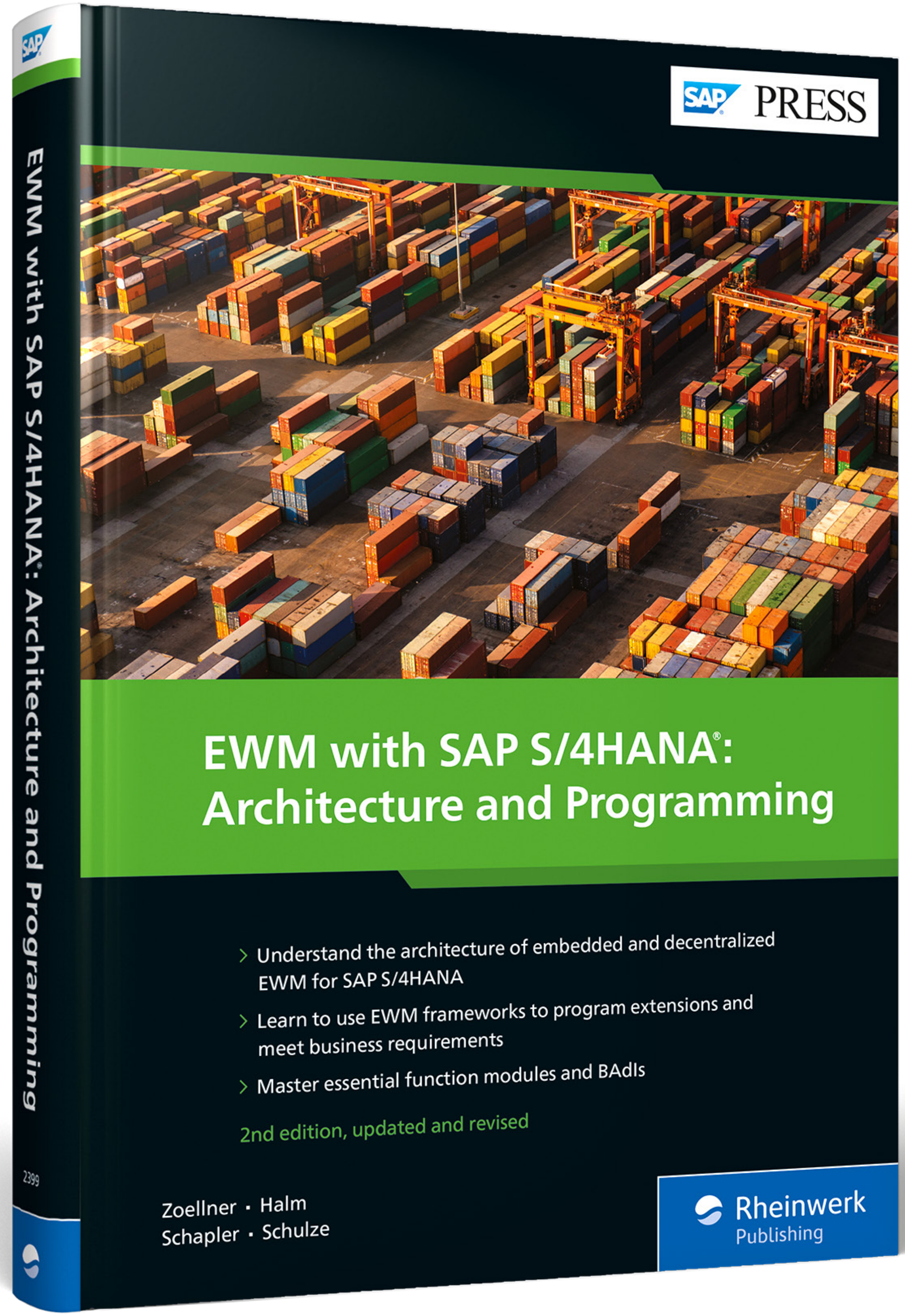


Reading Sample

This selection walks through a number of custom developments, based on SAP Best Practices for embedded EWM, that can be adopted within a warehouse. The first section details the functionality used within the SAP Best Practices processes, technically referred to as scope items, and how to install them. Then it covers how to locate and test custom developments within the selected processes. Each later section deals with a single, specific process scenario.

- "Enhancing SAP Best Practices for Embedded EWM"
- Contents
- Index
- The Authors

Zoellner, Halm, Schapler, Schulze

EWM with SAP S/4HANA: Architecture and Programming

493 pages | 07/2023 | $89.95 | ISBN 978-1-4932-2399-2

www.sap-press.com/5670

Chapter 4
Enhancing SAP Best Practices for Embedded EWM

In this chapter, we provide a number of custom developments that you can adopt and use for your own projects. The foundation for these example enhancements is SAP Best Practices for embedded EWM with SAP S/4HANA.

In the previous chapters, you learned about the architecture of EWM and the enhancement frameworks that it offers. Now we would like to give you the opportunity to put into practice what you have learned.

Skilled project personnel that are both functionally and technically oriented, and who dare to look beyond their own expertise, provide the foundation of a successful EWM implementation project. Robust custom developments are most likely to come about when all parties of the development process bring to the table a clear understanding of software requirements and a solid solution approach for optimal implementation. It is from this perspective that we have written this chapter.

As both solution architects and developers alike, you will get an overview of the processes in the warehouse and an understanding of core processes in EWM. In addition, you will get to know the use of features, frameworks, and best practices that should be considered for the realization of custom developments in EWM. This chapter therefore applies to more functionally oriented consultants and to technically oriented developers who want to broaden their horizons for custom development options in EWM.

We have decided to use the available SAP Best Practices for embedded EWM scenarios and processes as the foundation on which to present custom development examples for EWM. SAP Best Practices for embedded EWM contain the core functionalities used in warehouses and can be quite rapidly installed on any SAP S/4HANA system landscape. We will enlarge the functionality within some of these processes by custom developments.

In the first section of this chapter, you will learn about the functionality used within the SAP Best Practices processes, technically referred to as *scope items*, and how to install them. Then you will learn how to locate and test custom developments within the selected processes. Further sections deal with single, specific process scenarios for warehouse management. We'll cover two processes for goods receiving, two for goods issue, and one for warehouse internal activities:

- Basic warehouse inbound processing from a supplier (Section 4.2)
- Warehouse inbound processing from a supplier with batch management (Section 4.3)
- Basic warehouse outbound processing to a customer (Section 4.4)
- Advanced warehouse outbound processing to a customer (Section 4.5)
- Physical inventory in the warehouse (Section 4.6)

For each process, we will pay special attention to the custom developments for you to reenact.

Adopting and Testing the Custom Developments in a Project

Consider the example custom developments as templates. If you plan to use one or another custom development presented in this chapter within your project, you should thoroughly check its fit and correctness within your project's context. In particular, you should pay attention to how the performance of your custom developments fit with the data volumes processed and the overall system load.

4.1 Introduction to SAP Best Practices for Embedded EWM

SAP Best Practices for SAP S/4HANA have been made available to allow for quick implementation of core standard based organizational structures and process flows. They contain preconfiguration and optional demo data content that can easily be activated in a system using the respective solution implementation tools. SAP Best Practices align to SAP S/4HANA versions and span several countries and languages.

In the area of supply chain management, and warehouse management specifically, the SAP Best Practices content is provided for both embedded and decentralized EWM. Embedded EWM-based processes will be used and described within this chapter.

A subset of the preconfigured processes for embedded EWM forms the basis for a variety of custom developments that we will explain in the following sections. You can certainly try out and use the examples within your self-configured processes, but they might have to be adapted first to the individual situation. However, if you have the opportunity to install SAP Best Practices for EWM in your system before you start programming, we recommend doing so, because it forms the basis upon which our custom developments were built and tested.

We will now look at the functional scope of the SAP Best Practices for embedded EWM and present an overview of the enhancements within the scope items, which will be described in much detail in the following sections. Last but not least, we will provide some information on how to get your system prepared to reenact the processes and enhancements.

4.1.1 Functional Scope of SAP Best Practices for Embedded EWM

As mentioned earlier, SAP Best Practices for SAP S/4HANA represent a library of scope items. The contained scope items for embedded EWM include typical warehouse management scenarios of inbound, outbound, and internal operation within one warehouse number. Fundamental settings have been included for organizational structure, master data, and resource management, as well as for integration with the SAP S/4HANA system, which is critical in ensuring the operation of all processes. The 11 scenarios and processes shown in Table 4.1 are available in SAP Best Practices for embedded EWM in SAP S/4HANA 2022. During activation of the SAP Best Practices solution, items from this set of processes can be chosen for custom installation.

Scenario	ID	Process
Inbound	1FS	Basic warehouse inbound processing from supplier
	1V5	Warehouse inbound processing from supplier with batch management
	1V9	Basic warehouse inbound processing from supplier with quality management
Outbound	1G2	Basic warehouse outbound processing to customer
	1V7	Warehouse outbound processing to customer with batch management
	1VD	Advanced warehouse outbound processing to customer
	1G0	Scrapping in warehouse
Internal	1FY	Replenishment in warehouse
	1FW	Physical inventory in warehouse
Cross	1FU	Initial stock upload for warehouse
	1VB	Production integration—component consumption and receipt in warehouse

Table 4.1 Scope Items of SAP Best Practices for Embedded EWM

In the following paragraphs, we will provide you with an overview of SAP Best Practices for embedded EWM in terms of organizational structures and integration with other SAP S/4HANA functionality—predominantly materials management and logistics execution.

Figure 4.1 provides an overview of the organizational structures used for the integration with SAP S/4HANA. The SAP S/4HANA warehouse number is directly linked to the EWM warehouse number, whereas the SAP S/4HANA storage locations are mapped to EWM stock types via availability groups. You can find further details on the mapping of SAP S/4HANA and EWM stock models in Chapter 2, Section 2.5. When setting up the SAP

S/4HANA integration, you will be able to choose the organizational structures of the SAP S/4HANA modules that you would like to integrate with the EWM warehouse number.

How-to Guide for Basic Settings for SAP EWM in SAP S/4HANA (Embedded EWM)

We recommend reading the *How-To Guide for Basic Settings for SAP EWM in SAP S/4HANA* when setting up the SAP S/4HANA to EWM integration configuration for using embedded EWM in SAP S/4HANA. It contains valuable information on configuration of qRFC communication, warehouse creation, and integration into the enterprise organizational structure. Alternatively, search for “Pre-Activation Settings for Embedded EWM Scope Items” in the SAP Help Portal (*https://help.sap.com/*).

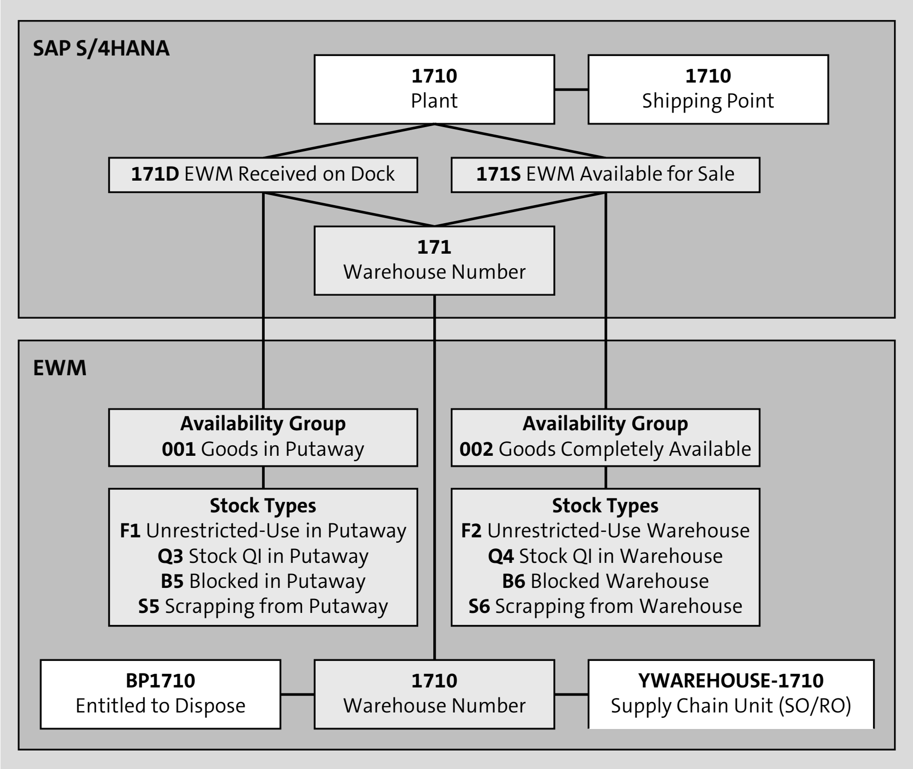

SAP S/4HANA
1710 Plant
1710 Shipping Point
171D EWM Received on Dock
171S EWM Available for Sale
171 Warehouse Number
EWM
Availability Group 001 Goods in Putaway
Availability Group 002 Goods Completely Available
Stock Types
F1 Unrestricted-Use in Putaway
Q3 Stock QI in Putaway
B5 Blocked in Putaway
S5 Scrapping from Putaway
Stock Types
F2 Unrestricted-Use Warehouse
Q4 Stock QI in Warehouse
B6 Blocked Warehouse
S6 Scrapping from Warehouse
BP1710 Entitled to Dispose
1710 Warehouse Number
YWAREHOUSE-1710 Supply Chain Unit (SO/RO)

Figure 4.1 SAP Best Practices for Embedded EWM: Organizational Structure for Integration with SAP S/4HANA

Figure 4.2 shows the organizational structure of warehouse number 1710 as used in the individual processes. Warehouse number 1710 is assigned a variety of storage types that are used for the different material movements within the processes. These primarily differ by product size. Inside the storage types you can find storage sections, which again are broken down by product demand—namely, fast, medium, and slow movers. Doors, staging areas, and work centers can be found for inbound and outbound operations.

Warehouse number 1710 also includes storage types for clarification and scrapping. The use of storage types is documented in detail in the process descriptions of the various processes, which are included with the documentation of the respective SAP Best Practices scope items.

You can find additional information for the use of master data and settings for resource management in the test cases of SAP Best Practices scope items as well as individual step-by-step process descriptions. They contain detailed information on the procedures and transactions used for each of the 11 available scope items. You can find them in the Process Navigator by SAP on the SAP for Me platform at *https://me.sap.com/processnavigator*. You need registration with SAP (S-User) to be able to access this content.

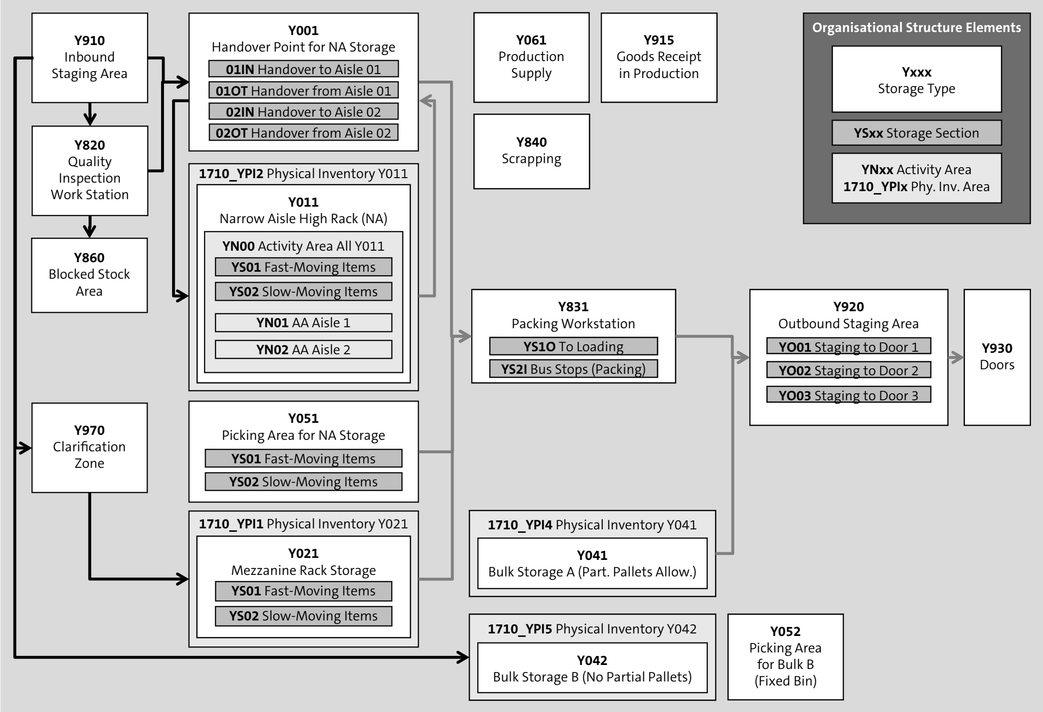

Y910 Inbound Staging Area
Y820 Quality Inspection Work Station
Y860 Blocked Stock Area
Y970 Clarification Zone
Y001 Handover Point for NA Storage
01IN Handover to Aisle 01
01OT Handover from Aisle 01
02IN Handover to Aisle 02
02OT Handover from Aisle 02
1710_YPI2 Physical Inventory Y011
Y011 Narrow Aisle High Rack (NA)
YN00 Activity Area All Y011
YS01 Fast-Moving Items
YS02 Slow-Moving Items
YN01 AA Aisle 1
YN02 AA Aisle 2
Y051 Picking Area for NA Storage
YS01 Fast-Moving Items
YS02 Slow-Moving Items
1710_YPI1 Physical Inventory Y021
Y021 Mezzanine Rack Storage
YS01 Fast-Moving Items
YS02 Slow-Moving Items
Y061 Production Supply
Y915 Goods Receipt in Production
Y840 Scrapping
Y831 Packing Workstation
YS1O To Loading
YS2I Bus Stops (Packing)
1710_YPI4 Physical Inventory Y041
Y041 Bulk Storage A (Part. Pallets Allow.)
1710_YPI5 Physical Inventory Y042
Y042 Bulk Storage B (No Partial Pallets)
Y052 Picking Area for Bulk B (Fixed Bin)
Y920 Outbound Staging Area
YO01 Staging to Door 1
YO02 Staging to Door 2
YO03 Staging to Door 3
Y930 Doors
Organisational Structure Elements
Yxxx Storage Type
YSxx Storage Section
YNxx Activity Area
1710_YPIx Phy. Inv. Area

