Università degli Studi di Padova

Department of Information Engineering

Master's degree in
Computer Engineering

Data Warehousing for Financial Processes in the Fashion Industry

Supervisor:
Ch.mo Prof. Fabio Vandin

Candidate:
Riccardo Toffano
2023929

Academic Year 2022/2023
Date of Graduation 05/12/2023

Abstract

Business intelligence (BI) refers to the process of collecting, analyzing, and presenting data. It is a valuable tool for organizations to make informed business decisions, and improve their overall performance. By analyzing data from various sources, organizations can identify trends, understand customer behavior, and gain insights into their operations, making strategic decisions that give them an edge over their competitors. Given the datacentric nature of BI, there is the need of data warehousing systems supporting it, with its related financial processes. A data warehouse is a central repository that stores historical data in a format that is optimized for analysis and reporting. The use of specialized software tools is required, and SAP is one of the leaders in the industry. BW/4HANA is the data warehousing solution from SAP, specifically optimized for HANA, their database management system (DBMS).
This document presents the work carried out during the internship at PwC Italy, for one of the largest Italian fashion groups: in particular, it describes how the project is organized, the main objectives, the initial state of the system, and the detailed description of the work done.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the current digital world, business intelligence is crucial to make informed business decisions. Data warehouses are often the backbone of business intelligence systems, deriving insights from large amounts of data. Thus, the designed solutions must provide an efficient data management.

The PwC Enterprise Process Management (EPM) team, specialized in data warehousing solutions for planning and reporting purposes, has been engaged by one of the largest Italian fashion groups, which for privacy reasons, is simply referred as "*FG*". The collaboration between the two parties began during the COVID-19 pandemic, optimizing and migrating a variety of financial processes, to a new single platform. standardizing the data format.

After the migration, FG requested the design and implementation of the Closing and Consolidation process and the design and development of the stand-alone WBS Planning process. The goal of FG with the Closing and Consolidation process, is to produce single companies and Group financial statements. These documents keep track of the companies and group trends and are presented to the board of directors, to plan the budget for future fiscal years. The Closing and Consolidation process is spit into two distinct sub-processes, with this work focusing on the Consolidation aspects. The goal of FG with the WBS Planning process, is to allocate yearly budget for job orders. The process consists of two main steps: there is an initial budget allocation, then, in the middle of the fiscal year, the initial budget can be revised.

A reporting phase allows to check the budget trend with respect to the true values.

Initially, within FG, several companies used different tools for overlapping processes. The adoption of a single EPM platform (i.e. a single warehousing system) to work on multiple processes, acting as "single source of the truth", helps removing the redundancy caused by the upload on different systems. It has been opted for SAP BW/4HANA as the warehousing tool. This choice, followed the fact that FG already employed some SAP tools, such as the HANA database and the SAP ECC ERP system, so it makes the integration with different components easier. Furthermore, FG is considering to upgrade the ERP system to SAP S/4HANA. In this way it would be to have a real-time connection between SAP S/4HANA and the warehousing system, avoiding data transfers, and reading data on the fly, directly from the ERP system. FG wants to keep this option open.

SAP BW/4HANA provides all the components to set up the infrastructure for the various processes, managing the data and configuring multiple planning and reporting steps. Process Chains automate the Extract, Transform, Load (ETL) pipeline to format the data of the two processes in the desired format, while Planning Sequences are employed to structure the execution of complex planning routines, manipulating and generating new data. Planning Sequence are for example employed to manage cost allocations (i.e. redistribution of costs), and intercompany elisions (i.e. generation of new records to write off intra-company trades).

This work presents the key aspects of the applications developed for these two processes, highlighting the importance of having an efficient EPM solution, that helps support the BI aspects. The design decisions and implementation of the WBS Planning process successfully met the FG requirements, receiving positive feedback. The application is already deployed and running in the production environment, and has already been adopted for the first planning cycle. Also the Closing and Consolidation process successfully met the FG requirements. The new Closing process revolutionized some of the key aspects, and required a lot of testing time from the financial users. Only some of the key features are already used in the production environment, such as the new allocation engine, which new design and implementation results a lot faster than the older version. At the time of writing, however, the

Consolidation part of the process is yet to be tested.

From the technical standpoint, the two applications successfully achieved an efficient management of large data volumes, optimizing the design of the process workflows. Furthermore, the end users are provided with a centralized tool to produce requested financial outputs. From the development of the two applications, emerged the significance of the EPM system to provide smoother and efficient process flows, lowering the FG manual effort, and improving the user experience during the execution of the different financial processes. Limiting the manual steps of the users, keeping it only to the necessary aspects, and avoiding overly-complicated logics, coupled with the technical aspects of SAP BW/HANA allows to maximize the automation, producing a positive impact on the business operations.

After an overall introduction, Chapter 2, introduces the involved parties, namely PricewaterhouseCoopers (PwC), one of the most important consultancy firms worldwide, engaged by one of the largest Italian fashion groups, to work for the design and implementation of a data warehousing system. PwC and the client established a Non-disclosure agreement (NDA), so the client will be simply called with the fictional name "*Fashion Group*" *(FG)*. After the presentation of the involved parties, the chapter gives an overview of the project with its requirements, starting from the initial state of the system, the new processes to implement and the overall goals.

Chapter 3, presents the technological stack involved, which includes the central warehousing tool, SAP BW/4HANA, and all the supporting technologies on which the application is based. It first introduces the DBMS on which the warehousing system relies, then the ERP system that provides the starting data, and finally the tool used for dashboarding. Furthermore, it presents another last tool, which has been used to deal with the configurations aspects, such as the user access control.

Since SAP BW/4HANA is the adopted data warehousing tool, Chapter 4 details its major components. This section focuses on the data modeling aspect, describing the components necessary for structuring the data, starting from InfoObjects, used to represent the characteristics of the data and ADSOs, which are used for the physical transactional data storage. Then, the chapter introduces the necessary elements to build and manage effective data flows. These elements can be exploited to import data into the warehousing system from external sources, or to move data between structures within the SAP BW/4HANA environment. Finally, Planning Sequences allows the end-users to perform complex calculations and data manipulations in a controlled manner. The components presented in this chapter are needed to better understand the implementation details, provided in the later chapters.

Chapter 5 presents the technical aspects and implementation details of the Consolidation process. The Consolidation process depends on the Closing process, so it first presents the data model, keeping in mind that the two process shares the majority of the characteristics of the data. Are then highlighted the major InfoObjects relevant for the Consolidation. The Consolidation process does not have a specific data flow, which is implemented for the Closing, but the data is manually imported into its structures through custom calculations. This section gives a general idea on how the data coming from the ERP system are managed and elaborated, through the data flow, discussing the relevant points for the Consolidation.The next section presents how the application is designed and developed to follow the process flow, focusing on the SAP BW/4HANA Components (Queries and Planning Sequences) and the dashboards developed in SAC. Finally, the last section describes how the securities and the data access are managed with SAP BPC.

After the Consolidation Process, Chapter 6 details the WBS Planning process and application, with some consideration on the implementation. The goal of this process is to plan the job order expected expenses, and compare the planned (budget) amounts with the actual amounts. First, this chapter introduces the data model, with the main characteristics employed to structure the data. After that, are presented the main architectural aspects and how the data flow is designed, in order to manage master data and transactional data. The WBS Planning process is executed two times for every fiscal year, with the second time consisting on a review of the budget, usually after the first six months. After the data flow, this chapter describes the major details concerning the technical flow of the process and the corresponding implementation details. Finally, it shortly discusses how the security and the data access are managed.

The final chapter, Chapter 7, reports the major considerations on the work done, further discussing the focal aspects and the next steps.

1.1 Contributions

The major contributions are reflected in the work done to structure the Consolidation process and the WBS Planning process as applications, detailed in Chapter 5 and Chapter 6.

Following the Business Blueprint (BBP), a document specifying all the functionalities to develop, for the Consolidation process, I started creating the necessary components to model and store the data in the warehouse system. Recalling that the Consolidation depends on the Closing process, this components consist of the missing InfoObjects to represent the data characteristics, and the ADSOs to store the transactional data of the process. Furthermore, I also implemented the Composite Provider, and Aggregation level on top of it, which are respectively used for reporting purposes and to store new data, through custom calculations, developed using ABAP and SQL.

Following the process flow, I implemented a custom calculation importing the budget data and the actual data, from the Closing structures to the Consolidation environment. Then, I modeled a query to check the imported data.

The Consolidation phase consists of two major steps, namely, *currency conversion* and *intercompany elisions*. Since the Consolidation process consists in creating and analyzing the results of the entire group, there is the need to rewrite all the financial operations, which are also performed in different countries, into an homogeneous view, with predefined currencies. From this view are then removed the intra-group transactions, i.e. the transactions involving two companies within the group. For this two steps I developed two custom calculations and two queries, to present the data obtained from the corresponding steps to the financial users. At the end of the Consolidation phase, a reporting phase allows to check the results of the process. I built one *profit and loss statement* query for the *RETAIL*, *INDUSTRIAL*, *AGGREGATED* division, organizing the data into specific categories that represent the type of transaction. The query developed for the reporting phase are then imported into SAC, for the data visualization phase, where I developed different dashboards, highlighting the trends of the key performance indicators through various charts and plots.

Finally I worked more on the configuration aspects, defining the process flow in SAP BPC, where all the reports and requested functionalities can be accessed by the users through the web interface. Then, the data access is specified, by means of data access rules, along with how to manage the workstatus, selecting which InfoObjects are provided to the users to lock the data once confirmed and definitive. Furthermore I also prepared the user manual.

At the time of writing, the testing phase of the Closing application just finished, so the Consolidation application has not been tested yet.

The application supporting the WBS Planning process has been implemented from scratch, so I was involved in more aspects of the development. I was involved in the initial analysis phase, discussing the feasibility of the FG requests. From that, I started working on the data model, implementing the needed structures, namely the InfoObjects to structure the data and the two ADSOs to store the actual data and the budget data.

Once created the various structures, I was involved in the design and implementation of the data flow. Specifically, I developed the different derivation routines for the master data and to import the transactional data into the WBS structures. One of the most important routines is used to build the *hierarchy* of the job orders. In the later stages of the developments, I have employed this hierarchy in various aspects of the process implementation, from the modeling of the queries, to the setup of the DAP and the workstatus. Then, I organized the various components of the data flow into a process chain, scheduled to run at night, keeping the master data and transactional data up to date.

The WBS Planning is executed two times every fiscal year, one in June and one in February, with the second one revising the budget initially estimated. I contributed on the logic to manage the data from this two different planning periods. The main idea is that if the budget for a specific job order is revised, the system stores a record representing the delta between the revised budget and the initial budget. This helps avoiding to overwrite the records containing the initial budget. Furthermore, in this way, the data is stored in a format which allows to perform an analysis on the deviation between the July and the February budget.

Following the process flow, I configured an import task for the end users, to import the already planned job orders before the go-live through a *csv* file, and then modeled a query to show the data imported into the system. Next, in the setup phase, there is the mapping step, which allows the process manager to activate new editable cells for the compilers. This step is fundamental for example, when a new job order is created and it has no actual data, or when a new Vendor is employed in an already existing job order. Since the 0-value records are not shown and it would not be possible to insert the planned amount. I achieved this behavior with an input query, where the users can enable a new record by the combination WBS-Cost Element-Vendor and flagging it as active (writing *'1'* in the flag field). If this procedure is done correctly then a new line, corresponding to the new activated record, is added in the table of the planning phase.

After the setup, there is the crucial phase of WBS Planning. The planning cycle is organized in two phases: initial budget planning, and revision of the budget. For the initial budget planning, I modeled an input query, that allows the compilers to store the job order budgets by fiscal year and to carry-over the unspent budget from the previous planning cycle, consequently writing off this amount from the past budget. I managed the write-off mechanic with a copy planning sequence, which is configured to be triggered when a record representing the carry-over is stored into the physical structure (ADSO) containing the job orders budget data. The planned budget of the previous planning cycle is not overwritten: the various reporting queries are designed to give an aggregated view considering budget, carry-over and write off.

For the REVISED budget phase I modeled an input query, pre-populated with the values inserted during the June phase, i.e the sum of initial budget and carry-over. When the compiler inserts a new budget value, the value initially stored is not updated, but the system generats the delta between the original value and the new value. I set up this behavior through a query feature called *inverse formula.*

A specific InfoObject (MIDTS) allows to distinguish the data belonging to different phases. It is used to manage the input and the reporting phase. To consider only data from the July phase (initial budget), the MIDTS InfoObject is filtered to include the values INPUT, CARRY OVER, WRITE OFF. Instead, if there is the need to consider also the revision phase, it requires to consider also the DELTA among the other values of the filter. In this way, the organization of the data from the planning cycle allows an easy data management and query design, and keeping in mind eventual future developments.

