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 Implementation |  | 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 data processing steps implemented within the program, which is designed to perform comprehensive data extraction and processing routines across various SAP standard and custom tables. The scope includes the definition of selection screens, data declarations, modular subroutine logic for data retrieval and transformation, and preparation of final datasets for reporting or further processing. The intended audience for this document includes SAP ABAP developers, technical consultants, and project stakeholders who require a detailed understanding of the program's technical design and implementation.

# 3. Business Requirement Overview

The business requirement addressed by the ZRCOPY\_SAMPLE\_ECC\_CODE\_V1 program is to automate and streamline the extraction, validation, and processing of key business data from multiple SAP tables, including material, plant, delivery, billing, accounting, and customer information. The objective is to provide a modular, reusable, and efficient solution that fetches, validates, and prepares data for reporting or downstream processes. The solution addresses the need for accurate and timely data retrieval, transformation, and aggregation, supporting business operations that depend on integrated information from logistics, finance, and sales modules.

# 4. Business Process Flow

Start

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

-> System performs fetch\_and\_check\_plant (validates and retrieves plant data if provided)

-> System performs fetch\_material (retrieves material master data for selected materials)

-> System performs fetch\_delivery\_items (fetches delivery item data based on material and plant)

-> System performs fetch\_konv (retrieves condition records from prcd\_elements)

-> System performs fetc\_vbrk (fetches billing document header where draft is blank)

-> System performs fetch\_vbrp (fetches billing document items where draft is blank)

-> System performs fetch\_bsak (fetches accounting document data from acdoca)

-> System performs fetch\_j1m0cust (fetches customer numbers from KNA1)

-> System performs fetch\_marc\_stawn (fetches MARC fields and commodity code details)

-> System performs fetch\_dzaehk (fetches condition counter from prcd\_elements)

-> System performs fetch\_jbbranch (fetches business place data from P\_BusinessPlace)

-> System performs fetch\_vbuk (counts entries in VBAK)

-> System performs fetch\_marc\_mard (fetches material numbers from MARC and lsobs from MARD)

-> System performs fetch\_orderby (fetches and processes material data with ordering and message logic)

-> System performs prepare\_final\_data (transfers and prepares final dataset from delivery items)

-> System performs populate\_salary (extracts monetary value from acdoca)

-> 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 of a selection screen allowing users to input a plant (p\_werks) and select material numbers (s\_matnr) for filtering data.

- Data declarations and setup of internal tables for processing, including tables for condition records (ikonv), sales document details (ifinal), plant data (i\_t001w), delivery items (ilips), and material master data (imara).

- Fetching and validating plant data from T001W based on user input.

- Fetching material data from MARA based on selected material numbers.

- Fetching delivery item data from LIPS, filtered by material and plant, and populating internal tables accordingly.

- Fetching condition records from PRCD\_ELEMENTS into the ikonv internal table.

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

- Fetching financial data from ACDOCA, including company code, fiscal year, document number, line item, GL account, amounts, currency, and posting date.

- Fetching customer numbers from KNA1, ordered by customer number.

- Fetching commodity code and related details from MARC and using /SAPSLL/CL\_MM\_CLS\_SERVICE class methods for commodity code classification and details.

- Fetching a condition counter from PRCD\_ELEMENTS.

- Fetching business place data from P\_BusinessPlace, including company code and branch.

- Counting the number of entries in VBAK (sales document headers).

- Fetching material numbers from MARC based on substrings and retrieving storage location data from MARD.

- Fetching material data from MARA with dynamic substring and type-based filtering, and constructing messages based on material number patterns.

- Fetching a single material number from MARC based on a substring of another material number.

- Preparing final data by processing delivery items and populating the final output internal table.

- Populating salary data by extracting monetary values from ACDOCA.

- Definition and minimal implementation of a local class (lcl\_data) with a public data variable (gv\_vbrk) and a method (get\_data) for further data processing.

- Modularization of logic using includes for variable declarations, selection screen, 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 (no ALV grid, list output, or custom screens are defined).