Figure 4.2 SAP Best Practices for Embedded EWM: Organizational Structures Used per Scope Item

Let’s get an overview of the inbound functionality of the EWM system used in the scope items for embedded EWM:

- **Delivery processing**
  - Creation of inbound delivery in SAP S/4HANA with purchase order reference
  - Creation of inbound delivery in EWM with purchase order reference
  - RF-based receiving of handling units
  - (Partial) goods receipt posting on handling unit level

- **Warehouse logistics**
  - Automatic warehouse task creation for inbound delivery via PPF action
  - Process-oriented storage control
  - Layout-oriented storage control
  - Usage of quality inspection work center
  - Printing of warehouse order
  - Warehouse task confirmation with difference correction (supplier/warehouse)
  - Warehouse task confirmation with put-away physical inventory
  - Warehouse task/warehouse order confirmation with RF
- **Quality inspection**
  - Warehouse task creation as follow-up activity of quality inspection usage decision
- **Stock management**
  - Automatic posting change from received on dock to available for sales stock
  - Automatic posting change from quality to un-restricted use stock
  - Automatic posting change from quality to scrap stock

Let's also look at the functionality of outbound scope items, as follows:

- **Delivery processing**
  - Creation of outbound deliveries in SAP S/4HANA with sales order reference
  - Batch selection in outbound delivery item
  - Printing of outbound delivery note
- **Warehouse logistics**
  - Manual warehouse task creation via PPF action
  - Automatic wave assignment via PPF action (advanced)
  - Process-oriented storage control
  - Layout-oriented storage control
  - Warehouse order creation with packing profile
  - Picking partial handling unit quantities
  - Packing and confirmation of pick handling units as ship handling units via RF
  - Printing of shipping labels
  - Warehouse task confirmation via RF
  - Warehouse task confirmation with exception handling for differences in picking and packing
  - Movement of handling units to staging area
  - Controlled loading of handling units to transportation unit

- **Shipping**
  - Manual creation of transportation units (advanced)
  - Manual assignment of doors
  - Manual assignment of deliveries to transportation units
  - Printing of loading lists
- **Stock management**
  - Manual goods issue posting for outbound delivery
  - Manual goods issue posting for transportation unit (advanced)

The internal warehouse processes use the standard functionality as provided for inventory procedures (periodic and cycle counting), scrapping, and automatic replenishment.

4.1.2 Overview of Enhancements within the Scope Items

The basic inbound and outbound scope items of the SAP Best Practices for embedded EWM already contain much of the core warehouse management functionality, focusing on delivery processing, SAP S/4HANA integration, and a variation of different put-away and picking procedures. Further scope items then broaden the scope of warehouse management to include the advanced functionality of batch management, shipping and receiving, and more complex warehouse logistics supported by the use of wave management and quality management. We have tried to position the example enhancements within the scope items accordingly. Table 4.2 gives an overview of the enhancements that we present in this chapter, indicating the framework or enhancement technique used for each development.

ID	Custom Development	Realized By
1FS	1FSa: Automatic handling unit creation without packaging specifications	BAdI
	1FSb: Simplify the screen flow for RF putaway	RF framework
1V5	1V5a: Permit activation of the transportation unit only after entering the license plate and pager	BAdI
	1V5b: Putaway depending on quarantine period	BAdI
	1V5c: Enhancing the warehouse monitor	Warehouse monitor
	1V5d: Delay inbound delivery with missing batch	BAdI

Table 4.2 Overview of Enhancements

ID	Custom Development	Realized By
1G2	1G2a: Enhancing the delivery interface by custom data	EEW, BAdI
	1G2b: Transfer of custom data from outbound delivery order to warehouse task	EEW, BAdI
	1G2c: Showing custom data in the form view of the outbound delivery order item	BAdI
	1G2d: Determination and transfer of handling unit type from packaging specification to pick warehouse task	BAdI
	1G2e: Determination of the operative UOM by packaging specification of goods receipt	BAdI
	1G2f: Prohibit goods issue for incomplete packing	BAdI
1V7	1V7a: Take over transportation unit after unloading	PPF, BAdI
	1V7b: Print picking labels on a mobile printer	Condition technique, PPF
1FW	1FWa: Enhancing the goods movement interface by additional data	BAdI

Table 4.2 Overview of Enhancements (Cont.)

4.1.3 Installation of SAP Best Practices for Embedded EWM

For setting up the SAP Best Practices scope items in an on-premise EWM system, you will need to use the *solution builder* (Transaction /N/SMB/BBI). We recommend following the *Administration Guide to Implementation of SAP S/4HANA 2022 with SAP Best Practices*, which you can find in the help portal for SAP S/4HANA at *https://help.sap.com/*. The guide will take you through prerequisite settings, implementation, and upgrade procedures for loading, scoping, and activation of SAP Best Practices in your on-premise system. There is also a dedicated section on SAP S/4HANA–based scope items that you will need to activate on top of the EWM scope items so as to enable end-to-end processes.

4.1.4 Activating an SAP Cloud Appliance Library Instance

The fastest and easiest way to run SAP Best Practices processes and reenact the provided development examples would be for you to activate an SAP Cloud Appliance Library instance. This will specifically be easy if you already have an account at one of the following available cloud service providers:

- Amazon Web Services
- Microsoft Azure
- Google Cloud Platform

If you do not have an account with one of these providers yet, you might consider getting a trial account for a limited time, which often comes with a free budget allowing you to run the appliance actively for a decent number of hours.

With your cloud provider's account available, it should be fairly easy to create a trial instance of the latest SAP S/4HANA version. Go to *http://cal.sap.com* and explore the available fully activated appliance templates for SAP S/4HANA. To create an SAP Cloud Appliance Library instance, you will need a valid SAP ID before selecting your cloud provider, and then will need to specify your account credentials for the provider accordingly. Thereafter, you should be ready for the instance activation. If you run into activation issues, check the provided error log. You might likely need to request a parameter change at your cloud provider to fit the requirements of the SAP recommended virtual machine (VM) sizes. Once this is settled, it will only take a few hours before you have an SAP Cloud Appliance Library instance available for activation and connection. Make sure to suspend the instance when not actively using it to save some budget. Check out the SAP Cloud Appliance Library support page for further information and for how-to videos about instance activation.

Let's now continue with the process enhancements per SAP Best Practice scope item.

4.2 Basic Warehouse Inbound Processing from Supplier: 1FS

In this section, we will first introduce the complex goods receipt process of the SAP Best Practices, in which the functions of the transport control and the RF-based processing of handling units in the processing of incoming goods are in the foreground. Second, we will explore the following variants through custom developments and enhancements:

- **1FSa: Automatic handling unit creation without packaging specifications**
  You will get familiar with the BAdI for automatic handling unit creation.
- **1FSb: Simplify the screen flow for RF putaway**
  We will adjust the RF framework Customizing to skip the entry screen of the RF putaway transaction.

With most enhancements that we introduce in this section, you will find the `BREAK-POINT ID` ABAP statement in the coding. For each enhancement, we use a corresponding checkpoint group. You can create and activate the checkpoint groups using Transaction SAAB (Checkpoints that Can Be Activated).

4.2.1 Process Description of Scope Item 1FS

In Table 4.3, you will find in an overview of the steps for scope item 1FS (Basic Warehouse Inbound Processing from Supplier). We list the physical activities and the system activities in separate columns. The process steps in SAP S/4HANA modules are skipped, and we start with the description of the EWM steps. A purchase order was created in materials management in SAP S/4HANA as preparation.

Steps 3, 5, and 8 are completed by a goods receipt office clerk with system access via a desktop or a tablet PC using either SAP GUI or SAP Fiori–based apps. The warehouse operators who are physically moving the pallets from the truck (step 2), packing at the packing station (step 4) and moving to the final bin (step 7) work with mobile RF guns so that they post the movements in the system as they occur using RF based apps.

During the putaway operation, there are a couple of variants to be found in the scope item. These variants are based on different products requiring different packaging, storage concepts, and final destination storage types. They include piece- or carton-based small part rack storage in a mezzanine, pallet-based narrow aisle high rack storage (including handover locations for the narrow aisle truck) for medium size parts, and ground floor–based bulk storage (mainly full pallets stacked in front and on top of each other) for large parts, allowing for either full pallet or partial pallet putaway. Another process variant includes clarification and repacking of small parts that arrived in the wrong packaging.

Step	Physical Activity	System Activity
1: A truck arrives at the checkpoint and drives to the door.	A truck arrives. The warehouse clerk receives the delivery paper and communicates the door. The truck drives to the door.	
2: Unload the truck and check the goods.	A warehouse worker unloads the truck. A warehouse clerk checks the goods against the delivery note.	
3: Create EWM inbound delivery via the Create Inbound Delivery app (SAP Fiori app F1705).		The warehouse clerk creates the inbound delivery based on the delivery paperwork.
4: Pack the goods, creating handling units and posting goods receipt (EWM).		A warehouse worker creates and labels the handling units, posts the good receipt per handling unit.

Table 4.3 Steps in Inbound Scope Item 1FS

Step	Physical Activity	System Activity
5: Create warehouse orders (EWM).		The warehouse clerk creates putaway warehouse tasks and warehouse orders per the good receipts posted for the handling unit.
6: The truck leaves.	The truck leaves.	
7.1: Putaway the handling units to the mezzanine (EWM).		A warehouse worker logs on as a resource. The warehouse worker scans the handling units. The system determines the open warehouse orders for the handling units.
	The warehouse worker moves the handling units to the final bin in the mezzanine.	The warehouse worker confirms the warehouse orders.
7.2: Putaway the handling units to narrow aisle high bay racks (EWM).		A warehouse worker logs on as a resource. The warehouse worker scans the handling units. The system determines the open warehouse orders for the handling units.
	The warehouse worker moves the handling units to the handover zone for the narrow aisle high bay.	The warehouse worker confirms the warehouse orders. The system activates the next warehouse task for final putaway in the background.
		A narrow aisle forklift driver scans the handling units. The system determines the open warehouse orders for the handling units.
		The forklift driver confirms the warehouse orders.

Table 4.3 Steps in Inbound Scope Item 1FS (Cont.)

Step	Physical Activity	System Activity
7.3: Putaway the handling units to bulk ground floor area (EWM).		
7.4: Putaway the handling units to the clarification zone and repacking.		
	The packer identifies the goods and creates putaway handling units.	
		The packer closes the putaway handling units. The system prints handling unit labels and creates warehouse orders.
	The packer labels the putaway handling units.	
		A warehouse worker logs on as a resource. The warehouse worker scans a putaway handling unit. The system determines the open warehouse order for the putaway handling unit.
The warehouse worker moves the putaway handling unit to the final bin.		
	The warehouse worker confirms the warehouse order.	
8: Check the inbound delivery (EWM).		

Table 4.3 Steps in Inbound Scope Item 1FS (Cont.)

The process description of scope item 1FS can also be found in the flow chart and test script of the scope item.

4.2.2 Enhancement 1FSa: Automatic Handling Unit Creation without Packaging Specifications

In custom development 1FSa, we will enhance the inbound steps with a function called Automatic Handling Unit Creation without Packaging Specifications. Step 3 of standard scope item 1FS will change with this custom development, making step 4 obsolete (see Table 4.4). All other process steps stay the same as described in Table 4.3.

Consider Unified Package Builder Functionality

EWM offers a rather new feature that allows you to activate data sources other than the packaging specification for packaging requirements or proposals. These include SAP S/4HANA–based packing instructions and SAP Supply Chain Management–based package builder rules (e.g., alternative units of measure from the global material data).

Step	Physical Activity	System Activity
3: Create EWM inbound delivery (EWM) via GUI Transaction /SCWM/PRDI and create handling units.	N/A	The warehouse clerk checks the quantities, creates handling units in the system, and prints the new handling unit labels. She then posts the goods receipt.

Table 4.4 Process Steps with Deviation for Enhancement 1FSa

The main difference between scope item 1FS and enhancement 1FSa is that the pallets arriving from the office clerk create the handling unit labels instead of the warehouse operator, while the latter will apply them on the shop floor. The handling unit label will stay with the pallet as long as possible and can, for example, be used for an internal move, replenishment, or stock removal.

In standard EWM, you could let the system automatically create handling units in the inbound process based on packaging specifications or unified package builder profiles. So for each product (or reference product), you have to maintain a packaging specification that defines the pallet quantity, the handling unit type, and the packaging material for the pallet. If you do not have packaging specifications for each product, and the unified package builder might not be an option, the solution shown in this chapter might be a suitable alternative in your project.

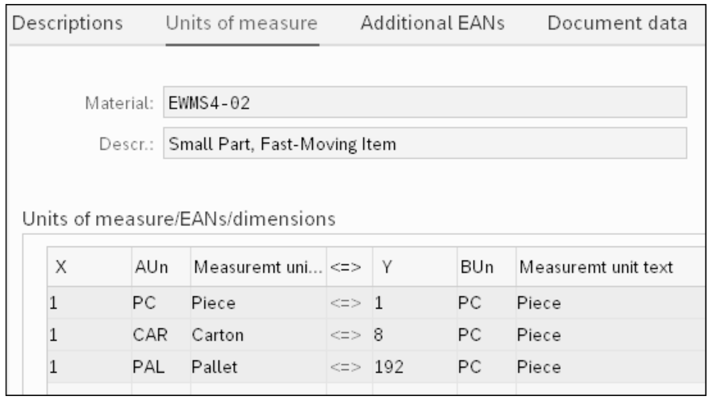
To realize enhancement 1FSa, the following steps are necessary:

1. Create a new master data table `ZHU_PMAT` to determine the packaging material for the handling units.
2. Implement the `HU_PROPOSAL` method of BAdI `/SCWM/EX_HU_BASICS_AUTOPACK`.
3. Switch on automatic packing for the inbound delivery.
4. Create a condition record to print handling unit labels. Deactivate the printing for warehouse order labels.

We describe the details for these four steps in the following sections. First, however, let's look at the pallet building process in 1FS and discuss the prerequisites for this enhancement. We'll also close the section by discussing how to test process 1FSa.

Pallet Building in Process 1FS

Process 1FS uses the *quantity classification* based on alternative UOM to determine the warehouse task quantity when putting away product from the clarification area. The alternative UOMs are defined in the product (see Figure 4.3), so, for example, 192 EA is one pallet, and 8 EA is one carton.



Descriptions | Units of measure | Additional EANs | Document data

Material: EWMS4-02

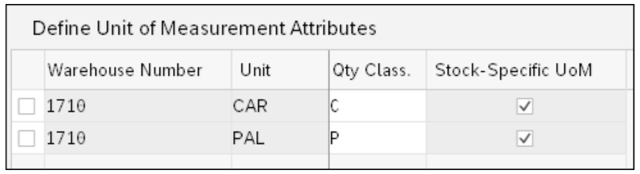
Descr.: Small Part, Fast-Moving Item

Units of measure/EANs/dimensions

X	AUn	Measuremt uni...	<=>	Y	BUn	Measuremt unit text
1	PC	Piece	<=>	1	PC	Piece
1	CAR	Carton	<=>	8	PC	Piece
1	PAL	Pallet	<=>	192	PC	Piece

Figure 4.3 Material Master with UOM: Additional Data

In the **EWM • Master Data • Product • Define Unit of Measure Attributes** IMG activity, the UOMs are assigned to quantity classifications (see Figure 4.4). So, for example, **Quantity Classification P** (Pallets) is assigned to **Unit PAL**. Based on this setting, the system will split the warehouse task quantities during warehouse task task creation in put-away and could also determine different storage types for putaway.



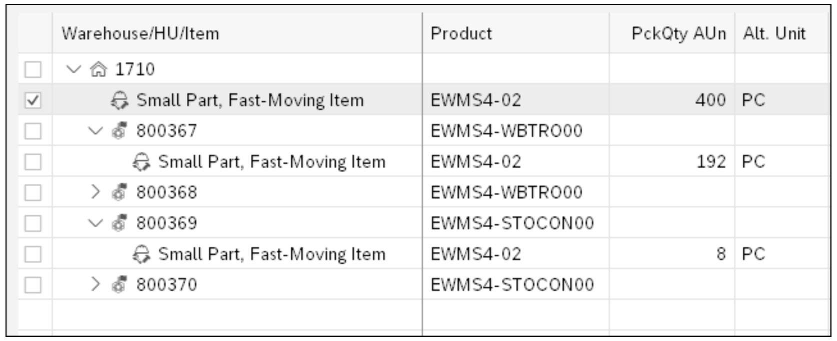
Define Unit of Measurement Attributes

	Warehouse Number	Unit	Qty Class.	Stock-Specific UoM
□	1710	CAR	C	☑
□	1710	PAL	P	☑

Figure 4.4 IMG Activity to Assign Quantity Classification to Units

So if, for example, an inbound delivery with a quantity of 400 EA is received in the warehouse, the system would create four product warehouse tasks (see Figure 4.5, where we simulate inbound delivery packing in Transaction /SCWM/PRDI):

- Two warehouse tasks with 192 EA = 1 PAL
- Two warehouse tasks with 8 EA = 1 CAR



	Warehouse/HU/Item	Product	PckQty AUn	Alt. Unit
□	⌄ 1710			
☑	Small Part, Fast-Moving Item	EWMS4-02	400	PC
□	⌄ 800367	EWMS4-WBTRO00		
□	Small Part, Fast-Moving Item	EWMS4-02	192	PC
□	› 800368	EWMS4-WBTRO00		
□	⌄ 800369	EWMS4-STOCON00		
□	Small Part, Fast-Moving Item	EWMS4-02	8	PC
□	› 800370	EWMS4-STOCON00		

Figure 4.5 Automatic Handling Unit Creation in Inbound Delivery Packing

Prerequisites for Enhancement 1FSa

Enhancement 1FSa is useful in a warehouse if these prerequisites are met:

- No packaging specifications by product exist.
- The unified package builder is not a valid option.
- The pallet and/or carton quantity is available as an alternative UOM by product.

