of the Limit definition (SR0440.8.5) process parameter (page 48).
	20	If the limit is violated, phase creates the Limit violation (SR0440.3.2.1) system-triggered exception (page 56).
	20.1	If the Limit violation (SR0440.3.2.1) system-triggered exception (page 56) is signed and recorded, the box that holds the actual value is no longer editable, but can only be updated by using the Override value (SR0440.3.1.1) user-triggered exception (page 59).
	20.2	If the Limit violation (SR0440.3.2.1) system-triggered exception (page 56) is triggered upon the post-completion correction of a value, both exceptions are combined to one exception record, according to the Correct value (SR0440.3.3.1) post-completion exception (page 62).
	30	If the check is not violated, phase can be completed, see Confirm phase (SR0440.2.5) function (page 41).

Option Value Bundle

Get Option Value (SR0440.2.3)

- Function: Get an Option Value
- Trigger: Phase becomes active
- Postcondition: Option Value is documented

Step	#	Description
Phase activation	10	Phase displays its user interface according to the Active mode (SR0440.1.2) layout (page 39).
	20	<ul style="list-style-type: none"> ■ If no default value is set, operator can select a value. (If the value is not editable, the operator can only select a value by using the Override value (SR0440.3.1.2) user-triggered exception (page 60).) ■ If a default value is set and the value is editable, the operator can accept the default value or select another value. ■ If a default value is set and the value is not editable, the operator can only accept the default value or override the default by using the Override value (SR0440.3.1.2) user-triggered exception (page 60).
Operator selects Option Value	30	If the check is enabled according to the Expected value configuration (SR0440.8.10) process parameter (page 50), phase triggers the Validate Option Value (SR0440.2.4) function (page 44).

Validate Option Value (SR0440.2.4)

- Function: Validate an Option Value
- Trigger: Check is enabled and operator selects an Option Value or phase is confirmed
- Postcondition: Option Value is validated

Step	#	Description
Validation	10	Phase checks the value against the settings of the Expected value definition (SR0440.8.11) process parameter (page 51).
	20	If the expected value is violated or cannot be selected at all, phase creates the Violation of expected value (SR0440.3.2.2) system-triggered exception (page 57).
	20.1	If the Violation of expected value (SR0440.3.2.2) system-triggered exception (page 57) is signed and recorded, the value selection is no longer editable, but can only be updated by using the Override value (SR0440.3.1.2) user-triggered exception (page 60).
	20.2	If the Violation of expected value (SR0440.3.2.2) system-triggered exception (page 57) is triggered upon the post-completion correction of a value, both exceptions are combined to one exception record, according to the Correct value (SR0440.3.3.2) post-completion exception (page 64).
	30	If the check is not violated, phase can be completed, see Confirm phase (SR0440.2.5) function (page 41).

Boolean Value Bundle

Get Boolean Value (SR0440.2.6)

- Function: Get a Boolean Value
- Trigger: Phase becomes active
- Postcondition: Boolean Value is documented

Step	#	Description
Phase activation	10	Phase displays its user interface according to the Active mode (SR0440.1.2) layout (page 39).
	20	<ul style="list-style-type: none"> ■ If no default value is set, operator can select a value. (If the value is not editable, the operator can only select a value by using the Override value (SR0440.3.1.3) user-triggered exception (page 61).) ■ If a default value is set and the value is editable, the operator can accept the default value or select another value.

Step	#	Description
		<ul style="list-style-type: none"> ■ If a default value is set and the value is not editable, the operator can only accept the default value selection or override the default by using the Override value (SR0440.3.1.3) user-triggered exception (page 61).
Operator selects Boolean Value	30	If the check is enabled according to the Expected value configuration (SR0440.8.10) process parameter (page 50), phase triggers the Validate Option Value (SR0440.2.4) function (page 44).

Validate Boolean Value (SR0440.2.7)

- Function: Validate a Boolean Value
- Trigger: Check is enabled and operator selects a Boolean Value or phase is confirmed
- Postcondition: Option Value is validated

Step	#	Description
Validation	10	Phase checks the value against the settings of the Expected value definition (SR0440.8.17) process parameter (page 54).
	20	If the check is violated, phase creates the Violation of expected value (SR0440.3.2.3) system-triggered exception (page 58).
	20.1	If the Violation of expected value (SR0440.3.2.3) system-triggered exception (page 58) is signed and recorded, the value selection is no longer editable, but can only be updated by using the Override value (SR0440.3.1.3) user-triggered exception (page 61).
	20.2	If the Violation of expected value (SR0440.3.2.3) system-triggered exception (page 58) is triggered upon the post-completion correction of a value, both exceptions are combined to one exception record, according to the Correct value (SR0440.3.3.3) post-completion exception (page 65).
	30	If the check is not violated, phase can be completed, see Confirm phase (SR0440.2.5) function (page 41).

Process Parameters (SR0440.8+)

The following process parameters define the behavior of the phase.

BASIC PARAMETERS

Instruction (SR0440.8.1)

➤ For recent changes, see revision history (page [85](#)).

Attribute	Type	Comment
Column 1	HTML text	Phase-related instruction text to be displayed during execution. Restriction: Maximum length is 2000 characters (including HTML tags).
Column 2	HTML text	Not used.
Column 3	HTML text	

Measured Value Bundle

Bundle process parameters (Framework capability)

For the master process parameter of a bundle, its internal identifier is populated from the bundle identifier.

For all other process parameters of the bundle, their internal identifier is a concatenation of the bundle identifier and the process parameter name.

This framework capability refers to **Bundle Process Parameters (SR3146.9.7.4.1)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page [83](#)).

BASIC BUNDLE PARAMETERS

Master (Bundle identifier) (SR0440.8.2)

Attribute	Type	Comment
Description	Text	Value-related instruction text to be displayed during execution. By default, the description is taken from the bundle identifier. If no description is defined, the system displays the short description; if no short description is defined, the system displays the bundle identifier. Maximum length is 250 characters.

Attribute	Type	Comment
Short description	Text	Defines the header title of the table in the Completed mode (SR0440.1.3) layout (page 40) and, if applicable, of the linked Show values (SR0450+) (page 71) phase. Example: Weight

Unit of measure (SR0440.8.3)

Attribute	Type	Comment
UoM	Unit of measure	Must match a unit of measure available within PharmaSuite. See also attributes of the Limit definition (SR0440.8.5) process parameter (page 48).

CONFIGURATION OF SYSTEM-TRIGGERED EXCEPTIONS

Limit configuration (SR0440.8.4)

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the Lower limit and Upper limit attributes of the Limit definition (SR0440.8.5) process parameter (page 48) are set.
Display	Flag	Controls if the limit range is displayed during execution.
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.

See also **Limit violation (SR0440.3.2.1)** system-triggered exception (page 56).

Limit definition (SR0440.8.5)

The unit of measure must be of the same system of measurement as the one used for the **Unit of measure (SR0440.8.3)** process parameter (page 47) (e.g. weight: mg, kg, pound; length: mm, m, inch).

Attribute	Type	Comment
Lower limit	MeasuredValue	Defines the value of the lower limit (including the values themselves). Limit values with more than 7 digits are truncated at the end in the Phase Preview of Recipe and Workflow Designer and Production Execution Client.
Upper limit	MeasuredValue	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 of Recipe and Workflow Designer and Production Execution Client.
Default value	MeasuredValue	Defines the default value.
Value editable	Flag	Controls if the displayed value is editable during execution. Default setting: Yes

CONFIGURATION OF USER-TRIGGERED EXCEPTIONS

Override value (SR0440.8.6)

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.

See also **Override value (SR0440.3.1.1)** user-triggered exception (page 59).

CONFIGURATION OF POST-COMPLETION EXCEPTIONS

Correct value (SR0440.8.7)

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.

See also **Correct value (SR034410.3.3.1)** post-completion exception (page 62).

Option Value Bundle

Bundle process parameters (Framework capability)

For the master process parameter of a bundle, its internal identifier is populated from the bundle identifier.