Next, I imported the planning queries into SAC to be available to the compilers. I built two *stories*, one for the June phase and one for the February revision phase, integrating the imported queries with some charts and plots to compare planned (budget) amount with the aactual amount, and see the trends of the major key performance indicators. Furthermore, following the same logic I developed also two reports shaped in the same way. The reports contain a custom expandable structure, which categorizes the type of Vendor into company within the group, job orders developed internally to the reference Company, and external providers.

The last step of the Consolidation process allows the process managers to select which job order to download to the OPEX application, and launch such download. For this step, I developed an input query with the same logic of the activation step in the Setup phase. Furthermore, from this query allows to specify a destination cost center, detail needed in the OPEX application. The download to the OPEX application is executed through a custom planning sequence, for which I implemented a custom calculation, using a mix of ABAP and SQL. The calculation derives some of the fundamental characteristics, which are not used in the WBS Planning application, but are required in the OPEX application. For example the WBS budget is planned with respect to the fiscal years, but the calculation extract the fall-winter and spring-summer season with the corresponding months. Then the budget is evenly split into the months of the seasons, to have an homogeneous representation with the OPEX data. All the queries and custom calculations of the process are either imported into Excel or SAC, to be available for the end users.

Finally, as for the Consolidation application, I worked on the configuration aspects, making available all the developed functionalities to the financial users from the SAP BPC web interface. Next, I configured the data securities by means of data access rules, and how to manage the workstatus. Furthermore I also prepared the user manual.

After the implementation I followed the testing phase. The application is already deployed and running in the production environment.

Chapter 2

Project Overview

This chapter provides a short illustration of the project and the work done. First it briefly introduces the involved parties, i.e. PwC and one of the largest Italian fashion groups, which, for privacy reasons, wishes to remain anonymous. These two entities collaborated for the development of a data warehouse system, with PwC working on the solution following the Fashion Group (FG) requests. The chapter, then, gives an overview of the project, highlighting its initial state and critical points, with insights into the strategic choices, followed by the newly implemented processes.

2.1 Involved Companies

2.1.1 PricewaterhouseCoopers (PwC)

PwC is a multinational professional services network. It is one of the "Big Four" accounting firms, but provides a wide range of professional services, including audit and assurance, consulting, tax, and legal services, to clients in various sectors, such as financial services, technology, healthcare, and energy. PwC was formed in 1998 by the merging of two accounting firms, Price Waterhouse and Coopers&Lybrand and now operates in over 157 countries with a network of more than 328.000 employees.

As just mentioned, PwC is a network of firms, and the main entity involved for the project is the Financial Transformation team, operating under the Italian

PwC branch. This team offers technology consulting services paired with financial consulting services to improve the client business processes. The goal of the team is to enable organizations to become more efficient, and effective in managing their finances, while also improving the technological aspects.

2.1.2 Fashion Group (FG)

Client who issued the warehousing project. As already mentioned, for privacy and security reasons, this entity will be referred as "*Fashion Group (FG)*". Furthermore, all the FG information are replaced with fictional one to avoid the leak of sensitive data.

FG consists of several companies providing different services, which are categorized into three different divisions:

- **Industrial**: designing and manufacturing of clothing and new materials.
- **Diffusion**: distribution of clothing and materials
- **Retail**: selling to the end-customers through physical and online stores.

It is worth noting that there are also intra-group exchanges, i.e. transactions between companies within the group. For example, an "*Industrial*" company could buy materials from a "*Diffusion*" company.

2.2 Initial Project Status

The collaboration between the two parties began during the COVID-19 pandemic, migrating a variety of financial processes, with the related old systems, to a new single platform. Furthermore the client requested the extension of the migrated system functionalities, implementing new custom capabilities and improving the existing ones. The goal of FG is to keep improving the efficiency and effectiveness of the future processes and EPM (Enterprise Process Management) systems involving strategic functionalities, such as Planning and Consolidation. The migration was not a simple "copy and paste", but the idea was to evolve the implicated processes, for example standardizing the data format and avoiding data redundancy, for a future-proof solution, considering the future implementation of a new ERP system based on SAP S/4HANA (as briefly explained in the end of Section 3.2). The main advantage would be to have a real-time connection between SAP S/4HANA and the warehousing system, avoiding data transfers, and reading data on the fly, directly from the ERP system, however, this topic is still under evaluation.

Back to the actual work that has been done, the major benefits of having a single EPM platform (i.e. a single warehousing system) to work on multiple processes, are the data consistency between those different processes, with a “single source of the truth” and the optimization of the involved systems. Furthermore, it is also beneficial for the end financial users, reducing the number of tools that they have to use and learn.

The Financial budgeting and Planning represents the overall financial position and goals of an organization. This practice creates a baseline to see how actual results vary from expected performance. Initially, within FG, each company used different tools for the same process, but, as already mentioned, FG wanted to simplify the data flow and the number of steps to complete the planning process. The main concern was to eliminate the data redundancy caused by the upload on different systems. As result of the design, FG and PwC decided to opt for a single system to perform different processes: distinct process flows based on the aspect of the planning phase.

The different planning applications make use of actual data and budget data. Actual data refers to the true transactional data, recorded through the ERP system, while the budget data refers to the values planned by the users.

The processes already implemented in the new warehouse environment are:

- **OPEX budgeting**: the term OPEX (Operating Expense) refers to the costs that a business incurs in order to maintain its day-to-day operations and generate revenue, for example employee salaries, advertising expenses and utilities like electricity, water, and internet services. The related application provides an automated procedure to plan the various expenses through the SAC web interface.

- **Sales and Margin**: the related planning and reporting application gives a view on the profit made thanks to the sales of products, considering all the expenses related to the cost of goods, i.e. materials, labor, and so on. This analysis can be done at different levels of granularity.

- **Indirect Costs Workflow (WFI)**: the related reporting application, among others, allows the interested users to have a clear and immediate vision on the approved or to be approved requests and purchase orders related to the indirect costs. This application consists only of reporting, so it is based only on actual data, no planning functionalities are involved.

Each process is further divided in versions according to the company division, hence each process is slightly adjusted depending on the division. Furthermore, in each process, thanks to SAP BPC, a Business Process Flow is defined, guiding the user through the steps of the process in question. SAP BPC allows also to manage the data access, defining which user has access to which data.

The processes just listed are out of the scope of this work, but they demonstrate an effective and well-established technological landscape, which will be exploited also for the new processes to implement, introduced in the next sections. The major developments are on the data warehousing side, but the technological stack comprise:

- **SAP ECC - ERP system**: tool for management of day-to-day business processes, which provides a continuously updated view on transactional data.

- **SAP BW/4HANA - EPM System**: warehousing system which provides budgeting, forecasting and financial management capabilities.

- **SAP BPC and SAC - User Applications**: software applications which interface the users with the underlying warehousing system.

For a more detailed view on the used tools refer to Chapter 3.

2.3 Consolidation Process

The Consolidation process involves combining the financial statements of subsidiary companies into a single set of financial statements for the entire group. The Financial Consolidation is part of the management Closing process, and hence part of a larger homonym application. The management Closing is a financial and accounting process that organizations usually go through at the end of their fiscal year, to evaluate the past performance and to plan the choices of the next fiscal year. The results are then reported to the board of directors.

The main difference between the Consolidation and the Closing process are the outputs produced: the Consolidation aims at giving an overall view on the group, while the Closing aims at providing the financial details of every for every single entity within the group. Furthermore, in this case, the Consolidation is performed by a single user, while the management Closing is performed by different controllers for each company.

The overall activities of the management Closing (including the Consolidation subprocess) are:

- **COGS re-evaluation**: Cost of goods sold (COGS) refers to the direct costs of producing the goods sold by a company that are purchased by customers. This amount includes the cost of the materials and labor directly used to create the good. COGS differs from OPEX, in that OPEX includes expenditures that are not directly tied to the production of goods or services, such as distribution costs. This step first allows to give value to the *standard* production costs of a specific item, and then, by recomputing the standard COGS with *actual* unit costs, to perform a variance analysis between standard COGS and actual COGS. The standard costs are predetermined at budget, before production. They constitute a target, that the enterprise wants to achieve, and they are used as a benchmark for the actual costs. The result of this step provides an indicator, to assess the financial performance of the companies within FG.

  This process is organized in the following steps:

- Import of the actualized costs;
- Computation of standard COGS;
- Re-evaluation of standard COGS using actual unit costs;
- COGS variance analysis between standard and actual total costs to produce the same quantity of items.

Let's see an example for a single model. In the real world the process is performed for every model. Table 2.1a shows that two types of standard costs (for example material and labour) affect the production of a specific quantity of $Model_{01}$ items. Then it is possible to derive the standardized unit cost. Suppose that at a certain point during the fiscal year, the cost of the material to produce the model of a particular clothing item increase. The actual unit cost increases, and, hence, also the total cost to produce the same quantity of $Model_{01}$ items increases, see Table 2.1b. Finally, the variance between the total standard COGS and the total actual COGS is used as efficiency indicator.

	$\boldsymbol{Cost_1}$	$\boldsymbol{Cost_2}$	$\boldsymbol{Total}$	$\boldsymbol{Qnt}$	$\boldsymbol{Unit\ Cost\ STD}$
$Model_{01}$	10 €	25 €	35 €	7 pcs	5 €

(a) *Standard Costs*

	$\boldsymbol{Unit\ Cost\ ACT}$	$\boldsymbol{Qnt}$	$\boldsymbol{Total}$
$Model_{01}$	6 €	7 pcs	42 €

(b) *Actual Costs*

	$\boldsymbol{Delta\ Total\ Cost}$
$Model_{01}$	7 €

(c) *COGS Variance*

Table 2.1: *COGS re-evaluation procedure - Table 2.1a shows that the production of a specific quantity of $Model_{01}$ items is affected by two standard costs. Considering the total of these two costs, the next step of the process is to derive standardized unit cost of $Model_{01}$. If at a certain point during the fiscal year, the unit cost of $Model_{01}$ changes, also the total cost to produce the same quantity of $Model_{01}$ items is affected, as shown in Table 2.1b. The COGS re-evaluation procedure shows how a variation on the unit costs affects the total costs of the sold items, furthermore the variance between the total standard COGS and the total actual COGS is used as efficiency indicator, to compare the planned trend and the true trend.*

- **Cost Allocations**: usually most of the data records have maximum level of details, however, in some cases this is not true and the end user must allocate some of the missing details manually. For example, consider the costs of running an e-commerce. They are not specific of a brand or a country, so the costs are allocated according to user defined logics, based on what should be allocated, on which dimension. This step is broken down into 3 parts:

- *Working Version Creation*: snapshot of the actual data, which is then used in the allocation process. This step allows to keep the independence between the true data and the data processed by the final users.
- *Allocation Rules (Drivers) Definition*: creation of mapping rules to extract the percentages for the next step. In other words which data to use to generate the percentages.
- *Allocation Percentage Generation*: based on the rules entered in the step previous step, are calculated the allocation percentages, which are then applied to compute the actual cost allocation.

Let's consider the following example to better explain the allocation computation. Assume the following scenario: the user responsible for the Closing of the company M005 wants to allocate the costs of dimension Country for a certain set of parameters, such as fiscal year and month. The data produced by this set of parameters may be as displayed in Table 2.2a, where the data to be allocated is highlighted in gray. The symbol '#' specify that no country is assigned. Then, suppose that a single driver is defined, which indicates that the country costs should be allocated according to the country revenues. Table 2.2b provides an example of data which may be produced by the driver. Next, the percentages are computed as the weight of each country over the total revenues. Finally the amount to allocate is distributed to the various countries, according to the computed percentages, as shown in Table 2.2c. In the real applications, the classification of *costs* and *revenues*, is more fine grained. Using a characteristic called *Profit&Loss Line* (*P&L* Line), costs and revenues are classified based on financial operations. Furthermore, there is also a different type of allocation. Instead of allocating only the 'not assigned' values, all the values in the Table 2.2a would be redistributed, using same percentage

Country	***Costs***
C_{01}	10 €
C_{02}	20 €
#	80 €

(a) *Data to Allocate*

Country	***Revenues***	%***Revenues***
C_{01}	30 €	20%
C_{02}	20 €	13%
C_{03}	30 €	20%
C_{04}	70 €	47%
Total	150 €	100%

(b) *Drivers*

Country	***Post Alloc Costs***
C_{01}	10 + 16 €
C_{02}	20 + 10.4 €
C_{03}	0 + 16 €
C_{04}	0 + 37.6 €

(c) *Allocation Results*

Table 2.2: *Cost allocation procedure - Table 2.2a displays the costs by county, where '#' represents the costs that need to be allocated (no country assigned). These costs are allocated following the drivers from Table 2.2b. The costs to allocate are split according to the revenue impact by country on the overall revenues. Finally, Table 2.2c displays the new cost by country after the execution of the allocation.*

logic given by Table 2.2b. In this way the original values in Table 2.2a are overridden.