Create New Master Data Table

To create a handling unit in the system, a packaging material is mandatory. The system takes over the tare weight, volume, and handling unit type from the packing material master and also the kind of numbering (e.g., number range, Serial Shipping Container Code [SSCC]) specified via the packaging material. Hence, we will use a new, simple master data table to determine the packaging material based on the quantity classification. We will also determine a handling unit type that could be used to influence the putaway strategy and to find the optimal bin type.

Create a new database in ABAP Dictionary (Transaction SE11). Enter the name, such as ZHU_PMAT, and a description. In **Delivery Class**, choose option **A (master and transactional data)**, and in the **Data Browser/Table View Maintenance** field, make sure you choose option **X—Display/Maintenance Allowed**. On the **Fields** tab, maintain the fields as they are listed in Table 4.5.

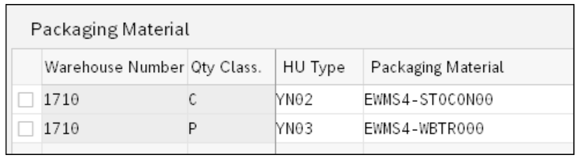
Field	Data Element	Key	Check Table/Search Help
CLIENT	MANDT	Yes	
LGNUM	/SCWM/LGNUM	Yes	/SCWM/SH_LGNUM
QUANCLA	/SCWM/DE_QUANCLA	Yes	/SCWM/TQUANCLA
HUTYP	/SCWM/DE_HUTYP	No	/SCWM/THUTYP
PACKMAT	/SCWM/DE_PMAT	No	/SCWM/SH_PMAT_ONLY

Table 4.5 Fields of Custom Table ZHU_PMAT

For the technical settings of the new table, choose “APPL0” for **Data Class** ,“0” for **Size Category**, and switch on the buffering (fully buffered).

After you have saved and activated the table, navigate to the table maintenance generator (Transaction SE55) and generate a maintenance view (e.g., with authorization group SCEA and function group ZHU_PMAT).

Last but not least, maintain a few entries in the new table using Transaction SM31 (Maintain Table Views; see Figure 4.6).



Packaging Material

	Warehouse Number	Qty Class.	HU Type	Packaging Material
□	1710	C	YN02	EWMS4-STOCON00
□	1710	P	YN03	EWMS4-WBTR000

Figure 4.6 Maintain Packaging Material and Handling Unit Type in Table ZHU_PMAT

Implement the HU_PROPOSAL Method

To implement method HU_PROPOSAL, you have to start the BAdI Builder (Transaction SE19) and create an implementation for enhancement spot /SCWM/ES_HU_BASICS. First enter the name of the enhancement implementation (e.g., ZEI_HU_BASICS) and then choose BAdI definition /SCWM/EX_HU_BASICS_AUTOPACK. As a name for the BAdI implementation, you can enter, for example, ZEX_HU_BASICS_AUTOPACK, and as a class name, choose ZCL_IM_HU_BASICS_AUTOPACK. Navigate to the HU_PROPOSAL method and enter the coding as shown in Listing 4.1. Define one static, private attribute, ST_TUOM_QCLA, of type /SCWM/TT_UOM_QCLA for the class.

Activate the coding and also the BAdI implementation:

- In coding paragraph “1, we first fetch the quantity classification table (/SCWM/TUOM_QCLA) from the IMG.
- We loop over all delivery items, and for each product we determine the list of alternative UOMs (see paragraph “2).
- By using standard function module /SCWM/QUANCLA_DET_UOM, we determine the quantity classification depending on the unpacked, open quantity (see “3).
- Based on the quantity classification, we determine in our new table ZHU_PMAT the packaging material and handling unit type (see “4).
- To determine the handling unit target quantity, we look up the first alternative unit that matches the required quantity classification. The numerator of this unit becomes the target quantity (see paragraph “5).
- In the last two paragraphs, “6 and “7, we finally create a handling unit and pack the delivery item into it.
- Note that we do not use the save method or the commit statement. As this is a BAdI implementation, we expect the calling environment to take care of saving and database commits.

```
METHOD /scwm/if_ex_hu_basics_autopack~hu_proposal.

    DATA: lt_mat_uom TYPE /scwm/tt_material_uom,
          lv_quancla TYPE /scwm/de_quancla.

    BREAK-POINT ID zewmdevbook_1fsa.
    DATA(lo_pack)  = CAST /scwm/cl_hu_packing( io_pack_ref ).
    DATA(lo_stock) = NEW /scwm/cl_ui_stock_fields( ).
    "1 Get quantity classification (prefetch)
    IF st_tuom_qcla IS INITIAL.
      SELECT * FROM /scwm/tuom_qcla
               INTO TABLE st_tuom_qcla
               WHERE lgnum = lo_pack->gv_lgnum.
      IF st_tuom_qcla IS INITIAL.
        RETURN.
      ENDIF.
    ENDIF.

    LOOP AT ct_pack ASSIGNING FIELD-SYMBOL(<pack>).
      CLEAR: lt_mat_uom.
      "2 Get product master for each delivery item
      TRY.
          CALL FUNCTION '/SCWM/MATERIAL_READ_SINGLE'
            EXPORTING
              iv_matid   = <pack>-matid
            IMPORTING
              et_mat_uom = lt_mat_uom.
        CATCH /scwm/cx_md.
          io_pack_ref->go_log->add_message( ).
          cv_severity = sy-msgty.
          CONTINUE.
      ENDTRY.
      WHILE <pack>-quan > 0.
        "3 Get quantity classification based on open quantity
        TRY.
            CALL FUNCTION '/SCWM/QUANCLA_DET_UOM'
              EXPORTING
                iv_lgnum   = lo_pack->gv_lgnum
                iv_matid   = <pack>-matid
                iv_batchid = <pack>-batchid
                iv_quan    = <pack>-quan
                iv_unit    = <pack>-unit
                it_mat_uom = lt_mat_uom
              IMPORTING
                ev_quancla = lv_quancla.
          CATCH /scwm/cx_core.
            io_pack_ref->go_log->add_message( ).
            cv_severity = sy-msgty.
            CONTINUE.
        ENDTRY.
        "4 Determine packmat and hu_typ for the quantity classification
        SELECT SINGLE * FROM zhu_pmat
          INTO @DATA(ls_zhu_pmat)
          WHERE lgnum = @lo_pack->gv_lgnum
          AND quancla = @lv_quancla.
        IF ls_zhu_pmat-packmat IS INITIAL.
          "Error: No Packaging Material maintained for Quan.Class. &1.
          MESSAGE e001(zewmdevbook_1fsa) WITH lv_quancla.
          io_pack_ref->go_log->add_message( ).
          EXIT.
        ENDIF.
        DATA(lv_packmatid) = lo_stock->get_matid_by_no(
                             iv_matnr = ls_zhu_pmat-packmat ).
        "5 Determine target quantity and UoM
        LOOP AT st_tuom_qcla INTO DATA(ls_quancla) WHERE quancla = lv_quancla.
          DATA(ls_mat_uom) = VALUE #( lt_mat_uom[ matid = <pack>-matid
                                                  meinh = ls_quancla-unit ] ).
          IF sy-subrc IS NOT INITIAL.
            EXIT.
          ENDIF.
        ENDLOOP.
        "6 Create new handling unit
        DATA(ls_hu_crea) = VALUE /scwm/s_huhdr_create_ext(
          hutyp = ls_zhu_pmat-hutyp ).
        DATA(ls_huhdr) = io_pack_ref->create_hu(
          EXPORTING
            iv_pmat      = lv_packmatid
            is_hu_create = ls_hu_crea ).
        IF sy-subrc <> 0.
          io_pack_ref->go_log->add_message( ).
          EXIT.
        ENDIF.
        "7 Pack item
        DATA(ls_quan) = CORRESPONDING /scwm/s_quan( <pack> ).
        IF <pack>-quan >= ls_mat_uom-umrez.
          ls_quan-quan = 1.
```

```
        ls_quan-unit = ls_mat_uom-meinh.
        <pack>-quan  = <pack>-quan - ls_mat_uom-umrez.
      ELSE.
        ls_quan-quan = <pack>-quan.
        <pack>-quan = 0.
      ENDIF.
      DATA(ls_mat) = CORRESPONDING /scwm/s_pack_stock( <pack> ).
      io_pack_ref->pack_stock(
        EXPORTING
          iv_dest_hu  = ls_huhdr-guid_hu
          is_material = ls_mat
          is_quantity = ls_quan ).
      IF sy-subrc <> 0.
        io_pack_ref->go_log->add_message( ).
        EXIT.
      ENDIF.
      CLEAR: ls_mat, ls_quan, ls_hu_crea, lv_quancla,
             ls_huhdr, ls_quancla, ls_mat_uom, ls_zhu_pmat.
    ENDWHILE.
  ENDLOOP.

ENDMETHOD.
```

Listing 4.1 Coding of HU_PROPOSAL Method

Switch on Automatic Packing

In IMG activity **EWM • Goods Receipt Process • Inbound Delivery • Manual Settings • Define Document Types for Inbound Delivery Process**, choose document type INB for document category PDI, change the following settings, and save:

- **Packaging Material Proposal Procedure: OIBD**
  This packaging specification procedure entry is required to switch on the automatic packing in general. We enter a procedure, although we will not use packaging specifications. Make sure you enter a procedure for which you do not use packaging specifications in your project.
- **No Automatic Packing: Yes**
  With this setting, we switch off the automatic packing during inbound delivery creation. As in our process variant, we first want the user to verify the quantity and then create the handling unit labels in the office. We want the user to start the automatic packing manually when he is done with checking.

Create a Condition Record

As we want the system to print the labels automatically when the user creates the handling units, check and create new condition records in SAP menu **EWM • Work Scheduling • Print • Settings • Create Condition Records for Printing (HUs)**. Enter application and maintenance group PHU and select existing condition records for condition type 0HU1. Check the results list. If one does not exist yet, create one entry for each packaging material you use as in Figure 4.6 with the following values:

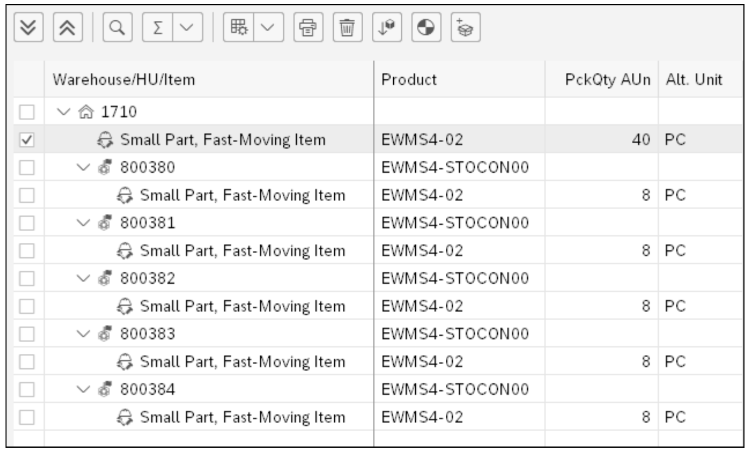
- **Condition Type**: 0HU1
- **Warehouse**: 1710
- **HU Step: I—Create**
- **Packaging Material**: for example, EWMS4-STOCON00 (EWM Default Storage Container/Box)
- **HU Type**: for example, YN02 (EWM Carton/Box)
- **Form**: /SCWM/HU_CONTENT
- **Printer**: for example, LP01
- **Spool**: 01
- **Action**: HU_LABEL_GENERAL_AND_RF

Testing of Process 1FSa

You can now test the handling unit creation and handling unit printing (see Table 4.6). In Figure 4.7, you can see the result of test step 4.1. For an inbound delivery quantity of 40 EA, the system created five handling units. All were created as boxes of 8 EA.

Step	Step Description	Input Data and Expected Results
4.2	Create EWM Inbound Delivery	Use Transaction /SCWM/GR (Goods Receipt). Search for the inbound delivery using the PO number or using the ASN number. Check the delivery data against the revised delivery note and adapt the quantities if necessary. Use the **Pack** button to navigate to the **Work Center Packing for Inbound Delivery** screen for inbound deliveries, select the delivery items, and use the **Pack Automatically** button (see Figure 4.7). Navigate back and use the **Post Goods Receipt** button. *Expected results:* The system generates handling units. The system prints handling unit labels.

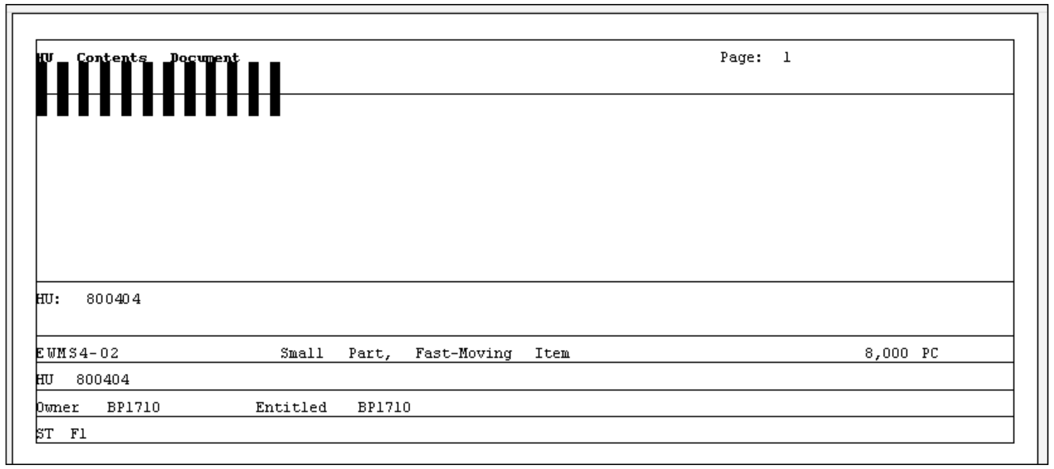
Table 4.6 Test Steps for Enhancement 1FSa



	Warehouse/HU/Item	Product	PckQty AUn	Alt. Unit
□	1710			
☑	Small Part, Fast-Moving Item	EWMS4-02	40	PC
□	800380	EWMS4-STOCON00		
□	Small Part, Fast-Moving Item	EWMS4-02	8	PC
□	800381	EWMS4-STOCON00		
□	Small Part, Fast-Moving Item	EWMS4-02	8	PC
□	800382	EWMS4-STOCON00		
□	Small Part, Fast-Moving Item	EWMS4-02	8	PC
□	800383	EWMS4-STOCON00		
□	Small Part, Fast-Moving Item	EWMS4-02	8	PC
□	800384	EWMS4-STOCON00		
□	Small Part, Fast-Moving Item	EWMS4-02	8	PC

Figure 4.7 Automatic Packing in Inbound Deliveries

After the user saves the results of automatic packing, the system will automatically print the handling unit labels, shown in Figure 4.8. The standard form shows a barcode, but it is very likely that you will have to adjust this form to your printer size and your barcode type.



```
HU  Contents  Document                                         Page:  1

HU:   800404

EWMS4-02                 Small   Part,   Fast-Moving   Item              8,000  PC
HU   800404
Owner   BP1710          Entitled   BP1710
ST  F1
```

Figure 4.8 Printout of Handling Unit Content Label

For the putaway step 4 of the test case, you have now two options: either execute the step as described in the standard process, or continue with the information in the next section to use enhancement 1FSb.

4.2.3 Enhancement 1FSb: Simplify the Screen Flow for Radio Frequency Putaway

In enhancement 1FSb, we will change step 7, Put Away the Goods, of process 1FS. The changes are described in Table 4.7. In the standard process, the putaway is paper-driven and without RF support. The warehouse operators move the pallets to the final bins and hand over the papers to the office clerk. Several times per day, the office clerk confirms the warehouse orders in the system based on the returned papers. This way, the stock increases and becomes available for sale. In process variant 1FSb, the warehouse operators will use RF devices and confirm each move of stock in the system immediately.

In this section, we give an instruction on how to simplify the Putaway by Warehouse Order RF transaction (logical RF Transaction PTWOSI). With an adjustment in the RF framework, we reduce the number of UIs and eliminate one RF screen. So for each pallet the user scans, there is a warehouse order barcode from the paper and the destination bin barcode.

Step	Physical Activity	System Activity
4.6.5: Move Products from Clarification Zone to Mezzanine	The warehouse worker takes the warehouse order printouts and sticks one printout on the goods. The warehouse worker moves the goods from the clarification zone to the final storage in the mezzanine.	The warehouse operator confirms each putaway via RF.

Table 4.7 Process Steps with Deviation for Enhancement 1FSb

To realize the variant 1FSb, Simplify the Screen Flow for RF Putaway, you have to do the following steps:

1. Create RF presentation profile 1710 and copy the standard menu.
2. Change the RF step flow for logical Transaction PTWOSI and presentation profile 1710 such that the first screen is skipped.