For all other process parameters of the bundle, their internal identifier is a concatenation of the bundle identifier and the process parameter name.

This framework capability refers to **Bundle Process Parameters (SR3146.9.7.4.1)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

BASIC BUNDLE PARAMETERS

Master (Bundle identifier) (SR0440.8.8)

Attribute	Type	Comment
Description	Text	Value-related instruction text to be displayed during execution. By default, the description is taken from the bundle identifier. If no description is defined, the system displays the short description; if no short description is defined, the system displays the bundle identifier. Maximum length is 250 characters.
Short description	Text	Defines the header title of the table in the Completed mode (SR0440.1.3) layout (page 40) and, if applicable, of the linked Show values (SR0450+) (page 71) phase. Example: Appearance

List of options (SR0440.8.9)

Attribute	Type	Comment
Options	Text (structured)	Defines the available options as key/display text value pairs. Both keys and display texts are unique within a phase.

Option List editor (Framework capability)

The system provides an Option List editor for entering choice items as key/display text value pairs.

CONFIGURATION OF SYSTEM-TRIGGERED EXCEPTIONS

Expected value configuration (SR0440.8.10)

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the Expected value attribute of the Expected value definition (SR0440.8.11) process parameter (page 51) is set.

Attribute	Type	Comment
Display	Flag	Controls if an expected value is displayed during execution. The value is marked as underlined text. Ensure that the Expected value attribute of the Expected value definition (SR0440.8.11) process parameter (page 51) is set.
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.

See also **Violation of expected value (SR0440.3.2.2)** system-triggered exception (page 57).

Expected value definition (SR0440.8.11)

Attribute	Type	Comment
Expected value	String	Defines the expected value.
Default value	String	Defines the pre-selected item in the list of options.
Value editable	Flag	Controls if the displayed value is editable during execution. Default setting: Yes

CONFIGURATION OF USER-TRIGGERED EXCEPTIONS

Override value (SR0440.8.12)

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.

See also **Override value (SR0440.3.1.2)** user-triggered exception (page 60).

CONFIGURATION OF POST-COMPLETION EXCEPTIONS

Correct value (SR0440.8.13)

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.

See also **Correct value (SR0440.3.3.2)** post-completion exception (page 64).

Boolean Value Bundle

Bundle process parameters (Framework capability)

For the master process parameter of a bundle, its internal identifier is populated from the bundle identifier.

For all other process parameters of the bundle, their internal identifier is a concatenation of the bundle identifier and the process parameter name.

This framework capability refers to **Bundle Process Parameters (SR3146.9.7.4.1)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

BASIC BUNDLE PARAMETERS

Master (Bundle identifier) (SR0440.8.14)

Attribute	Type	Comment
Description	Text	Value-related instruction text to be displayed during execution. By default, the description is taken from the bundle identifier. If no description is defined, the system displays the short description; if no short description is defined, the system displays the bundle identifier. Maximum length is 250 characters.
Short description	Text	Defines the header title of the table in the Completed mode (SR0440.1.3) layout (page 40) and, if applicable, of the linked Show values (SR0450+) (page 71) phase. Example: Hardness

Boolean options (SR0440.8.15)

Attribute	Type	Comment
Display text for TRUE	String	Defines the string displayed as TRUE option. Maximum length is 8 characters. Default setting: Yes
Display text for FALSE	String	Defines the string displayed as FALSE option. Maximum length is 8 characters. Default setting: No

CONFIGURATION OF SYSTEM-TRIGGERED EXCEPTIONS

Expected value configuration (SR0440.8.16)

Attribute	Type	Comment
Enabled	Flag	Controls if a check is performed. If so, ensure that the Expected value attribute of the Expected value definition (SR0440.8.17) process parameter (page 54) is set.
Display	Flag	Controls if an expected value is displayed during execution. The value is marked as underlined text. Ensure that the Expected value key attribute of the Expected value definition (SR0440.8.17) process parameter (page 54) is set.
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.

See also **Violation of expected value (SR034410.3.2.3)** system-triggered exception (page 58).

Expected value definition (SR0440.8.17)

Attribute	Type	Comment
Expected value	String	Defines the expected value.
Default value	String	Defines the pre-selected item in the list of options.
Value editable	Flag	Controls if the displayed value is editable during execution. Default setting: Yes

CONFIGURATION OF USER-TRIGGERED EXCEPTIONS

Override value (SR0440.8.18)

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.

See also **Override value (SR0440.3.1.3)** user-triggered exception (page 61).

CONFIGURATION OF POST-COMPLETION EXCEPTIONS

Correct value (SR0440.8.19)

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.

See also **Correct value (SR034410.3.3.3)** post-completion exception (page 65).

Exceptions (SR0440.3+)

The phase supports user-defined, user-triggered (page 59), system-triggered (page 56), and post-completion exceptions (page 62) and their configuration by means of process parameters (page 45).

User-defined exceptions cannot be configured by process parameters since they are provided by the framework and independent of phases.

System-triggered Exceptions (SR0440.3.2+)

➤ For recent changes, see revision history (page 85).

A system-triggered exception is represented in a message dialog along with an **Exception** button, in the Exception Window as the read-only description of the exception, and in the batch report.

The following system-triggered exceptions are available.

MEASURED VALUE BUNDLE

Limit violation (SR0440.3.2.1)

Representation of the exception:

Exception dialog

- <Exception text>
(taken from **Limit configuration (SR0440.8.4)** process parameter (page 47))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
<Limit name>: <expected value>
Actual value: <value>

Exception Window

- <Exception text>
(taken from **Limit configuration (SR0440.8.4)** process parameter (page 47))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
<Limit name>: <expected value>
Actual value: <value>
- Example:
Limit violation confirmed
Speed
Lower limit: 300 rpm
Actual value: 200 rpm

Limit violation - Logic (SR0440.3.2.1.1)

- Trigger: Value is not within the defined limits
- Postcondition: Exception is recorded

Step	#	Description
Operator accepts exceptional situation	10	Phase shows exception description to be signed.
Operator signs exception	20	Phase records exception.

OPTION VALUE BUNDLE**Violation of expected value (SR0440.3.2.2)**

Representation of the exception:

Exception dialog

- <Exception text>
(taken from **Expected value configuration (SR0440.8.10)** process parameter (page 50))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
Expected key/text: <expected key>/<expected text>
Actual value: <selected value>

Exception Window

- <Exception text>
(taken from **Expected value configuration (SR0440.8.10)** process parameter (page 50))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
Expected key/text: <expected key>/<expected text>
Actual value: <selected value>
- Example:
Expected value check failed.
Production key
Expected key/text: B/Biotech
Actual key/text: M/Microbiology

Violation of expected value - Logic (SR0440.3.2.2.1)

- Trigger: Operator confirms phase
- Postcondition: Phase is completed

Step	#	Description
Operator confirms phase	10	Phase creates Violation of expected value (SR0440.3.2.2) system-triggered exception.
Operator triggers exception	20	Phase records exception.

BOOLEAN VALUE BUNDLE

Violation of expected value (SR0440.3.2.3)

Representation of the exception:

Exception dialog

- <Exception text>
 (taken from **Expected value configuration (SR0440.8.16)** process parameter (page 54))
 <Short description> or, if not maintained, <Bundle identifier>
 (taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
 Expected value: <expected value>
 Actual value: <selected value>

Exception Window

- <Exception text>
 (taken from **Expected value configuration (SR0440.8.16)** process parameter (page 54))
 <Short description> or, if not maintained, <Bundle identifier>
 (taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
 Expected value: <expected value>
 Actual value: <selected value>
- Example:
 Expected value check failed.
 Solution dissolved?
 Expected value: Yes
 Actual value: No

Violation of expected value - Logic (SR0440.3.2.3.1)

- Trigger: Operator confirms phase
- Postcondition: Phase is completed

Step	#	Description
Operator confirms phase	10	Phase creates Violation of expected value (SR0440.3.2.3) system-triggered exception.
Operator triggers exception	20	Phase records exception.