- **Currency Conversion**: convert the aggregated view of all the companies to predefined currencies (in this case USD, GBP, EUR), to have the final group view in different currencies.
- **Intercompany Elisions**: the goal of this point is to write off the exchanges between companies within the group from the final view, highlighting the extra-group exchanges.

For an easier management the Closing and the Consolidation are treated as two different steps, and at the end of the two steps there is a reporting phase, enhanced with various dashboards implemented in SAC. The tasks specific of the Consolidation phase are currency conversion and intercompany elisions, and the corresponding application was implemented to compare the actual data with the budget data, over a specific period of time, to evaluate the performance of FG.

Further details about the Consolidation process and application are provided in

Chapter 5, while the Closing process is out of the scope of this. Are only provided the notions that also affect the Consolidation phase.

2.4 WBS Planning Process

A WBS (work breakdown structure) is a hierarchical deconstruction of a project into smaller components, for example by scope. This process, and hence the application, aims at planning the budget of such smaller components, which identify different job orders. Then, a reporting phase allows to see how the actual costs compare with respect to the planned costs. The job orders are hierarchically organized by company, and parent project, and for each parent project a single responsible person is identified.

The application is specific for the budgeting of IT job orders. There are three different families of IT job orders, where the budget collection is carried out:

- **Implementation/Projects (IM)**, e.g. new projects.
- **Operations (OP)**, e.g. licenses.
- **Evolutions (EV)**, e.g. post release new client requests.

The budget for the current fiscal year is gathered in two different moments, July and February. The first budget collection is performed in July, then, in February, a budget review is done. Next, depending on the job order, the budget collected needs to be integrated in the OPEX application (not every job order is going to be considered as OPEX).

The budget is compiled by the responsible in two different steps. First, the compiler enters the total budget allocated for each WBS. Afterwards, the compiler enters the planned expected quota for the current year. After the input phase, the local controllers of the various companies, select which job orders to download on the OPEX application, and they will follow its process flow.

The main technologies involved are SAP BW/4HANA to model the input query, then SAC, with a live data connection to BW/4HANA to insert the budget. Furthermore, SAC is also used to build dashboards for the reporting phase.

Chapter 3

Technologies Employed

This chapter aims to provide an in-depth understanding of the key technologies employed, highlighting the significance of these tools and their functionalities. Furthermore, it explores the interplay between these technologies, from the database management system to the data analytics platform.

3.1 SAP High-Performance Analytic Appliance (HANA)

SAP HANA is the database platform developed by SAP and adopted by the customer. It is designed to process large volumes of data in real time, keeping in memory large volumes of data, thus return the results of queries in fractions of seconds. In practice multi-temperature data is used.

Some key aspects of SAP HANA are:

- **In-memory Computing**: unlike traditional databases that store data on disk, SAP HANA stores data in memory, which allows for extremely fast data processing and retrieval. By keeping the data in memory, HANA eliminates the need to access it from slower disk-based storage systems, resulting in significantly improved performance.

- **Multi-temperature Data**: data is divided by frequency of access. In business applications, recent information must be readily available and it is usually more requested than old information. With this in mind, hot data represents the information frequently queried and it is stored in the RAM memory, while cold data is rarely queried, and stored outside SAP HANA, hence resulting in a longer retrieval time. One of the SAP solutions for external storage is SAP IQ which enables extreme data compression. The last category is warm data, where records are store in SAP HANA, but not in primary memory.

- **Hybrid Data Storage**: combination of column-based and row-based storage techniques within the SAP HANA DBMS. This hybrid approach take the strengths of both storage methods, optimizing performance and efficiency for different types of data processing and analytical workloads. Column-based Storage is used for large tables with many repeated values, that go through bulk operations. Queries that involve aggregation, filtering, and reporting benefit from column-based storage because it allows for faster access to specific columns needed for the query. Row-based storage is used for data that is frequently updated or accessed individually, such as transactional data. It allows for faster access to complete rows of data and it is ideal for scenarios where real-time processing is critical. In addition to column-based and row-based storage, SAP HANA employs a delta storage concept to efficiently handle changes to data. When data is updated or inserted into the system, the changes are not immediately written to the main storage. Instead, the changes are temporarily stored in a separate area called Delta Storage, which is optimized for high-speed writes.
  This flexibility enables SAP HANA to support a wide range of use cases, making it a versatile platform for data-driven businesses.

SAP HANA integrates with a wide range of structured and unstructured data from various systems and sources, including the SAP applications introduced below, adopted by the client. SAP HANA is available in both, on-premise and cloud.

3.2 SAP ERP Central Component (ECC)

SAP ECC is the core ERP system adopted by the client. It is a business application suite, comprehensive of various modules that support key business functions within organizations. The software provides a centralized way for managing and automating such business processes. SAP ECC contains all the transactional information of the company, that is, every economic transaction such as invoices, requests and purchase orders.

Some of the most important modules and features of SAP ECC include:

- **Controlling (CO)**: comprise various controlling functions related to cost and revenue management;
- **Controlling Profitability Analysis (COPA)**: sub-module within CO, that is specifically dedicated to the profitability analysis of the various business segments;
- **Financial Accounting (FI)**: manages financial transactions;
- **Materials Management (MM)**: handles procurement and inventory management;
- **Human Capital Management (HCM)**: manages personnel administration, payroll, and organizational management.

The main modules utilized by the client are FI, CO, COPA.

Over the years, SAP ECC has been a widely adopted ERP solution by organizations of various sizes and industries worldwide, providing them with a robust and scalable platform to manage their business processes. Since the support of SAP ECC is going to end in 2030, IFG is planning the migration to the newest SAP ERP platform, namely S/4HANA.

Some of the key performance-wise and feature-wise improvements of SAP S/4HANA are:

- **In-memory Computing**: SAP S/4HANA is built on the SAP HANA in-memory database platform, whereas SAP ECC uses traditional disk-based storage. In-memory computing allows faster data processing and quicker access to information.

- **Enhanced User Experience**: S/4HANA provides a modern and intuitive user interface through SAP Fiori, an user experience (UX) framework, making it easier for users to navigate, access information, and perform tasks efficiently.

- **Embedded Analytics and Machine Learning**: users can perform predictive tasks, directly within the ERP system, reducing the need for separate analytics tools.

3.3 SAP Business Warehouse for HANA (BW/4HANA)

SAP BW/4HANA is the Data Warehousing System employed by the client, designed to run on the SAP HANA in-memory database platform. SAP BW/4HANA is specifically optimized for SAP HANA and leverages its in-memory computing capabilities to process large volumes of data, resulting in fast data loading and transformation processes. SAP BW/4HANA integrates with other SAP applications as well as non-SAP systems. This allows organizations to consolidate data from various sources and gain a comprehensive view of their business. This is also one of the goals of the client: arrange all the companies within the group to have consistent data processes, providing a unified view of business operations.

For the project, SAP ECC has been used as the ERP source system, and with the use of built in extractors, it allows to transfer data from the SAP ECC to the SAP BW/4HANA infrastructure. On chapter 4 are presented more in details the major design components of SAP BW/4HANA.

3.4 SAP Analytics Cloud (SAC)

SAC is a cloud-based business intelligence and analytics platform. It offers the flexibility of cloud-based deployment, enabling rapid implementation, scalability, and accessibility from any device. SAC provides a comprehensive set of tools for planning and reporting and allows users to connect, visualize, and explore data from various sources in real-time.

Some of the key features of SAC include:

- **Data Visualization**: SAC allows users to create charts, graphs, tables, and geo-maps. The interactive nature of these visualizations enables users to explore data, and gain valuable insights.
- **Planning and Budgeting**: SAC offers advanced planning capabilities, allowing users to create and manage financial plans, budgets, and forecasts. Collaborative planning features enable multiple users to work together in real-time on planning processes.
- **Integration**: SAC integrates with other SAP products, such as the warehousing tool SAP BW/4HANA. This integration enables users to access real-time data directly from these systems, for analytics, planning and reporting.

For this specific work, a live connection from SAP BW/4HANA sources is used, allowing real-time data reporting/visualization with interactive dashboards and budgeting. Users can interact with SAC through a web-based interface.

3.5 SAP Business Planning and Consolidation (BPC)

SAP BPC is a central platform designed to assist organizations streamline financial planning, budgeting, forecasting, and consolidation processes across various entities and subsidiaries. This platform is extremely customizable, especially for more complex infrastructures.

SAP BW/4HANA queries can be imported into Microsoft Excel, then, thanks to the SAP BPC Business Process Flow, the end-users have a single access point to all the reports (either SAC or SAP BW/4HANA) related to a specific process. The BPFs define the sequence of steps involved in various planning, budgeting, and consolidation processes, providing a guided approach to executing these processes. Furthermore, a workstatus mechanism defines different stages for the data, such as "*In Progress*", "*Submitted*", "*Approved*" or "*Locked*". Each status represents a specific phase of the planning or consolidation cycle. This mechanism can enhance data governance, preventing unauthorized changes, and ensures that the data undergoes the appropriate approval and review processes before being finalized.

Major functionalities of SAP BPC relevant for the project are:

- **Workflows and Collaboration**: the system supports collaborative planning and consolidation processes by enabling workflows and approvals (thanks to the workstatus). Users can collaborate across departments and roles to ensure data accuracy and consistency in the planning and reporting cycle.

- **Data Integration and Connectivity**: SAP BPC integrates with various data sources, including SAP and non-SAP systems, to access and consolidate financial data from different parts of the organization. This ensures data consistency and reduces manual data entry efforts.

- **Data Security and Governance**: the solution offers robust data security and governance features to control user access to sensitive financial information. It allows administrators to define roles and permissions to maintain data integrity and comply with regulatory requirements (Data Access Profiles).

- **Real-time Monitoring and Alerts**: the system can trigger alerts and notifications based on predefined thresholds or events, allowing users to respond quickly to critical financial situations.

Chapter 4

SAP BW/4HANA Components

This chapter presents some of the various tools of the SAP BW/4HANA environment. These tools are especially relevant for the modelling aspect, the development of data flows and the implementation of various calculations, either native or custom.

4.1 Modeling

4.1.1 InfoObject

InfoObjects are the smallest components used to define and categorize data attributes, in a way that supports meaningful reporting and analysis. They can be seen as an "enhanced" field of the standard DBMS. They serve as the building blocks for creating InfoProviders (such as Advanced DataStore Objects), and queries used for reporting and analysis. InfoObjects define the structure and semantics of data within the warehousing system.

InfoObjects can be classified into different categories:

- **Characteristics**: represent descriptive attributes that provide context to the data. Examples of characteristics include company, product, and region. Characteristics help categorize and organize data.
- **Time Characteristics**: special type of characteristics specific to time-related aspects, such as year, month, week, and date. These are crucial for time-based

analysis and reporting.

- **Technical Characteristics**: this type of InfoObjects are managed by SAP BW/4HANA directly, and are used for technical purposes, such as the generation of timestamps.

- **Key Figures**: measurable data elements that represent business metrics or values. Examples of key figures include sales revenue and quantity sold.

- **Units**: represent the measurement units associated with key figures. For example, if a key figure is "*Sales Revenue*" the corresponding unit might be "*USD*" for US dollars or "*EUR*" for Euro.

InfoObjects also provide the *Master Data* option: this option allows to list the admissible values that a specific InfoObject can have. It is particularly useful for data consistency, and when dealing with aspects that have a limited set of possible values, such as currencies and the companies within the Group. Along with the admissible values, the Master Data option allows to define also the corresponding texts, used by the final users. Furthermore, SAP BW/4HANA makes every InfoObject with enabled Master Data, accessible like a normal table.

When the Master Data option is activated, InfoObjects can be further enriched with *Navigation Attributes*, represented by other InfoObjects. For example, the InfoObject representing the POS, could be enriched by the Country where the shop is located. When reading the InfoObject data structure, the attributes will be the columns of the table. One of the possible use-cases is to derive missing attributes of the records in the system: by looking at the POS attributes, the developers are able to retrieve the reference location and set up a report by Country.

