Functional Specification Document

# Index

1. Document Information 1

2. Introduction 1

3. Business Requirement Overview 1

4. Business Process Flow 1

5. Functional Scope 1

6. Functional Solution Approach 1

7. Functional Requirements 1

8. Interfaces & Integration 1

9. Output 1

10. UI Requirement 1

11. Authorization & Security 1

12. Error Handling & Notifications 1

13. Assumptions & Dependencies 1

14. Test Scenario 1

15. Sign-Off 1

# 1. Document Information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document Title | Project Name | Client Name | Prepared By (Author, Department) | Document Version & Date |
| ZRCOPY\_SAMPLE\_ECC\_CODE\_V1 Technical Specification | ZRCOPY\_SAMPLE\_ECC\_CODE\_V1 |  | PWC AI Asset |  |

# 2. Introduction

This document provides the technical specification for the SAP ABAP program ZRCOPY\_SAMPLE\_ECC\_CODE\_V1. The purpose of this document is to outline the structure, logic, and components of the program, which is designed to perform comprehensive data extraction and processing routines within the SAP ECC environment. The scope includes detailed explanations of the program’s modular subroutines, data declarations, selection screen setup, and processing logic for handling material, delivery, billing, and financial data. The intended audience for this document includes SAP ABAP developers, technical consultants, and project stakeholders who require an in-depth understanding of the program’s technical implementation and its integration with standard SAP tables and processes.

# 3. Business Requirement Overview

The business requirement addressed by the ZRCOPY\_SAMPLE\_ECC\_CODE\_V1 program is to enable efficient extraction, processing, and preparation of data from various SAP ECC modules, including materials management, sales and distribution, and finance. The program is structured to fetch and process data from standard SAP tables such as MARA, LIPS, VBRK, VBRP, ACDOCA, and others, based on user input provided via a selection screen. The objective is to streamline the retrieval of relevant business data—such as material master records, delivery items, billing documents, and financial postings—and to prepare this data for further reporting or integration purposes. The solution addresses the need for modular, reusable, and maintainable ABAP code that can handle complex data processing requirements across multiple business areas.

# 4. Business Process Flow

Start

-> User enters Plant (p\_werks) and Material Numbers (s\_matnr) on the selection screen

-> fetch\_and\_check\_plant: Validate and fetch plant data from T001W if p\_werks is provided

-> fetch\_material: Retrieve material master data from MARA for selected materials

-> fetch\_delivery\_items: Fetch delivery item data from LIPS based on materials and plant

-> fetch\_konv: Retrieve condition records from prcd\_elements

-> fetc\_vbrk: Fetch billing document header (vbeln) from VBRK where draft is blank

-> fetch\_vbrp: Fetch billing document items from VBRP where draft is blank

-> fetch\_bsak: Retrieve financial postings from ACDOCA

-> fetch\_j1m0cust: Fetch customer numbers from KNA1

-> fetch\_marc\_stawn: Fetch MARC fields and call commodity code service classes

-> fetch\_dzaehk: Retrieve condition\_counter from prcd\_elements

-> fetch\_jbbranch: Fetch company code and business place from P\_BusinessPlace

-> fetch\_vbuk: Count entries in VBAK (sales document headers)

-> fetch\_marc\_mard: Retrieve material numbers from MARC and lsobs from MARD

-> fetch\_orderby: Process and fetch material data from MARA with substring logic and message construction

-> prepare\_final\_data: Process ilips data and populate ifinal internal table

-> populate\_salary: Extract dmbtr (amount) from acdoca to local variable

-> End

# 5. Functional Scope

1. In-Scope items

1. The program 'ZRCOPY\_SAMPLE\_ECC\_CODE\_V1' is designed to perform a series of data retrieval and processing tasks using subroutines (PERFORM statements) and includes. The following functionalities are in scope:

- Definition and handling of selection screen parameters and select-options for plant (p\_werks) and material numbers (s\_matnr), allowing user input for filtering data.

- Declaration and setup of internal tables and variables for processing data from SAP standard tables such as T001W (plant), MARA (material master), LIPS (delivery items), BSEG (accounting document segment), and others.

- Data retrieval and validation logic for plant data using the subroutine 'fetch\_and\_check\_plant', which checks if a plant parameter is provided and fetches the corresponding plant record from T001W.

- Fetching material data from MARA based on user-selected material numbers using the 'fetch\_material' subroutine.

- Fetching delivery item data from LIPS based on materials and plant using 'fetch\_delivery\_items', including sorting and mapping of data to internal tables.

- Retrieval of condition records from the PRCD\_ELEMENTS table into an internal table using 'fetch\_konv'.

- Fetching billing document header (VBRK) and item (VBRP) data, including filtering for non-draft documents.

- Fetching financial data from ACDOCA (accounting documents) and customer data from KNA1, including sorting and mapping to internal tables.