User-triggered Exceptions (SR0440.3.1+)

A user-triggered exception is represented in the list of available user-triggered exceptions in the Exception Window, as the description of the exception, and in the batch report.

The following user-triggered exceptions are available.

MEASURED VALUE BUNDLE

Override value (SR0440.3.1.1)

The **Override value** exception allows an operator to override the value in case it is set to **read-only** (**Value editable** attribute of the **Limit definition (SR0440.8.5)** process parameter (page 48)).

Representation during exception handling:

Exception instruction

- <Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
Enter a new value.
<Old value with unit of measure>
Box for new value (with unit of measure)
Confirm button.

Recorded exception

- <Exception text>
(taken from **Override value (SR0440.8.6)** process parameter (page 48))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
Old value: <old value> <UoM>
New value: <new value> <UoM>

- Example:
Speed value corrected.
Speed
Old value: 20 rpm
New value: 25 rpm

Override value - Logic (SR0440.3.1.1.1)

- Trigger: Exception is selected
- Postcondition: Value is overridden

Step	#	Description
Operator triggers exception	10	Phase displays Exception Window.
	20	Operator enters new value.
Operator confirms exception	30	Phase shows exception description to be signed according to Override value (SR0440.8.6) process parameter (page 48).
Operator signs exception	40	Phase records exception.

OPTION VALUE BUNDLE

Override value (SR0440.3.1.2)

The **Override value** exception allows an operator to override the value in case it is set to **read-only** (**Value editable** attribute of the **Expected value definition (SR0440.8.11)** process parameter (page 51)).

Representation during exception handling:

Exception instruction

- <Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
Select another option. Old value: <Old text>
Display of list of options according to the **Active mode (SR0440.1.2)** layout (page 39)
Confirm button.

Recorded exception

- <Exception text>
(taken from **Override value (SR0440.8.12)** process parameter (page 52))
<Short description> or, if not maintained, <Bundle identifier>

(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))

Old key/text: <Old key>/<Old text>

New key/text: <New key>/<New text>

- Example:
Selection corrected.
Appearance
Old key/text: Yellow/Yellow appearance of test strip
New key/text: Blue/Blue appearance of test strip

Override value - Logic (SR0440.3.1.2.1)

- Trigger: Exception is selected
- Postcondition: Value is overridden

Step	#	Description
Operator signs exception	10	Phase displays Exception Window.
	20	Operator selects another value.
Operator confirms exception	30	Phase shows exception description to be signed according to Override value (SR0440.8.12) process parameter (page 52).
Operator signs exception	40	Phase records exception.

BOOLEAN VALUE BUNDLE

Override value (SR0440.3.1.3)

The **Override value** exception allows an operator to override the value in case it is set to **read-only** (**Value editable** attribute of the **Expected value definition (SR0440.8.17)** process parameter (page 54)).

Representation during exception handling:

Exception instruction

- <Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
Select another option. Old value: <Old value>
Display options (Yes, No)
Confirm button.

Recorded exception

- <Exception text>
(taken from **Override value (SR0440.8.18)** process parameter (page 55))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
Old value: <Old value>
New value: <New value>
- Example:
Option corrected.
Dissolved?
Old value: No
New value: Yes

Override value - Logic (SR0440.3.1.3.1)

- Trigger: Exception is selected
- Postcondition: Value is overridden

Step	#	Description
Operator signs exception	10	Phase displays Exception Window.
	20	Operator selects another Boolean Value.
Operator confirms exception	30	Phase shows exception description to be signed according to Override value (SR0440.8.18) process parameter (page 55).
Operator signs exception	40	Phase records exception.

Post-completion Exceptions (SR0440.3.3+)

A post-completion exception is accessible via the Navigator and represented in the list of available post-completion exceptions in the Exception Window, as the description of the exception, and in the batch report.

The following post-completion exceptions are available.

MEASURED VALUE BUNDLE

Correct value (SR0440.3.3.1)

The **Correct value** exception allows an operator to correct the recorded value from the Navigator after the completion of the phase.

TIP

A recorded value could be used within branching. The correction of a value **does not influence** already processed branching decisions.

Representation of the exception:

Exception instruction

- <Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
Enter a new value.
<Old value with unit of measure>
Box for new value (with unit of measure)
Confirm button.

Recorded exception

- <Exception text>
(taken from **Correct value (SR0440.8.7)** process parameter (page 49))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.2)** process parameter (page 46))
Old value: <old value> <UoM>
New value: <new value> <UoM>
- Example:
Speed value corrected (after phase completion).
Speed
Old value: 20 rpm
New value: 25 rpm

Correct value - Logic (SR0440.3.3.1.1)

- Trigger: Phase is completed
- Postcondition: Value is corrected

Step	#	Description
Operator triggers action	10	Phase displays Exception Window.
	20	Operator enters corrected value.
Operator confirms exception	30	If the related check is enabled, phase checks the value against the settings of the Limit definition (SR0440.8.5) process parameter (page 48).

Step	#	Description
	30.1	If the limit is violated, only one combined exception (post-completion exception) is displayed including both, exception text from the correction and from the limit violation. The recorded risk of the combined exception and its required signature are controlled by the highest risk of the two underlying exceptions.
	30.2	If the limit is not violated or no check applies, the corrected value-related exception is displayed.
Operator signs exception	40	Phase records the new value and its related exception.

OPTION VALUE BUNDLE

Correct value (SR0440.3.3.2)

The **Correct value** exception allows an operator to correct the selected value from the Navigator after the completion of the phase.

TIP

A recorded value could be used within branching. The correction of a value **does not influence** already processed branching decisions.

Representation of the exception:

Exception instruction

- <Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
Select another option. Old value: <Old text>
Display of list of options according to the **Active mode (SR0440.1.2)** layout (page 39)
Confirm button.

Recorded exception

- <Exception text>
(taken from **Correct value (SR0440.8.13)** process parameter (page 52))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.8)** process parameter (page 50))
Old key/text: <Old key>/<Old text>
New key/text: <Old key>/<Old text>

- Example:
Selection corrected (after phase completion).
Appearance
Old key/text: Yellow/Yellow appearance of test strip
New key/text: Blue/Blue appearance of test strip

Correct value - Logic (SR0440.3.3.2.1)

- Trigger: Phase is completed
- Postcondition: Value is corrected

Step	#	Description
Operator triggers action	10	Phase displays Exception Window.
	20	Operator selects corrected value.
Operator confirms exception	30	If the related check is enabled, phase checks the value against the settings of the Expected value definition (SR0440.8.11) process parameter (page 51).
	30.1	If the expected value is violated, only one combined exception (post-completion exception) is displayed including both, exception text from the correction and from the violation of the expected value. The recorded risk of the combined exception and its required signature are controlled by the highest risk of the two underlying exceptions.
	30.2	If the expected value is not violated or no check applies, the corrected value-related exception is displayed.
Operator signs exception	40	Phase records the new value and its related exception.

BOOLEAN VALUE BUNDLE

Correct value (SR0440.3.3.3)

The **Correct value** exception allows an operator to correct the selected value from the Navigator after the completion of the phase.

TIP

A recorded value could be used within branching. The correction of a value **does not influence** already processed branching decisions.

Representation of the exception:

Exception instruction

- <Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
Select another option. Old value: <Old value>
Display options (Yes, No)
Confirm button.

Recorded exception

- <Exception text>
(taken from **Override value (SR0440.8.19)** process parameter (page 55))
<Short description> or, if not maintained, <Bundle identifier>
(taken from **Master (bundle identifier) (SR0440.8.14)** process parameter (page 53))
Old value: <Old value>
New value: <New value>
- Example:
Option corrected (after phase completion).
Dissolved?
Old value: No
New value: Yes

Correct value - Logic (SR0440.3.3.3.1)

- Trigger: Phase is completed
- Postcondition: Value is corrected

Step	#	Description
Operator triggers action	10	Phase displays Exception Window.
	20	Operator selects corrected value.
Operator confirms exception	30	If the related check is enabled, phase checks the value against the settings of the Expected value definition (SR0440.8.17) process parameter (page 54).
	30.1	If the expected value is violated, only one combined exception (post-completion exception) is displayed including both, exception text from the correction and from the violation of the expected value. The recorded risk of the combined exception and its required signature are controlled by the highest risk of the two underlying exceptions.
	30.2	If the expected value is not violated or no check applies, the corrected value-related exception is displayed.