Finally, InfoObjects can be organized into hierarchical structures to provide different levels of granularity for analysis. For instance, a product hierarchy might include product categories, subcategories, and individual products.

4.1.2 Advanced DataStore Object (ADSO)

As seen in the previous subsection, InfoObjects are used to define the characteristics, attributes, and key figures that provide context and meaning to the data. They serve as dimensions for structuring the data and are used as dimensions within ADSOs to categorize the data. ADSOs are the primary persistent objects used to store transactional and historical data.

In SAP BW/4HANA, the data storage of an Advanced DataStore Object (ADSO) is organized in a way that doesn't directly correspond to traditional relational database tables. Instead, an ADSO employs a unique architecture for storing and managing data, composed of three tables, according to the type of ADSO selected:

- **Inbound Table**: the Inbound table maintains information about the individual data requests that have been loaded into the ADSO. This table includes metadata about the requests, such as timestamps and technical information. Once activated, the data is moved to the second table.
- **Active Data Table**: the Active Data table represents the central data persistence table. It contains the activated data after a load request.
- **Change Log Table**: the Change Log stores historical changes to the data in the ADSO. It records all changes made to the data, including inserts, updates, and deletes.

Within SAP BW/4HANA are provided tools and interfaces to interact with ADSOs, such as the Query Designer or Data Modeling tools. These tools allow to define queries, perform data transformations, and create reports without being concerned with the underlying storage intricacies.

Building an ADSO requires to define all the InfoObjects that make up the data records, and the InfoObjects within the ADSO can be seen as the column of a table. Furthermore, the ADSO must have selected the characteristics to use as key. Depending on the type of ADSO, each dimension is used as key, or technical dimensions are exploite. Another important feature is that, if an ADSO goes under restructuring operations (hence changing the data format), a refactoring task is run on all the records stored, changing them to the new format, while maintaining the data consistency.

There are four different types of ADSO available in SAP BW/4HANA:

- **Standard**: most popular type of ADSO. It makes use of all the three tables, Inbound table, Active Data table and Change Log table. So, before reporting, a fundamental requirement is to activate the data, in order to have the Active Data table up to date.

- **Staging**: these ADSOs serve as intermediary step in the data integration and transformation process. Staging ADSOs are used to temporarily hold data, before it undergoes further transformations and is loaded into target ADSOs. In this case the Active Data table is not used, exploiting only the Inbound table. Usually it is not used for reporting.

- **Data Mart**: Data Mart ADSOs are designed to address the specific requirements of data marts, where data is tailored to the needs of a particular business function. Instead of the traditional insert, update, delete logic, Data Mart ADSOs can support either Insert-Only mode or Delta Updates. In *Insert-Only* mode, the Data Mart ADSO is designed to accept only new data records for insertion. Once data is loaded into the ADSO, it cannot be updated or modified. This mode is suitable for scenarios where historical data doesn't change, and the focus is on adding new data records. In *Delta Update mode*, the Data Mart ADSO supports incremental data updates. This means that can be loaded both, new data records and updates on existing records. The update is done through a new record, containing the difference between the original value and the new value (delta). Then, by aggregating the two records, it will be displayed the updated value. This mode is useful when need to keep your data mart ADSO up-to-date with changes in the source system.

- **Direct Update**: this type of ADSO is designed for real-time data processing scenarios, enabling immediate data availability for reporting and analysis. It does not use the Inbound table, so the Active Data table is directly updated on each load request.

Finally, from the ADSO's settings, there is also the possibility to enable the multi-temperature data storage.

As explained in the next subsection, ADSOs can be combined as a basis for Composite Providers, allowing the creation of comprehensive logical data models.

4.1.3 Composite Provider

A Composite Provider is a virtual InfoProvider that allows to combine data from multiple InfoProviders, such as InfoObjects, ADSOs, or other Composite Providers, into a single unified view, via UNION and JOIN operations. These operations are performed "on the fly" in SAP HANA, taking advantage of its in-memory processing capabilities, without the need to physically move or replicate the data.

Composite Providers are especially useful to create a single access point for reporting and analysis, regardless of the physical location of the data. This enables the end users to access data from different sources as if they were working with a single InfoProvider, avoiding the need to navigate through different sources separately. Furthermore, Composite Providers support also aggregations "on the fly", at different levels of granularity and navigating the various dimensions available.

Next, an example of use-case of a Composite Provider. Suppose there is the need to compare the *budget* data (i.e. the planned data) and the *actual* data (i.e. the true data) to derive some KPIs, and visualize these KPIs by Country in a geo map. Within the SAP BW/4HANA environment, the budget data and the actual data are stored into two different ADSOs, and a Country InfoObject stores the geographical coordinates. Then, the Composite Provider can be modeled as the UNION of the two ADSOs, JOINING the Country InfoObject.

For reporting purposes, it is good practice to use a Composite Provider even if there is a single underlying InfoProvider (i.e a single InfoObject or a single ADSO). Composite Providers do not allow data manipulation, since they are not physical tables within SAP BW/4HANA. For this reason these objects are specifically used for reporting and data visualization.

4.1.4 Aggregation Level

Given the virtual nature of the Composite Provider, it can't be directly used to manipulate data. For this purpose, an additional component, called Aggregation Level, is added on top of the Composite Provider. Thus, this additional component is specifically employed for manual input tasks by the final users, and for storing the data produced by planning sequences. Aggregation Levels represent a combination of dimensions, characteristics and key figures, that defines how data is stored. A single Composite Provider can have multiple Aggregation Levels on top of it, defined at various levels of granularity, to address different types of tasks.

4.1.5 Queries

When the source of a query is a Composite Provider, the query is the primary tool used for reporting, presenting the information in a structured and meaningful format. In this way the end users can also interact with the data stored, to perform various analytical operations. Instead, when the source of a query is an Aggregation Level, the query has still access to the underneath InfoProviders for reporting, but also allows the end-users to manually insert data records.

The Query Designer provides a graphical interface for defining query structures, selecting characteristics, key figures, filters, and calculations. Next, are listed and briefly explained the major query components:

- **Query Elements**: they consists of characteristics, key figures and calculated key figures. Characteristics provide context and dimensions for analysis, while key figures represent the values being measured. Calculated key figures enable the creation of new calculated values for reporting.

- **Query Structure**: it describes how to arrange the query elements in a meaningful way, determining the rows and columns. Furthermore, it allows to define some elements to be treated as "*Free Characteristic*". These elements are initially not visible, but are freely available to the final users, to enhance the basic query structure and explore such details if needed.

- **Filters**: used to restrict the data that is retrieved by the query, based on specific conditions. Filters allow to focus on specific subsets of data. Each Filter must refers to an InfoObject. Once defined in the modeling phase, Filters are fixed, and cannot be modified by the final users. For example, let's consider a query designed to analyse specifically the budget/planned data. The final user are not allowed change the "*Fiscal Year Category*" to actual/true data, otherwise they would be able to change the scope of the exploration. To allow the end-users to interact with the data source, SAP BW/4HANA provides other components, called Variables, or the already mentioned Free Characteristics.

- **Variables**: they allow users to select values dynamically at runtime, when executing a query. There are three major types of Variables: User-input Variables, Customer Exits and Security Variables. Each Variable must refers to an InfoObject. User-input Variables allow the end-users to manually delimit the data to retrieve. They can also leverage hierarchies defined in InfoObjects, to enable drill-down capabilities, allowing to analyze different levels of granularity. Customer Exits are particular types of Variables, that define the value of a characteristic without the direct interaction of the user. Such value is derived by a custom algorithm, which depends on the user selection of other Variables. Security Variables control which data the end-users can access, depending on the security configurations.

Queries can be imported into Excel, with the Analysis for Office (AFO) plugin developed by SAP. Then, thanks to SAP BPC, the Excel sheets are hooked to a Business Process Flow, where they can be accessed by the financial users.

On a final note, from the end-user point of view, queries can be considered as Pivot Tables, where rows and columns can be rearranged based on the necessity, and the available characteristics can be filtered as needed.

4.2 Data Flows

4.2.1 DataSource

A DataSource object (also called Extractor) is a component used to extract data from source systems and make it available for further processing within the SAP BW/4HANA environment. DataSource objects act as connectors that allow to retrieve data from various external source systems, such as SAP systems, non-SAP databases, and flat files. For this specific project, Extractors are specifically used to connect the ERP system (SAP ECC) to the warehousing system (SAP BW/4HANA). Any data structure represented by a table can be used as the source of a DataSource.

DataSource objects are an essential component for the data integration process, however they only extract the data from the source systems. To actually manipulate and transfer the data to SAP BW/4HANA InfoProviders, they are then combined with Transformations and DTPs.

4.2.2 Transformation

Transformations are a modeling tool used to convert raw source data into a structured and cleansed format, that is suitable for the target InfoProvider, using a rule-based logic. A Transformation is defined between a source object and a target object. Source objects and target objects can be DataSources, InfoObjects and ADSOs. When the data is sourced from multiple systems, Transformations help aligning the formats. Within Transformations can also be defined filtering rules, discarding unwanted data.

Through a graphical interface, the source fields are connected to the target fields, and then each field is coupled with the type of each assignment:

- **Direct**: field copied as-is.

- **Constant**: assign fixed value to target field.

- **Formula**: this type of assignment allows to have multiple inputs connected to the target and to apply basic data manipulations, such as concatenation.

- **Expert Routine**: used when there is the need to have complex manipulations. Expert Routines allow to write custom code, using ABAP, the SAP proprietary language. They dispose of all the Formula's functionalities, but with the possibility to retrieve additional data from InfoObjects or other sources, within the warehousing environment.

- **Start Routine and End Routine**: special cases of Expert routine. Start Routines are executed before the data transfers, so no Transformation rule is applied. On the other hand, End Routines are executed after the data transfers, so after the Transformation rules are applied.

Source and target can coincide, especially when dealing with InfoObjects, building hierarchies through Expert Routines. Transformations can be thought as the definition of rules, and when the actual data transfer is executed, such rules are applied to each field of each source record.

4.2.3 Data Transfer Process (DTP)

DTPs are used to extract data from source objects, such as InfoProviders or DataSource objects, to target InfoProviders, like ADSOs, within the SAP BW/4HANA system. Each DTP is associated with a Transformation, which, during the data transfer, applies the Transformation rules. DTPs can be configured to use parallel processing, optimizing the data transfer performance. They can also be scheduled, to run at specific times or intervals, automating the data integration process. Of course, there is the possibility to filter the source data, to select which data to extract, loading only the portion of interest.
There are two types of extraction:

- **Full**: this type of DTP extracts and loads all data from the source to the target.

- **Delta**: Delta DTPs extract and load only the new or changed data since the last extraction. The unchanged data is not be extracted. Delta loads are used to keep the target InfoProvider updated with incremental changes.

Overall, DTPs (in combination with Transformations) are essential components to efficiently move and manipulate data from source systems to target InfoProviders.

4.2.4 Process Chain

A Process Chain is a tool for automating and orchestrating a sequence of processes that are related to data extraction, transformation, and loading. Process Chains enable to create and manage end-to-end data workflows. Thanks to a graphical interface, a visual representation of the workflow, allows to define the logical data flow and track the progress of each task. Furthermore, Process Chains can be triggered based on other events, or can be scheduled to run at a specific time. They also include error handling mechanisms.

4.3 Calculations

4.3.1 Planning Sequence

A Planning Sequence is a set of defined steps and operations that provides a structured framework for the planning process, guiding the end-users on data manipulation. Planning sequences allow to specify a group of Planning Functions and execute them sequentially. Each Planning Function can be seen as one step of the entire process. For this project, Planning Sequences have been exploited for one, or at most two steps.

To define a planning sequence, are required the following components:

- **Function Type**: within each step of a Planning Sequence, the Function Type is used to define the type of task to perform. Specifically, it specifies the target Aggregation Level and which InfoObjects should be treated as parameters during runtime. Furthermore, each Function Type is associated with an ABAP class, which provides the implementation details of the operation to perform. The Function Type can be standard (directly provided by SAP), and hence coupled with a standard Planning Function, or a custom type can be created from scratch, coupling it with a custom ABAP class. During the developments, the latter option was adopted almost everywhere.

- **Filter**: it initially locks the portion of data to be overwritten, then monitors that the data produced by the corresponding planning function is within the specified scope, preventing incorrect data from being stored.

- **Planning Function**: Each Planning Sequence consists of a sequence of steps, each of which defines a specific operation or calculation to be performed. Each Planning function is matched with a Function Type to determine how to get the parameters (for example, through a User-input Variable) and execute the code.

