SAP Manufacturing Execution

How-To Guide

How To Set Up and Use the SAP ME Basic Routing Features

Applicable Release: ME 15.0

Version 1.0

January 2013

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided “as is” without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

SAP ME “How-to” Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP ME. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Document History

Document Version	Description	Author
1.0	Initial version	Mike Houston

Table of Contents

1 Introduction

1.1 Purpose

The purpose of this document is to describe how to use the Basic Routing features in SAP ME. These scenarios are supported in all versions of SAP ME.

Some new SAP ME 6.2 Non-serialized scenarios are described in 'SAP ME How-To-Guide – Non-Serialized Processing'. That document covers some of the more flexible 'flow' scenarios.

All of the scenarios in this document assume the routing does not have the ME 6.2 relaxed flow routing flag set.

1.2 Glossary

2 Feature Overview

2.1 Description and Applicability

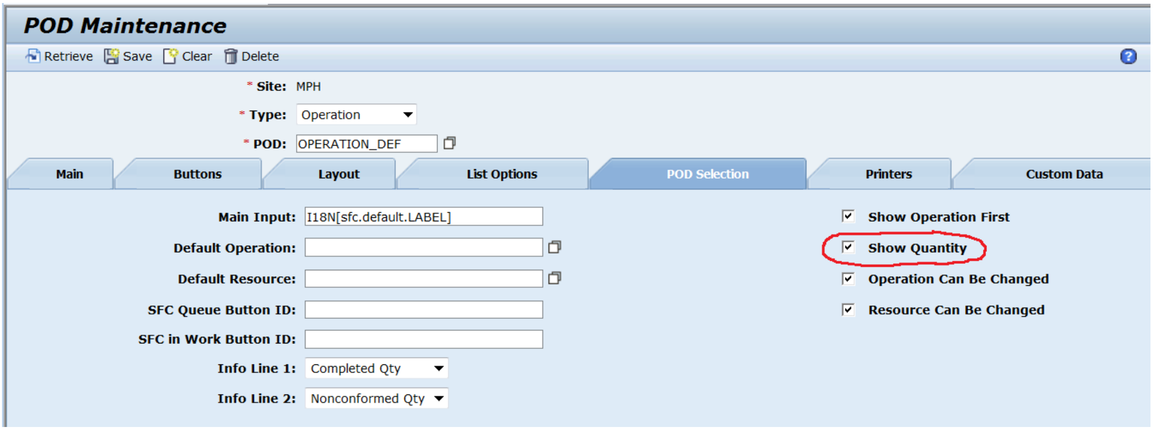
The routings defined in SAP ME provide a very flexible way to process materials on the shop floor. This includes a large number of scenarios and behaviors. The behaviors described in the document apply to all supported SAP ME releases.

This document will contain a number of scenario examples. These are meant to document the expected behavior for the many routing scenarios supported by SAP ME.

3 Setup

3.1 POD Maintenance

If partial quantity processing on SFCs is needed, then the POD definition in POD Maintenance needs to enable the 'Show Quantity' field to allow the operator to enter the quantity to be processed.



* Site: MPH

* Type: Operation

* POD: OPERATION_DEF

Main Input: I18N[sfc.default.LABEL]

Default Operation:

Default Resource:

SFC Queue Button ID:

SFC in Work Button ID:

Info Line 1: Completed Qty

Info Line 2: Nonconformed Qty

☑ Show Operation First

☑ Show Quantity

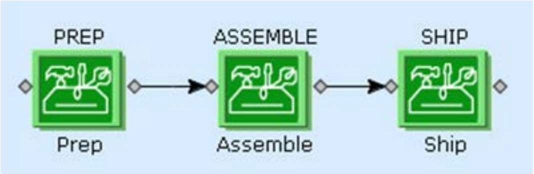
☑ Operation Can Be Changed

☑ Resource Can Be Changed

4 Usage Scenario Examples

4.1 Simple Linear Routing

A simple linear routing is shown below. This routing moves the SFC from one step to another when the SFC fully completes the previous steps. This routing does not contain any nonconformance routing logic within the routing itself.



4.1.1 Scenario Flow

The SFC is started and completed at each step in turn. As each step is completed, the full SFC quantity is placed in queue at the next step.

The SFC may not be started at the second step until the first step is fully completed. Upon initial release, the SFC quantity is placed in queue at the first step. The SFC cannot be started at the second step.

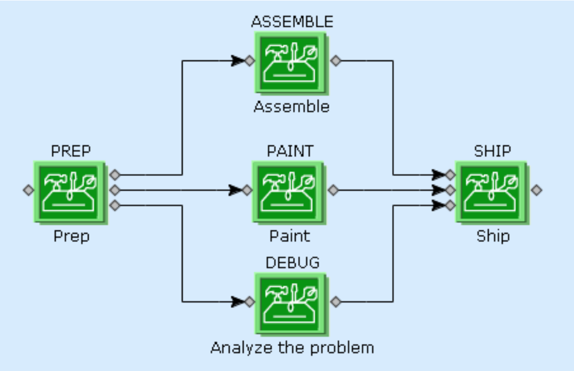
Once the SFC has moved to the second step, it cannot be started at the first or third step.

Completing the last step marks the SFC as done.

Partial starts and completes are allowed if the POD is configured to show the Quantity field. When partial quantities are completed, they remain at the completed step until the entire quantity is completed. These completed quantities are not in queue or in work anywhere.

4.2 Operator Chooses Next Step

Routings may have alternate paths in SAP ME. In this scenario, the completing operator chooses which step to process the entire SFC at upon completion.



4.2.1 Scenario Flow

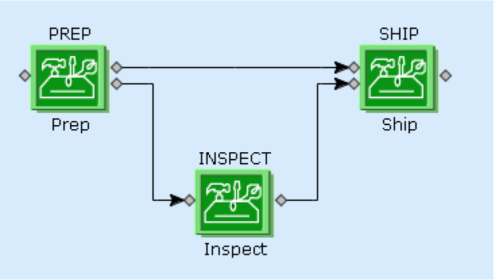
The SFC is started and completed at each step in turn. As each step is completed, the full SFC quantity is placed in queue at the next step. When the full quantity is completed at the PREP step, then the completing operator is prompted to pick the next step. The SFC will then be placed in queue at that step.