The details for these two steps are described in the following sections. As usual, we will begin by discussing the prerequisites and end by discussing testing of this enhancement.

Prerequisites for Enhancement 1FSb

Enhancement 1FSb has the following prerequisites:

- Wi-Fi and mobile devices are supported in the warehouse
- Warehouse order barcode is on the printout of the warehouse order

Create the Presentation Profile

Now create a new presentation profile (see also Chapter 3, Section 3.3) to make warehouse-specific changes in the RF framework Customizing in the next step:

- Create a new presentation and personalization profile. This is done in the **EWM • Mobile Data Entry • Define Steps in Logical Transactions** IMG activity. In the **Define Presentation Profiles** folder, create a new entry, “1710”, by copying the existing **** entry.
- In the **RF Menu Manager** IMG activity, keep the default values on the entry screen and use the **Copy Menu** function. In the popup, enter presentation profile 1710, then continue and save.
- Assign the new presentation profile 1710 to warehouse number 1710. This is done in IMG activity **Assign Presentation Profile to Warehouse**.

Change the RF Step Flow

We now change one entry in the RF framework Customizing so that the system will skip the putaway source screen.

Go to IMG activity **EWM • Mobile Data Entry • Define Steps in Logical Transactions**. In the **Define Logical Transactions** folder, select logical Transaction PTWO** and navigate to subfolder **Define Logical Transaction Step Flow**. Select and copy the entry with the following keys:

- **Presentation profile**: ****
- **Logical transaction**: PTWO**
- **Step**: PTHUSC
- **Function code**: PBO1

Before you save, change the following fields:

- **Presentation profile**: 1710
- **Next step**: PTHUDS
- **Processing mode**: 1—Background
- **Function code background**: PBO1

With the changed settings, the system will skip the screen for step PTHUSC and immediately continue with step PTHUDS.

Copy Instead of Change

We recommend not changing settings for presentation profile ****. For each project/template project, create a separate presentation profile as shown in step 3 and copy the standard settings from profile **** to your own profile before changing them. You can then use the unchanged settings for profile **** as reference. If there is a problem in RF, you can change back to **** before you ask SAP Support for help.

Testing of Enhancement 1FSb

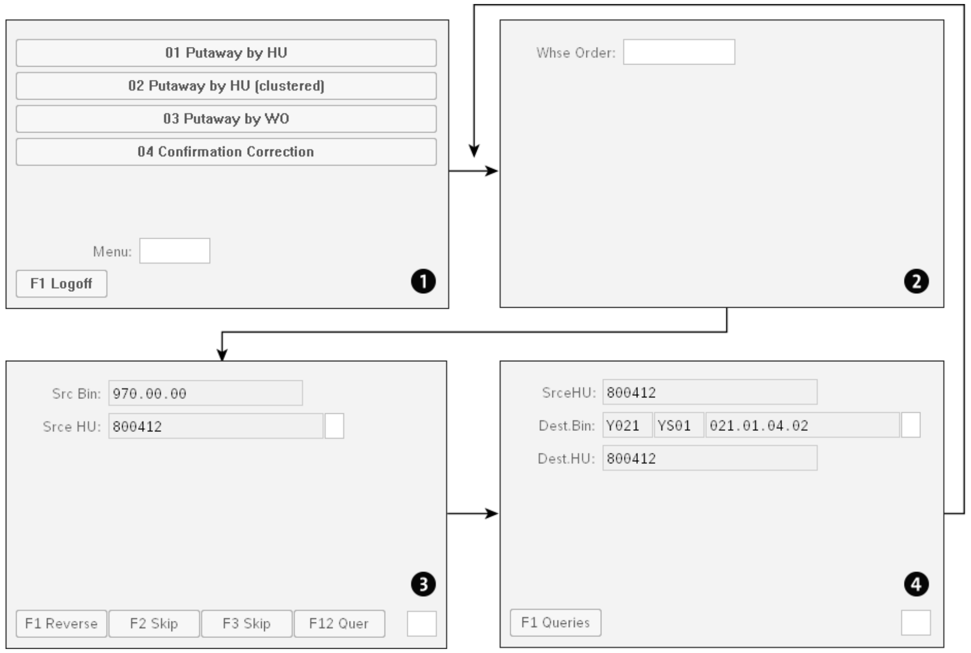
Start your test with test steps 1 to 5 from process 1FS. Then replace step 6 with the test described in Table 4.8.

Step	Step Description	Input Data and Expected Results
4.6.5	Move Products from Clarification Zone to Mezzanine—Choose Menu	■ Use Transaction /SCWM/RFUI (Log onto RF Environment) and log onto warehouse 1710 with resource YREC-1 and presentation device YE00. ■ Navigate to menu item **03 Inbound Process • 03 Putaway • 03 Putaway by WO** (fast access via 333). ■ Scan the warehouse order barcode. ■ Scan the destination bin. *Expected result:* The warehouse order is confirmed, and the stock is available for sale.

Table 4.8 Test Steps for Enhancement 1FSb

In Figure 4.9, we want to show you how the RF transaction works for standard presentation profile ****, as follows:

In the RF menu, the user chooses **03 Putaway by WO**.



Src Bin: 970.00.00
Srce HU: 800412

SrceHU: 800412
Dest.Bin: Y021 YS01 021.01.04.02
Dest.HU: 800412

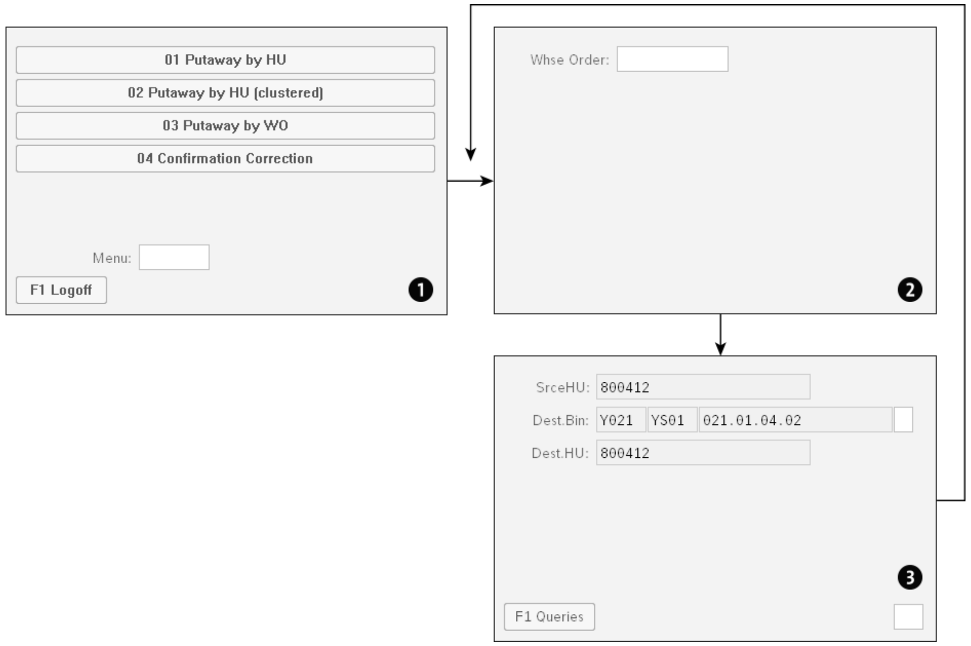
Figure 4.9 Putaway by Warehouse Order with Presentation Profile ****

On the entry screen, the user scans the barcode for the warehouse order (RF step PTWOSL).

On the source screen (RF step PTHUSC), the user confirms the pickup of the stock with several presses of Enter.

On the destination screen (RF step PTHUDS), the user scans the destination bin after he puts the stock there.

When using the new presentation profile 1710, you'll see that step PTHUSC is skipped by the system. The flow of screens is shown in Figure 4.10. So for each putaway transaction, the user will save one screen and one Enter press. There is also a difference in the number of warehouse tasks: the system will not use two tasks (one for posting stock from the source bin to the resource and another for posting stock from the resource to the destination bin). It will post only one task—that is, the moment the user confirms the destination bin.



SrceHU: 800412
Dest.Bin: Y021 YS01 021.01.04.02
Dest.HU: 800412

Figure 4.10 Putaway by Warehouse Order with Presentation Profile 1710

To have only one instead of two warehouse tasks at the end of the transaction is also a benefit in case the user changes his mind and steps back with F7 on the destination screen (step PTHUDS). With the standard settings, the stock would stay on the resource and hence no other resource would be able to perform this warehouse order. If you work with high-level forklifts and low-level forklifts, it can happen quite often that only after seeing the destination screen can the operator decide if he is capable of doing the putaway at that destination bin.

In this section, you learned how to skip screens in RF and how to process steps in background mode without coding adjustments. There are more RF transactions in the standard where you might have the need to skip a screen if the screen does not provide valuable information to your users.

4.3 Warehouse Inbound Processing from Supplier with Batch Management: 1V5

In this section, we will introduce inbound scope item 1V5 (Warehouse Inbound Processing from Supplier with Batch Management). We will ensure that we create two variants of this process by adding custom developments and Customizing settings:

- **1V5a: Permit activation of the transportation unit only after entering the license plate and pager**
  We will show how you can realize additional checks during the activation of transportation units.
- **1V5b: Putaway depending on quarantine period**
  With this custom development, we show you how to affect the putaway strategy using the `/SCWM/EX_CORE_PTS_TYPSQ` BAdI.
- **1V5c: Enhancing the warehouse monitor**
  We will show you how to place your own node in the warehouse management monitor for advanced data selection.
- **1V5d: Delay inbound delivery with missing batch**
  You will learn to recognize an opportunity by using the `/SCWM/EX_ERP_INT_CONF` BAdI to influence the creation of the inbound delivery notification during batch capturing within inbound processing.

Both process variants start in the same way: an inbound delivery without packing information is created in EWM. The vendor is not labeling the pallets in such a way that the warehouse can reuse the labels, and furthermore, no label information is passed electronically to EWM. All of the described enhancements are independent of each other.

4.3.1 Process Description of Scope Item 1V5

In Table 4.9, you will find in an overview of the steps for process 1V5, Warehouse Inbound Processing from Supplier with Batch Management. We list the physical activities and the system activities in separate columns. The process steps in SAP S/4HANA are skipped, and we start with the description of the EWM steps. A purchase order and inbound delivery (optional) were created in the SAP S/4HANA system as preparation steps.

Steps 2, 4, and 6 are completed by a goods receipt office clerk with system access. The warehouse operators who are physically moving the pallets from the truck to the final bin are without system access. They use the printouts of the office clerk to find out which pallet needs to be moved to which bin. When they are done with the physical work, they hand the printouts back to the clerk, who then confirms the work in the system.

We have extended the scope item with incoming truck handling using the shipping and receiving module of EWM. Be aware that this module falls under the advanced features of embedded EWM and might require additional licensing. The extended steps will work out of the box in the respective SAP Best Practices processes however, so no additional configuration will be required.

Step	Physical Activity	System Activity
1: A truck arrives at the checkpoint and drives to the door.	A truck arrives. A checkpoint clerk communicates the door to the truck driver. The truck drives to the door.	*Scope item extension:* A checkpoint clerk creates a transportation unit, assigns the transportation unit to a door, and confirms the arrival of the truck at the door.
		Scope item extension: A goods receipt office clerk assigns the inbound delivery items to the transportation unit.
2: Check the delivery note and find or create an inbound delivery.		A good receipt office clerk finds or creates an inbound delivery.
3: Unload the truck and check the goods.	A warehouse worker unloads the truck. A warehouse clerk checks the goods against the delivery note.	
4: Post the good receipt and create put-away warehouse orders.		The goods receipt office clerk checks the quantities and posts the goods receipt. The system creates and prints warehouse orders.
5: The truck leaves.	The truck leaves.	*Scope item extension:* The goods receipt office clerk confirms the departure of the truck.

Table 4.9 Process Description of Inbound Process 1V5

Step	Physical Activity	System Activity
6: Putaway the goods.	The warehouse worker takes the warehouse order printouts and sticks one printout to the goods. The warehouse worker moves the goods to the final bin or to the clarification zone (exceptional case).	The warehouse clerk confirms the warehouse order.

Table 4.9 Process Description of Inbound Process 1V5 (Cont.)

The process description of scope item 1V5 can also be found in the flow chart and test script of the scope item.

4.3.2 Enhancement 1V5a: Permit Activation of the Transportation Unit Only after Entering the License Plate and Pager

In variant 1V5a, we will show you how you can realize additional checks during the activation of transportation units. With this custom development, you can ensure that during the creation of the transportation units or, at the latest, during the posting of the **Arrival at Checkpoint** transportation unit action, a license plate of the truck and a pager number have to be entered.

Quite often it is the case in larger warehouse facilities that a truck driver has to register first at a central check-in point of the plant in order to pick up or deliver goods. Now the truck is being maintained in the EWM system via Transaction /SCWM/TU (Maintain Transportation Unit). For the sake of traceability and facility protection, the license plate number is also retained. At this time, however, the corresponding door does not necessarily need to be assigned. This may be because another department does the door assignment (e.g., the goods receipt office). Therefore, the project requirement might be that if the truck driver is provided with the door, he should go to it at any time after check-in. The following custom development ensures that the employee at the plant cannot forget to retain the pager number and license plate of the truck.

To perform enhancement 1V5a, the following steps are necessary:

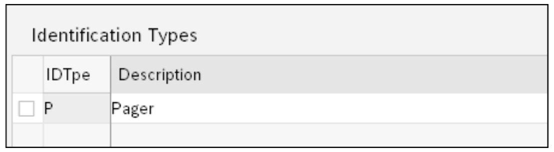
1. Define an identification type in the IMG to collect the pager number.
2. Create three messages in the `ZEWMDEVBOOK_1V5A` message class.
3. Implement the `BEFORE_SAVE` method of BAdI `/SCWM/EX_SR_SAVE`.

Next, we explain in detail how to proceed with each of these steps before finishing with information on testing this enhancement.

Define an Identification Type

To create the identification type for collecting the pager number, follow these steps:

1. Navigate to IMG activity **EWM • Cross-Process Settings • Shipping and Receiving • Define Identification Type for Transportation Unit/Vehicle**.
2. Under identification types, in the **IDTpe** column, retain the abbreviation P and assign a description ("Pager"; see Figure 4.11).
3. Save your entries.



Identification Types

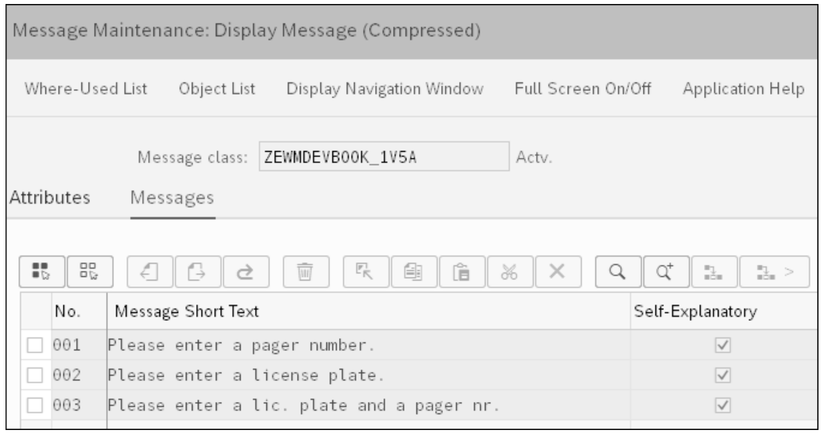
	IDTpe	Description
□	P	Pager

Figure 4.11 Identification Type for Pager Number

Create Three Messages

To create the three new messages, follow these steps:

1. Start Transaction SE91 (Message Maintenance).
2. Create three new messages in the `ZEWMDEVBOOK_1V5A` message class, as shown in Figure 4.12.
3. Save your entries.



Message Maintenance: Display Message (Compressed)

Message class: ZEWMDEVBOOK_1V5A Actv.

	No.	Message Short Text	Self-Explanatory
□	001	Please enter a pager number.	☑
□	002	Please enter a license plate.	☑
□	003	Please enter a lic. plate and a pager nr.	☑

Figure 4.12 Messages for Enhancement 1V5a

Implement the BEFORE_SAVE Method

To implement the `BEFORE_SAVE` method of the `/SCWM/EX_SR_SAVE` BAdI, proceed as follows:

1. Start the BAdI builder (Transaction SE19) and create an enhancement implementation (e.g., `ZEWM_EI_SR_SAVE`) for enhancement spot `/SCWM/ES_SR_READ_SAVE`.