Planning Sequences can be scheduled to run at specific intervals or triggered by events. However, they can also be connected to Excel sheets and triggered through macros by the final users. This is the approach adopted for the project.

Chapter 5

Consolidation Process

This chapter explaines more in depth the technical aspects and implementation details of the Consolidation process. First it introduces the data model, describing the relevant aspects of transactional data, with the corresponding InfoObjects. Furthermore, it gives a general idea on how the data coming from the ERP system are managed and elaborated, through the data flow. Then, this chapter presents how the application is designed and developed to follow the process flow. Next, it presents the tools that the users interact with, and how they are built using the SAP BW/4HANA Components (Queries and Planning Sequences). The majority of the developments were done using the Eclipse IDE. Finally it describes how the securities and the data access are managed with SAP BPC.

5.1 Data Model

As already mentioned in Chapter 2, the financial Consolidation can be seen as part of the management Closing process. For an easier management they are treated as two distinct processes, however the Consolidation data depends on the Closing data. This is also reflected in the Data Model, which shares a lot of dimension between Closing and Consolidation, but it is integrated with some InfoObjects specific for the Consolidation process. In Table 5.1 are presented the main InfoObjects used to structure the transactional data. Each InfoObject is used to represent a particular aspect of the data.

FG requested the possibility to perform the Consolidation process only on specific subsets of companies within the group. This is controlled thanks to the SCOPE InfoObject. Enabling the Master Data option, this InfoObject is enriched by a *Navigation Attribute* which specifies the target currencies of the process. Among the characteristic inherited by the Closing data model, there is the Intercompany, which identify the second company involved in a financial operation. This InfoObject is fundamental for the elision step.

Another InfoObject directly involved in the elision step is the *Profit* & *Loss* Line (*P*&*L* Line). A *P*&*L* Line is an accounting entity, used to classify the financial operations. FG provided how every P&L line should be treated. This aspect is modeled thanks to the MIDTS InfoObject, used as navigation attribute, which identifies the type of elision. Furthermore, MIDTS is used to distinguish data between the various process steps. The Data Model dimensions can be categorized into three macro groups: temporal, business and technical. Temporal and business characteristics are selected together with the FG controllers. Instead the technical dimension are employed to manage the implementation of the process.

Finally, a technical dimension called *Version* allows to distinguish the data coming from the Closing process, fundamental for the setup step.

5.2 Data Flow

There is no data flow specific for this process and its corresponding application. The Closing and the Consolidation process can be seen as a larger macro-process, but they are managed independently, dividing their Business Process Flows and functionalities. The idea is to have modular tool for the Consolidation process, keeping independent the structures between the two subprocesses. Even if the physical structures are separated, the Consolidation starts from data produced in the Closing. The output data of this process is distinguished thanks to Working Version InfoObject. As explained below, in Section 5.3, the data is imported through a custom calculation and it is not updated through a data flow.

Characteristic	INFOOBJECT	Description
Company	MICOMPCOD	Company within the group who
Scope	MISCOPE	Set of companies and target currencies
Intercompany	MINTERCMP	Trading partner involved in the transaction
Brand	MIBRAND	Brand of the material involved in the transaction
Division	MIDIV	division related to the transaction, e.g. *RETAIL*, *DIFFUSION*, *INDUSTRIAL*
Country	MICOUNTRY	Geographical information of the transaction
PL Line	MIPL_LN	Accounting entity used to classify the financial operations
Channel	MICHNTYP	Define the type of sales channels used by the company
Cost Center	MICOSTCEN	Business unit to which costs can be allocated
POS	MIWNPOS	Physical or virtual shops
Category	MICTGRY	Type of transactional data, i.e. *actual* or *budget*
Fiscal Period	0FISCPER	Fiscal Period (Month)
Fiscal Year	0FISCYEAR	Fiscal Year
Season	MISEASON	Particular characteristic of the fashion industry, i.e. *fall-winter* or *spring-summer*
Amount	MIAMOUNT	Amount of the operation represented by the record
Currency	0CURRENCY	Currency related to the AMOUNT
Data Source	MIDTS	Technical InfoObject to identify the step where the data belong within the process

Table 5.1: *Consolidation: Transactional Data Model - This table represents the main characteristics of the transactional data, relevant for the Consolidation process, with the corresponding InfoObjects. The majority of the data model is inherited from the Closing, which is then integrated with specific characteristics, such as the scope.*

In the warehouse environment there is one main data flow, that keeps up to date the data from majority of the applications, including the Closing, on which the Consolidation data depends. Table 5.1 displays a simplified version of the main data flow. The last ADSO MCBP2URC in the flow, stores the transactional data in the final form, as basis for the allocation engine and COGS re-evaluation. Then, two auxiliary ADSOs are populated through custom calculations: MCCO1CRV, specific for the COGS re-evaluation, while MCCO1URP hosts data for the allocation phase. In this way, the data flow, and the data manipulated by the users from this two phases of the Closing process, are kept separate. The data stored in the allocation ADSO is the starting point for the Consolidation phase.

Next are reported the focal points of the transactional flow from SAP ECC, which have an effect on the Closing and hence, on the Consolidation.

- **ECC → MDCO0PAMM**: the first step consists on retrieving the data from the ERP system into the SAP BW/4HANA environment. This is done thanks to a DataSource object, which allows SAP BW/4HANA to detect data from external systems. The target ADSO *MDCO0PAMM* is treated as staging layer, so the data is imported as-is, without further elaborations.

- **ECC → MICORATE**: exchange rate ADSO, used in the currency conversion for the Consolidation process.

- **MDCO0PAMM → MCCO1PAA**: the imported data is sparse, with a column representing each value field. In this step the data is transposed, to represent the value fields with one specific column. The different value fields represents costs, revenues and amounts.

- **MCCO1PAA → MCCO2PAA**: derivation of generic dimension missing from the data, such as the Trading Partner division.

- **MCCO2PAA → MCBP2URC**: derivation of specific dimensions for the Closing process, such as business area and market type. Furthermore, this step also computes an *year to date (YTD)* view on the data, which is the one used for the Closing and the Consolidation processes. The YTD is simply the cumulative value up to the corresponding date. So for example, the YTD of the specific cost x in at the end of March is $x_{YTD} = x_{jan} + x_{feb} + x_{mar}$. Fundamental step, since the Consolidation process is performed using YTD view.

The derivations and manipulations just listed are implemented by means of transformation layers and Data Transfer Processes (details in Chapter 4). These components are then scheduled into a Process Chain which runs every night, to automate the entire data flow, keeping the transactional data up to date.

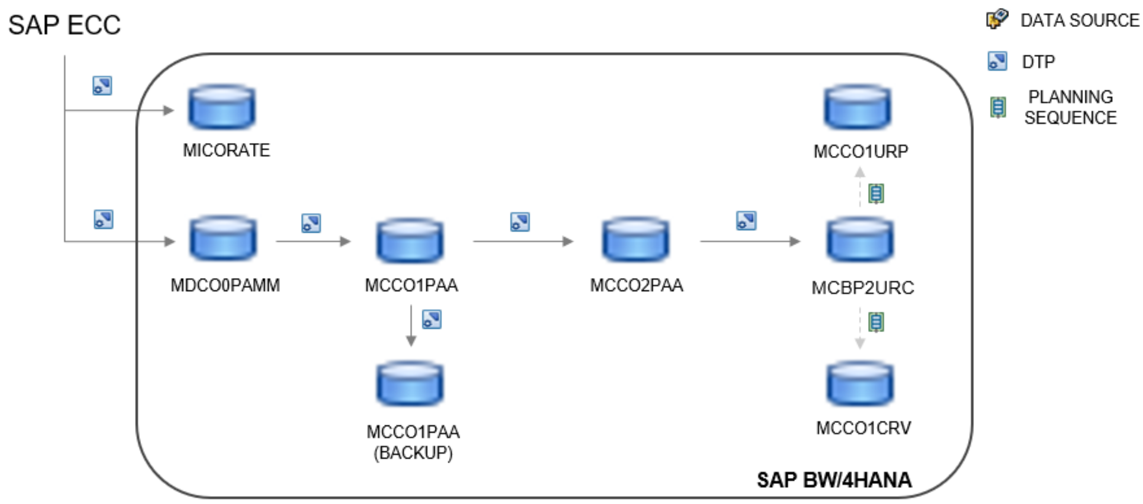


Figure 5.1: *Consolidation: Transactional Data Flow - Simplified schema of how the transactional data is moved from SAP ECC, the ERP system, into the warehouse environment. This flow moves and manipulates millions of records every night. It populates the structures involved in the majorities of the processes, including the Closing, which then affects the Consolidation aspects.*

5.3 Business Process Flow

This section provides further details about the Consolidation process with the most relevant implementation aspects. The technical flow of the process consists of Setup phase, Consolidation phase and finally, the Reporting and Dashboarding phase. All these phases are accessible through the specific Business Process Flow of the application, specified in SAP BPC, which redirect the user to the correct resources. This process is performed and made available to the controllers of the holding, that have access to all the data of the subsidiaries. The financial users are interfaced to SAP BW/4HANA, thanks to Excel and the AFO plugin. Then, at the end of the Consolidation process are developed some dashboards using SAC, to highlight the trend of the most important key performance indicators. For the description of technologies and components used, refer to Chapter 3 and Chapter 4.

5.3.1 Setup

This step includes all the mandatory operations for the correct execution of the Consolidation process. As briefly introduced in the data flow, the Closing and the

Consolidation processes are modular, i.e. they have separate storage structures. Therefore it follows, that the first step is to import the target data into the Consolidation structures. Then, a setup phase allows to manage the different Consolidation scopes, specifying the holding ownership percentage with respect to all the companies in the scope. Below are presented more details on these operations.

- **Import Data**: This sub-step allows to import the actual and budget data from the Closing process into a specific ADSO for the Consolidation step. Since the Closing process involves hundred of millions of data records, and it was requested the possibility to perform the Consolidation not only on the entire group, but also on subsets of companies (e.g for specific markets or divisions). FG opted to manually import requested data through a custom calculation. This custom calculation is implemented with combination of ABAP and SQL, and linked to a planning sequence which can be launched with an Excel Macro. Setting the MIDTS InfoObject as *'IMPORT'* helps to distinguish the initial imported data with, no elaborations applied. More details on how this planning sequence is setup:
  - **Function Type**: define the parameters needed by the function, can be seen as its signature. In this case, the parameters used to define the perimeter of the data to copy are Company, fiscal period and category (actual or budget).
  - **Aggregation Level**: aggregation level used to physically store the data in the target ADSO.
  - **Filter**: check that the data produced is in the correct perimeter. For this case the data is imported with the value IMPORT for the MIDTS InfoObject. So the filter checks that the data records have this specific value for MIDTS and have Company, fiscal period and category selected by the user.
  - **Planning Function**: specify where to get the parameters values. This is done linking linking the parameters to the corresponding variables or fixing a constant.

The values for the Filter and the Planning Sequence parameters correspond to the variables of the query, which has been modeled keeping in mind two purposes: checking the data imported in the selected perimeter and using the variable values selected by the users as parameters. This query and the custom import calculation are imported into an Excel workbook to be available to the users.

- **Setup Ownership**: Definition of ownership percentages of the holding company with respect to the subsidiaries, in the selected Scope for the selected fiscal period. This setup operation consists of a BPC page, represented in Figure 5.2, where the users, depending on data perimeter selected, can add and remove companies from a scope, and specify the holding ownership percentage with respect to the companies of such scope. The data from this step is automatically stored into an ADSO which is automatically generated by the system, when configuring this step. The ownership percentages are used in the currency conversion step, to consider only the right proportion of the final converted value. Generally, the holding company owns 100% of all the subsidiaries, so it considered the entirety of the final amount, and the currency conversion is unaffected.