- Fetching commodity code and related details from MARC and using class-based service calls to '/sapsll/cl\_mm\_cls\_service' for commodity code classification and details.

- Fetching single fields or counters from PRCD\_ELEMENTS and other tables as required.

- Fetching business place data from P\_BusinessPlace and sales document status from VBAK.

- Fetching material and storage location data from MARC and MARD, including substring operations on material numbers.

- Performing ordered data retrieval and message construction based on material number and type substrings.

- Preparing final data by mapping and appending processed delivery item data into a final internal table, counting entries, and clearing the table for reuse.

- Populating salary-related data by extracting monetary values from ACDOCA.

- Use of object-oriented ABAP constructs, including class definition and implementation for encapsulating data and methods (e.g., lcl\_data with method get\_data).

- Modularization of code using includes for variable declarations, selection screens, and subroutine implementations.

2. Out-of-Scope items

1. The following functionalities are explicitly or implicitly out of scope:

- Any user interface beyond the selection screen (e.g., ALV grid display, custom screens, or interactive reports) is not included.

- Data manipulation or business logic beyond the retrieval, mapping, and basic processing of data (e.g., no complex calculations, updates, or transactional processing).

- Error handling, exception management, and user messaging are not detailed or implemented in the provided logic.

- Authorization checks, security validations, and user-specific logic are not addressed.

- Integration with external systems, interfaces, or non-SAP data sources is not covered.

- Output formatting, printing, or exporting of results (e.g., to PDF, Excel) is not included.

- Advanced performance optimization, buffering, or parallel processing techniques are not present.

- No implementation of custom business rules, workflow triggers, or event-driven processing.

- No persistence of processed data back to the database (i.e., no INSERT, UPDATE, or DELETE operations).

- No detailed implementation of the logic inside the class method 'get\_data' or other methods beyond the provided stubs.

- No dynamic selection screen elements or runtime modifications to the selection screen.

# 6. Functional Solution Approach

The business requirement is addressed by designing an ABAP program that orchestrates a series of data retrieval and processing steps, leveraging SAP standard tables and modular subroutines. The solution begins with a user-friendly selection screen, allowing users to specify a plant and a range of material numbers, ensuring that only relevant data is processed. Upon execution, the program sequentially performs a set of subroutines, each responsible for a distinct functional area such as plant validation, material and delivery item retrieval, pricing condition extraction, billing document processing, financial data gathering, and customer and branch information fetching.

Each subroutine is designed to encapsulate its logic, making use of internal tables and structures that mirror the underlying SAP data models. For example, material and delivery data are fetched and stored in internal tables structured after MARA and LIPS, while pricing and billing information is retrieved from KONV, VBRK, and VBRP tables. The program also includes logic for handling commodity code classification, number range counters, and storage location data, ensuring comprehensive coverage of the business process.

The final steps involve preparing the consolidated dataset for output or further processing, including the calculation or population of salary-related data where applicable. Throughout, the program maintains modularity and clarity, allowing for easy maintenance and scalability. The approach ensures that all relevant business data is accurately retrieved, validated, and processed in alignment with the specified user inputs and business rules.

# 7. Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement ID | Requirement Description | Business Rule | Priority | Comments |
| To Be Filled | To Be Filled | To Be Filled | To Be Filled | To Be Filled |

# 8. Interfaces & Integration

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Interface ID | Source System | Target System | Data/Message Type | Frequency/Mode | Description |  |
|  | -------------- | -------------- | -------------- | ------------------- | --------------- | ------------- |  |

# 9. Output

Based on the provided payload, the ABAP program 'ZRCOPY\_SAMPLE\_ECC\_CODE\_V1' is structured to perform a series of data retrieval and processing tasks through multiple subroutines. However, there is no explicit mention of specific output formats (such as Excel, PDF, CSV), layouts, or destinations (such as SAP AL11, email, or spool) in the available information. The program prepares and processes data from various SAP tables (e.g., MARA, LIPS, VBRK, VBRP, ACDOCA, KNA1, etc.) and organizes it into internal tables such as 'ifinal', 'ilips', and others.

The functional purpose of the outputs is to extract, process, and prepare data related to materials, deliveries, billing documents, financial postings, customers, plants, and other business objects for further use within the SAP system. The final data is likely intended for reporting or subsequent processing, but the exact output medium, format, and destination are not specified in the payload.

In summary, the program generates internal datasets as outputs for further processing or reporting, but no explicit external report, extract, or output file format or destination is defined in the provided information.

# 10. UI Requirement

The selection screen consists of the following UI elements:

1. Field Name: p\_werks

- Type: Input (Single-value input field)

- Data Element: t001w-werks

- Description: Single-value input field for plant (WERKS) based on table T001W.

- Default Value: None specified.

- Mandatory: Not specified.

- Business Purpose/Validation: Used to input a specific plant code. No additional validation or dependencies mentioned.