The SFC may not be started at the middle steps until the first step is fully completed. In the example above, once the full quantity is queued at PAINT, it may not be started at ASSEMBLE or DEBUG.

Partial starts and completes are allowed if the POD is configured to show the Quantity field. When partial quantities are completed, they remain at the completed step until the entire quantity is completed. These completed quantities are not in queue or in work anywhere.

4.3 Optional Steps

Routings may have optional steps. These are steps that can be bypassed by some operator's decision. In the case of the completing operator making this decision, the behavior is the same as the previous scenario (Operator Chooses Next Step). It is also possible to next operator to decide to bypass the optional step or not. For example:



In this scenario, the PREP step is defined with the 'Queue Decision' set to 'Next Operator'. This means the SFC is actually queue at both INSPECT and SHIP. The first operator to start this SFC at either of those steps will remove it from queue at the other steps.

4.3.1 Scenario Flow

The SFC is started and completed at each step in turn. As each step is completed, the full SFC quantity is placed in queue at the next step. When the full quantity is completed at the PREP step, it is placed in queue at INSPECT and SHIP.

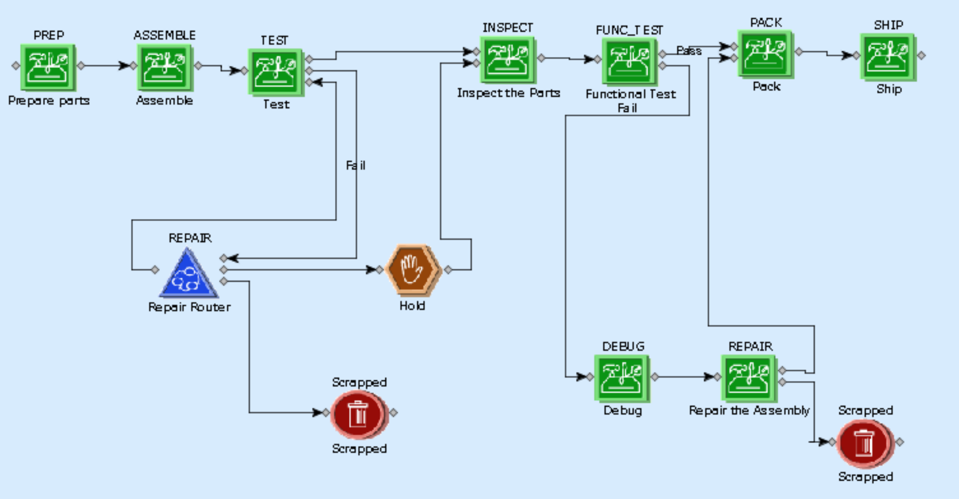
The first operator to start the SFC at INSPECT or SHIP will be able to start the SFC, Then the SFC will be removed from the queue at the other step.

The SFC may not be started at INSPECT or SHIP until it completes PREP.

Partial starts and completes are allowed if the POD is configured to show the Quantity field. When partial quantities are completed, they remain at the completed step until the entire quantity is completed. These completed quantities are not in queue or in work anywhere.

4.4 Embedded Rework Routings

This scenario is much more complex and involves multiple routings. The SFC may descend into the nested routing or may trigger specific actions such as scrap or hold. This routing has a nested sub-routing and an embedded debug/repair loop.



4.4.1 Scenario Flow

As the SFC completes out of TEST, it will go to the REPAIR sub-routing or to INSPECT. The entire SFC must travel to the REPAIR routing. Once it completes the REPAIR routing, it will return to TEST, be placed on hold or will be immediately scrapped.

Later processing may also trigger debug and scrapping as needed.

The special Hold and Scrap steps will be discussed in more detail in later scenarios in this document.

Completions out of the Repair Routing only support the Queue Decision of Completing Operator.

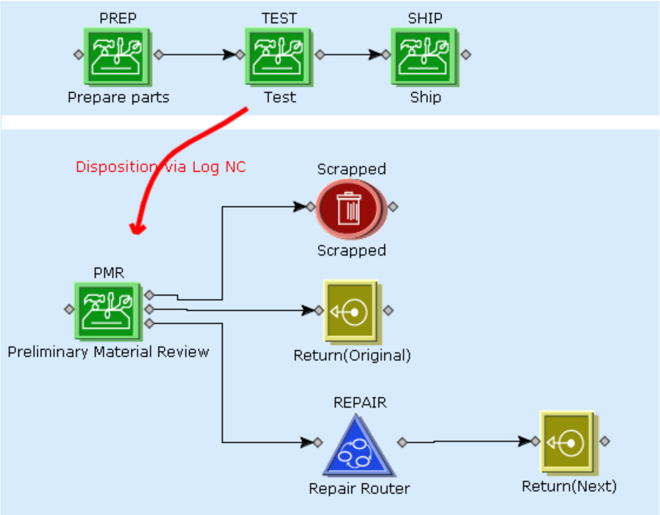
4.5 NC Disposition Routings

The SAP ME (NC) nonconformance feature is described in the SAP ME How-To-Guide – NC.

ME supports a type of routing that can be used as a disposition for an NC code as a special routing, an NC routing or a disposition routing. These all behave basically the same way.

- An NC code is defined.
- A disposition routing or disposition group is associated with the NC code.
- When the NC code is logged, the SFC can be dispositioned to the routing.

The example below shows how the built-in PMR routing can be used as a disposition routing. In this example, the main production routing has 3 steps and the PMR routing has a complex repair scenario.



4.5.1 Scenario Flow

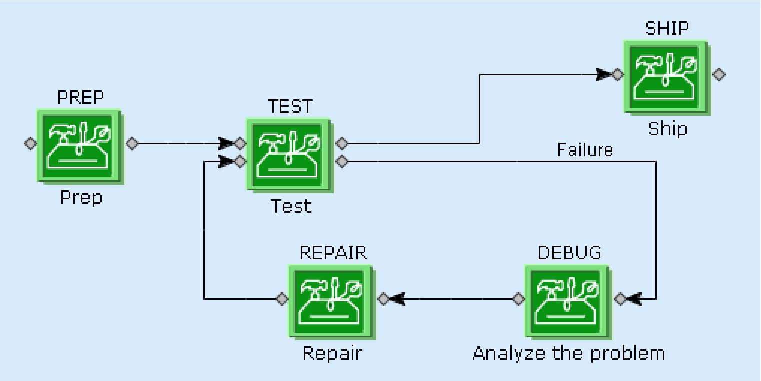
When the NC is logged and dispositioned at the TEST step, the entire SFC is sent to PMR routing. At the PMR step, the completing operator chooses whether to scrap the SFC, return it to the original routing or to repair the SFC.