2. Then create a BAdI implementation (e.g., ZEX_SR_SAVE) for BAdI definition /SCWM/EX_SR_SAVE and an implementing class (e.g., ZCL_IM_SR_SAVE).
3. Now you can program the /SCWM/IF_EX_SR_SAVE~BEFORE_SAVE method with the sample code from Listing 4.2:
   - In our example, we first check whether the transportation unit is currently being changed. This we determine based on the *changing indicator* of the business object (wmesr_objstate_new or wmesr_objstate_chg).
   - Next, we must determine which business process for the transportation unit is currently running. Our check should only be carried out in the case of activating the transportation unit. The BAdI, and thus our sample implementation, however, is executed for each action of the transportation unit (on creation, change, save, delete, cancel, etc.).
   - After verifying that all requirements are met, we can now check if the user actually entered a license plate and a pager number. At this point, we would like to accurately determine which entry may be missing in order to be able to issue a specific error message. If a value has not been entered, we write the corresponding message in the log object (coding section “6) and trigger the /SCWM/CX_SR_ERROR exception. With the exception, the save operation is canceled, and a popup will display the respective error context for the user.

```
METHOD /scwm/if_ex_sr_save~before_save.

    BREAK-POINT ID zewmdevbook_1v5a.

    LOOP AT it_bo_tu ASSIGNING FIELD-SYMBOL(<fs_bo_tu>).
      IF <fs_bo_tu>-bo_ref IS BOUND.
        TRY.
            <fs_bo_tu>-bo_ref->get_data(
               IMPORTING
                 ev_objstate = DATA(lv_state)
                 et_ident    = DATA(lt_ident) ).
          CATCH /scwm/cx_sr_error.
            CONTINUE.
        ENDTRY.
        "1. Check changing indicator of object
        CHECK lv_state = wmesr_objstate_new
        OR    lv_state = wmesr_objstate_chg.
        "2. Determine context
        CHECK <fs_bo_tu>-bo_ref->get_sr_act_state( ) =
        wmesr_act_state_active.
        CHECK <fs_bo_tu>-bo_ref->get_status_change_by_id(
        wmesr_status_check_in ) = abap_true.
        "3. Check for pager
```

```
        TRY.
            DATA(ls_ident) = lt_ident[ idart = 'P' ].
          CATCH cx_sy_itab_line_not_found.
            DATA(lv_check) = 1.
        ENDTRY.
        IF ls_ident-ident IS INITIAL.
          lv_check = 1. "no pager
        ENDIF.
        "4. Check for lic_plate
        <fs_bo_tu>-bo_ref->get_data(
          IMPORTING es_bo_tu_data = DATA(ls_bo_tu_data) ).
        IF ( ls_bo_tu_data-lic_plate         = ''
        OR   ls_bo_tu_data-lic_plate_country = '' )
        AND  lv_check = 1.
          lv_check = 3. "no pager and no lic_plate
        ELSEIF ( ls_bo_tu_data-lic_plate         = ''
        OR       ls_bo_tu_data-lic_plate_country = '' ).
          lv_check = 2. "no lic_plate
        ENDIF.
        "5. Raise message
        CASE lv_check.
          WHEN 1. "no pager
            MESSAGE e001(zewmdevbook_1v5a) INTO DATA(lv_msg).
          WHEN 2. "no lic_plate
            MESSAGE e002(zewmdevbook_1v5a) INTO lv_msg.
          WHEN 3. "no pager and lic_plate
            MESSAGE e003(zewmdevbook_1v5a) INTO lv_msg.
        ENDCASE.
        "6. add message to current log and raise exception
        IF NOT lv_check IS INITIAL.
          /scwm/cl_sr_bom=>so_log->add_message( ).
          RAISE EXCEPTION TYPE /scwm/cx_sr_error.
        ENDIF.
      ENDIF.
      CLEAR: lv_msg, lv_check.
    ENDLOOP.

  ENDMETHOD.
```

Listing 4.2 Sample Code for BAdI /SCWM/EX_SR_SAVE

MESSAGE Statement

Make sure that you *never* use the MESSAGE statement without the INTO addition in your own BAdI implementations (see coding section “5 in Listing 4.2).

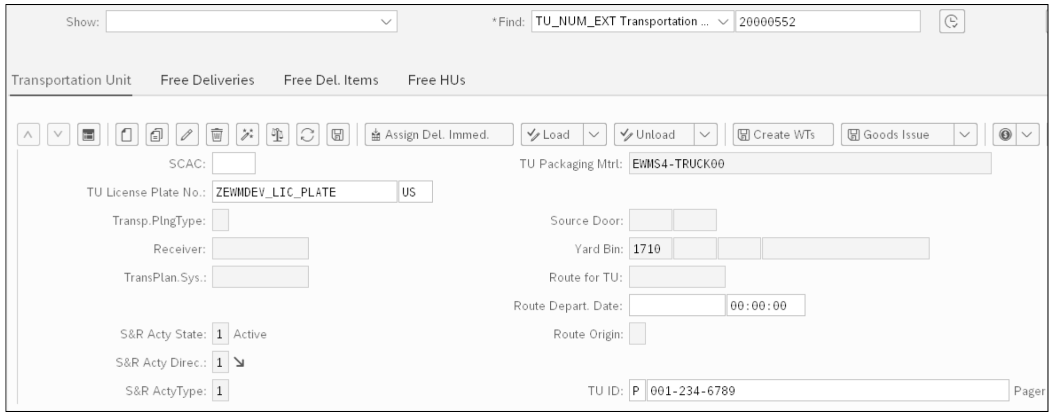
Testing Enhancement 1V5a

To test enhancement 1V5a, add the steps in Table 4.10 to the preparation of the test case, using the test instructions provided in the table.

Step	Step Description	Input Data and Expected Results
1	A truck arrives at the checkpoint and drives to the door (EWM).	
1.1	Create a transportation unit and assign it to a free door.	Start Transaction /SCWM/TU (Maintain Transportation Unit) and choose **Create**. Enter the data as stated in the test case description. On the **Assigned Doors** tab, choose **Add Door Assignment** and enter warehouse number 1710 and an empty door. Save your entries. *Expected result:* The transportation unit has been created and is assigned to a door.
1.2: Variant A	Confirm the arrival of the truck at the door.	Now select the transportation unit in Transaction /SCWM/TU and choose menu path **Action • Door • Arrival at Door**. Then save your action. *Expected result:* You receive an error message stating that the action was canceled because you did not enter a pager number and a license plate number.
1.2: Variant B	Confirm the arrival of the truck at the door.	Now select the transportation unit in Transaction /SCWM/TU. Switch to form view and enter the following data: ■ **License Plate Number** ■ **Country of Registration Number** ■ **Identification Type**: P ■ **Pager Number** (alternative transportation unit identification) Then choose the menu path **Action • Door • Arrival at Door** and save your action. *Expected result:* The transportation unit has both the status **Arrival at Checkpoint** and **Docked at Door**.

Table 4.10 Additional Test Steps for Enhancement 1V5a

Figure 4.13 shows an example of how you can enter the license plate and the pager number in Transaction /SCWM/TU.



*Find: TU_NUM_EXT Transportation ... 20000552

TU Packaging Mtrl: EWMS4-TRUCK00

TU License Plate No.: ZEWMDEV_LIC_PLATE US

Yard Bin: 1710

Route Depart. Date: 00:00:00

S&R Acty State: 1 Active

S&R Acty Direc.: 1 ↘

S&R ActyType: 1

TU ID: P 001-234-6789

Figure 4.13 Sample for 1V5a

4.3.3 Enhancement 1V5b: Putaway Depending on Quarantine Period

With this custom development, we want to influence the putaway strategy. It often happens in projects that products must be stored separately. Normally such product-specific settings are controlled via master data attributes of the warehouse product (e.g., putaway control indicator, storage section indicator). However, if a product is to be treated more flexibly—for example, due to technical production requirements—it might make sense to use a case distinction. In our example, the product is stored under normal conditions in the storage type that was determined during the putaway strategy based on the putaway control indicator. In the event that a quarantine period is set for the product, it should be different from the normal case and moved into a different storage type for maturation. The enhancement contains this as a rather simple extension to the standard provided putaway strategy and highlights the ease of integrating custom messages into the warehouse task creation log, which allows the business user to understand why certain destination bins might have been (sometimes unexpectedly) determined in a specific situation.

Enhancement 1V5b, Putaway Depending on Quarantine Period, requires the following steps:

1. Define new storage type search sequences in the IMG and assign corresponding storage types.
2. Create the `ZEWMDEVBOOK_1V5B` message class and add a new message.
3. Implement the `STORAGE_TYPE_SEQ` method of the `/SCWM/EX_CORE_PTS_TYPSQ` BAdI.
4. Create the `C_PTS_TYPSQ` constant in the implementing class.

Next, we will explain in detail how to proceed with each of these steps. As usual, we will begin by discussing the prerequisites and end by discussing testing of this enhancement.

Prerequisites for Enhancement 1V5b

The prerequisite for the successful implementation of enhancement 1V5b is that you have maintained a quarantine period in the corresponding **Quarantine Period** field on the **WM Execution** tab of the material master.

> **Field Quarantine Period**
>
> The **Quarantine Period** field is not used within EWM standard (SAP S/4HANA 2022). You can maintain the field in the material master (**WM Execution** tab). Note for decentralized EWM that the field is part of the MATMAS IDoc and will hence be distributed during the master data transfer from SAP S/4HANA to EWM.

Define and Assign Storage Type and Search Sequences

To create the new storage type, search sequences, and assign a storage type, follow these steps:

1. In the **EWM • Goods Receipt Process • Strategies • Storage Type Search • Define Storage Type Search Sequence for Putaway** IMG activity, create the entry in Table 4.11 by copying the existing storage type search sequences (e.g., YE02) and prefix them with a *Q* instead of a *Y*. Feel free to add more storage type search sequences to your liking in a similar fashion.

Warehouse Number	Storage Type Search Sequence	Description
1710	QE02	Bin Determination in Mezzanine for Q

Table 4.11 Storage Type Search Sequence for Enhancement 1V5b

2. Now assign the appropriate storage types (e.g., storage type Y970, Clarification Zone) to the new search sequences in the **EWM • Goods Receipt Process • Strategies • Storage Type Search • Assign Storage Types to Storage Type Search Sequence** IMG activity. Feel free to create new storage types if you prefer; for our demo and test purposes, using the clarification zone seems sufficient.

Create the Message Class

To create the message, start Transaction SE91 (Message Maintenance) and type in the ZEWMDEVBOOK_1V5B message class. Create the new message 001. As a short text message, enter "Storage type search sequence & 1 changed to & 2".

Implement the STORAGE_TYPE_SEQ Method

To make the BAdI implementation, proceed as follows:

1. Start the BAdI builder (Transaction SE19) and create an enhancement implementation (e.g., ZEWM_EI_CORE_PTS_TYPSQ) for enhancement spot /SCWM/ES_CORE_PTS.
2. Then create a BAdI implementation (e.g., ZEX_CORE_PTS_TYPSQ) for BAdI definition /SCWM/EX_CORE_PTS_TYPSQ and an implementing class (e.g., ZCL_IM_CORE_PTS_TYPSQ).
3. Now program the /SCWM/IF_EX_CORE_PTS_TYPSQ~STORAGE_TYPE_SEQ method with the sample code from Listing 4.3:
   - Begin to take over the transferred storage type search sequence (IV_PUT_SSEQ) and the storage control (IV_PUT_RULE) as a return parameter.
   - Next check if the quarantine period is maintained in the product master and whether a storage type search sequence has already been determined. You can assume that this is the case because the putaway strategy in the SAP Best Practices warehouse scenario is based entirely on the fact that for all products a corresponding putaway control indicator is maintained.
   - Then change the storage type search sequence so that the putaway strategy later finds the other storage types based on the Customizing; for example, search sequence YE02 is changed into QE02.
   - Finally, document your changes in the application log by writing a message as information in table ET_BAPIRET. This is very important so that the user can also understand the system behavior.

```
METHOD /scwm/if_ex_core_pts_typsq~storage_type_seq.

    BREAK-POINT ID zewmdevbook_1v5b.
    "1 Set return values
    ev_put_sseq = iv_put_sseq.
    ev_put_rule = iv_put_rule.
    "2 Changing the stor.type search seq.
    IF  NOT is_mat_global-qqtime IS INITIAL
    AND NOT ev_put_sseq IS INITIAL.
      REPLACE SECTION LENGTH '1' OF ev_put_sseq
      WITH c_pts_typsq.
      "3 Raise message
      MESSAGE i001(zewmbookdev_1v5b)
      WITH iv_put_sseq ev_put_sseq INTO DATA(message).
      DATA(ls_bapiret) = VALUE bapiret2( type       = sy-msgty
                                         id         = sy-msgid
                                         number     = sy-msgno
                                         message_v1 = sy-msgv1
```

```
                                    message_v2 = sy-msgv2
                                    message_v3 = sy-msgv3
                                    message_v4 = sy-msgv4 ).
        APPEND ls_bapiret TO et_bapiret.
      ENDIF.

    ENDMETHOD.
```

Listing 4.3 Sample Coding for BAdI /SCWM/EX_CORE_PTS_TYPSQ

Create the C_PTS_TYPSQ Constant

Create the new static constant `C_PTS_TYPSQ` on the **Attributes** tab in the implementing class tab of the BAdI with type `CHAR1` and assign the initial value `Q`.

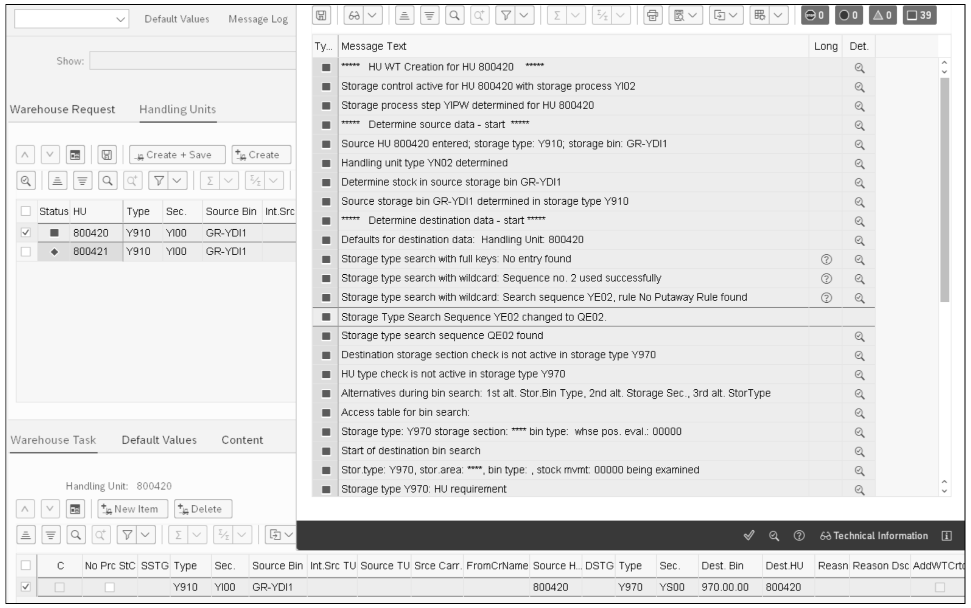
Testing the Enhancement 1V5b

To test enhancement 1V5b, run the 1V5 test case with material EWMS4-01 for direct putaway to the mezzanine storage type. Ensure that the quarantine period is set for material EWMS4-01 on the **Warehouse Execution** tab of the material master. At step 4.3.2, use the test instructions in Table 4.12.

Step	Step Description	Input Data and Expected Results
4.3.1	Process Goods Receipt (creating the final putaway warehouse orders)	Trigger putaway warehouse task while receiving the newly created handling units.
4.3.2	Check Warehouse Orders	*Expected result:* Warehouse orders for putaway have been created. In contrast to the normal case of putaway to the mezzanine, the warehouse tasks point to the destination bins of your alternative Q storage type (e.g., 970.00.00 in destination storage type Y970).

Table 4.12 Test Steps for Enhancement 1V5b

After a successful test, you should see the custom message for changing the storage type search sequences come up in the message log. In addition, the mezzanine destination storage type should be replaced by the defined quarantine storage type from the storage type search strategy—in our case, clarification zone Y970. See Figure 4.14 for successful testing results.



Status	HU	Type	Sec.	Source Bin	Int.Src
■	800420	Y910	YI00	GR-YDI1	
◆	800421	Y910	YI00	GR-YDI1	

```
Message Text
*****  HU WT Creation for HU 800420  *****
Storage control active for HU 800420 with storage process YI02
Storage process step YIPW determined for HU 800420
*****  Determine source data - start  *****
Source HU 800420 entered; storage type: Y910; storage bin: GR-YDI1
Handling unit type YN02 determined
Determine stock in source storage bin GR-YDI1
Source storage bin GR-YDI1 determined in storage type Y910
*****  Determine destination data - start *****
Defaults for destination data:  Handling Unit: 800420
Storage type search with full keys: No entry found
Storage type search with wildcard: Sequence no. 2 used successfully
Storage type search with wildcard: Search sequence YE02, rule No Putaway Rule found
Storage Type Search Sequence YE02 changed to QE02.
Storage type search sequence QE02 found
Destination storage section check is not active in storage type Y970
HU type check is not active in storage type Y970
Alternatives during bin search: 1st alt. Stor.Bin Type, 2nd alt. Storage Sec., 3rd alt. StorType
Access table for bin search:
Storage type: Y970 storage section: **** bin type:  whse pos. eval.: 00000
Start of destination bin search
Stor.type: Y970, stor.area: ****, bin type: , stock mvmt: 00000 being examined
Storage type Y970: HU requirement
```

Handling Unit: 800420

C	No Prc StC	SSTG	Type	Sec.	Source Bin	Int.Src TU	Source TU	Srce Carr.	FromCrName	Source H...	DSTG	Type	Sec.	Dest. Bin	Dest.HU	Reasn	Reason Dsc	AddWTCrt
□	□		Y910	YI00	GR-YDI1					800420		Y970	YS00	970.00.00	800420			□

Figure 4.14 Successful Testing Results of Enhancement 1V5b

4.3.4 Enhancement 1V5c: Enhancing the Warehouse Monitor

With the help of variant 1V5c, we want to show you how you can place your own node into the warehouse management monitor for advanced data selection using a detailed guide. The SAP Best Practices scope items are also completely executable without batch functionality.

In practice, however, most warehouses handle batch stocks at least partially. Because these batches are generally classified—that is, batch characteristics (production date, vendor batch, etc.) are included—we want to display the selection of the available stock in the warehouse management monitor in addition to the **Production Date** and **Vendor Batch** batch characteristics.