2. Field Name: s\_matnr

- Type: Input (Range input field)

- Data Element: mara-matnr

- Description: Range input for material number (MATNR) based on table MARA.

- Default Value: None specified.

- Mandatory: Not specified.

- Business Purpose/Validation: Allows the user to specify a range of material numbers for selection. No additional validation or dependencies mentioned.

No further user interactions, default values, mandatory flags, or validation logic are specified in the provided payload.

# 11. Authorization & Security

|  |  |  |  |
| --- | --- | --- | --- |
| Role/Profile | Authorization Object | Access Level | Description |
| [To Be Filled] | [To Be Filled] | [To Be Filled] | [To Be Filled] |

# 12. Error Handling & Notifications

# 13. Assumptions & Dependencies

# 14. Test Scenario

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Objective | Input Data | Expected Output | Actual Result/Status | Sign-off/Comments |
| TC01 | Validate plant selection screen input and plant data fetch | p\_werks = '1000' | Plant data for WERKS '1000' is fetched from T001W and stored in i\_t001w |  |  |
| TC02 | Validate material number range selection and material data fetch | s\_matnr = ['MAT001', 'MAT002'] | Material data for MATNR in ['MAT001', 'MAT002'] is fetched from MARA and stored in imara |  |  |
| TC03 | Fetch delivery items for selected materials and plant | p\_werks = '1000', s\_matnr = ['MAT001', 'MAT002'] | Delivery items from LIPS for MATNR in ['MAT001', 'MAT002'] and WERKS '1000' are fetched and stored in ilips |  |  |
| TC04 | Fetch condition records from prcd\_elements | None | All KNUMV values from prcd\_elements are fetched and stored in ikonv, sorted by KNUMV |  |  |
| TC05 | Fetch billing document header where draft is empty | None | A single VBELN from VBRK where DRAFT = space is fetched and stored in lv\_vbeln |  |  |
| TC06 | Fetch billing document items where draft is empty | None | VBELN and POSNR from VBRP where DRAFT = space are fetched and stored in lt\_vbrk |  |  |
| TC07 | Fetch accounting document data from ACDOCA | None | BUKRS, GJAHR, BELNR, BUZEI, HKONT, DMBTR, WRBTR, WAERS, BUDAT from ACDOCA are fetched and stored in lt\_data, ordered by all fields |  |  |
| TC08 | Fetch customer numbers from KNA1 | None | KUNNR from KNA1 are fetched and stored in lt\_data, ordered by KUNNR |  |  |
| TC09 | Fetch MARC commodity code and call classification services | None | STAWN and EXPME from MARC are fetched; /sapsll/cl\_mm\_cls\_service->get\_commodity\_code\_cls and get\_commodity\_code\_details are called |  |  |
| TC10 | Fetch condition counter from prcd\_elements | None | CONDITION\_COUNTER from prcd\_elements is fetched and stored in lv\_dzaehk |  |  |
| TC11 | Fetch business place data from P\_BusinessPlace | None | CompanyCode and BusinessPlace from P\_BusinessPlace are fetched and stored in lt\_data, ordered by BUKRS and BRANCH |  |  |
| TC12 | Count number of sales document headers in VBAK | None | Number of entries in VBAK is counted and stored in lv\_vbak\_cnt |  |  |
| TC13 | Fetch MARC material numbers by substring and single MARD field | lv\_matnr = 'MAT1234567' | MARC MATNR matching substring(3,4) of lv\_matnr are fetched into lt\_data; first LSOBS from MARD is fetched into lv\_lsobs |  |  |
| TC14 | Fetch material data with ordering and message construction | lv\_matnr, lv\_mtart set; lv\_matnr\_chk4 = '1234' | MARA MATNR, MTART, MATKL matching substring and type are fetched into lt\_table; message is constructed if lv\_matnr\_chk4 = '1234' |  |  |
| TC15 | Prepare final data from delivery items | ilips populated with delivery items | For each entry in ilips, corresponding fields are moved to ifinal and appended; number of lines counted and ifinal is refreshed |  |  |
| TC16 | Populate salary from ACDOCA | acdoca-dmbtr = 1000 | lv\_salary is set to 1000 |  |  |
| TC17 | Fetch single MARC material number by substring | lv\_matnr = 'MAT1234567' | MARC MATNR matching substring(3,3) of lv\_matnr is fetched into lv\_marc\_matnr |  |  |
| TC18 | Class method get\_data clears billing document variable | gv\_vbrk previously set | After calling get\_data, gv\_vbrk is cleared |  |  |

# 15. Sign-Off

|  |  |  |  |
| --- | --- | --- | --- |
| Role | Name | Signature | Date |
| Prepared By |  |  |  |
| Approved By |  |  |  |
| Client Sign-Off |  |  |  |

Document generated by PWC AI-powered ABAP Tech Spec Assistant.