Step	#	Description
Operator signs exception	40	Phase records the new value and its related exception.

Information Messages

There are no information messages available.

Questions

There are no questions available.

Decisions

There are no decisions available.

Error Messages (SR0440.3.6+)

Error messages are represented in an error message dialog containing a message type-specific icon, the error message, and an **OK** button.

The following error messages are available to inform the operator about error conditions.

No value entered or selected (SR0440.3.6.1)

UI text	Comment
<Short description> (or, if not maintained, <Bundle identifier>, taken from Master (bundle identifier) You have to enter or select a value before you can confirm the phase.	Message pack: PhaseGetValues<version> Message ID: EmptyValue_ErrorMsg

Output Variables (SR0440.9+)

The following output variables are available to reference the phase's output.

Instance count (Framework capability)

- Data type: Long
- 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 (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the start time of the phase.

Completion time (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the completion time of the phase.

Identifier (Framework capability)

- Data type: String
- Usage: The output variable provides the identifier of the phase.

Data reference (SR0440.9.7)

- Data type: PhaseDataReference
- Usage: The output variable provides a reference to a data collection phase.

Measured Value Bundle

Bundle output variable (Framework capability)

For all output variables of the same bundle, the output variable identifier is a concatenation of the bundle identifier and the output variable name.

This framework capability refers to **Bundle Output Variable (SR3146.9.7.4.2)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

Value (SR0440.9.1)

- Data type: MeasuredValue
- Usage: The output variable provides the complete process value as a **MeasuredValue** object.

Unit of measure (SR0440.9.2)

- Data type: String
- Usage: The output variable provides the unit of measure of the process value.

Option Value Bundle

Bundle output variable (Framework capability)

For all output variables of the same bundle, the output variable identifier is a concatenation of the bundle identifier and the output variable name.

This framework capability refers to **Bundle Output Variable (SR3146.9.7.4.2)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

Option text (SR0440.9.3)

- Data type: String
- Usage: The output variable provides the display text of the selected option.

Option key (SR0440.9.4)

- Data type: String
- Usage: The output variable provides the key value of the selected option.

Boolean Value Bundle

Bundle output variable (Framework capability)

For all output variables of the same bundle, the output variable identifier is a concatenation of the bundle identifier and the output variable name.

This framework capability refers to **Bundle Output Variable (SR3146.9.7.4.2)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

Option text (SR0440.9.5)

- Data type: String
- Usage: The output variable provides the display text of the selected option.

Option key (SR0440.9.6)

- Data type: Boolean
- Usage: The output variable provides the selected boolean option.

Show Values Phase (SR0450+)

The **Show values** phase allows an operator to view the data that has been collected along with the **Get values** phase (page 37) across multiple runs, which are based on loops or ETO templates. The data is represented in a table format and supports up to five columns with values of the following data types:

- Measured Value (measured value): value and unit of measure,
- Option Value (choice list): choice from a pre-defined list of options, and
- Boolean Value: choice between Yes and No (`true` and `false`).

Example use cases are:

- Recording of tablet weights across multiple IPC runs.
- Recording of visual appearance across multiple IPC runs (e.g. Transparent, Cloudy, Dark).
- Recording of the execution of mandatory checks across multiple IPC runs.

For Measured Values, the result of statistical calculations can be presented as well, like Average, Min, Max, Sum, and Standard Deviation.

All data is stored in the batch record, thereby becoming available for documentation purposes in the sub-report and batch report (page 74).

After completion the phase displays the data in the Execution Window.

The Navigator displays the phase completion timestamp.

Analyze and review the collected IPC data.

Data collected from: Stand-alone Tableting Run/Tableting (Solo)/Tableting IPC/Collect IPC Data

Run	Tablet size	Tablet weight	Equipment check	Output mat. ready	Materials returned	Signature / Time
1.1	4.20 mm	298 mg	Smooth run without issues	Yes	Yes	07/11/2014 10:39:22 AM
1.2	4.20 mm	302 mg	Smooth run without issues	Yes	Yes	07/11/2014 10:41:29 AM
1.3	4.20 mm	300 mg	Smooth run without issues	Yes	Yes	07/11/2014 10:43:06 AM
1.4	4.20 mm	299 mg	Smooth run without issues	Yes	Yes	07/11/2014 10:45:13 AM
1.5	4.20 mm	304 mg	Smooth run, maintenance to be scheduled	Yes	Yes	07/11/2014 10:53:53 AM
1.6	4.20 mm	304 mg	Smooth run without issues	Yes	Yes	07/11/2014 10:57:45 AM
Average	4.200 mm	301.2 mg	N/A	N/A	N/A	
Minimum	4.20 mm	298 mg	N/A	N/A	N/A	
Maximum	4.20 mm	304 mg	N/A	N/A	N/A	
Sum	25.20 mm	1,807 mg	N/A	N/A	N/A	
Standard deviation	0.0 mm	2.555 mg	N/A	N/A	N/A	




Refresh   Confirm 

Figure 12: Show values during execution

Layout

The phase provides individual layouts for its representation during execution (page 72), in the Navigator (page 74), and in the sub-report (page 74).

Representation during Execution (SR0450.1+)

The representation during execution depends on the phase mode.

Preview mode (SR0450.1.1)

1. <Instruction text>
(taken from **Instruction (SR0450.8.1)** process parameter (page 77))
2. Data collected from:
(no path information available during preview)
3. Empty table header for value-related columns
4. Empty table with statistics-related rows
(according to the **Master (bundle identifier) (SR0450.8.3)** process parameter (page 78))
5. **Refresh** button (disabled).
6. **Enable** button (disabled).
The button is not visible in case a phase completion signature was configured during authoring.
7. **Confirm** button (disabled).

Active mode (SR0450.1.2)

1. <Instruction text>
(taken from **Instruction (SR0450.8.1)** process parameter (page 77))
2. Data collected from: <Path information of referred **Get values** phase>
3. Table of values with **Run** column, up to five value-related columns, and **Signature / Time** column.
(Header titles of the value-related columns are populated from the **Get values (SR0440+)** phase (page 37) that is referenced by the **Data collection reference** attribute of the **Definition (SR0450.8.2)** process parameter (page 77) and from the **Short description** attributes of the related **Master (bundle identifier)** process parameters (**SR0440.8.2, SR0440.8.8, SR0440.8.14**).
Signature / Time values are populated from the completion times of the **Get values** phase.)
4. Table with statistical data.
(Visibility is controlled by the **Master (bundle identifier) (SR0450.8.3)** process parameter (page 78).)
5. **Refresh** button.
6. **Enable** button (unlocks the **Confirm** button).
The button is not visible in case a phase completion signature was configured during authoring.
7. **Confirm** button.

Completed mode (SR0450.1.3)

1. <Instruction text>
(taken from **Instruction (SR0450.8.1)** process parameter (page 77))
2. Data collected from: <Path information of referred **Get values** phase>
3. Table of values with **Run** column, up to five value-related columns, and **Signature / Time** column.
(Header titles of the value-related columns are populated from the **Get values (SR0440+)** phase (page 37) that is referenced by the **Data collection reference** attribute of the **Definition (SR0450.8.2)** process parameter (page 77) and from the **Short description** attributes of the related **Master (bundle identifier)** process parameters (**SR0440.8.2, SR0440.8.8, SR0440.8.14**).
Signature / Time values are populated from the completion times of the **Get values** phase.)
4. Table with statistical data.
(Visibility is controlled by the **Master (bundle identifier) (SR0450.8.3)** process parameter (page 78).)