We implement our new node and the necessary ABAP logic based on the standard functions that are behind the **Available Stock** monitor node. These standard functions can be found primarily in the `/SCWM/AVLSTOCK_NO_BINS_MON` function module and in the includes of the associated function group, `/SCWM/STOCK_OVERVIEW_MON`. Here, however, exists a variety of embedded monitor nodes that might seem a little confusing at first. Therefore, we copy some coding sections particularly for this enhancement.

Custom development 1V5c, Enhancing the Warehouse Management Monitor, requires the following steps:

1. Enhance the `/SCWM/INCL_EEW_AQUA` extension structure with structure `ZEWM_S_AQUA_ALL_MON` (with fields `ZZ_PROD_DAT` and `ZZ_VEND_BATCH`).
2. Create the `Z_OVERVIEW_MONITOR` function group and implement the includes.

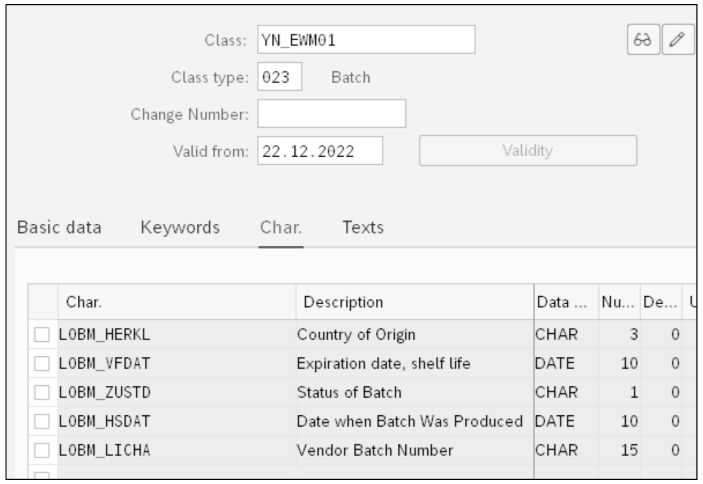
3. Implement the Z_YEWM_AVLSTOCK_NO_BINS_MON function module within function group Z_OVERVIEW_MONITOR.
4. Create the text elements.
5. Define a new **Available Stock (1710)** node in the IMG and assign function module Z_YEWM_AVLSTOCK_NO_BINS_MON.

Next, we explain in detail how to proceed with each of these steps. As usual, we will begin by discussing the prerequisites and end by discussing testing of this enhancement.

Prerequisites for Enhancement 1V5c

For the realization of enhancement 1V5c, the following prerequisites must be fulfilled with respect to the master data:

- At least one product is managed in batches (e.g., EWMS4-20 large part, fast-moving item, batches).
- You use the standard *production date* (LOBM_HSDAT) and *vendor batch* (LOBM_LICHA) characteristics as class characteristics. As an example, your batch class in EWM could look like the image in Figure 4.15 (class YN_EWM01). Feel free to extend the SAP Best Practices–provided batch class or create a new class and assign it to your newly created batch-managed material.



Class: YN_EWM01
Class type: 023 Batch
Change Number:
Valid from: 22.12.2022

	Char.	Description	Data ...	Nu...	De...
☐	LOBM_HERKL	Country of Origin	CHAR	3	0
☐	LOBM_VFDAT	Expiration date, shelf life	DATE	10	0
☐	LOBM_ZUSTD	Status of Batch	CHAR	1	0
☐	LOBM_HSDAT	Date when Batch Was Produced	DATE	10	0
☐	LOBM_LICHA	Vendor Batch Number	CHAR	15	0

Figure 4.15 SAP Best Practices Batch Class YN_EWM01, Extended with Characteristics for Enhancement 1V5c

  We will not describe how to set up the batch management required for a product or how to create batch classes and their validations at this point. There is plenty of information available on this subject, as well as several SAP Notes (SAP Note 990638, SAP Note 1305698, etc.) on the SAP Support Portal (*https://support.sap.com*).

- These characteristics are recorded during the goods receipt process.

Enhance the Extension Structure

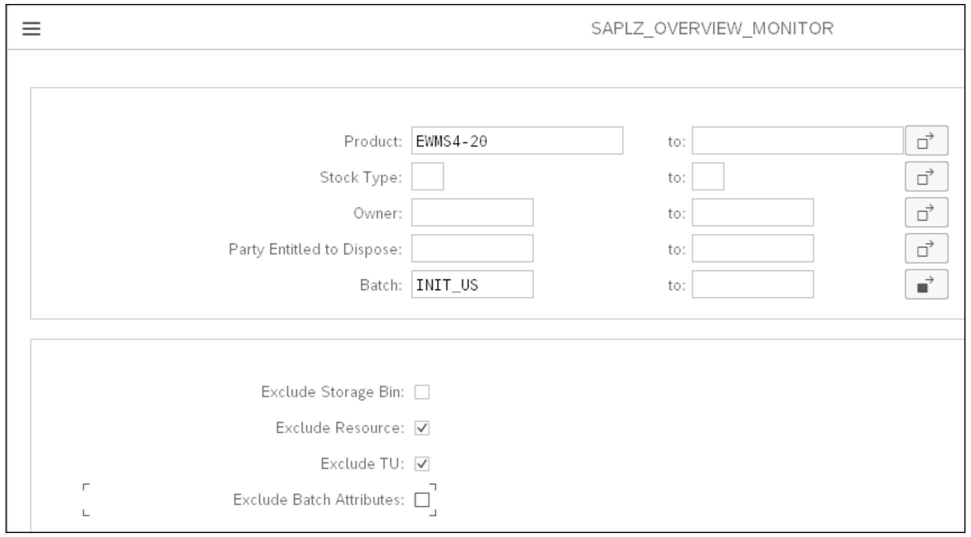
To create the new append structure, proceed as follows:

1. First, extend the standard structure that displays available stock in the warehouse management monitor. Start Transaction SE11 (ABAP Dictionary Maintenance) and select structure /SCWM/INCL_EEW_AQUA.
2. Click the **Append Structure** button and create the new append structure, ZEWM_S_AQUA_ALL_MON.
3. Declare two fields, ZZ_PROD_DAT (data element /SCWM/DE_BPROD_DATE) and ZZ_VEND_BATCH (data element /SCWM/DE_VENDOR_BATCHNO). The structure can be enhanced (character-type or numeric).
4. Activate your append structure.

Create the Function Group

To create and implement the master program, proceed like this:

1. Start the ABAP workbench (Transaction SE80) and choose your development package. Create function group Z_OVERVIEW_MONITOR using the context menu. If you want to create more project-specific monitor nodes, we recommend that you also embed the needed function modules in this function group.
2. Next, declare in the LZ_OVERVIEW_MONITORTOP TOP include of the function group the global variables and then make the selection screen 100 (see Listing 4.4), which appears when you select the monitor node in Transaction /SCWM/MON by double-clicking. You can see that for the image design, neither the screen painter nor any other settings or logic (process after input [PAI], process before output [PBO]) need to be considered. Our selection screen will end up looking like the one in Figure 4.16.



SAPLZ_OVERVIEW_MONITOR

Product: EWMS4-20 to:
Stock Type: to:
Owner: to:
Party Entitled to Dispose: to:
Batch: INIT_US to:

Exclude Storage Bin: ☐
Exclude Resource: ☑
Exclude TU: ☑
Exclude Batch Attributes: ☐

Figure 4.16 Adjustments to Stock Selection Screen for Enhancement 1V5c

3. Finally, include two standard includes as most form routines, which we use for our example, are included there.

```
FUNCTION-POOL z_overview_monitor.           "MESSAGE-ID ..

* INCLUDE LZ_OVERVIEW_MONITORD...            " Local class definition

"1. Global data
TYPE-POOLS: rsds, wmegc, abap, icon.
TABLES: sscrfields, /scwm/s_ui_mon_suom_change.
TYPES:
  BEGIN OF lsty_btch_val,
    matid         TYPE /scwm/de_matid,
    batchid       TYPE /scwm/de_batchid,
    zz_vend_batch TYPE /scwm/de_charg,
    zz_prod_dat   TYPE /scwm/de_bprod_date,
  END OF lsty_btch_val,
  ltty_btch_val TYPE STANDARD TABLE OF lsty_btch_val.
CONSTANTS: gc_vbtch TYPE atnam VALUE 'LOBM_LICHA',
           gc_hsdat TYPE atnam VALUE 'LOBM_HSDAT'.
DATA:
  functxt        TYPE smp_dyntxt,
  gs_tabname     TYPE /scwm/s_tabname_alias,
  gt_tabname     TYPE /scwm/tt_tabname_alias,
  gt_wherecl_tab TYPE rsds_twhere,
  gt_whereclause TYPE rsds_where_tab,
  gv_matnr_sel   TYPE /scwm/s_lagp_mon_f4-matnr,
  gv_matid_sel   TYPE /scwm/de_matid,
  gv_stcat_sel   TYPE /scwm/s_lagp_mon_f4-stcat,
  gv_entit_sel   TYPE /scwm/s_lagp_mon_f4-entitled,
  gv_owner_sel   TYPE /scwm/s_lagp_mon_f4-owner,
  gv_batch_sel   TYPE /scwm/s_lagp_mon_f4-charg,
  gv_free_sel    TYPE char255,
  go_mon_stock   TYPE REF TO /scwm/cl_mon_stock,
  gt_btchval     TYPE ltty_btch_val.
"Additional declarations S4
DATA gv_txt02.
DATA p_altme.
DATA p_quana.
DATA p_meins.
DATA p_quan.
"2. Build selection screen 100
SELECTION-SCREEN BEGIN OF SCREEN 100 AS WINDOW.
  SELECTION-SCREEN BEGIN OF BLOCK warehouse.
    PARAMETERS:
```

```
    p_lgnum TYPE /scwm/s_lagp_mon_f4-lgnum NO-DISPLAY.
  SELECTION-SCREEN END OF BLOCK warehouse.
  SELECTION-SCREEN BEGIN OF BLOCK stock_batch WITH FRAME TITLE TEXT-001.
    SELECT-OPTIONS:
    s_matnr FOR gv_matnr_sel,
    s_stcat FOR gv_stcat_sel,
    s_owner FOR gv_owner_sel,
    s_entit FOR gv_entit_sel,
    s_batch FOR gv_batch_sel,
    s_matid FOR gv_matid_sel NO-DISPLAY.
  SELECTION-SCREEN END OF BLOCK stock_batch.
  SELECTION-SCREEN BEGIN OF BLOCK exclude
    WITH FRAME TITLE TEXT-002.
    PARAMETERS:
      p_lgpla TYPE xfeld,
      p_rsrc  TYPE xfeld,
      p_tu    TYPE xfeld,
      p_ybtch TYPE xfeld.
  SELECTION-SCREEN END OF BLOCK exclude.
  SELECTION-SCREEN BEGIN OF BLOCK free.
    SELECT-OPTIONS:
    s_free FOR gv_free_sel NO-DISPLAY.
  SELECTION-SCREEN END OF BLOCK free.
  SELECTION-SCREEN: FUNCTION KEY 1.
  SELECTION-SCREEN: FUNCTION KEY 2.
  SELECTION-SCREEN: FUNCTION KEY 3.
SELECTION-SCREEN END OF SCREEN 100.
"3. Include of standard routines
INCLUDE:
/scwm/lstock_overview_monf01,
/scwm/lstock_overview_monf02.
```

Listing 4.4 TOP Include

4. Insert the `LZ_OVERVIEW_MONITORS01` include by copying the `/SCWM/LSTOCK_OVERVIEW_MONS01` include from the `/SCWM/STOCK_OVERVIEW_MON` function group. The logic in this part of the source code (Listing 4.5) is responsible for the screen controller of the selection screen.
5. Adjust the standard logic in the following section (see Listing 4.5):
   – Change the code within the `CASE` statement for image number 100. The other distinctions you can remove because we only use one screen in our example.
   – If you want to use several different selection screens in the project environment due to several individual monitor nodes, you can add case distinctions at this point.

```
CASE sy-dynnr.
  WHEN ’0100’.
* Fill alias table
    PERFORM aqua_alias.
ENDCASE.
* Display dynamic selections dialog
* (and take S_FREE into account)
CALL FUNCTION ’/SCWM/DYN_SEL4MON’
  EXPORTING
    it_tabname     = gt_tabname
    it_selopt      = s_free[]
  IMPORTING
    et_wherecltab  = gt_wherecl_tab
    et_whereclause = gt_whereclause.
```

Listing 4.5 Sample for LZ_OVERVIEW_MONITORS01

6. Next define two form routines, `BATCH_VAL` and `YBTCH_MAPPING`, within in the new `LZ_OVERVIEW_MONITORF01` include:
   - In the `BATCH_VAL` subroutine (see Listing 4.6), first determine if the call mode is a refresh of the screen or whether it is a new selection. So long as the user does not enter new select conditions, we hold the already determined batch characteristics in the buffer (`gt_btchval`).
   - Then build a table containing the required product-batch combinations. Products without batches and multiple entries are sorted out. If we do not have a matching record in the buffer already, we read the entire batch master for each of the product-batch combinations using the `GET_BATCH` method. With this feature, you can also determine other batch attributes. For our custom development, we use only the production date and the vendor batch.
   - We need the second `YBTCH_MAPPING` subroutine (see Listing 4.6) for the fielded control for our selection screen. We use an additional parameter (`p_ybtch`), which we will use later to give the end users the opportunity to decide for themselves whether the batch characteristics are to be read or not.
   - In Chapter 3, Section 3.1, we indicated the importance of performance within the warehouse management monitor. The additional reading of several hundred batch master records costs a lot of time. Through our control parameter, the user decides whether to take a longer runtime behavior into account.
   - Thus, the control parameters during the initialization of the screen do not retain their assigned values. We need to extend table `LT_MAPPING` accordingly.

```
*&---------------------------------------------------------------------*
*& Include          LZ_OVERVIEW_MONITORF01
*&---------------------------------------------------------------------*
```

```
*&----------------------------------------------------*
*& Form BATCH_VAL
*-----------------------------------------------------*
FORM batch_val USING
     it_stock_mon TYPE /scwm/tt_stock_mon
     iv_mode      TYPE /scwm/de_mon_fm_mode.
  DATA:
    lt_whbatch    TYPE /scwm/tt_batch,
    lo_batch_appl TYPE REF TO /scwm/cl_batch_appl.

  "1 Clear buffers
  /scwm/cl_batch_appl=>cleanup( ).
  "2 Check for REFRESH-Mode
  IF iv_mode NE 4.
    CLEAR gt_btchval.
  ENDIF.
  LOOP AT it_stock_mon ASSIGNING FIELD-SYMBOL(<ls_stock_mon>).
    DATA(ls_whbatch) = VALUE /scwm/s_batch( lgnum   = <ls_stock_mon>-lgnum
                                            batchid = <ls_stock_mon>-batchid
                                            matid   = <ls_stock_mon>-matid ).
    APPEND ls_whbatch TO lt_whbatch.
  ENDLOOP.
  SORT lt_whbatch BY lgnum batchid matid.
  DELETE ADJACENT DUPLICATES FROM lt_whbatch.
  CLEAR ls_whbatch.
  LOOP AT lt_whbatch INTO ls_whbatch
    WHERE matid   IS NOT INITIAL
    AND   batchid IS NOT INITIAL.
    "3 If we have the record in buffer, we use it
    IF iv_mode EQ 4.
      READ TABLE gt_btchval TRANSPORTING NO FIELDS
        WITH KEY matid   = ls_whbatch-matid
                 batchid = ls_whbatch-batchid
      BINARY SEARCH.
      CHECK NOT sy-subrc = 0.
    ENDIF.
    "4 Get instance
    TRY.
        lo_batch_appl ?= /scwm/cl_batch_appl=>get_instance(
        iv_productid = ls_whbatch-matid
        iv_batchid   = ls_whbatch-batchid
        iv_lgnum     = ls_whbatch-lgnum ).
      CATCH /scwm/cx_batch_management.
        CONTINUE.
    ENDTRY.
    "5 Get batch and values
    TRY.
        lo_batch_appl->get_batch(
        EXPORTING
          iv_no_classification = abap_true "remove if necessary
          IMPORTING
            es_batch    = DATA(ls_batch)
            et_val_num  = DATA(lt_val_num)
            et_val_char = DATA(lt_val_char)
            et_val_curr = DATA(lt_val_curr) ).
      CATCH /scwm/cx_batch_management.
    ENDTRY.
    "6 Fill global table
    READ TABLE lt_val_char ASSIGNING FIELD-SYMBOL(<val_char>)
      WITH KEY charact = gc_vbtch.
    IF sy-subrc = 0.
      DATA(ls_btchval) = VALUE lsty_btch_val( zz_vend_batch = <val_char>-
value_char ).
    ENDIF.
    IF ls_batch-vendrbatch_used = abap_true.
      ls_btchval-zz_vend_batch = ls_batch-vendrbatch.
    ENDIF.
    READ TABLE lt_val_num ASSIGNING FIELD-SYMBOL(<val_num>)
      WITH KEY charact = gc_hsdat.
    IF sy-subrc = 0.
      ls_btchval-zz_prod_dat = CONV date( <val_num>-value_from ).
    ENDIF.
    IF ls_batch-prod_date_used = abap_true.
      ls_btchval-zz_prod_dat = ls_batch-prod_date.
    ENDIF.
    ls_btchval = CORRESPONDING #( BASE ( ls_btchval ) ls_batch ).

    APPEND ls_btchval TO gt_btchval.

  ENDLOOP.
ENDFORM. " BATCH_VAL
*&---------------------------------------------------*
*& Form YBTCH_MAPPING
*&---------------------------------------------------*
FORM ybtch_mapping
CHANGING ct_mapping TYPE /scwm/tt_map_selopt2field.
  DATA: ls_mapping TYPE /scwm/s_map_selopt2field.
  MOVE: '/SCWM/AQUA' TO ls_mapping-tablename,
```

```
        'P_YBTCH'    TO ls_mapping-selname,
        'X_BTCH'     TO ls_mapping-fieldname.
  APPEND ls_mapping TO ct_mapping.
ENDFORM. " YBTCH_MAPPING
```