The entire SFC must be sent to the PMR routing and return when finished with the PMR routing. Partial processing is not allowed when moving the SFC to the PMR routing.

4.6 Rework Loop

ME supports a type of a rework loop capability within the main production routing. This allows test and repair actions without using nonconformances or any special routing logic.

The example below shows how a TEST/DEBUG/REPAIR cycle can be used to detect, diagnose and fix problems with SFCs.



4.6.1 Scenario Flow

When the SFC is completed at TEST, the completing operator decides if the SFC passes or fails. If it passes, it is sent to SHIP. If it fails, then it is sent to DEBUG to analyze the problem and then to REPAIR to fix the SFC. After REPAIR, the SFC is then re-tested. This can start the whole loop again.

It is possible to limit the number of times the SFC can loop through an operation (see Maximum Loop Count on the routing step properties).

This scenario can also use the Queue Decision: Next Operator, but that does not make much sense with this particular routing. Any operator picking up the SFC at SHIP could cause the SFC to bypass the DEBUG operation.

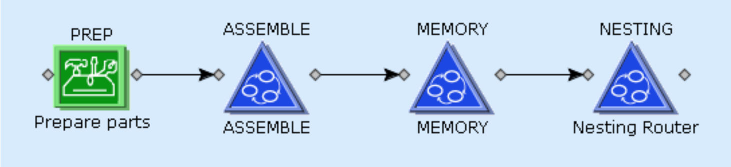
A Start followed by a Signoff at the TEST step leaves the loop counter (times processed in the database) unchanged.

4.7 Nested Routings

ME supports nested routings. These are routings that are made up of other routings. These routings are treated like normal steps in within a routing with the one main exception: the entire SFC must move into the nested routing.

These nested routings allow creation of re-usable building blocks for other routings.

The example below shows an ASSEMBLE, MEMORY and NESTING routing used on a main production routing.



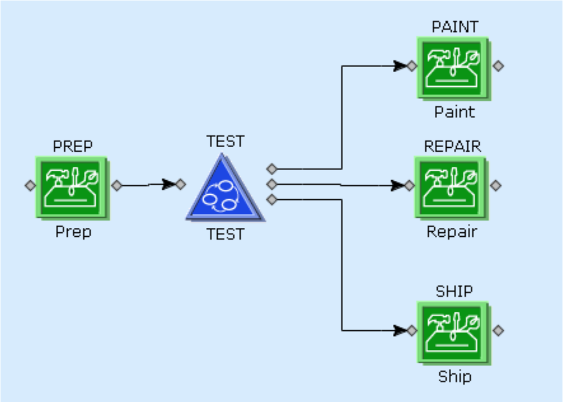
4.7.1 Scenario Flow

When the SFC is completed at PREP, the entire quantity moves to the first step on the ASSEMBLE routing. Once it completes the ASSEMBLE routing, it moves on to the MEMORY and the finally then NESTING routing. Once it completes the NESTING routing, it then moves to done.

4.8 Nested Routings with Multiple Next Steps

ME also supports nested routings with a few more complex scenarios. When returning from the nested routing, it is possible to have multiple next steps.

The example below shows a choice of 3 steps when returning from the TEST routing: PAINT, REPAIR or SHIP.



4.8.1 Scenario Flow

When the SFC is completed from the TEST routing, the completing operator or the next operator may decide which next step the SFC may move to.

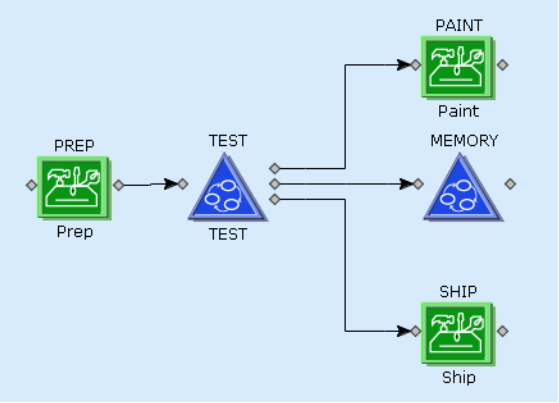
The entire SFC quantity must move to the nested routing at the same time and must leave the nested routing at the same time.

Partial processing is not fully supported as the SFC leaves the nested routing.

4.9 Nested Routings with Multiple Mixed Next Steps

ME also supports nested routings with a fairly complex configuration with different types of next steps. When returning from the nested routing, it is possible to have operations and other nested routings as next steps.

The example below shows a choice of 3 steps when returning from the TEST routing: the PAINT operation, the MEMORY routing or the SHIP operation.



4.9.1 Scenario Flow

When the SFC is completed from the TEST routing, the completing operator may decide which next step the SFC may move to.

The next operator cannot choose the next step since that would place the SFC in queue on more than one routing.

The entire SFC quantity must move to the nested routing at the same time and must leave the nested routing at the same time.

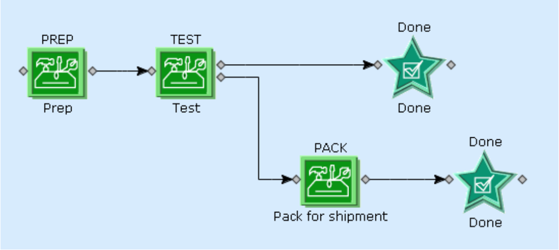
Partial processing is not fully supported as the SFC leaves the nested routing. When the final partial quantity is completed, the next step decision is made.

When the SFC is placed in queue on the MEMORY routing, the entire quantity is placed in queue at the first operation on the MEMORY routing.

4.10 Multiple Done Steps

ME provides a step to mark the entire SFC as done. There can be more than one of these types of steps on the routing.

The example below shows 2 choices. The completing operator at TEST will decide if the SFC is DONE or is sent to PACK.



4.10.1 Scenario Flow

When the SFC is completed from the TEST operation, the completing operator must decide which next step the SFC should move to. If the operator chooses the first 'Done' step, then the SFC is marked as DONE and no further processing is expected.

Partial processing is not fully supported with the Done step. When the final partial quantity is completed, the next step decision is made.