5. **Refresh** button (disabled).
6. **Enable** button (disabled).
The button is not visible in case a phase completion signature was configured during authoring.
7. **Confirm** button (completed).

Representation in Navigator (SR0450.4+)

The Navigator provides the following details:

Phase column (Framework capability)

- <Phase name>
 - Example:
Hardness trend

Information column (SR0450.4.1)

- <Phase completion timestamp>
 - Example: 03/03/2014 12:34:12 EDT

Action column

- There are no actions available.

Representation in Sub-report (SR0450.5+)

The sub-report contains the following information:

Common sub-report elements (Framework capability)

- <Start time>
- <Completion time>
- <Unit procedure> / <operation> / <phase>
- <Work center> / <station> / <device> - <phase completion user>

Sub-report elements (SR0450.5.1)

- Instruction text
- Data collected from: <Path information of referred **Get values** phase>
- Table of values with **Run** column, up to five value-related columns, and **Signature / Time** column.
(Header titles of the value-related columns are populated from the **Get values** (SR0440+) phase (page 37) that is referenced by the **Data collection reference**)

attribute of the **Definition (SR0450.8.2)** process parameter (page 77) and from the **Short description** attributes of the related **Master (bundle identifier)** process parameters (**SR0440.8.2, SR0440.8.8, SR0440.8.14**).

Signature / Time values are populated from the completion times of the **Get values** phase.)

(Visibility is controlled by the **Master (bundle identifier) (SR0450.8.3)** process parameter (page 78).)

- Table with statistical data.
(Visibility is controlled by the **Master (bundle identifier) (SR0450.8.3)** process parameter (page 78).)

Business Logic (SR0450.2+)

The phase implements the following business logic.

Phase activation (SR0450.2.1)

- Function: Phase is active
- Trigger: Previous phase is completed
- Postcondition: Phase is active

Step	#	Description
Phase is activated	10	Phase checks if data from the data collection reference phase and its instances is already available.
	10.1	If no data is available, the table of values and the table of statistics still remain empty. With Refresh , phase checks again if data is available, according to step 10.
	10.2	If data is available, phase displays the table of values and the table of statistics according to the Active mode (SR0450.1.2) layout (page 73). <ul style="list-style-type: none"> ■ In case the referenced Get values phase was completed with a phase completion signature, phase displays this signature and its timestamp in the Signature / Time column. ■ In case the referenced Get values phase was completed without a signature, phase displays the phase completion timestamp in the Signature / Time column.
Phase is refreshed	20	With Refresh , phase adds more run-related data, if available, and recalculates the statistical values.

Statistics calculation (SR0450.2.2)

- Function: Calculate statistics
- Trigger: Phase is activated or refreshed
- Postcondition: Statistics are calculated

Step	#	Description
Phase is activated or refreshed	10	If data is available, phase calculates statistical data according to the Master (bundle identifier) (SR0450.8.3) process parameter (page 78).

Prerequisite for calculation: The bundle identifier of the statistics bundle must match the bundle identifier of the respective value bundle of the referenced **Get values** phase (page 37).
During execution, values can only be calculated and displayed for the Measured Value data type.

Phase completion (SR0450.2.3)

- Function: Completion of phase
- Trigger: Operator confirms phase
- Postcondition: Phase is completed

Step	#	Description
Operator confirms phase	10	Phase refreshes the data automatically. <ul style="list-style-type: none"> ■ If new data is available, phase displays the Automatic refresh (SR0450.3.4.1) information message (page 79) and remains in the active status until the operator confirms the phase again. ■ If no new data is available, phase continues with step 20.
	20	Phase runtime data is recorded according to the Representation in Sub-report (SR0450.5+) layout (page 74).
	30	The result of all statistical calculations is available as output variables, including those calculations that are not shown during execution and in the batch report according to the Master (bundle identifier) (SR0450.8.3) process parameter (page 78).
	40	Phase is completed.

Process Parameters (SR0450.8+)

The following process parameters define the behavior of the phase.

BASIC PARAMETERS

Instruction (SR0450.8.1)

➤ For recent changes, see revision history (page 85).

Attribute	Type	Comment
Column 1	HTML text	Instruction text to be displayed. Restriction: Maximum length is 2000 characters (including HTML tags).
Column 2	HTML text	Not used.
Column 3	HTML text	

Definition (SR0450.8.2)

Attribute	Type	Comment
Data collection reference	Reference	Reference to a preceding data collection phase.
Scope of data collection	Choice list	Defines the scope of collected data. Available settings: Within operation run , Across operation runs . Default setting: Across operation runs .
Show table in batch report	Flag	Defines if the table of values is displayed in the batch report. Default settings: Yes Note: The table of statistics, if applicable, is always displayed in the batch report.

Statistics Bundle

Bundle process parameters (Framework capability)

For the master process parameter of a bundle, its internal identifier is populated from the bundle identifier.

For all other process parameters of the bundle, their internal identifier is a concatenation of the bundle identifier and the process parameter name.

This framework capability refers to **Bundle Process Parameters (SR3146.9.7.4.1)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

Master (Bundle identifier) (SR0450.8.3)

Attribute	Type	Comment
Show average	Flag	Defines if the average value is displayed during execution.
Show minimum	Flag	Defines if the minimum value is displayed during execution.
Show maximum	Flag	Defines if the maximum value is displayed during execution.
Show sum	Flag	Defines if the sum value is displayed during execution.
Show standard deviation	Flag	Defines if the standard deviation value is displayed during execution.

The bundle identifier of the statistics bundle must match the bundle identifier of the respective value bundle of the referenced **Get values** phase (page 37). During execution, values can only be calculated and displayed for the Measured Value data type.

Exceptions (SR0450.3+)

The phase can support user-defined, user-triggered (page 78), system-triggered (page 78), and post-completion exceptions (page 78) and their configuration by means of process parameters (page 77).

User-defined exceptions cannot be configured by process parameters since they are provided by the framework and independent of phases.

System-triggered Exceptions

There are no system-triggered exceptions available.

User-triggered Exceptions

There are no user-triggered exceptions available.

Post-completion Exceptions

There are no post-completion exceptions available.

Information Messages (SR0450.3.4+)

Information messages are represented in an information dialog containing a message type-specific icon, the information message, and an **OK** button.

The following information messages are available to inform the operator about how to proceed.

Automatic refresh (SR0450.3.4.1)

UI text	Comment
Cannot confirm, since there is new data available. Values are now being refreshed automatically. Confirm again to complete the phase.	The information message is displayed upon phase completion in case new data is available. Message pack: PhaseShowValues<version> Message ID: RefreshNecessary_ErrorMsg

Questions

There are no questions available.

Decisions

There are no decisions available.

Error Messages

There are no error messages available.

Output Variables (SR0450.9+)

The following output variables are available to reference the phase's output.

Instance count (Framework capability)

- Data type: Long
- 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 (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the start time of the phase.

Completion time (Framework capability)

- Data type: Timestamp
- Usage: The output variable provides the completion time of the phase.

Identifier (Framework capability)

- Data type: String
- Usage: The output variable provides the identifier of the phase.

Statistics Bundle

Bundle output variable (Framework capability)

For all output variables of the same bundle, the output variable identifier is a concatenation of the bundle identifier and the output variable name.

This framework capability refers to **Bundle Output Variable (SR3146.9.7.4.2)** in "Functional Requirement Specification Recipe and Workflow Management" [A2] (page 83).

Average (SR0450.9.1)

- Data type: MeasuredValue
- Usage: The output variable provides the average value as a **MeasuredValue** object.

Minimum (SR0450.9.2)

- Data type: MeasuredValue
- Usage: The output variable provides the minimum value as a **MeasuredValue** object.

Maximum (SR0450.9.3)

- Data type: MeasuredValue
- Usage: The output variable provides the maximum value as a **MeasuredValue** object.

Sum (SR0450.9.4)

- Data type: MeasuredValue
- Usage: The output variable provides the sum value as a **MeasuredValue** object.