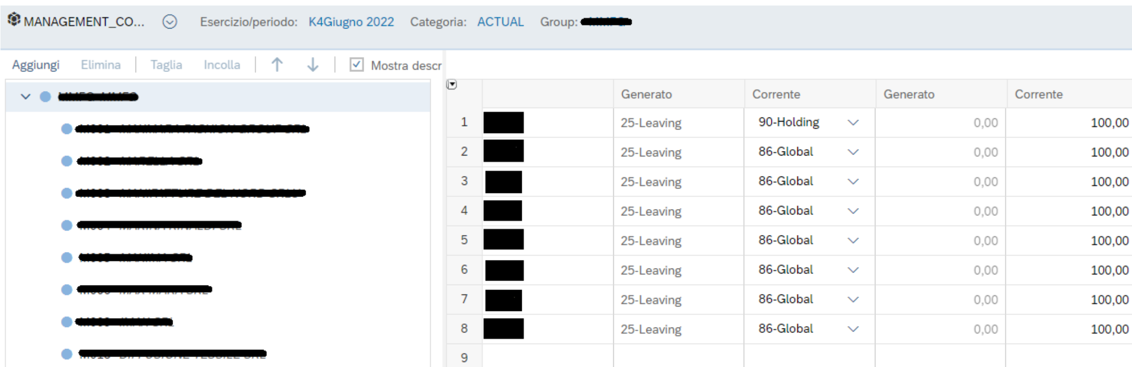


Figure 5.2: *Consolidation: Ownership Configuration - SAP BPC web page, employed for the management of the Consolidation scope and the ownership percentages. This graphical interface allows to add and remove companies from the scope in the selected perimeter, and to define the ownership percentages of the holding company with respect to the subsidiaries.*

5.3.2 Consolidation

This step includes how the actual Consolidation is performed, based on the setup from the previous step and the exchange rates stored on the corresponding ADSO. The Consolidation phase includes the two operations presented below.

- **Currency Conversion**: once the individual company Closing processes are finished, it will be possible to perform the conversion of values from the company currency to the group currencies, defined by the SCOPE InfoObject. The conversion is done using quarterly exchange rates, so the currency conversion is calculated following the formula, for each target currency:

$$FG_n = \sum_{i \in Scope} CMP_n^i * fx_n^i - CMP_{n-1}^i * fx_n^i + CMP_{n-1}^i * fx_{n-1}^i$$

  Where FG_n indicates the group value at the reference quarter n, CMP_n^i indicates the company i value at the reference quarter n and fx_n^i represents the exchange rate from the company i reference currency to the target group currency, at the reference quarter n. Considering a single company, Figure 5.3 highlights the three components to perform the conversion. This step consists

Quarter *n*			
Company Value	**Group Value**	**Exchange Rate**	
10	15	1.5	**YTD Value, Quarter *n*, Exchange Rate of Quarter *n***
-5	-7.5	1.5	**YTD Value, Quarter *n-1*, Exchange Rate of quarter *n***
5	10	2	**YTD Value, Quarter *n-1*, Exchange Rate of Quarter *n-1***
Overall	**17.5**		**YTD Group Value**

Figure 5.3: *Consolidation: Currency Conversion - Example of currency conversion of a single company, highlighting the components that affect the overall group value.*

  of a custom procedure developed with a combination of ABAP and SQL, and hooked to a Planning sequence, which is manually triggered by the financial users. Then, a query has been developed to set up an Excel report to check the conversion results. The report is organized with P&L Line elements in the rows, and companies with the target currencies of the scope in the columns. Below are presented the main steps of the conversion:

- When opening the report, the users specify Fiscal Period and Scope. This manual selection is configured during the modelling of the query. The selected values are then passed as parameters to the conversion procedure.
- Since the conversion is performed with respect to the selected quarter and to the previous quarter, it is kept only the data from these two quarters. Furthermore are kept only the data records from the companies specified in the selected scope. Ownership ADSO to get all the companies in the scope
- The target currencies are specified in the Scope Master Date and they are extracted using string comprehension. Then thanks to the RATE ADSO, a support structure containing only the requested conversion rates is created.
- Finally the filtered data is converted to each target currency using the formula presented above.

- **Intercompany Elisions**: this goal of this operation to remove the transactions between two companies within FG, from the aggregated view. As for the Currency Conversion also this step consists of a custom procedure developed with a combination of ABAP and SQL. The MIDTS InfoObject allows to distinguish the data generated from this calculation. The generation of the the counterparts depends on the P&L Line Master Data. FG provided three cases to manage:
  - **NONE**: no counterpart is generated.
  - **INTERDIV**: If a data record contains a PL Line of this type, the custom calculation generates a counterpart of that record, i.e a copy of that record but with negated amount and 'ELISION' MIDTS
  - **ICELIM**: If a data record contains a PL PL Line of this type, the custom calculation generates a counterpart of that record for the holding com-

pany, i.e a copy of that record but with the holding as company, negated amount and 'ELISION' MIDTS.

Furthermore, if a data record does not contains an intercompany value, no counterpart is generated. This calculation is manually triggered by the financial users, thanks to a planning sequence which can be launched through an Excel Macro. A query has been developed to set up a report to compare the original value (after the currency conversion) of the various P&L Lines and the amounts elided by the calculation, based on the type of elision specified in the Master Data. When the financial users access the workbook, they are presented with a prompt to manually select Fiscal Period, Scope, Category, Currency. The selected values are used to restrict data perimeter of the report and if the calculation is launched they are passed as parameters. The report is organized with P&L Line elements in the rows, and companies and the Companies of the selected Scope in the column. For each company there is an expandable structure, populated according to the type of elision specified in the PL master data.

5.3.3 Reporting

Through this step, the group controllers can access three different profit and loss statements. The three reports are organized based on the reference division: AGGREGATED, INDUSTRIAL and RETAIL, where the AGGREGATED report combines all the divisions.

SAP BW/4HANA allows to define a specific component called *structure*, which is an organization of data into categories. The *Profit&Loss* Line InfoObject has been already introduced. This InfoObject is exploited while building the structures, to distinguish the types of operations and group them into organized and coherent accounting voices. For example the row *Compensation and Contributions* of the retail structure, comprises the PL Lines *Clerks and Workers Labor Costs*, *Bonus*, *Collaborations/Temporary Staff*. Each division has a specific structure.

The Consolidation data is stored into a single ADSO, and the reporting queries are developed on top of the composite provider containing such ADSO. They are defined filtering the corresponding division and using variables on the following InfoObjects, providing control to the users on which aspects of the Consolidation to see in the reports:

- Fiscal Year and Period
- Scope
- Category
- Currency

The implemented structures are then imported into the corresponding query. Each query is imported into Excel where it is accessible by the users through the BPF. When an user access the report, the data corresponding to the selected perimeter is organized into the structure items.

5.3.4 Dashboarding

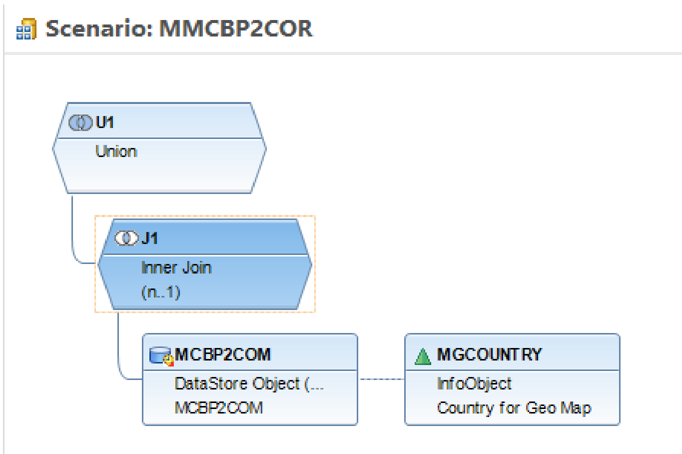


Figure 5.4: *Consolidation: Dashboard Composite Provider - In the reporting phase was requested to build various maps, visualizing different KPIs organized by country. Joining the transactional data and the Country InfoObject, on the country ID, allows to extract the coordinates from its master data. The Composite provider containing this enhanced version of the data can then be used as source, to model the queries for the map building.*

Thanks to this step, the end users are able to access three stories developed using SAC which shows the Consolidation results. A story is a collection of dashboards and tables, employed for the group evaluations. As for the reporting phase, FG requested to develop one story per division (i.e. RETAIL, INDUSTRIAL, AGGREGATED). The data is provided through a live connection with SAP BW/4HANA, which allows the queries developed in the warehouse to feed the various dashboards.

This step consists only on reporting functionalities, so the first step is to build a composite provider gathering all the details needed to develop queries the for the dashboards. Since in the dashboards are implemented also some maps, the transactional data need to be joined to the Country InfoObject, as shown in Figure 5.4: SAP BW/4HANA allows to automatically update the master data of geographical InfoObjects, for example with the coordinates of the country capital cities, as done in this case. For this purpose are reused the same composite provider and queries of the reporting step. Then, the SAC Graphical interface gives the possibility to select or create custom KPIs, and display details at different levels of granularity.

Below, some of the KPIs employed in the stories are:

- $EBIT\% = \frac{EBIT}{NetRevenues}\%$. Used in all the stories.
- $delta\%Ricavi = \frac{delta}{NetRevenues_{n-1}}\%$ where $delta = NetRev_n - NetRev_{n-1}$, with n representing a specific fiscal year. Net Revenues percentage change over the previous year. Used in the AGGREGATED and INDUSTRIAL stories.
- $delta\%EBIT = \frac{delta}{EBIT_{n-1}}\%$ where $delta = EBIT_n - EBIT_{n-1}$, with n representing a specific fiscal year. EBIT Percentage change over the previous year. Used in the AGGREGATED and INDUSTRIAL stories.
- $GrossOperatingResult\% = \frac{GrossOperatingResult}{NetRevenues}\%$. Used in the RETAIL story.

Figure 5.5, shows a part of the AGGREGATED story, consisting of a map which displays the EBIT (Earnings Before Interests and Taxes) by country for the selected period. There is also a lateral filter to further restrict the geographic area. Among The dimensions of analysis there are Country, Division and Brand. Each story has particular dashboards and tables depending on the division. As for the reporting phase, when opening the story the user is presented with a prompt to select the data perimeter. The prompt is configured during the design of the queries, setting

Figure 5.5: *Consolidation: Map Dashboard - Dashboard that highlights the Earnings before interest and taxes (EBIT) trend, organized by country.*

up which characteristic can be manually selected. In this case: Scope, Fiscal Period, Category and Currency.

5.3.5 Work Status

The work status mechanism is employed to freeze portions of data within the system and its configuration is managed through SAP BPC.

Each process and the corresponding application, first requires the definition of a BPC model, which includes the list of ADSOs used and then, which InfoObjects are employed to define the data perimeter to control. It is possible to use only dimensions that are shared by all ADSOs. For the Consolidation application, the perimeter consists of MICOMPCOD, FISCYEAR, CATEGORY and MIDTS, where MIDTS is used to distinguish the steps of the process, thus allowing to lock only data from specific steps. Specifically *IMPORT* locks the data import and the currency conversion, while *ELISION* locks the homonym phase. When the process manager access the Work Status step through the BPF, the pop up with the prompt in Figure 5.6, is presented to the user. The user can select the values for the perimeter to consider, change the status to "*Locked*" or revert previously configured statuses.

When, from SAP BPC, the Work Status option is actived for a specific application, the warehouse system automatically creates an ADSO to store the information

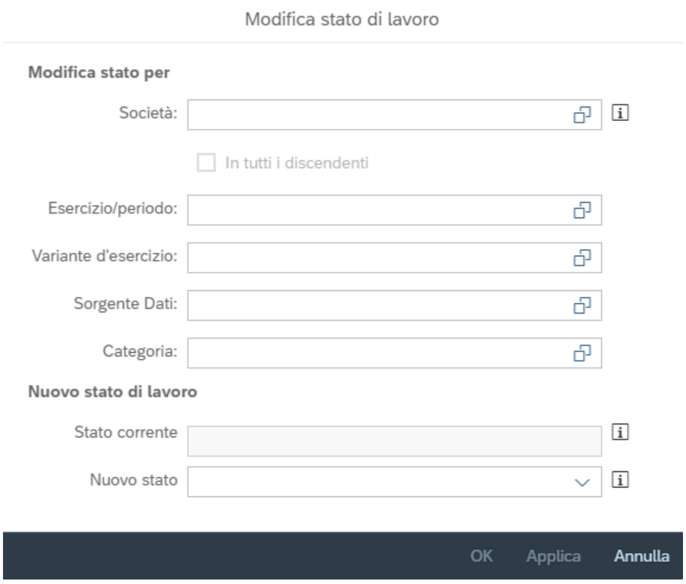


Figure 5.6: *Consolidation: Work Status Prompt - The first part of the prompt allows to select the data perimeter involved in the work status management operation. Then, from the drop down menu on the bottom, the final user selects the new status for the selected data.*

about the data locks. This technical ADSO is then exploited to build a query, reporting the characteristic of the locked portions of data. Thanks to the AFO plugin the query is linked to an Excel sheet provided to the end users.