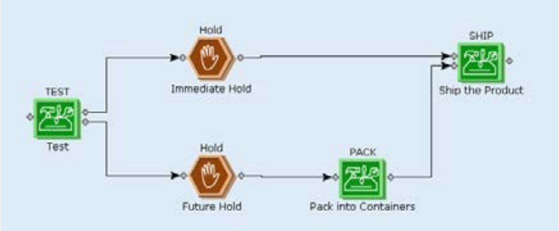
If there is no next step on the routing, then the SFC is also sent to done. The second Done step in the example above is optional.

4.11 Hold Steps

ME Routings have some immediate actions steps that trigger specific ME actions on an SFC. One such action step is the hold step.

Hold steps have two variant: immediate and future hold. Immediate hold steps will place the SFC on hold as soon as the SFC is queued at the hold step. Future hold steps will define a future hold for another step on the routing. When the SFC reaches that future hold step, the SFC will be placed on hold. In both cases, the routing must have a single operation step immediate following the hold step. The SFC will move through the hold step and be queued at that operation step.

The example router shows how the Hold step may be used to place an SFC on hold. The operator may choose to place the SFC on immediate hold or on a future hold at the SHIP step. The future hold will be enacted when the SFC is started (or attempted to start) at the SHIP step.



4.11.1 Scenario Flow

When the SFC is completed from the TEST operation, the completing operator must decide which next step the SFC should move to. If the operator chooses the 'Immediate Hold' step, then the SFC will be placed on hold and the queued at the SHIP operation. This hold must be released in order to process the SFC at the SHIP operation.

If the operator chooses the 'Future Hold' step, then the SFC will be queue at PACK and a future hold will be triggered when the SFC arrives at the SHIP step.

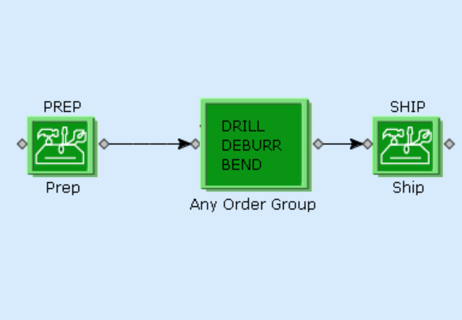
Partial processing is not allowed on hold steps. The entire SFC is processed and placed on hold.

Hold steps are not supported on production routings as the first step on the routing.

4.12 Any Order Steps

ME allows a group of steps to executed in any order. This is shown on the routing as a large Any Order step (green box) with the operations listed inside of the box. The steps within this box can be worked in any order, but only one step can be in work at a given time.

This router starts with an entry into the PREP operation. Upon release, the SFC is placed in queue at the PREP operation. When the SFC completes the PREP operation, the SFC is placed in queue at the DRILL, DEBURR and BEND operations. These operations can be performed in any order, but the SFC may only be work one at any given time. After it has completed the last of these operations, it is placed in queue at the SHIP operation. After completing the SHIP operation, the SFC is marked as done.



4.12.1 Scenario Flow

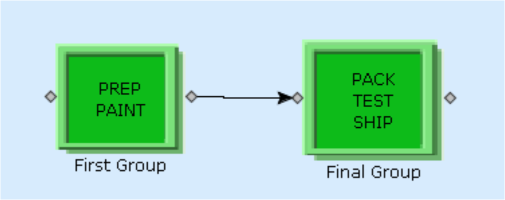
When the SFC is started at a step inside of the any order group, then quantity is removed from queue at all other steps in the group. This means no other user can start the SFC at the other operations.

If the SFC is has completed some of the steps in a step group, and it is pulled (or dispositioned) to another router, the return will be handled differently from normal returns. The SFC will be placed back in queue at all uncompleted steps in the group. This is because ME treats the whole step group as the step to return to.

Partial processing is supported inside of any order groups.

Signoff of a step within the any order group will return the quantity back into queue at the other steps.

It is possible to have two any order groups connect as shown below:



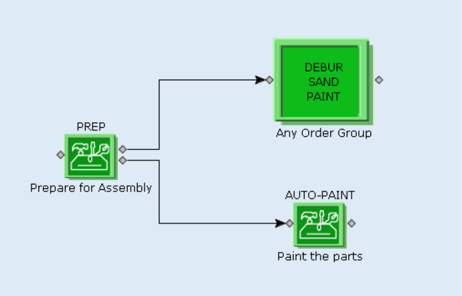
All steps in the first group must be completed before the second group is started.

4.13 Multiple Next Steps before Any Order Step

ME allows an operation step to feed into an any order step or a normal operation step. The Queue Decision supports only 'Completing Operator' when queuing at a step group.

The example below discusses the behavior when the Queue Decision is set to 'Completing Operator'.

Upon release, the SFC is placed in queue at the PREP operation. When the SFC completes the PREP operation the operator must choose if the SFC is to be placed in queue at the step group or the other operation.



4.13.1 Scenario Flow

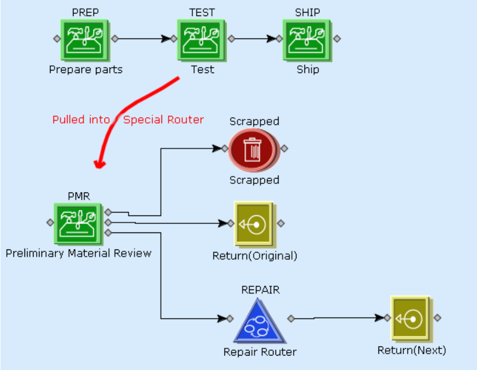
This next step logic works with any order group as possible next step.

This next step logic works with simultaneous group as possible next step.

Starting at one step and then performing signoff will return the SFC in queue to that step (or other steps within the any order group). The queue decision will not be allowed after the signoff.

4.14 Pulling Into Special Operations

A special operation can be used to 'pull' an SFC into an unplanned router. This is sometimes used for ad-hoc audits and inspections. Special operations are the first step on these special routings. When an operator starts the SFC at this special operation and the SFC is not already in queue at the operation, then the SFC is removed from its current routing and moved to the special routing.



An SFC is released to the production router (PREP, TEST, and SHIP). During the TEST operation, an auditor decides to pull the SFC into the PMR router for additional inspections. The SFC now travels down the PMR router and can eventually return to the production router.

Partial processing is not supported when pulling SFCs into special operations.