Standard deviation (SR0450.9.5)

- Data type: MeasuredValue
- Usage: The output variable provides the standard deviation value as a **MeasuredValue** object.

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Reference Documents

The following documents are available from the Rockwell Automation Download Site.

No.	Document Title	Part Number
A1	PharmaSuite Functional Requirement Specification Execution Framework	PSFRSEF-RM004E-EN-E
A2	PharmaSuite Functional Requirement Specification Recipe and Workflow Management	PSFRSRD-RM008E-EN-E

TIP

To access the Rockwell Automation Download Site, you need to acquire a user account from Rockwell Automation Sales or Support.

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

Document Information

The document information covers various data related to the document.

Approval

This document has been approved electronically via the Rockwell Automation Document Management System (DMS). The required approvers of this document include the following:

Name	Role
Martin Dittmer	Product Manager
Steffen Landes	Development Manager
Martin Irmisch	Test Manager

In addition, the electronic document approval via DMS is confirmed by a handwritten signature of all approvers in the Quality Document when the release is completed. The Quality Document summarizes the quality-related planning activities and results of a PharmaSuite release.

Version Information

Object	Version
PharmaSuite	8.4
Time-based trigger	1.0 MR5
Counter-based trigger	1.0 MR5
Get values	1.0 MR4
Show values	1.0 MR4
Functional Requirement Specification	1.1

Revision History

The following table describes the history of this document.

Changes related to the document:

Object	Description	Document
Document	The document applies to the Get values phase 1.0 MR4 and the Set values phase 1.0 MR4.	1.1

Changes related to "IPC-related Phases and Operations" (page 3):

Object	Description	Document
---	---	---

Changes related to "Time-based Trigger Phase" (page 15):

Object	Description	Document
Pause Trigger Processing (SR0400.2.3) (page 17)	Update Step 20.2 added: In case no relevant ETO template was active during the pause period and the timeout period has not elapsed yet: trigger processing is still waiting and the timeout clock is running, however, the timeout clock is reset upon resume of the pause. No change of code.	1.0

Changes related to "Counter-based Trigger Phase" (page 23):

Object	Description	Document
Pause Trigger Processing (SR0405.2.3) (page 26)	Update Step 20.2 added: In case no relevant ETO template was active during the pause period and the timeout period has not elapsed yet: trigger processing is still waiting and the timeout clock is running, however, the timeout clock is reset upon resume of the pause. No change of code.	1.0

Changes related to "Get Values Phase" (page 37):

Object	Description	Document
System-triggered Exceptions (SR0440.3.2+) (page 56)	Update The message dialog of a system-triggered exception no longer provide a Cancel button.	1.0
Instruction (SR0440.8.1) (page 46)	Update The maximum length of the Instruction process parameter is 2000 characters (including HTML tags). No change of code.	1.0

Changes related to "Show Values Phase" (page 71):

Object	Description	Document
Instruction (SR0450.8.1) (page 77)	Update The maximum length of the Instruction process parameter is 2000 characters (including HTML tags). No change of code.	1.0

-
-
- Rockwell Software PharmaSuite® 8.4 - Functional Requirement Specification IPC Phases
-
-

C

Compliance-related

- SR0400.3+ - Exceptions (Time-based trigger) • 20
- SR0405.3+ - Exceptions (Counter-based trigger) • 32
- SR0440.3+ - Exceptions (Get values) • 56
- SR0450.3+ - Exceptions (Show values) • 78

Conventions (typographical) • 1

Counter-based trigger (SR0405+) • 23

- Automation error - Logic (SR0405.3.2.2.1) • 33
- Automation error - Resume (SR0405.3.2.2.2) • 33
- Automation error (SR0405.3.2.2) • 33
- Automation error exception (SR0405.8.7) • 31
- Batch record-related elements (SR0405.5.2) • 24
- Business logic (SR0405.2+) • 24
- Common sub-report elements (Framework capability) • 23
- Completion time (Framework capability) • 35
- Counter reset - Logic (SR0405.3.2.3.1) • 34
- Counter reset (SR0405.3.2.3) • 34
- Counter reset exception (SR0405.8.8) • 32
- Exceptions (SR0405.3+) • 32
- Fire triggers (SR0405.2.2) • 25
- Identified equipment entity (SR0405.8.5) • 29
- Identifier (Framework capability) • 35
- Instance count (Framework capability) • 34
- Interval definition (SR0405.8.1) • 30
- Numeric property (SR0405.8.6) • 29
- Output variables • 34
- Pause trigger processing (SR0405.2.3) • 26
- Phase activation (SR0405.2.1) • 24
- Phase completion (SR0405.2.5) • 29
- Process parameters (SR0405.8+) • 29
- Property Selection editor (SR0405.8.6.1) • 29
- Reading cycle (SR0405.8.9) • 30
- Representation during execution • 23
- Representation in Navigator • 23
- Representation in sub-report (SR0405.5+) • 23

- Resume trigger processing (SR0405.2.4) • 28
- Start time (Framework capability) • 34
- Sub-report elements (SR0405.5.1) • 24
- System-triggered exceptions (SR0405.3.2+) • 32
- Timeout - Logic (SR0405.3.2.1.1) • 32
- Timeout (SR0405.3.2.1) • 32
- Timeout exception (SR0405.8.4) • 31
- Timeout period (SR0405.8.3) • 30

D

- Data collection phase • 12
- Data representation phase • 12

E

- ETO • 3
- Event-triggered operation • 3

F

Framework capability

- Bundle output variable (Get values, Boolean value) • 69
- Bundle output variable (Get values, Measured value) • 68
- Bundle output variable (Get values, Option value) • 69
- Bundle output variable (Show values) • 80
- Bundle process parameters (Get values, Boolean value) • 53
- Bundle process parameters (Get values, Measured value) • 46
- Bundle process parameters (Get values, Option value) • 49
- Bundle process parameters (Show values) • 77
- Common sub-report elements (Counter-based trigger) • 23
- Common sub-report elements (Get values) • 41
- Common sub-report elements (Time-based trigger) • 15
- Common sub-report elements (Show values) • 74
- Completion time (Completion-based trigger) • 35
- Completion time (Get values) • 68

Completion time (Show values) • 80
Completion time (Time-based trigger) • 21
Identifier (Counter-based trigger) • 35
Identifier (Get values) • 68
Identifier (Show values) • 80
Identifier (Time-based trigger) • 22
Instance count (Counter-based trigger) • 34
Instance count (Get values) • 68
Instance count (Show values) • 79
Instance count (Time-based trigger) • 21
Option list editor (Get values) • 50
Phase column (Get values) • 40
Phase column (Show values) • 74
Start time (Completion-based trigger) • 34
Start time (Get values) • 68
Start time (Show values) • 80
Start time (Time-based trigger) • 21

G

Get values (SR0440+) • 37
Action column (SR0440.4.2) • 41
Active mode (SR0440.1.2) • 39
Boolean options (SR0440.8.15) • 53
Bundle output variable (Boolean value, Framework capability) • 69
Bundle output variable (Measured value, Framework capability) • 68
Bundle output variable (Option value, Framework capability) • 69
Bundle process parameters (Boolean value, Framework capability) • 53
Bundle process parameters (Measured value, Framework capability) • 46
Bundle process parameters (Option value, Framework capability) • 49
Business logic (SR0440.2+) • 41
Common sub-report elements (Framework capability) • 41
Completed mode (SR0440.1.3) • 40
Completion time (Framework capability) • 68
Confirm phase (SR0440.2.1) • 41