5.4 Securities

To make the right data available to the right users, securities and data access are managed thanks to SAP BPC. As for the Work Status, the data access rules are defined with respect to a BPC model, which is coupled with the corresponding application, and selecting which InfoObjects are employed to define such access rules. Users are usually grouped in teams, where a team represent a specific task for a specific company. This means that users in the same team have access to the same tasks and the same portions of data. The only special case is for central controllers, which must access all the subsidiaries of FG, while controllers of a specific company must access only the data of such company.

What specifies the securities are the *Data Acces Profiles* (DAP), set up through SAP BPC web interface. Each DAP, defines the perimeter of data whose access is granted, and the corresponding type of privilege (i.e. read or write). Finally, one or more data access profiles can be assigned to either teams or specific users.

Since the process is specific for the holding controllers, all the users are organized into a single team. Then, a data access rule defines the access to the data of all companies, imported in the ADSOs specified in the BPC model, corresponding to the Consolidation application.

Chapter 6

WBS Panning Process

This chapter explains more in depth the technical aspects and implementation details of the WBS Planning process. First, it presents the major architectural aspects, such as the data model and the data flow within SAP BW/4HANA. Then, it details the design of the application procedure, with the respective calculations. Finally, it describes how the securities and the data access are managed with SAP BPC.

6.1 Data Model

The identification of a correct data model is fundamental to carry out a good design for the data flow, which is explained in the next section. Furthermore, it also helps to improve the application scalability, in case of future developments.

The WBS Planning process is smaller than the Closing and Consolidation process, and not all the companies within the group are involved, thus resulting in a simpler data model and a lower amount of data involved. Since the process consists in planning the job orders, an InfoObject is needed to represent such characteristic. Initially, it is estimated the overall costs of each Job order, then the total costs are budgeted by year, requiring another InfoObject to depict this aspect of the data. It is also needed to register the company that commissioned WBS. Furthermore, a technical dimension is used to distinguish the data corresponding to the various steps of the process.

Table 6.1 contains the main InfoObjects, with the corresponding aspects that they represent, used for the design of the WBS Planning application. In the design

INFOOBJECT	Description
MICOMPCOD	Company within the group who commissioned the Job Order
MIWBSELEP	WBS/Job order
MIVENDOR	Vendor/Supplier of the job order
MICOSTELM	Cost Element, characteristic used to organize the final financial statements
MICTGRY	Type of data, i.e. *actual* or *budget*
0FISCPER	Fiscal Period (Month)
0FISCYEAR	Fiscal Year
MISEASON	Season, i.e. *fall-winter* or *spring-summer*
MIAMOUNT	Amount of the operation represented by the record
0CURRENCY	Currency related to the AMOUNT
MIDTS	Technical InfoObject to identify the step where the data belong within the process

Table 6.1: *WBS Planning: Transactional Data Model - This table represents the main characteristics of the transactional data, relevant for the WBS Planning process, identified by the corresponding InfoObjects.*

phase, there was also a particular focus on the MIWBSELEP InfoObject, which plays a critical role also on the custom functionalities and the execution flow of the different process steps (more details in the next sections). The Master Data option for the job orders has been activated. The main reason is to apply basic data checks. For example the transactional data must contain records referring to job orders/WBS existing within the warehouse environment. As already mentioned in Chapter 4, when the Master Data option is activated, InfoObjects can be further enriched with *Navigation Attributes*, represented by other InfoObjects. Instead of having a single list of admissible values, InfoObjects became a data structure represented as a table, where the attributes are the columns of such table.

Table 6.2 reports the structure of the InfoObject used to identify the job orders. The use of navigation attributes allow to further perform other data consistency checks. For example, the transactional data can contain only records where the pair of ID_WBS and MICOMPCOD exists in the Master Data of the Job Orders.

In addition to identifying the job order, ID_WBS combined with MIPRJWBS, are

MIWBSELEP	
ID_WBS	Job Order code
MICOMPCOD	Company within the group who commissioned the Job Order
MITYPWBS	Job order Type
MIPRJWBS	Parent Project of the Job Order
MICOSTCEN	Original Cost Center of the budget data
MICDCDST	Target Cost Center of the budget data
MIDWNLWBS	Flag to import the Job Order into the OPEX application

Table 6.2: *WBS Planning: Job Order InfoObject Structure - This table represents the structure of the WBS InfoObject and the corresponding master data. ID_WBS distinguishes the job orders in the transactional data, while the other characteristic are all navigation attributes, exploited in different part of the process. For example, MICOSTCEN, MICDCDST and MIDWNLWBS can be modified by the user through a graphical interface, to set up the download of the WBS planned data into the ADSO of the OPEX application, which provides a view on the overall group costs.*

employed to build the WBS hierarchy, which is utilized for the security management and, in general, as manual filter on the data to query. Then, MITYPWBS categorizes the type of job order. Finally, the last three attributes are employed in the last step of the WBS Planning process. Specifically they are used to manage the import into the OPEX application. The Cost Centers identify parts of an organization to which costs are charged for accounting purposes (for example a physical store).

It is also important to mention a fundamental aspect of the application technical design: the WBS Planning application employs the use of actual and budget transactional data. The two types of data are stored into two different ADSOs and are managed differently. The budget data is manually inserted by the end users, while the actual data is managed through an automated data flow. The main goal is to keep the independence between the structures storing the two types of data. If needed, it can be built a combined view of the two ADSOs, using a Composite Provider.

6.2 Data Flow

This section discuss the main architectural aspects along with the data flow implemented. The smaller size of the process and the smaller amount of data involved, is also reflected on the architecture complexity, which results less intricate than the data flow shared among the majority of the processes, briefly presented in Chapter 5. Only two ADSOs are employed, one storing the actual data coming from SAP ECC, and one for storing the planning data, which are also uploaded to the OPEX application. More details on this interaction with the OPEX application are provided in the next section.

In the *Indirect Costs Workflow (WFI)* application, mentioned in Section 2.2, were already implemented some functionalities related to the job orders. These functionalities are only limited to reporting aspects of actual data, however some components (InfoObjects and ADSOs) can still be exploited to simplify the architecture and the data flow of the *WBS Planning* application.

Initially, multiple data flows travel from SAP ECC to SAP BW/4HANA. The two systems are connected thanks to different DataSource objects, coupled with ECC tables. DataSource objects allow SAP BW/4HANA to detect data from external systems, that are then elaborated on the fly, and imported into its environment, thanks to transformation layers and DTPs, as presented in Chapter 3. After that, the data flow is managed entirely within SAP BW/4HANA. The flow controls two categories of data: master data and Transactional data. The master data specifies what values the features with this option enabled can take. Furthermore, since the InfoObjects that represent such features usually have some navigation attributes, the master data can be used for consistency checks on the values simultaneously assumed by different characteristics. Finally, the master data can also be exploited to build mapping tables, allowing to derive missing dimensions (more details below and in the next section). Instead, the transactional data represents records of financial operations, associated to amounts or quantities. In the WBS Planning process, such records represent budget and actual for the respective job orders. Transactional data is stored within ADSOs, while master data is stored into InfoObjects.

Figure 6.1 depicts the data flow of the WBS Planning application. It is possible to see that the goal is to collect the actual data, from SAP ECC into the SAP BW/4HANA target ADSO, and to format and derive all the dimensions of the WBS element master data. Custom operations are defined in the transformation layers, and implemented in ABAP, the SAP proprietary language. ABAP can be seen as a variant of SQL, where the operations are performed row wise. For example the SQL *SELECT* statement is structured as a loop using ABAP. As just mentioned, the design of the data flow takes advantage of WBS structures of the WFI, however in the planning application the job order InfoObject have different details, so before dealing with the transactional data, there is the need to derive all the attributes of the MIWBSELEP master data. Next are summarized the DTPs composing the Master Data flow, with the respective manipulations specified in the corresponding transformation lavels.

- **ECC → MIPRJWBS**: this step populates the master data of the InfoObject representing the WBS projects. Direct assignment from SAP ECC to SAP BW/4HANA, hence the master data of the projects is simply replicated within the warehouse. The MIPRJWBS InfoObject supports the correct population of the job orders.

- **ECC → MIOBJNRW**: this data transfer uses the external table of the MIPRJWBS InfoObject to build a mapping between WBS and the corresponding WBS parent project. Among its navigation attributes, MIPRJWBS contains also MIOBJNRW, so it used to join the two InfoObject tables, deriving the mentioned mapping

- **ECC → MIWBS_ELE**: MIWBS_ELE is the job order InfoObject used in the WFI application. Direct assignment from SAP ECC to SAP BW/4HANA of the used attributes in the master data.

- **MIPRJWBS → MIWBSELEP**: the WBS projects are treated as normal job orders, so here, they are added to the master data of the target InfoObject, with 'PR' assigned to the project type attribute.

- **MIOBJNRW → MIWBSELEP**: in this step, the job orders that are not projects are added to the master data of MIWBSELEP. Furthermore, are derived the job order types, other than projects (PR). Job orders have a standardized naming convention, so this type of derivation is done simply checking the if the WBS code contains a specific substring. The different types are Operations (OP), Evolutions (EV) and Implementations (IM). In the end, the MIWBSELEP master data contains a list of job orders and parent projects, where the job orders have the navigation attribute MIPRJWBS populated with the corresponding parent project id.

- **MIWBS_ELE → MIWBSELEP**: assign the full name and description of the job orders to their corresponding ID_WBS.

- **MIWBSELEP → MIWBSELEP**: self-loop, no data manipulation is performed. This step is only employed to compute a hierarchy object, starting from the complete master data of the job orders. This object can be seen as a tree, where selecting a node gives access to the subtree below such node (according on how the data access rules are defined). Suppose, for simplification, that each row of the WBS master data has the following format (ID_WBS, PARENT_PRJ, TYPE, MICOMPCOD). Furthermore, each node of the resulting tree is represented by a data structure with 4 attributes: *node_id* and *parent_id* are used to organize the nodes within the tree, while *iobj_name* and *hnode_name* defines additional characteristic of the nodes, namely, the InfoObject to use for future filtering and its corresponding ID in the master data. WBS projects represent a collection of job orders, however they are treated as a normal job order to populate the masterdata. In the hierarchy they are located in the penultimate layer, while the single job orders are the leafs of the tree structure. Algorithm 1 is structured in four steps, computing the hierarchy with depth-first logic:

  - first, the root node is manually built;
  - then, a node for each company is created;
  - next, for each company, appended its corresponding commissioned projects;

– finally, the single job orders are added the respective projects as leaf nodes. to the parent project

The computed hierarchy structure is then coupled with a *variable* object. Variables act on InfoObjects and are specifically used on queries. In this case, the hierarchy acts on the MIWBSELEP InfoObject, so among the other functionalities, it is employed by the end users to manually filter the job orders.

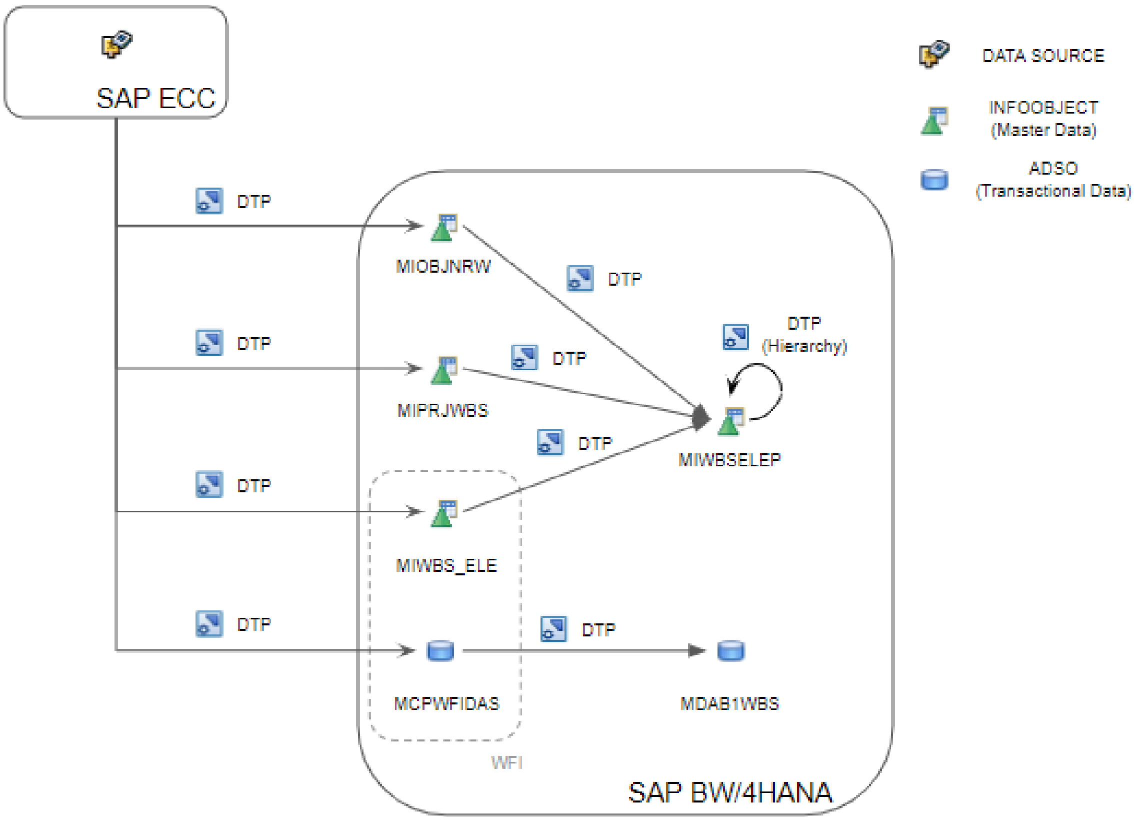


Figure 6.1: *WBS Planning: Data Flow - Structure of the data flow designed to populate the target ADSO and InfoObject, starting from SAP ECC, the ERP system, importing the data into the warehouse environment correctly formatted. This flow makes use of some of the components already developed for the Indirect Workflow (WFI) application, avoiding to duplicate the same steps. This flow runs every night, keeping the transactional data and the master data up to date.*