4.14.1 Scenario Flow

The SFC is worked down the production routing as normal.

An auditor can start the SFC at the PMR routing at any time. When this happens, the SFC is pulled from the current step on the production routing.

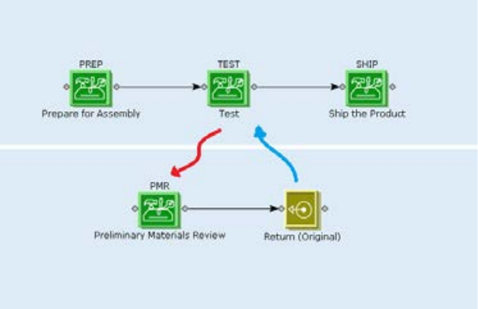
If the SFC is in work on the production routing, then it is signed-off from the production routing before it is moved to the special routing.

The production routing can have operation, any order or simultaneous group steps. See the scenario on returning to groups for more details.

See the return step scenarios for details on how the SFC returns to the original production routing.

4.15 Return Steps

Return steps are used to return to a step on the previous router. This is frequently used with NC Disposition routers and Special routers, but it can be used with other types of sub-routings. These return steps allow the customer to decide which step(s) the SFC can return to. There are options to return to the routing to the original step, next step or previous steps. There are also options to return to any step on the original routing.



An SFC is released to the production router (PREP, TEST, and SHIP). During the TEST operation, an NC is logged and the SFC is sent to the PMR routing. The SFC is completed at PMR and returned to the original step for re-test.

4.15.1 Scenario Flow

The SFC is worked down the production routing as normal.

The SFC is tested at TEST and an NC can be logged. The SFC is dispositioned to the PMR routing.

Partial processing at the original step is allowed. The entire SFC is sent to the PMR routing and the entire SFC is returned.

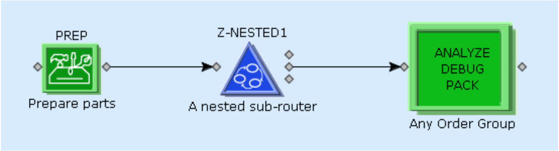
The return steps have these options:

- Return (Original) – Returns to the original step. The operator is not allowed to choose any steps.
- Return (Next) – Returns to the next step after the original step. The operator is not allowed to choose any steps.
- Return (Previous) – Returns to the step before the original step. The operator is not allowed to choose any steps.
- Return (Any) – Returns to any step on the original routing. The operator is allowed to choose any step on the original routing.
- Return (Any Previous) – Returns to any step on the original routing that the SFC has been processed on. The operator is allowed to choose any of these steps on the original routing.
- Return (Original and Previous) – The SFC may return to the original or the previous step. The operator must choose the specific step.

4.16 Return from Nest Routing to Step Group

After a nested routing, ME can use operation steps or a step group after the nested routing.

In the example below, the SFC returns from the nested routing into the any order group. This SFC is placed in queue at all steps in the any order group.



4.16.1 Scenario Flow

When returning from a nested sun-routing, the SFC is placed in queue at all steps in the step group.

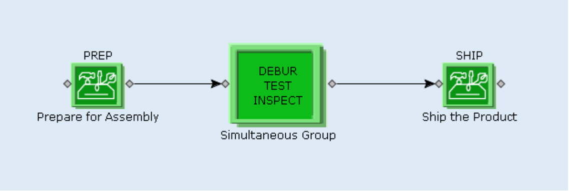
This also works the same with simultaneous groups (if the successors allow it).

4.17 Simultaneous Groups

SAP ME allows several operations to be performed at the same time within a given routing. The Simultaneous step group defines a list of steps that can be performed independently of each other by different operators.

In the example below, the SFC is releases to the PREP operation. When it is completed at PREP, the SFC is placed in queue at all steps in the simultaneous group (DEBUR, TEST and INSPECT). Any operator can start the SFC at any of these 3 steps independently of each other. All 3 steps can be active ay the same time.

When all 3 steps are completed, the SFC is then sent to the SHIP operation.



Some more complex scenarios can use successors to control how steps within the step group depend on each other. These successors will be described later.

4.17.1 Scenario Flow

The SFC is released to PREP. When completed at PREP, then entire SFC quantity is placed in queue at all steps in the step group.

The SFC can be started at one or more of the step group's steps at a given time.

When each step in the group is completed, a check is made on the other steps in the group. If all steps are completed, then the SFC leaves the step group and can be sent to the next step.

A step group can be the entry step on a routing.

A step group can be the last step on a routing.

A step group supports the normal queue decision options.

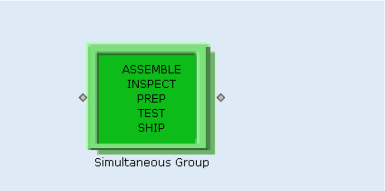
A step group can be the last step on a nested routing. This means the SFC will return to the original routing when the last step in the group is completed.

Partial processing can be used at a single simultaneous step.

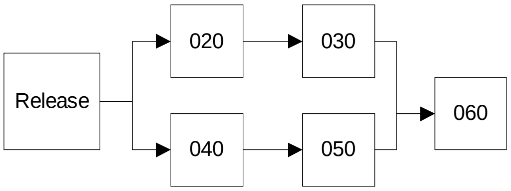
4.18 Simultaneous Group with Simple Successors

SAP ME allows several operations to be performed at the same time within a given routing. The Simultaneous step group defines a list of steps that can be performed independently of each other by different operators. The successors allow finer control over which steps are dependent on other steps within the group.

In the example below, the SFC is releases to the entry group, but the SFC is not placed in queue at all steps within the group. The successors defined which steps must be worked before a specific step can be started.



The successors force the SFC down two simultaneous paths as shown below.

Step	Operation	Successors	**Successor Map** (not visible in Routing Maintenance)
010	(Group)		
020	ASSEMBLE	030	
030	INSPECT	060	
040	PREP	050	
050	TEST	060	
060	SHIP		

Step 030 cannot be worked until step 020 is completed. Step 050 cannot be completed until step 040 is completed.

4.18.1 Scenario Flow

When an SFC enters a simultaneous group, the successors are checked to make sure the SFCs are placed in queue at all steps that do not depend on other steps within the group.

As steps are completed, the SFC is queued at other steps as the successor dependencies are satisfied.