Correct value - Logic (SR0440.3.3.1.1) • 62
Correct value - Logic (SR0440.3.3.2.1) • 64
Correct value - Logic (SR0440.3.3.3.1) • 65
Correct value (SR0440.3.3.1) • 62
Correct value (SR0440.3.3.2) • 64
Correct value (SR0440.3.3.3) • 65
Correct value (SR0440.8.13) • 52
Correct value (SR0440.8.19) • 55
Correct value (SR0440.8.6) • 49
Data reference (SR0440.9.7) • 68
Decisions • 67
Error messages (SR0440.3.6+) • 67
Exceptions (SR0440.3+) • 56
Expected value configuration (SR0440.8.10) • 50
Expected value configuration (SR0440.8.16) • 54
Expected value definition (SR0440.8.11) • 51
Expected value definition (SR0440.8.17) • 54
Get Boolean Value (SR0440.2.6) • 44
Get Measured Value (SR0440.2.1) • 42
Get Option Value (SR0440.2.3) • 43
Identifier (Framework capability) • 68
Information column (SR0440.4.1) • 40
Information messages • 67
Instance count (Framework capability) • 68
Instruction (SR0440.8.1) • 46
Limit configuration (SR0440.8.4) • 47
Limit definition (SR0440.8.5) • 48
Limit violation - Logic (SR0440.3.2.1.1) • 56
Limit violation (SR0440.3.2.1) • 56
List of options (SR0440.8.9) • 50
Master (Bundle identifier) (SR0440.8.14) • 53
Master (Bundle identifier) (SR0440.8.2) • 46
Master (Bundle identifier) (SR0440.8.8) • 50
No value entered or selected (SR0440.3.6.1) • 67
Option key (SR0440.9.4) • 69
Option key (SR0440.9.6) • 70
Option list editor (Framework capability) • 50
Option text (SR0440.9.3) • 69
Option text (SR0440.9.5) • 69
Output variables (SR0440.9+) • 67
Override value - Logic (SR0440.3.1.1.1) • 59

Override value - Logic (SR0440.3.1.2.1) • 60
 Override value - Logic (SR0440.3.1.3.1) • 61
 Override value (SR0440.3.1.1) • 59
 Override value (SR0440.3.1.2) • 60
 Override value (SR0440.3.1.3) • 61
 Override value (SR0440.8.12) • 52
 Override value (SR0440.8.18) • 55
 Override value (SR0440.8.6) • 48
 Phase column (Framework capability) • 40
 Post-completion exceptions (SR0440.3.3+) • 62
 Preview mode (SR0440.1.1) • 38
 Process parameters (SR0440.8+) • 45
 Questions • 67
 Representation during execution (SR0440.1+) • 38
 Representation in Navigator (SR0440.4+) • 40
 Representation in sub-report (SR0440.5+) • 41
 Start time (Framework capability) • 68
 Sub-report elements (SR0440.5.1) • 41
 System-triggered exceptions (SR0440.3.2+) • 56
 Unit of measure (SR0440.8.3) • 47
 Unit of measure (SR0440.9.2) • 69
 User-triggered exceptions (SR0440.3.1+) • 59
 Validate Boolean Value (SR0440.2.7) • 45
 Validate Measured Value (SR0440.2.2) • 42
 Validate Option Value (SR0440.2.4) • 44
 Value (SR0440.9.1) • 69
 Violation of expected value - Logic (SR0440.3.2.2.1) • 57
 Violation of expected value - Logic (SR0440.3.2.3.1) • 58
 Violation of expected value (SR0440.3.2.2) • 57
 Violation of expected value (SR0440.3.2.3) • 58

I

IPC • 3

S

Show values (SR0450+) • 71
 Action column • 74
 Active mode (SR0450.1.2) • 73
 Automatic refresh (SR0450.3.4.1) • 79

Average (SR0450.9.1) • 80
 Bundle output variable (Framework capability) • 80
 Bundle process parameters (Framework capability) • 77
 Business logic (SR0450.2+) • 75
 Common sub-report elements (Framework capability) • 74
 Completed mode (SR0450.1.3) • 73
 Completion time (Framework capability) • 80
 Decisions • 79
 Definition (SR0450.8.2) • 77
 Error messages • 79
 Exceptions (SR0450.3+) • 78
 Identifier (Framework capability) • 80
 Information column (SR0450.4.1) • 74
 Information messages (SR0450.3.4+) • 79
 Instance count (Framework capability) • 79
 Instruction (SR0450.8.1) • 77
 Master (Bundle identifier) (SR0450.8.3) • 78
 Maximum (SR0450.9.3) • 80
 Minimum (SR0450.9.2) • 80
 Output variables (SR0450.9+) • 79
 Phase activation (SR0450.2.1) • 75
 Phase column (Framework capability) • 74
 Phase completion (SR0450.2.3) • 76
 Post-completion exceptions • 78
 Preview mode (SR0450.1.1) • 72
 Process parameters (SR0450.8+) • 77
 Questions • 79
 Representation during execution (SR0450.1+) • 72
 Representation in Navigator (SR0450.4+) • 74
 Representation in sub-report (SR0450.5+) • 74
 Standard deviation (SR0450.9.5) • 81
 Start time (Framework capability) • 80
 Statistics calculation (SR0450.2.2) • 76
 Sub-report elements (SR0450.5.1) • 74
 Sum (SR0450.9.4) • 81
 System-triggered exceptions • 78
 User-triggered exceptions • 78
 SR0400.2.1 - Phase activation (Time-based trigger) • 16
 SR0400.2.2 - Fire triggers (Time-based trigger) • 17

SR0400.2.3 - Pause trigger processing (Time-based trigger) • 17	SR0405.3.2.2.2 - Automation error - Resume (Counter-based trigger) • 33
SR0400.2.4 - Resume trigger processing (Time-based trigger) • 18	SR0405.3.2.3 - Counter reset (Counter-based trigger) • 34
SR0400.2.5 - Phase completion (Time-based trigger) • 19	SR0405.3.2.3.1 - Counter reset - Logic (Counter-based trigger) • 34
SR0400.2+ - Business logic (Time-based trigger) • 16	SR0405.3.2+ - System-triggered exceptions (Counter-based trigger) • 32
SR0400.3.2.1 - Timeout (Time-based trigger) • 21	SR0405.3+ - Exceptions (Counter-based trigger) • 32
SR0400.3.2.1.1 - Timeout - Logic (Time-based trigger) • 21	SR0405.5.1 - Sub-report elements (Counter-based trigger) • 24
SR0400.3.2+ - System-triggered exceptions (Time-based trigger) • 20	SR0405.5.2 - Batch record-related elements (Counter-based trigger) • 24
SR0400.3+ - Exceptions (Time-based trigger) • 20	SR0405.5+ - Representation in sub-report (Counter-based trigger) • 23
SR0400.5.1 - Sub-report elements (Time-based trigger) • 16	SR0405.8.1 - Interval definition (Counter-based trigger) • 30
SR0400.5.2 - Batch record-related elements (Time-based trigger) • 16	SR0405.8.3 - Timeout period (Counter-based trigger) • 30
SR0400.5+ - Representation in sub-report (Time-based trigger) • 15	SR0405.8.4 - Timeout exception (Counter-based trigger) • 31
SR0400.8.1 - Delay time (Time-based trigger) • 19	SR0405.8.5 - Identified equipment entity (Counter-based trigger) • 29
SR0400.8.2 - Cycle time (Time-based trigger) • 19	SR0405.8.6 - Numeric property (Counter-based trigger) • 29
SR0400.8.3 - Timeout period (Time-based trigger) • 20	SR0405.8.6.1 - Property Selection editor (Counter-based trigger) • 29
SR0400.8.4 - Timeout exception (Time-based trigger) • 20	SR0405.8.7 - Automation error exception (Counter-based trigger) • 31
SR0400.8+ - Process parameters (Time-based trigger) • 19	SR0405.8.8 - Counter reset exception (Counter-based trigger) • 32
SR0400+ - Time-based trigger • 15	SR0405.8.9 - Reading cycle (Counter-based trigger) • 30
SR0405.2.1 - Phase activation (Counter-based trigger) • 24	SR0405.8+ - Process parameters (Counter-based trigger) • 29
SR0405.2.2 - Fire triggers (Counter-based trigger) • 25	SR0405+ - Counter-based trigger • 23
SR0405.2.3 - Pause trigger processing (Counter-based trigger) • 26	SR0440.1.1 - Preview mode (Get values) • 38
SR0405.2.4 - Resume trigger processing (Counter-based trigger) • 28	SR0440.1.2 - Active mode (Get values) • 39
SR0405.2.5 - Phase completion (Counter-based trigger) • 29	SR0440.1.3 - Completed mode (Get values) • 40
SR0405.2+ - Business logic (Counter-based trigger) • 24	SR0440.1+ - Representation during execution (Get values) • 38
SR0405.3.2.1 - Timeout (Counter-based trigger) • 32	SR0440.2.1 - Get Measured Value (Get values) • 42
SR0405.3.2.1.1 - Timeout - Logic (Counter-based trigger) • 32	SR0440.2.2 - Validate Measured Value (Get values) • 42
SR0405.3.2.2 - Automation error (Counter-based trigger) • 33	SR0440.2.3 - Get Option Value (Get values) • 43
SR0405.3.2.2.1 - Automation error - Logic (Counter-based trigger) • 33	