Listing 4.6 Sample Coding for Include LZ_OVERVIEW_MONITORF01

Implement the Function Module

To implement the advanced function module, do the following:

1. Mark the function group in the ABAP workbench and create a function module via the context menu. Here we use the standard /SCWM/AVLSTOCK_NO_BINS_MON function module as the template for our new function module, Z_YEWM_AVLSTOCK_NO_BINS_MON.
2. In Chapter 3, Section 3.1, we briefly presented the rough structure that you should consider when creating your own function module for the warehouse management monitor. Based on the Z_YEWM_AVLSTOCK_NO_BINS_MON function module, we show this structure in an example. The numbered code sections (see Listing 4.7) serve the following purposes:
   - “1: Check if selection variant is used
   - “2: Clear screen elements
   - “3: Map select options and parameters to database tables and fields
   - “4: Fill selection criteria based on the selection variant, if used
   - “5: Check if selection screen is to be displayed
   - “6: Pass selection criteria
   - “7: Convert selection results to UI appearance; selection will be carried out completely by method GET_AVAILABLE_STOCK
3. Following the selection of the available stock, we determine the batch characteristics for the selection result, if necessary, using the batch_val form routine. We set control parameter p_ybtch by default to “X.” This facilitates fast selection for the user without batch characteristics.
4. Finally, we bring the available stocks and the corresponding batch characteristics together in the return table ET_DATA. This table refers to the declaration of the /SCWM/S_AQUA_ALL_MON structure that we have extended in the first step.

```
FUNCTION z_ewm_avlstock_no_bins_mon.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(IV_LGNUM) TYPE  /SCWM/LGNUM
*"     REFERENCE(IV_VARIANT) TYPE  VARIANT OPTIONAL
*"     REFERENCE(IV_MODE) TYPE  /SCWM/DE_MON_FM_MODE DEFAULT '1'
*"  EXPORTING
*"     REFERENCE(ET_DATA) TYPE  /SCWM/TT_AQUA_ALL_MON
*"     REFERENCE(EV_RETURNCODE) TYPE  XFELD
*"     REFERENCE(EV_VARIANT) TYPE  VARIANT
*"  CHANGING
*"     REFERENCE(CT_TAB_RANGE) TYPE  RSDS_TRANGE OPTIONAL
*"  RAISING
*"      /SCWM/CX_MON_NOEXEC
*"----------------------------------------------------------------------

  DATA:
    lv_repid   TYPE sy-repid,
    lt_mapping TYPE /scwm/tt_map_selopt2field.

  lv_repid = sy-repid.
  CLEAR gt_tabname.
  "1 Only display popup and exit
  IF iv_mode = '3'.
    CALL FUNCTION 'RS_VARIANT_CATALOG'
      EXPORTING
        report               = lv_repid
        dynnr                = '0100'
      IMPORTING
        sel_variant          = ev_variant
      EXCEPTIONS
        no_report            = 1
        report_not_existent  = 2
        report_not_supplied  = 3
        no_variants          = 4
        no_variant_selected  = 5
        variant_not_existent = 6
        OTHERS               = 7.
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
      WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
    RETURN.
  ENDIF.
  "2 Initialization (clear screen elements)
  PERFORM initialization
    USING
      iv_lgnum
      lv_repid
    CHANGING
      et_data.
  "3 Fill mapping table
```

```
PERFORM aqua_mapping    CHANGING lt_mapping.
PERFORM bin_ind_mapping CHANGING lt_mapping.
PERFORM ybtch_mapping   CHANGING lt_mapping.
IF iv_variant IS NOT INITIAL.
  "4 Use selection criteria
  CALL FUNCTION 'RS_SUPPORT_SELECTIONS'
    EXPORTING
      report               = lv_repid
      variant              = iv_variant
    EXCEPTIONS
      variant_not_existent = 1
      variant_obsolete     = 2
      OTHERS               = 3.
  IF sy-subrc <> 0.
    MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
    WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
  ENDIF.
ENDIF.
IF lines( ct_tab_range ) > 0.
  CALL FUNCTION '/SCWM/RANGETAB2SELOPT'
    EXPORTING
      iv_repid     = lv_repid
      it_mapping   = lt_mapping
    CHANGING
      ct_tab_range = ct_tab_range.
ELSEIF iv_variant IS INITIAL.
  p_rsrc  = 'X'.
  p_tu    = 'X'.
  p_ybtch = 'X'.
ENDIF.
IF iv_mode = '1'.
  "5 Show selection screen
  p_lgnum = iv_lgnum.
  CALL SELECTION-SCREEN '0100' STARTING AT 10 10
  ENDING AT 130 30.
  IF sy-subrc <> 0.
    ev_returncode = 'X'.
    RETURN.
  ENDIF.
ENDIF.
"Prepare WHERECLAUSE
CLEAR gt_tabname.
PERFORM aqua_alias.
"Convert free select options to where clause
CALL FUNCTION '/SCWM/SFREE2WHERE4MON'
```

```
  EXPORTING
    it_tabname     = gt_tabname
    it_selopt      = s_free[]
  IMPORTING
    et_whereclause = gt_whereclause.
"Export selection criteria
CALL FUNCTION '/SCWM/SELOPT2RANGETAB'
  EXPORTING
    iv_repid     = lv_repid
    it_mapping   = lt_mapping
  IMPORTING
    et_tab_range = ct_tab_range.
"6. Select the data according to selection criteria
"7. Convert UI fields
CALL METHOD go_mon_stock->get_available_stock
  EXPORTING
    iv_skip_bin      = p_lgpla
    iv_skip_resource = p_rsrc
    iv_skip_tu       = p_tu
    it_matnr_r       = s_matnr[]
    it_cat_r         = s_stcat[]
    it_owner_r       = s_owner[]
    it_entitled_r    = s_entit[]
    it_charg_r       = s_batch[]
    it_whereclause   = gt_whereclause
  IMPORTING
    et_stock_mon     = DATA(lt_stock_mon)
    ev_error         = ev_returncode.
"Fill extensions (1V5c)
CHECK NOT lt_stock_mon IS INITIAL.
IF p_ybtch IS INITIAL.
  PERFORM batch_val USING lt_stock_mon iv_mode.
  SORT gt_btchval BY matid batchid.
ENDIF.
"Fill exporting table
LOOP AT lt_stock_mon ASSIGNING FIELD-SYMBOL(<fs_stock_mon>).
  DATA(ls_data) = CORRESPONDING /scwm/s_aqua_all_mon(
                    <fs_stock_mon> MAPPING
                    unit        = meins
                    cat_txt     = cat_text
                    doccat      = stref_doccat
                    stock_docno = stock_docno_ext ).
  READ TABLE gt_btchval ASSIGNING FIELD-SYMBOL(<fs_batch>)
    WITH KEY matid   = <fs_stock_mon>-matid
             batchid = <fs_stock_mon>-batchid
```

```
      BINARY SEARCH.
    IF sy-subrc EQ 0.
      ls_data = CORRESPONDING #( BASE ( ls_data ) <fs_batch> ).
    ELSE.
      CLEAR: ls_data-zz_prod_dat, ls_data-zz_vend_batch.
    ENDIF.
    APPEND ls_data TO et_data.
  ENDLOOP.
ENDFUNCTION.
```

Listing 4.7 Function Module Z_YEWM_AVLSTOCK_NO_BINS_MON

Create Text Elements

To create the new text elements, navigate to the selection texts and text symbols via **Go to • Text Elements**. The text elements you should create here can be found in Table 4.13 and Table 4.14.

Text Symbols	Text
001	Stock Attributes
002	Exclude Stock and Additional Info
DYN	Filled

Table 4.13 Text Symbols

Selection Texts	Text	Dictionary Reference
P_LGPLA	Exclude Storage Bin	
P_RSRC	Exclude Resource	
P_TU	Exclude TU	
P_YBTCH	Exclude Batch Attributes	
S_BATCH	Batch	X
S_ENTIT	Party Entitled to Dispose	X
S_MATNR	Product	X
S_OWNER	Owner	X
S_STCAT	Stock Type	X

Table 4.14 Selection Texts

Define a New Node

To define the new monitor node in Customizing, proceed as follows:

1. In Chapter 3, Section 3.1, we described how you can create a new monitor in Customizing and how to get along in the node hierarchy. Now place the new node profile ZAQUA018 in the IMG under **EWM • Monitoring • Warehouse Management Monitor • Define Nodes • Define Node Profiles** as a copy of the existing P0000018 profile.
2. Replace in the **List Funct. Module** field the function module /SCWM/AVLSTOCK_NO_BINS_MON with the new function module Z_YEWM_AVLSTOCK_NO_BINS_MON. Also change the following values:
   - **Dynpro Program**: SAPLZ_OVERVIEW_MONITOR (name of the master program of the function group)
   - **Dynpro No.**: 0100
   - **Text**: Available Stock (1710)
3. Now select the **Define node** point in the same IMG activity and create the new ZAQUALL18 node using node profile ZAQUA018. Save your settings.
4. Now go to the **EWM • Monitoring • Warehouse Management Monitor • Define Monitor** IMG activity. Select the monitor you would like to enhance, such as **SAP**, and then jump by double-clicking in the left dialog structure to **Define Node Hierarchy**. Our node should be placed under the main **Stock and Bin** node. Click the **Position** button and look for the parent node C000000011 (**Stock and Bin**). You will notice that there are exactly 11 subnodes. Now put in the 16th sequence by using the following values:
   - **Higher Node**: C000000011
   - **Lower Node**: ZAQUALL18
   - **Sequence**: 16
   - Leave both checkboxes empty.
5. At last, save your entries.

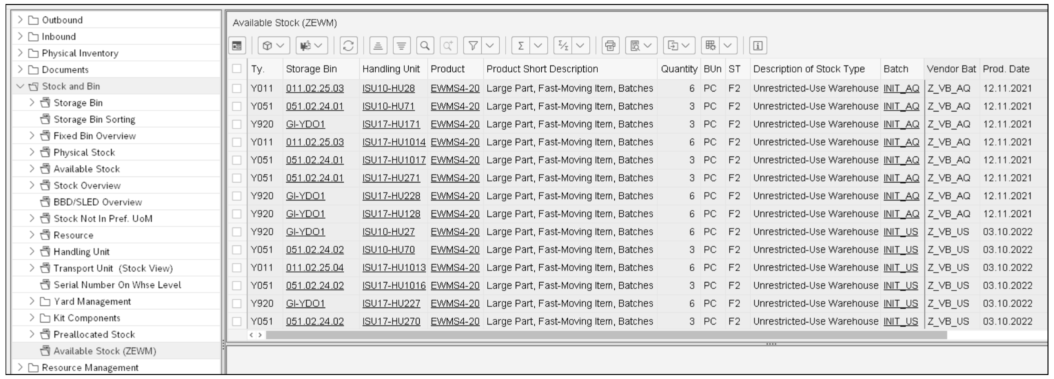
Testing Enhancement 1V5c

To test enhancement 1V5c, run the 1V5 test case with a batch-managed material for which the **Production Date** (LOBM_HSDAT) and **Vendor Batch** (LOBM_LICHA) batch characteristics are maintained. Then use the test instructions in Table 4.15 to test the custom development.

After the selection, the result of the custom development may look like the image in Figure 4.17.

Step	Step Description	Input Data and Expected Results
1	Selection of the available stock	■ Start Transaction /SCWM/MON (Warehouse Management Monitor) and navigate to monitor node **Stock and Bin • Available Stock (1710)**. ■ Open the selection screen by double-clicking this node, enter the batch managed product, and choose **Execute** F8. *Note:* Make sure that you have unchecked the **Exclude Batch Attributes** checkbox. *Expected result:* Both batch characteristics are displayed in the selection result.

Table 4.15 Test Steps for Enhancement 1V5c



Available Stock (ZEWM)

Ty.	Storage Bin	Handling Unit	Product	Product Short Description	Quantity	BUn	ST	Description of Stock Type	Batch	Vendor Bat	Prod. Date
Y011	011.02.25.03	ISU10-HU28	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y051	051.02.24.01	ISU10-HU71	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y920	GI-YDO1	ISU17-HU171	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y011	011.02.25.03	ISU17-HU1014	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y051	051.02.24.01	ISU17-HU1017	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y051	051.02.24.01	ISU17-HU271	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y920	GI-YDO1	ISU17-HU228	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y920	GI-YDO1	ISU17-HU128	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_AQ	Z_VB_AQ	12.11.2021
Y920	GI-YDO1	ISU10-HU27	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_US	Z_VB_US	03.10.2022
Y051	051.02.24.02	ISU10-HU70	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_US	Z_VB_US	03.10.2022
Y011	011.02.25.04	ISU17-HU1013	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_US	Z_VB_US	03.10.2022
Y051	051.02.24.02	ISU17-HU1016	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_US	Z_VB_US	03.10.2022
Y920	GI-YDO1	ISU17-HU227	EWMS4-20	Large Part, Fast-Moving Item, Batches	6	PC	F2	Unrestricted-Use Warehouse	INIT_US	Z_VB_US	03.10.2022
Y051	051.02.24.02	ISU17-HU270	EWMS4-20	Large Part, Fast-Moving Item, Batches	3	PC	F2	Unrestricted-Use Warehouse	INIT_US	Z_VB_US	03.10.2022

Figure 4.17 Monitor Available Batch Stock Enhancement 1V5c

4.3.5 Enhancement 1V5d: Delay Inbound Delivery with Missing Batch

The following custom development shows a way to influence the creation of the inbound delivery notification when entering the batch during inbound delivery processing in SAP ERP or SAP S/4HANA. This scenario might only occur in decentral EWM, where replication of batches from SAP ERP or SAP S/4HANA to EWM is still necessary. However, we still mention it here in context of the SAP Best Practices for embedded EWM, in case you might find yourself in a decentral EWM scenario and will specifically not be able to turn on direct batch replication in the **Define Enhanced Settings for Transfer to Decentralized EWM** Customizing transaction (view /SPE/V_EWM_DEST).

For example, it may happen that a truck delivers goods for which a purchase order indeed exists, but no inbound delivery has yet been created in the SAP ERP or SAP S/4HANA system. The employee in the goods receipt office creates an inbound delivery in the SAP ERP or SAP S/4HANA system with reference to the purchase order using

Transaction VL31N (Create Inbound Delivery) based on the supplier's delivery note. In this transaction, the employee can also create a batch for each delivery item that holds a batch-managed product. Because the distributions of the inbound delivery and the batch are carried out asynchronously and separately, the inbound delivery may be created in decentral EWM before the batch. The activation of the inbound delivery document will fail, and the document will remain in status *inactive*. With the help of the enhancement 1V5d, the creation of the inbound delivery document is delayed so that this situation does not occur.

Alternative Solutions

SAP Notes 1344366 and 2863720 describe in detail other ways you can work around the problem of inactive inbound delivery documents due to batch data missing during the inbound delivery processing. They might try to activate inactive documents once missing batches arrive in EWM instead of delaying inbound delivery document creation until batch data becomes available.

Enhancement 1V5d, Delay Inbound Delivery with Missing Batch, requires the following steps:

1. Implement the `DET_DOCTYPE` method of BAdI `/SCWM/EX_ERP_INT_CONF`.
2. Create a remote-enabled function module in SAP ERP or SAP S/4HANA.

Next, we explain in detail how to proceed with each of these steps. As usual, we will begin by discussing the prerequisites and end by discussing testing of this enhancement.

Prerequisites for Enhancement 1V5d

To achieve enhancement 1V5d, three requirements must be met:

- You are using decentralized EWM and transfer of batches via ALE.
- You must use SAP batch management. At this point, we will not describe how to set up the batch management required for a product and how to create batch classes and their validations. On this subject, there is plenty of information available, including several SAP Notes, on the SAP Support Portal (*http://support.sap.com*).
- You must not use automatic batch creation in EWM.
- Similar to the SAP Best Practices scope item, you must use alphanumeric product numbers.

Implement the DET_DOCTYPE Method

To create a BAdI implementation, proceed as follows:

1. Start the BAdI builder (Transaction SE19) and create an enhancement implementation (e.g., `ZEWM_EI_ERP_INT_CONF`) for enhancement spot `/SCWM/ES_ERP_INT_CONF`.

2. Then create a BAdI implementation (e.g., `ZEX_ERP_INT_CONF`) for BAdI definition `/SCWM/EX_ERP_INT_CONF` and an implementing class (e.g., `ZCL_IM_ERP_INT_CONF`).
3. Program the `/SCWM/IF_EX_ERP_INT_CONF~DET_DOCTYPE` method with the sample code from Listing 4.8:
   - First you integrate the standard logic of the `/SCWM/IF_EX_ERP_INT_CONF~DET_DOCTYPE` method of the `/SCWM/CL_DEF_IM_ERP_INT_CONF` fallback class.

BAdI /SCWM/EX_ERP_INT_CONF

Once you create an implementation for BAdI `/SCWM/EX_ERP_INT_CONF`, the default implementation of the `/SCWM/CL_DEF_IM_ERP_INT_CONF` fallback class is no longer executed. The methods of this class are responsible for document/item type determination in delivery documents. If you only want to substitute one of these determinations or none of them, you should first implement all three methods—`DET_DOCTYPE`, `DET_ITEMTYPE`, and `DET_ERP_DLVTYPE`—and call the corresponding standard methods of the fallback class. After the standard logic, you can then deploy and execute your own ABAP logic in one of the methods.