- Data manipulation or updates to SAP tables (all operations are data retrieval only; no INSERT, UPDATE, or DELETE statements).

- Error handling, logging, or exception management beyond basic SY-SUBRC checks.

- Authorization checks or user validation.

- Integration with external systems or non-SAP data sources.

- Detailed business logic for calculation, aggregation, or transformation of data beyond the described data movement and assignment.

- Implementation details of the get\_data method in lcl\_data (only the CLEAR statement is present).

- Any HR-specific logic beyond the simple population of a salary variable.

- Output formatting, reporting, or data export functionalities.

- Performance optimization, parallel processing, or background job scheduling.

- Custom enhancements, BADIs, or user exits.

- Any logic not explicitly described in the provided payload, such as workflow triggers, notifications, or advanced validations.

# 6. Functional Solution Approach

The business requirement is addressed by designing an ABAP program that orchestrates a series of data retrieval and processing steps, each encapsulated in modular subroutines. The program begins by presenting a selection screen to the user, allowing input of a plant code and a range of material numbers. Based on these inputs, the program sequentially performs the following functional steps:

The program first validates and fetches plant data if a plant code is provided, ensuring that subsequent data retrievals are contextually accurate. It then retrieves material master data for the selected material numbers, followed by fetching delivery item data that matches the selected materials and plant. Condition records and pricing information are fetched to support further business logic.

Billing document headers and items are retrieved to provide a comprehensive view of sales and billing activities. Vendor open item data and customer master data are also fetched to support financial and customer-related reporting. The program further retrieves material storage and commodity code information, as well as number range or counter data for tracking purposes.

Branch-specific data and sales document status information are collected to provide organizational and process visibility. Additional material and storage location data are fetched for inventory and logistics analysis. The program also includes logic to fetch and process data in a specific order, ensuring that dependencies and business rules are respected.

Finally, the program prepares the final dataset for output or further processing, and includes logic to calculate or populate salary-related data if required. Throughout, the solution leverages internal tables, modular subroutines, and object-oriented constructs to ensure maintainability, scalability, and clarity in addressing the business requirement.

# 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 program 'ZRCOPY\_SAMPLE\_ECC\_CODE\_V1' is structured to perform a series of data retrieval and processing tasks through various subroutines. However, there is no explicit mention of specific reports, output formats (such as Excel, PDF, or CSV), or output destinations (such as SAP AL11, email, or spool) in the available information. The functional purpose of the program is to extract and process data from multiple SAP tables (such as MARA, LIPS, VBRK, VBRP, ACDOCA, KNA1, etc.) using subroutines, and to prepare final datasets for further processing.

While the program prepares internal tables and processes data (e.g., through 'prepare\_final\_data'), the payload does not specify any concrete output generated for users, nor does it describe the layout, format, or destination of any reports or extracts. The functional purpose of the outputs, inferred from the subroutine names and explanations, is to provide processed datasets related to materials, deliveries, billing, financials, customers, and other business objects for further use within the SAP system.

In summary, the program is designed to extract, process, and prepare data, but the specific details regarding the format, layout, and destination of any generated outputs are not provided in the payload.

# 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 users to specify a range of material numbers for selection. No additional validation or dependencies mentioned.

No default values, mandatory flags, or inter-field dependencies 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

1. No explicit error handling or notification requirements are specified in the provided payload.

# 13. Assumptions & Dependencies

1. The solution assumes the existence and accessibility of SAP standard tables such as T001W, MARA, LIPS, BSEG, VBRK, VBRP, ACDOCA, KNA1, PRCD\_ELEMENTS, MARC, MARD, VBAK, and P\_BusinessPlace.

2. It is assumed that the selection screen parameters (p\_werks and s\_matnr) are provided by the user for data filtering.

3. The program depends on the availability of custom includes (e.g., ZRCOPY\_SAMPLE\_ECC\_CODE\_TOP\_V1, ZRCOPY\_SAMPLE\_ECC\_CODE\_F01\_V1) for global declarations and subroutine implementations.