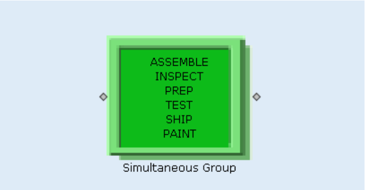
These simultaneous groups can be the entry step on a routing, a middle step or the final step on a routing.

Partial processing can be used at a single simultaneous step. When the full quantity at that step is completed, the step is considered completed. This is when the check for successors is made.

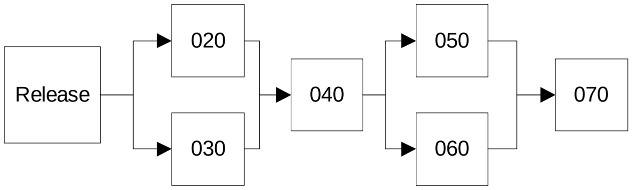
4.19 Simultaneous Group with Complex Successors

SAP ME allows several operations to be performed at the same time within a given routing. The Simultaneous step group defines a list of steps that can be performed independently of each other by different operators. The successors allow finer control over which steps are dependent on other steps within the group. There can be multiple step successors that can be used for more complex dependencies.

In the example below, the SFC is released to the entry group, but the SFC is not placed in queue at all steps within the group. The successors defined which steps must be worked before a specific step can be started.



The successors force the SFC down two simultaneous paths but have one common step on the two paths (step 040). Also, the last step (070) must wait until both of the precursor steps are completed.

Step	**Operation**	**Successors**	**Successor Map** (not visible in Router Maintenance)
010	(Group)		
020	ASSEMBLE	040	
030	INSPECT	040	
040	PREP	050, 060	
050	TEST	070	
060	PAINT	070	
070	SHIP		

4.19.1 Scenario Flow

When an SFC enters a simultaneous group, the successors are checked to make sure the SFCs are placed in queue at all steps that do not depend on other steps within the group.

As steps are completed, the SFC is queued at other steps as the successor dependencies are satisfied.

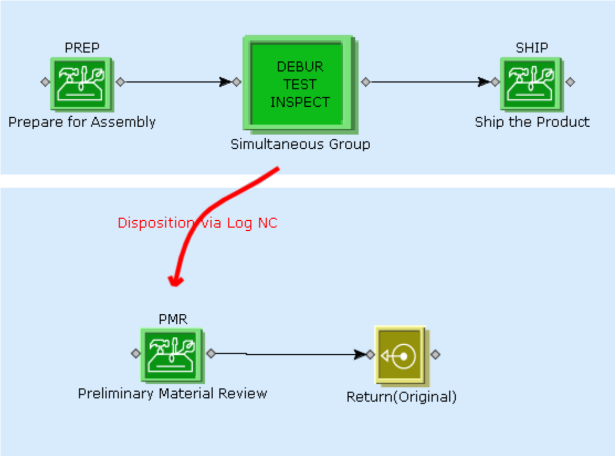
These simultaneous groups can be the entry step on a routing, a middle step or the final step on a routing.

Partial processing can be used at a single simultaneous step. When the full quantity at that step is completed, the step is considered completed. This is when the check for successors is made.

4.20 Return to Simultaneous Group

SAP ME allows several operations to be performed at the same time within a given routing. The Simultaneous step group defines a list of steps that can be performed independently of each other by different operators. When returning to a simultaneous group from another routing, ME does not know where to place the quantity in queue. ME will treat this scenario just like an initial entry into the group.

In the example below, even if DEBUG and TEST a recompleted, when the SFC is returned to the group, it will be placed in queue at all 3 steps.



4.20.1 Scenario Flow

When the SFC is pulled from the simultaneous group, the quantity is removed from all steps in the group. When returned to the main routing, it placed in queue at all steps in the simultaneous group (no successors scenarios).

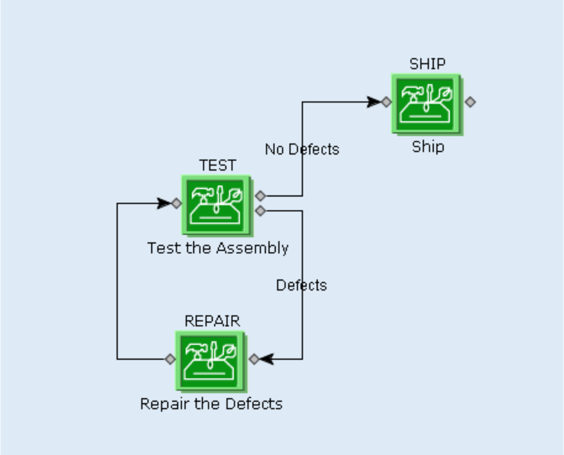
If there are successors defined for the group, then the return action will honor those successors as outlined for simultaneous groups (see previous scenarios).

When the SFC is finished with the group, it proceeds to the next step (if any).

4.21 Next Step Decisions with Scripting

SAP ME can define routings with multiple path options. This means that some decision must be made when leaving a step that has multiple next steps. This decision can be made by an operator or by a routing script.

Routing scripts are attached to the next step links. In the example below, a script is attached to the links coming from the TEST operation. In this case, the scripts are on the next step links themselves ('No Defects' and 'Defects' script in the example).



The scripting logic can be one of the pre-defined scripts or can be a custom script that is written in Javascript. A typical script decision is shown above. If not defects are logged at the TEST operation, the SFC moves to the SHIP operation. If any defects are found, then the SFC is sent to the REPAIR operation.

Scripts can be one of the pre-defined scripts are shipped with SAP ME or can be a custom script that is configurable at each next step link in the routing. Double clicking the line that represents the path will display the scripting dialog box.

4.21.1 Next Step Decisions

The next step decision is made when there are more than two possible next steps and the Queue Decision is 'Completing Operator'. The scripts are only executed when there are two or more possible next steps.

The scripting language is interpreted JavaScript. The scripts can make Enterprise Java Beans (EJB) calls or Structured Query Language (SQL) queries if needed. Each script is associated with the paths between steps.

There are a number of convenience methods (*callEJB()* and *getEJBProperties*(), *getItemProperties*(), *getCustomItemProperties*()) that make direct calls to EJBs. A large amount of data is also available to these scripts to help make next step decisions. The data and methods are discussed below.