- SR0440.2.4 - Validate Option Value (Get values) • 44
- SR0440.2.5 - Confirm phase (Get values) • 41
- SR0440.2.6 - Get Boolean Value (Get values) • 44
- SR0440.2.7 - Validate Boolean Value (Get values) • 45
- SR0440.2+ - Business logic (Get values) • 41
- SR0440.3.1.1 - Override value (Get values) • 59
- SR0440.3.1.1.1 - Override value - Logic (Get values) • 59
- SR0440.3.1.2 - Override value (Get values) • 60
- SR0440.3.1.2.1 - Override value - Logic (Get values) • 60
- SR0440.3.1.3 - Override value (Get values) • 61
- SR0440.3.1.3.1 - Override value - Logic (Get values) • 61
- SR0440.3.1+ - User-triggered exceptions (Get values) • 59
- SR0440.3.2.1 - Limit violation (Get values) • 56
- SR0440.3.2.1.1 - Limit violation - Logic (Get values) • 56
- SR0440.3.2.2 - Violation of expected value (Get values) • 57
- SR0440.3.2.2.1 - Violation of expected value - Logic (Get values) • 57
- SR0440.3.2.3 - Violation of expected value (Get values) • 58
- SR0440.3.2.3.1 - Violation of expected value - Logic (Get values) • 58
- SR0440.3.2+ - System-triggered exceptions (Get values) • 56
- SR0440.3.3.1 - Correct value (Get values) • 62
- SR0440.3.3.1.1 - Correct value - Logic (Get values) • 62
- SR0440.3.3.2 - Correct value (Get values) • 64
- SR0440.3.3.2.1 - Correct value - Logic (Get values) • 64
- SR0440.3.3.3 - Correct value (Get values) • 65
- SR0440.3.3.3.1 - Correct value - Logic (Get values) • 65
- SR0440.3.3+ - Post-completion exceptions (Get values) • 62
- SR0440.3.6.1 - No value entered or selected (Get values) • 67
- SR0440.3.6+ - Error messages (Get values) • 67
- SR0440.3+ - Exceptions (Get values) • 56
- SR0440.4.1 - Information column (Get values) • 40
- SR0440.4.2 - Action column (Get values) • 41
- SR0440.4+ - Representation in Navigator (Get values) • 40
- SR0440.5.1 - Sub-report elements (Get values) • 41
- SR0440.5+ - Representation in sub-report (Get values) • 41
- SR0440.8.1 - Instruction (Get values) • 46
- SR0440.8.10 - Expected value configuration (Get values) • 50
- SR0440.8.11 - Expected value definition (Get values) • 51
- SR0440.8.12 - Override value (Get values) • 52
- SR0440.8.13 - Correct value (Get values) • 52
- SR0440.8.14 - Master (Bundle identifier) (Get values) • 53
- SR0440.8.15 - Boolean options (Get values) • 53
- SR0440.8.16 - Expected value configuration (Get values) • 54
- SR0440.8.17 - Expected value definition (Get values) • 54
- SR0440.8.18 - Override value (Get values) • 55
- SR0440.8.19 - Correct value (Get values) • 55
- SR0440.8.2 - Master (Bundle identifier) (Get values) • 46
- SR0440.8.3 - Unit of measure (Get values) • 47
- SR0440.8.4 - Limit configuration (Get values) • 47
- SR0440.8.5 - Limit definition (Get values) • 48
- SR0440.8.6 - Override value (Get values) • 48
- SR0440.8.7 - Correct value (Get values) • 49
- SR0440.8.8 - Master (Bundle identifier) (Get values) • 50
- SR0440.8.9 - List of options (Get values) • 50
- SR0440.8+ - Process parameters (Get values) • 45
- SR0440.9.1 - Value (Get values) • 69
- SR0440.9.2 - Unit of measure (Get values) • 69
- SR0440.9.3 - Option text (Get values) • 69
- SR0440.9.4 - Option key (Get values) • 69
- SR0440.9.5 - Option text (Get values) • 69
- SR0440.9.6 - Option key (Get values) • 70
- SR0440.9.7 - Data reference (Get values) • 68
- SR0440.9+ - Output variables (Get values) • 67
- SR0440+ - Get values • 37
- SR0450.1.1 - Preview mode (Show values) • 72
- SR0450.1.2 - Active mode (Show values) • 73
- SR0450.1.3 - Completed mode (Show values) • 73
- SR0450.1+ - Representation during execution (Show values) • 72
- SR0450.2.1 - Phase activation (Show values) • 75
- SR0450.2.2 - Statistics calculation (Show values) • 76
- SR0450.2.3 - Phase completion (Show values) • 76
- SR0450.2+ - Business logic (Show values) • 75
- SR0450.3.4.1 - Automatic refresh (Show values) • 79

SR0450.3.4+ - Information messages (Show values) • 79
SR0450.3+ - Exceptions (Show values) • 78
SR0450.4.1 - Information column (Show values) • 74
SR0450.4+ - Representation in Navigator (Show values) • 74
SR0450.5.1 - Sub-report elements (Show values) • 74
SR0450.5+ - Representation in sub-report (Show values) • 74
SR0450.8.1 - Instruction (Show values) • 77
SR0450.8.2 - Definition (Show values) • 77
SR0450.8.3 - Master (Bundle identifier) (Show values) • 78
SR0450.8+ - Process parameters (Show values) • 77
SR0450.9.1 - Average (Show values) • 80
SR0450.9.2 - Minimum (Show values) • 80
SR0450.9.3 - Maximum (Show values) • 80
SR0450.9.4 - Sum (Show values) • 81
SR0450.9.5 - Standard deviation (Show values) • 81
SR0450.9+ - Output variables (Show values) • 79
SR0450+ - Show values • 71

Representation in sub-report (SR0400.5+) • 15
Resume trigger processing (SR0400.2.4) • 18
Start time (Framework capability) • 21
Sub-report elements (SR0400.5.1) • 16
System-triggered exceptions (SR0400.3.2+) • 20
Timeout - Logic (SR0400.3.2.1.1) • 21
Timeout (SR0400.3.2.1) • 21
Timeout exception (SR0400.8.4) • 20
Timeout period (SR0400.8.3) • 20
Trigger phase • 4

T

Time-based trigger (SR0400+) • 15
Batch record-related elements (SR0400.5.2) • 16
Business logic (SR0400.2+) • 16
Common sub-report elements (Framework capability) • 15
Completion time (Framework capability) • 21
Cycle time (SR0400.8.2) • 19
Delay time (SR0400.8.1) • 19
Exceptions (SR0400.3+) • 20
Fire triggers (SR0400.2.2) • 17
Identifier (Framework capability) • 22
Instance count (Framework capability) • 21
Output variables • 21
Pause trigger processing (SR0400.2.3) • 17
Phase activation (SR0400.2.1) • 16
Phase completion (SR0400.2.5) • 19
Process parameters (SR0400.8+) • 19
Representation during execution • 15
Representation in Navigator • 15