4. The solution relies on the existence of the class /sapsll/cl\_mm\_cls\_service for commodity code classification operations.

5. The solution assumes that the user executing the program has the necessary authorizations to access all referenced tables and objects.

# 14. Test Scenario

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Objective | Input Data | Expected Output | Actual Result/Status | Sign-off/Comments |
| TC01 | Validate plant selection input | p\_werks = '1000' | Plant '1000' is accepted and corresponding T001W data is fetched into i\_t001w |  |  |
| TC02 | Validate material number range selection | s\_matnr = ['MAT001', 'MAT002'] | Materials 'MAT001' and 'MAT002' are selected from MARA and loaded into imara |  |  |
| TC03 | Fetch delivery items for selected materials and plant | imara contains ['MAT001', 'MAT002'], i\_t001w-werks = '1000' | ilips is populated with LIPS entries matching matnr in imara and werks = '1000' |  |  |
| TC04 | Prepare final data from delivery items | ilips populated with delivery items | ifinal is populated with vbeln, posnr, matnr, werks from ilips; number of lines in ifinal is counted and table is refreshed |  |  |
| TC05 | Fetch condition records from prcd\_elements | No specific input | ikonv is populated with knumv from prcd\_elements, sorted by knumv |  |  |
| TC06 | Fetch billing document header where draft is space | VBRK table contains entries with draft = space | lv\_vbeln is set to vbeln of the first VBRK entry where draft = space |  |  |
| TC07 | Fetch billing document items where draft is space | VBRP table contains entries with draft = space | lt\_vbrk is populated with vbeln and posnr from VBRP where draft = space |  |  |
| TC08 | Fetch accounting document data from ACDOCA | ACDOCA table contains relevant entries | lt\_data is populated with bukrs, gjahr, belnr, buzei, hkont, dmbtr, wrbtr, waers, budat ordered accordingly |  |  |
| TC09 | Fetch customer numbers from KNA1 | KNA1 table contains customer data | lt\_data is populated with kunnr from KNA1, ordered by kunnr |  |  |
| TC10 | Fetch MARC stawn and call commodity code services | MARC table contains stawn, expme | ls\_marc is populated; /sapsll/cl\_mm\_cls\_service methods get\_commodity\_code\_cls and get\_commodity\_code\_details are called |  |  |
| TC11 | Fetch condition counter from prcd\_elements | prcd\_elements table contains condition\_counter | lv\_dzaehk is set to condition\_counter from prcd\_elements |  |  |
| TC12 | Fetch business place data from P\_BusinessPlace | P\_BusinessPlace table contains CompanyCode and BusinessPlace | lt\_data is populated with bukrs and branch, ordered by both |  |  |
| TC13 | Count number of sales document headers in VBAK | VBAK table contains entries | lv\_vbak\_cnt is set to the count of entries in VBAK |  |  |
| TC14 | Fetch MARC material numbers based on substring and MARD lsobs | lv\_matnr provided; MARC and MARD tables contain relevant data | lt\_data is populated with matnr from MARC matching substring; lv\_lsobs is set from MARD |  |  |
| TC15 | Fetch material data with ordering and substring checks | lv\_matnr, lv\_mtart provided | lt\_table is populated with matnr, mtart, matkl from MARA matching substring and mtart; message is set if substring condition met |  |  |
| TC16 | Fetch single MARC material number based on substring | lv\_matnr provided | lv\_marc\_matnr is set to matnr from MARC matching substring |  |  |
| TC17 | Populate salary from ACDOCA | acdoca-dmbtr has value | lv\_salary is set to acdoca-dmbtr |  |  |
| TC18 | Validate selection screen input fields | User enters p\_werks and s\_matnr on selection screen | Program accepts single-value plant and range of material numbers as input |  |  |
| TC19 | Validate global declarations and includes | Program initialization | All global tables, internal tables, variables, and includes are declared and available |  |  |
| TC20 | Validate class lcl\_data and method get\_data | Call lcl\_data=>get\_data | gv\_vbrk is cleared (set to initial value) |  |  |

# 15. Sign-Off

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

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