- Then determine, based on the business system group (`IV_ERPBSKEY`), the logical system and the corresponding RFC connection to the SAP ERP or SAP S/4HANA system from which the current inbound delivery has been distributed to EWM (coding sections “1 and “2).
- Then call the SAP ERP or SAP S/4HANA system by RFC to receive all material/batch combinations of the inbound delivery (coding section “3).
- Unless you get a return value (`LT_MATBTCH`), convert the SAP ERP or SAP S/4HANA material number in the technical key of SAP Supply Chain Management.
- With the product ID and the batch, check whether the batch master has already been created in EWM. If the batch master of the current material/batch combination exists, the entry of `LT_INTKEY` is deleted. This routine is repeated for all items. If there is still one entry in `LT_INTKEY` available at the end of the loop, wait for two seconds and start the checks for the remaining entries again (coding section “5).
- With the `DO` statement, we avoid an endless loop, if, for example, an error occurs during a specific batch distribution or creation.

```
METHOD /scwm/if_ex_erp_int_conf~det_doctype.

    DATA lo_std TYPE REF TO /scwm/cl_def_im_erp_int_conf.

    CREATE OBJECT lo_std.
    CALL METHOD lo_std->/scwm/if_ex_erp_int_conf~det_doctype
      EXPORTING
        iv_lgnum           = iv_lgnum
        iv_erpbskey         = iv_erpbskey
        it_bapidlvpartner   = it_bapidlvpartner
        it_header_deadlines = it_header_deadlines
        is_header           = is_header
        it_extension1       = it_extension1
        it_extension2       = it_extension2
        iv_doccat           = iv_doccat
      RECEIVING
        ev_doctype          = ev_doctype.

************* Enhancement 1V5d *******************

    TYPES:
      BEGIN OF lsty_mat_btch,
        matnr TYPE matnr,
        werks TYPE werks_d,
        charg TYPE /scwm/de_charg,
      END OF lsty_mat_btch.

    DATA:
      lt_matbtch          TYPE STANDARD TABLE OF lsty_mat_btch,
      lo_send_to_bussys   TYPE REF TO /scmb/cl_business_system,
      ls_receiving_system TYPE /scwm/s_recieving_system,
      ls_extkey           TYPE /scmb/mdl_ext_matnr_str,
      lt_extkey           TYPE /scmb/mdl_ext_matnr_tab,
      lt_extprod          TYPE /scmb/mdl_extprod_key_tab,
      ls_intkey           TYPE /scwm/dlv_matid_batchno_str,
      lt_intkey           TYPE /scwm/dlv_matid_batchno_tab,
      lo_stock_fields     TYPE REF TO /scwm/cl_ui_stock_fields.

    BREAK-POINT ID zewmdevbook_1v5d.

    CLEAR ls_receiving_system.

    "1. Get business object with receiver info
    ls_receiving_system-bskey = iv_erpbskey.
    TRY.
        lo_send_to_bussys =
        /scmb/cl_business_system=>get_instance( iv_erpbskey ).
      CATCH /scmb/cx_business_system. "#ec no_handler
        EXIT.
    ENDTRY.
    ls_receiving_system-logsys = lo_send_to_bussys->m_v_logsys.
    "2. Get RFC destination
```

```
    TRY.
        CALL METHOD /scwm/cl_mapout=>get_rfc_destination
          EXPORTING
            iv_erplogsys       = ls_receiving_system-logsys
          IMPORTING
            ev_rfc_destination = ls_receiving_system-rfc_destination.
      CATCH /scwm/cx_mapout. "#ec no_handler
        EXIT.
    ENDTRY.

    "3. RFC call: get product + batch from ERP delivery
    CALL FUNCTION 'Z_EWM_GET_BATCH_FROM_DLV'
      DESTINATION ls_receiving_system-rfc_destination
      EXPORTING
        iv_vbeln              = is_header-deliv_numb
      IMPORTING
        et_matnr_charg        = lt_matbtch
      EXCEPTIONS
        communication_failure = 1
        system_failure        = 2
        OTHERS                = 3.
    IF sy-subrc <> 0 OR lt_matbtch IS INITIAL.
      EXIT.
    ENDIF.

    "4. Convert matnr to matid (prefetch)
    LOOP AT lt_matbtch ASSIGNING FIELD-SYMBOL(<matbtch>).
      CLEAR ls_extkey.
      ls_extkey-ext_matnr = <matbtch>-matnr.
      COLLECT ls_extkey INTO lt_extkey.
    ENDLOOP.
    TRY.
        CALL FUNCTION '/SCMB/MDL_EXTPROD_READ_MULTI'
          EXPORTING
            iv_logsys = ls_receiving_system-logsys
            it_extkey = lt_extkey
          IMPORTING
            et_data   = lt_extprod.
      CATCH /scmb/cx_mdl. "#ec no_handler
        EXIT.
    ENDTRY.

    LOOP AT lt_matbtch ASSIGNING <matbtch>.
      "lt_extprod is sorted
```

```
  READ TABLE lt_extprod ASSIGNING FIELD-SYMBOL(<extprod>)
             WITH KEY ext_matnr = <matbtch>-matnr
  BINARY SEARCH.
  IF sy-subrc = 0.
    ls_intkey-productid = <extprod>-matid.
    ls_intkey-batchno = <matbtch>-charg.
    APPEND ls_intkey TO lt_intkey.
  ENDIF.
ENDLOOP.

IF lt_intkey[] IS INITIAL.
  EXIT.
ENDIF.

IF NOT lo_stock_fields IS BOUND.
  CREATE OBJECT lo_stock_fields.
ENDIF.

"5. Check if batch master exists
DO 10 TIMES.
  CALL METHOD lo_stock_fields->prefetch_batchid_by_no
    EXPORTING
      it_matid_charg    = lt_intkey
    IMPORTING
      et_batchid_extkey = DATA(lt_batch).
  SORT lt_batch BY batchno productid.
  LOOP AT lt_intkey ASSIGNING FIELD-SYMBOL(<intkey>).
    READ TABLE lt_batch ASSIGNING FIELD-SYMBOL(<batch>)
               WITH KEY batchno = <intkey>-batchno
                      productid = <intkey>-productid
               BINARY SEARCH.
    IF sy-subrc IS INITIAL AND
    <batch>-batchid IS NOT INITIAL. "batch exists
      DELETE lt_intkey.
    ENDIF.
  ENDLOOP.
  "Every batch exists -> lt_inkey is empty
  IF lt_intkey[] IS INITIAL.
    EXIT.
  ENDIF.
  WAIT UP TO 2 SECONDS.
  "Reset buffer for batches
  /scwm/cl_batch_appl=>cleanup( ).
  CLEAR lt_batch.
```

```
  ENDDO.

ENDMETHOD.
```

Listing 4.8 Sample Code for BAdI /SCWM/EX_ERP_INT_CONF

Create a Remote-Enabled Function Module

To implement the RFC function module in the SAP ERP or SAP S/4HANA system, proceed as follows:

1. Create a function group (e.g., Z_EWM_MISSING_BATCH) in the SAP ERP or SAP S/4HANA system that is connected to the EWM. To do this, start Transaction SE37 (Function Builder) and select the menu path **Goto • Function Groups • Create Group**.
2. Then create in this function group a new function module (e.g., Z_EWM_GET_BATCH_FROM_DLV) and implement the coding from Listing 4.9. Make sure that you choose, on the **Attributes** tab, the setting **Remote-Enabled Module** for the function module as the processing type:
   - First, set the read options for determining the delivery and then call the BAPI_DELIVERY_GETLIST function module with your SAP ERP or SAP S/4HANA delivery number.
   - Coding section "3 checks for which delivery items a batch has been entered and fills the appropriate material batch combination in the return table ET_MATNR_CHARG.

```
FUNCTION z_ewm_get_batch_from_dlv.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(IV_VBELN) TYPE  VBELN
*"  EXPORTING
*"     REFERENCE(ET_MATNR_CHARG) TYPE  MCHA_KEY_TABLE
*"----------------------------------------------------------------------
  DATA:
    ls_vbeln       TYPE bapidlv_range_vbeln,
    lt_vbeln       TYPE STANDARD TABLE OF bapidlv_range_vbeln,
    ls_dlv_item    TYPE bapidlvitem,
    lt_dlv_item    TYPE bapidlvitem_t,
    ls_dlv_control TYPE bapidlvbuffercontrol,
    ls_matnr_charg TYPE mcha_key,
    lv_lines       TYPE i.

  "1. Set read options
  ls_dlv_control-bypassing_buffer = abap_true.
  ls_dlv_control-item = abap_true.
  ls_vbeln-sign = 'I'.
  ls_vbeln-option = 'EQ'.
  ls_vbeln-deliv_numb_low = iv_vbeln.
  APPEND ls_vbeln TO lt_vbeln.

  "2. Get delivery
  CALL FUNCTION 'BAPI_DELIVERY_GETLIST'
    EXPORTING
      is_dlv_data_control = ls_dlv_control
    TABLES
      it_vbeln            = lt_vbeln
      et_delivery_item    = lt_dlv_item.

  "3. Fill return table
  LOOP AT lt_dlv_item INTO ls_dlv_item.
    MOVE-CORRESPONDING ls_dlv_item TO ls_matnr_charg.

    CHECK NOT ls_matnr_charg-charg IS INITIAL.

    READ TABLE et_matnr_charg TRANSPORTING NO FIELDS
    WITH KEY matnr = ls_matnr_charg-matnr
    charg = ls_matnr_charg-charg.
    IF sy-subrc NE 0.
      DESCRIBE TABLE et_matnr_charg LINES lv_lines.
      ADD 1 TO lv_lines.
      INSERT ls_matnr_charg INTO et_matnr_charg
      INDEX lv_lines.
    ENDIF.
  ENDLOOP.
ENDFUNCTION.
```

Listing 4.9 Example SAP ERP or SAP S/4HANA Function Module

Dynamic Processing

In our coding example (see Listing 4.9), we have done a rather lean implementation. You can use this code and adjust it according to your project requirements; for example, you can make the check depending on the warehouse number (IV_LGNUM). It may also be advantageous if the number of loops and the waiting time are determined dynamically. You can achieve this by reading, for example, a custom parameter at the beginning. This allows you also to regulate the delay in a production system without having to change the code.

Testing Enhancement 1V5d

To test enhancement 1V5d, run this technical test case first. Then you will be able to test the process using the 1V5 test case as outlined in Table 4.16.

Step	Step Description	Input Data and Expected Results
1	Deregister the EWM inbound queues for delivery processing.	Start Transaction SMQR (Registration of Inbound Queues) and deregister the queues. The first three characters of the queue name are DLV*.
2	Turn off immediate processing for BATMAS IDocs in EWM.	In Transaction WE20 set inbound processing for message type BATMAS to **Background Report**.
3	Activate checkpoint group in EWM.	Start Transaction SAAB (Checkpoints that Can Be Activated) and activate the breakpoints of checkpoint group ZEWMDEVBOOK_1V5D.
4	Create an inbound delivery with reference to a purchase order in SAP ERP or SAP S/4HANA.	■ Start Transaction VL31N (Create Inbound Delivery). ■ Enter the data (e.g., as stated in the test case description). ■ Enter a batch for a batch-managed item. ■ Press **Save**. *Expected result:* The inbound delivery was distributed to EWM.
5.1	Start the waiting queue for the inbound delivery in EWM.	Start Transaction SMQ2 (QRFC Monitor, Inbound Queue) and search for all queues. You will find a new queue: DLV* (contains the inbound delivery). Activate the queue of the inbound delivery. You will be holding in the debugger at statement BREAK-POINT ID ZEWMDEVBOOK_1V5D. *Expected result:* You will not receive the required batch ID via export table LT_BATCH from method prefetch_batchid_by_no.
5.2	Trigger BATMAS IDoc processing in EWM	■ Start a second mode and process the BATMAS IDoc containing the batch master record in Transaction BD87. Trigger batch replication from SAP ERP or SAP S/4HANA via IDoc using Transaction BD90 if no IDoc has been received yet. ■ Now you can start another round of the DO statement in the first mode. *Expected result:* You will receive the required batch ID via export table LT_BATCH from method prefetch_batchid_by_no.

Table 4.16 Test Steps for Enhancement 1V5d

Contents

3 Frameworks and Development Tools in Extended Warehouse Management 147

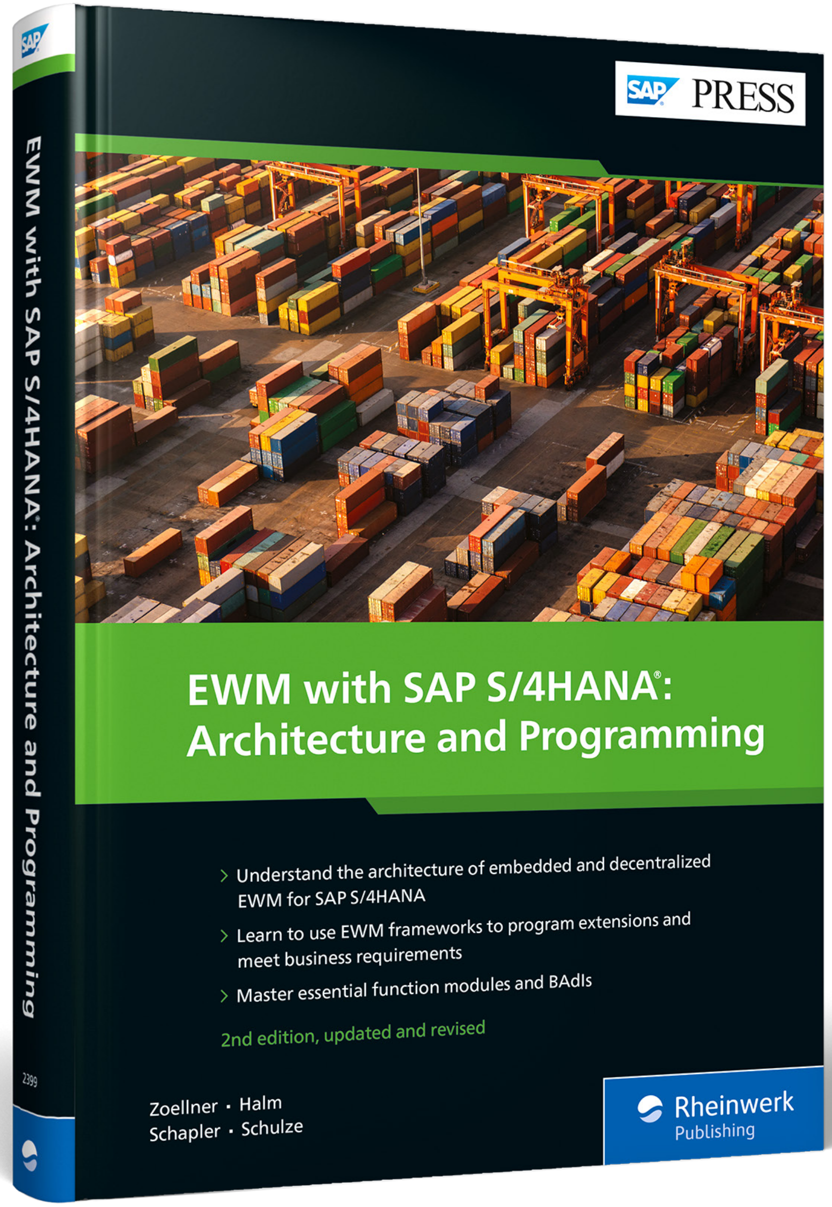
4 Enhancing SAP Best Practices for Embedded EWM 259

Appendices 463

Index

SAP PRESS First-hand knowledge.

Rheinwerk Publishing


SAP PRESS
EWM with SAP S/4HANA®: Architecture and Programming
› Understand the architecture of embedded and decentralized EWM for SAP S/4HANA
› Learn to use EWM frameworks to program extensions and meet business requirements
› Master essential function modules and BAdIs
2nd edition, updated and revised
Zoellner · Halm
Schapler · Schulze
Rheinwerk Publishing
EWM with SAP S/4HANA: Architecture and Programming

Zoellner, Halm, Schapler, Schulze

EWM with SAP S/4HANA: Architecture and Programming

493 pages | 07/2023 | $89.95 | ISBN 978-1-4932-2399-2

www.sap-press.com/5670

Peter Zoellner joined SAP Consulting in 2001 and has worked for both domestic and international customers from numerous industries, specializing in logistics execution and SAP EWM since its first introduction to the market. He holds a degree in economic studies (Diplom Kaufmann) from the Catholic University of Eichstätt, Germany. Peter is married with children, and he lives in the Kreuzberg district of Berlin.

Robert Halm is the head of portfolio management at prismat, where he focuses on mobile applications and SAP HANA solutions. His focus is on implementation and rollout projects in the field of integrated warehouse and distribution logistics with SAP EWM and workshops on special topics in supply chain management. While collaborating with highly specialized partners from several industries, he continuously drives innovation and go-to-market activities for leading-edge technologies integrated with SAP solutions.

Daniela Schapler is a solution architect at SAP SE. She joined SAP SE in 1996 as a developer in the area of logistics execution, following her study of physics at the Universität Tübingen. In her role as a solution architect, she was involved in the design of SAP EWM from the very beginning. She was instrumental in the development of handling unit management and has supervised several projects for companies running SAP. Since 2010, he has worked as part of the installed maintenance support for SAP EWM.

Karen Schulze has worked as a developer and consultant in the area of SAP EWM for more than 14 years. She is currently a senior consultant at abat, a company focusing on automotive industries and logistics. Karen started her professional career as a developer at SAP SE for warehouse management, joining the EWM development team. Starting in 2006, she worked as a consultant for SAP EWM projects across various industries. She is currently working on SAP EWM projects in automotive and furniture industries, where she implements SAP EWM standard processes.