When a step has multiple next steps, there will be multiple connectors exiting from the step. Each of the connectors can have a decision script associated with it. These scripts are evaluated when an SFC completes the step. The return values from all of these next steps are used to determine what the next step for the SFC will be.

Some example scripts are shown below:

Return	Next Step	Script
false	SHIP	`if (NC_CODE==null) exit( true;`
false	PMR ROUTER (Failure)	`if (NC_CODE!=null && LOOP_COUNT<3) exit( true;`
true	SCRAP (Failure – 3+ Passes)	`if (NC_CODE!=null && LOOP_COUNT>=3) exit( true;`
false	ASSEMBLE (Missing Component)	`if (NC_CODE.equals(“MISSING_COMP”) exit( true;`

Note: The scripting language is basically normal JavaScript, but there are some differences. In particular, all variables must be defined before using them. Also, there is no access to the GUI. Also, the exit() method is used to return a value to the next step logic.

4.21.1.1 Return Values

Each script returns a value indicating if the step should be the next step for the SFC or not. These return values can be simple true/false values or numeric to indicate importance. This allows script decisions to be based on the importance of the decision, not just a simple yes or no type decision.

The simplest case is when the scripts return **true** or **false**. This works well when the decision are mutually exclusive (e.g. no 2 steps are **true** at the same time). If several steps actually do return **true**, then user will still have to make a decision.

To make configuring the scripts easier, it is possible to return number value (a priority) to help decide which step the router should proceed to. This allows the customer to decide that one path will take priority over other valid (**true**) paths. This would allow the example above to return a higher priority for the “SCRAP (Failure – 3+ Passes)” path:

Return	Next Step	Script
false	SHIP	`if (NC_CODE==null) exit( true );`
true	PMR ROUTER (Failure)	`if (NC_CODE!=null) exit( true );`
600	SCRAP (Failure – 3+ Passes)	`if (NC_CODE!=null && LOOP_COUNT>=3) exit( 600 );`
false	ASSEMBLE (Missing Component)	`if (NC_CODE=="MISSING_COMP") exit( true );`

The numeric return keeps the scripts more independent of each other. For example, assume the customer changes the loop limit to 4 in the "SCRAP (Failure – 3+ Passes)" script. If the customer forgets to change the other script that checks the loop count, then the script may not handle all cases. By allowing one script to return a higher value (600), then the scripts are easier to maintain.

For comparison sakes, these return values are treated as priorities to allow some step scripts to return true or false and others to return numbers:

Return	Priority
True	500
False	0
(null)	499
(no script)	499

Note: Return steps with the Any Operation or Previous Operation set may have multiple next steps also. This means the same script can be executed multiple times, once for each possible return step.

Note: The scripts for the next steps are not executed in any specific order.

4.21.2 Data Available in Scripts

There are several variables defined that the scripts may use. Some of these are defined for the convenience of the script writer. They may exist inside of the *COMPLETE_DATA* or *NC_DATA* variables also.

The script variables are:

Variable Name	Description
COMPLETE_DATA	This contains the detailed information from the complete() action. See below for details (**Data**).
QTY	The Qty completed successfully (**int**). **Note:** This is not a fraction. If a fractional value is needed, then look at QTY in the COMPLETE_DATA (BigDecimal).
QTY_NON_CONFORMED	The Qty that was non-conformed (**int**). **Note:** This is not a fraction. If a fractional value is needed, then look at QTY in the COMPLETE_DATA (BigDecimal).
LOOP_COUNT and TIMES_PROCESSED	This is the loop count for the completing step. It is the TIMES_PROCESSED value from the SFC_STEP table (**int**).
OPERATION	The operation that is being completed. This is also available as a handle in COMPLETE _DATA (“OPERATION_BO”) (**String**).
TO_OPERATION	The operation that is the potential next operation. This is also available as a handle in COMPLETE _DATA (TO_OPERATION_BO) (**String**).
RESOURCE	The resource the step is being completed at. This is also available as a handle in COMPLETE _DATA (“RESOURCE_BO”) (**String**).
SITE	The site this complete took place in. (**String**).
SFC	The SFC being completed. This is also available as a handle in COMPLETE _DATA (“SFC_BO”) (**String**).
ITEM	The SFC’s Item. This is also available as a handle in COMPLETE _DATA (“ITEM_BO”) (**String**).
USER	The user completing the operation. This can be a resource if it is a process resource. This is also available as a handle in COMPLETE _DATA (“OWNER_GBO”) (**String**).
RETURN_OPERATION	If the next step is a Return step, then this will be the operation to return to on the original router (**String**).
RETURN_STEP_ID	If the next step is a Return step, then this will be the step to return to on the original router (**String**).
ELAPSED_TIME	The elapsed production time for the SFC (from start to complete) (**long** in milliseconds).
ELAPSED_QUEUE_TIME	The queue time for the SFC (before the start) (**long** in milliseconds).
NC_CODE	The NC Code for the failure at the completing operation. This is `null` if there was no failure logged. This is also available as a handle in NC_DATA (“NC_CODE_BO”) (**String**).
NC_DATA	The NC Data for the failure at the completing operation. This is `null` if there was no failure logged. This is a Data object containing the NC_DATA elements. See NCDataBOBean for details (**Data**).

FAILURE_ID	The Failure ID for the failure at the completing operation. This is `null` if there was no NC logged or if the NC had no Failure ID entered (**String**).
NC_DATA_REF_DES	This is a sub-Data object that contains the details of the component the NC was logged at. It contains the next two elements (REF_DES and COMPONENT_BO). This Data object can contain multiple rows when multiple REF_DES's are logged. **(Data**).
REF_DES	The Ref Des for the failure at the completing operation. This is `null` if the NC had no Ref Des entered. (**String**).
NC_COMPONENT_BO	The component that was failed (**BOHandle**).

This is the Data from the Complete() action:

Variable Name	**Description**
SITE	The site the complete took place in (**String**).
USER_BO	The user who performed the complete (**BOHandle**)
ACTIVITY	The activity that triggered the complete (**String**).
OPERATION_BO	The operation that was completed (**BOHandle**)
RESOURCE_BO	The resource the operation was completed at (**BOHandle**)
OWNER_GBO	The owner of the SFC (User or Resource if the resource is a process resource). (**BOHandle**)
SFC_BO	The SFC that was completed (**BOHandle**).
ITEM_BO	The item for the SFC (**BOHandle**)
TIMES_PROCESSED	The LOOP_COUNT (**BigDecimal**).
QTY	The Qty completed successfully.(**BigDecimal**).
QTY_NON_CONFORMED	The Qty that was non-conformed (failed).(**BigDecimal**).

4.21.3 PAPI Calls From Within Routing Scripts

Public API calls can be made within the routing scripts. This involves the following code:

```
importClass(Packages.com.visiprise.frame.configuration.ServiceReference);
importClass(Packages.com.visiprise.frame.service.ServiceLocator);
importClass(Packages.com.sap.me.common.ObjectReference);
ref=new ServiceReference("com.sap.me.productdefinition","ItemConfigurationService");
ser = ServiceLocator.getService(ref);
item=ser.readItem(new ObjectReference("ItemBO:START1,PC,A"));
```

4.21.4 Script Methods Available

The methods described below are available for scripts.

4.21.4.1 exit()

This method stops the script execution and returns a value to the Next Step logic. This value is used to determine which next step is used.

4.21.4.2 getItemProperty() [SFC, Operation, Resource, User]

This method allows the script to get a single property from the Item, SFC, Operation, Resource or User object. For example, getSFCProperty("QTY");

If more than one property is needed, the script should use the **getEJBProperties()** method instead.

4.21.4.3 getCustomItemProperty() [SFC, Operation, Resource]

This method allows the script to get a single custom property from the Item, SFC, Operation or Resource object. For example, getCustomSFCProperty("COLOR");

4.21.4.4 getEJBProperties()

This method allows the script to get column values from EJBs associated with database tables. Use this method to retrieve simple fields or sub-tables (e.g. Operation Certifications).

4.21.4.5 callEJB()

This method allows the script to call any method on an EJB. The method is somewhat complex and requires knowledge of the EJB being called. Therefore, use this method only as a last resort. Please consult Professional Services Organization (PSO) for more information about using this method.

4.21.4.6 print(value,…)

This method prints one or more values to the Netweaver console. This should only be used for debugging and testing purposes.

4.21.4.7 printAll()

This method prints all of the variables available to the script. This should only be used for debugging and testing purposes.

4.21.4.8 executeQuery(SQLString)

This method executes a single SQL query and returns the results in a Data object. This may be null if no records are found.

4.21.4.9 getOpenNCs()

This method returns any open NCs for the SFC being completed. This returns a Data object containing the NC_DATA information. See the javadoc for NCDataBOBean.getOpenNCs() for details.

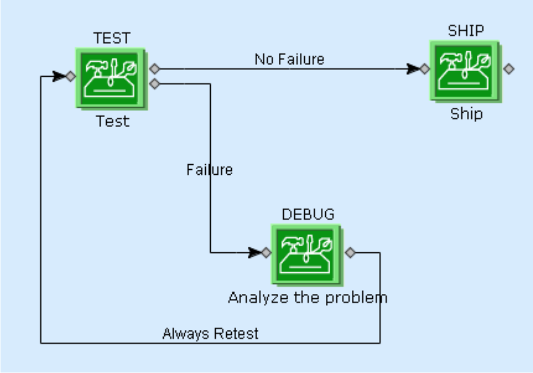
4.21.5 Exceptions and Errors

If an error is found executing a script, then the script engine will throw an exception. This will cause the ‘complete’ transaction to rollback with an error message. This rollback is also done if there are any other exceptions thrown by the script or any methods it calls. If the customer doesn't want to fail on script errors, then he/she must 'catch' the exceptions in the script. Script syntax errors cannot be caught this way.

4.21.6 Examples

4.21.6.1 Simple Test/Debug Loop

This is a simple router that starts with a TEST operation followed by either the SHIP or DEBUG step. The DEBUG step always returns to TEST for a re-test.



The script logic associated with the TEST step is show below:

From/To Steps	Script
TEST to SHIP *(No Failure)*	`if (NC_CODE==null) exit( true );`
TEST to DEBUG *(Failure)*	`if (NC_CODE!=null) exit( true );`

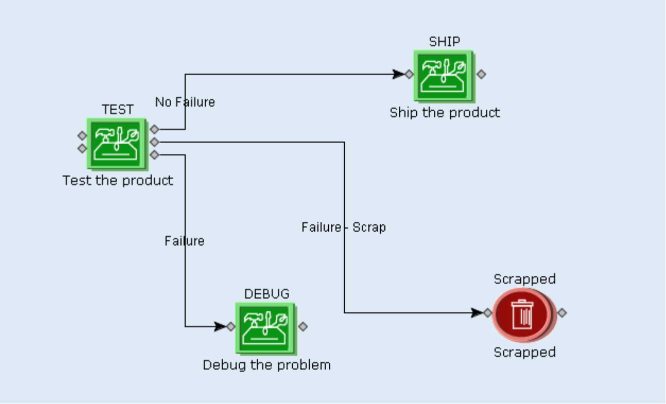
The other next steps have no scripts associated with them. There are two possible scenarios with this router:

- The SFC passes the TEST and proceeds to SHIP.
- The SFC fails the TEST and is sent to DEBUG. The SFC is repaired at DEBUG by adding a repair NC Code. The SFC is then sent back to TEST. It passes the TEST and is sent to SHIP.

4.21.6.2 Custom Data Fields

Various data values can be used in the next step decision. These can be core ME values from the SFC, Item, etc or they can be custom data fields associated with various elements in ME.

This example uses a simple routing:



The script logic associated with the TEST next step is show below:

From/To Steps	Script
TEST to SHIP *(No Failure)*	`if (NC_CODE==null) exit( true;`

TEST to DEBUG
(Failure)
	`if (NC_CODE==null) exit(false);`
	`// Get the custom property COST`
	`cost=getCustomItemProperty("COST");`
	`// If the item is worth more than $500, go to DEBUG`
	`if (cost>500)`
	`  exit(true);`
	`else`
	`  exit (false);`

TEST to DEBUG
(Failure - Scrap)
	`if (NC_CODE==null) exit(false);`
	`// Get the custom property COST`
	`cost=getCustomItemProperty("COST");`
	`// If the item is worth less than $500, go to SCRAP`
	`if (cost<500)`
	`  exit(true);`
	`else`
	`  exit (false);`

This logic bases the next step decision on the cost of the material when a failure is detected. If the material (Item) costs less than $500, then the SFC is scrapped rather than repaired.