Below are outlined the steps of the transactional actual data Flow. Generally, only the actual data is managed through DTPs, since imported from SAP ECC, and the data flow is automated. Instead, the planned data is usually managed through custom manipulations, which are provided to the end users and can be manually executed. As already mentioned in the previous section, the actual data and the

Algorithm 1: WBS Hierarchy Algorithm

Input : Set of Companies C with at least one job order commissioned, Set of Job Orders J

Output: Tree data structure WBS_H, representing the hierarchy

```
function BuildHier (C, J);
node_id ← 1; parent_id ← 0;
iobj_name ← '0HIER_NODE'; hnode_name ← 'ROOT_WBS';
root ← (node_id, parent_id,
        iobj_name, hnode_name) ;                    /* Root node */
WBS_H ← WBS_H ∪ {root};
foreach c ∈ C do
    node_id ← node_id + 1;
    tmp_comp_id ← node_id;
    parent_id ← 1;
    iobj_name ← 'MIWBSELEP';
    hnode_name ← c.MICOMPCOD ;
    hiernode ← (node_id, parent_id,
            iobj_name, hnode_name) ;     /* Node for each Company */
    WBS_H ← WBS_H ∪ {hiernode};
    foreach prj ∈ J where prj.MICOMPCOD = c.MICOMPCOD AND
      prj.TYPE == 'PR' do
        parent_id ← tmp_comp_id;
        node_id ← node_id + 1;
        tmp_prj_id ← node_id;
        iobj_name ← 'MIWBSELEP';
        hnode_name ← prj.ID_WBS ;
        hiernode ← (node_id, parent_id,
                iobj_name, hnode_name) ; /* Node for each Project */
        WBS_H ← WBS_H ∪ {hiernode};
        foreach wbs ∈ J where wbs.PARENT_PRJ = prj.ID_WBS AND
          wbs.TYPE != 'PR' do
            parent_id ← tmp_prj_id;
            node_id ← node_id + 1;
            iobj_name ← 'MIWBSELEP';
            hnode_name ← wbs.ID_WBS ;
            hiernode ← (node_id, parent_id,
                    iobj_name, hnode_name) ;  /* Leaf for each WBS */
            WBS_H ← WBS_H ∪ {hiernode};
        end
    end
end
return WBS_H;
```

budget data are stored into different ADSOs, keeping the independence between the data managed by the end users and the data managed through the data flow.

- **ECC → MCPWFIDAS**: MCPWFIDAS is the ADSO which stores the WBS actual data for the WFI application. The data is imported from the ERP system with minor adjustments, not relevant for the planning application.
- **MCPWFIDAS → MDAB1WBS**: the target structure is the WBS collector ADSO. The transactional actual data is already correctly formatted in the WFI application, so it is simply imported keeping only the needed details.

Lastly, all the components of the data flow are organized into a process chain. The process chain is scheduled to run at night, so the transactional data and master data are updated every night.

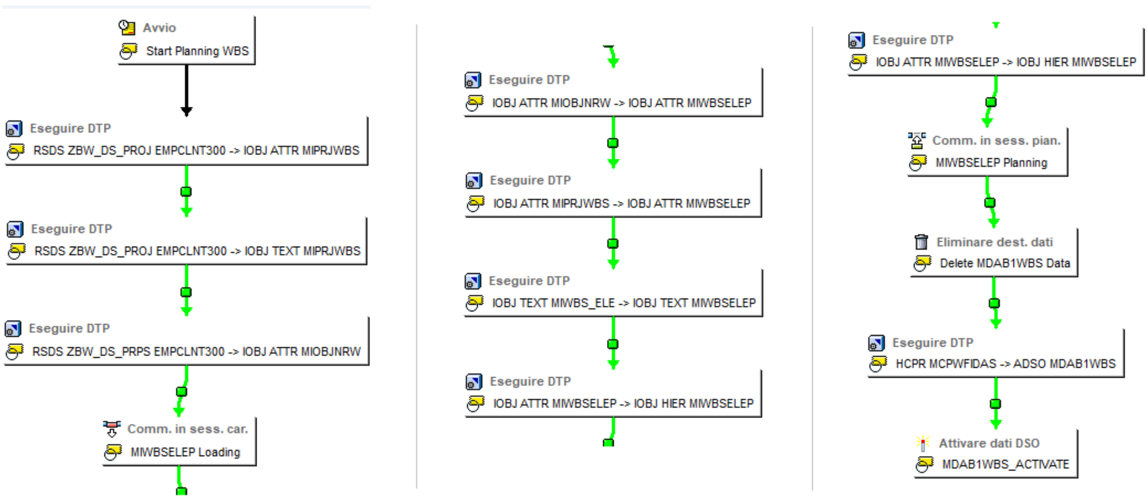


Figure 6.2: *WBS Planning: Process Chain - The represented Process Chain links all the single data transfer steps, automating the WBS Planning data flow. It consists of a sequence of operations to perform in the specified order.*

6.3 Business Process Flow

This section presents the Business Process Flow model of the WBS Planning process and application, with the most relevant implementation details. The technical flow of the process consists of Setup phase, Budget and Revised Budget phase and finally, the Work Status phase. The planning of the job orders is performed yearly. The

BPF is accessible through SAP BPC, and consists of a set of links which redirect the user to the correct page of the application.

For each company, there are two types of users: controllers and compilers. The controllers (which are the process managers) have access to the entire process, while the compilers have access only to the Budget and Revised Budget phase which is limited within SAC. The compilers are interfaced to SAP BW/4HANA through SAC, while the additional features provided to the controllers also employ Excel with the AFO plugin.

For the description of technologies and components used, refer to Chapter 3 and Chapter 4.

6.3.1 Setup

As already discussed in Chapter 5, the major developments consist on queries and planning functions. Queries are defined on Composite Providers for reporting purposes. Instead, they are defined on aggregation levels to create *'input'* queries, where the final users can manually input data to store into the system.

Planning functions can be native or custom, to perform complex data manipulations and they are launched by the financial users through Excel macros. Recall that a planning sequence is defined by *Function Type*, *Filter*, *Planning Function*.

The Setup phase of the WBS Planning process is organized in three steps. Their utility and implementation details are listed below.

- **Import Budget (OPTIONAL)**: while the the actual data is managed with the data flow, the budget data is manually inserted by the compilers. This optional step allows the controllers to import *csv* files for the job orders within the system. This step was used a single time, to store the budget data already planned before the go-live of the application. As usual, a query is built on the Composite Provider MMPLANWBS containing the data from the budget ADSO and the actual ADSO. This operation is performed with budget data so query restricted only on the budget data InfoProvider. The variables on Company and Fiscal Year are exploited for user manual input. The query is then imported into an Excel workbook, which is opened through the BPF. The procedure to import the file is started with an Excel macro. This is a standard functionality provided by SAP, so the definition of the planning function is straight-forward. The Aggregation Level used is AMPLANWBS, built on top of the Composite Provider just mentioned. The filter has the

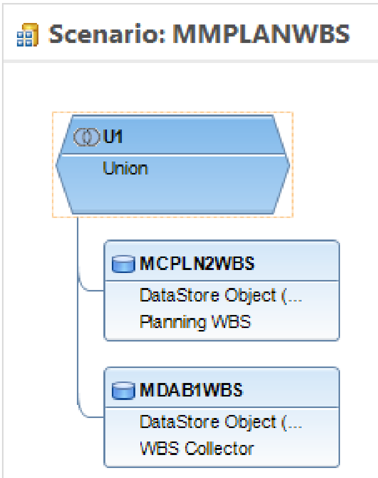


Figure 6.3: *WBS Planning: Composite Provider - The WBS Planning application consists of two main ADSOs. One collects the actual data from the data flow, while the the other one hosts the planned data stored by the users. The union of the two ADSOs allows to develop a reporting phase comparing the actual and the budget data.*

same variables used in the query, so the planning function is executed using the selections of the user. The purpose of the filter is to check that the new data stored respect the selected perimeter, otherwise an error is raised and no data is saved. If the user selects Company *C001* and Fiscal Year *2023*, the user cannot import a file with the job order budget for Company *C002* and Fiscal Year *2022*. The import is done through a native type of planning function. From the graphical interface it is first required to choose the type of update: overwrite or sum. Then, the InfoObjects are mapped to the corresponding columns of the file imported. This means that all the files to import must be structured in a fixed format. The data imported is stored into the budget ADSO. When opening the workbook, the controller is presented with a prompt to select company and year of the budget to import. The prompt acts also on the data visualized thanks to query. Then, the user can launch the import.

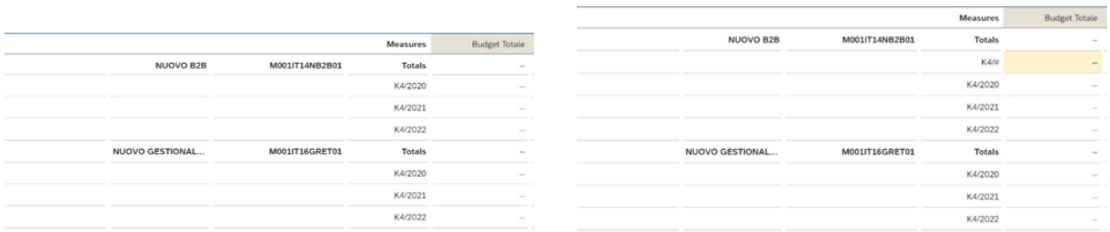
- **Input Mapping**: this step is modeled through an input query, that, when imported into Excel, is structured as Figure 6.4. This step is fundamental to enable the editable cells for the compilers, for example when a new job order is created and it has no actual data, or when a new Vendor is employed in an alraedy existing job order. The mapping phase consists on entering the values

							Abilitazione Input BDG Dettaglio	Abilitazione Input BDG Totale
WBS Planning		Voce di costo		Fornitore		Tipologia WBS	EUR	EUR

Figure 6.4: *WBS Planning: Mapping Table - This table presented to the process manager through SAP BPC, allows to activate a new combination of WBS-Cost Element-Vendor to be planned. This step is necessary if there is a new job order, without actual data, or if a new Vendor is involved in an existing project for the first time.*

for WBS-Cost Element-Vendor, that the process manager wants to enable for the compilers (the values for the other InfoObject are fixed in the query). The data entered from this phase is stored with the *'MAPPING'* value for the datasource InfoObject. If this operation is done correctly, a new row is added to the corresponding budget sheet, as shown in Figure 6.5. In particular, entering the value '1' in the *'Enable Input BDG Detail'* column, allows to compile the budget (and eventual balance carry-over) for the selected year and WBS. Instead entering '1' in the *'Enable Input BDG Total'* column, makes possible to enter the total budget for the entire job order, hence with fiscal year not assigned. The two types of budget are compiled through different queries.

- **Vendor Report**: the Vendor InfoObject is used also in the other processes, mentioned throughout the document. The master data values are imported from the ERP system, and already managed with the data flow of the OPEX application. The report simply lists the Vendor master data, and has the purpose to check if the SAP BW/4HAHA environment contains all the vendors. This point is exclusively reporting. A Composite Provider allows to get the Vendor master data, with a query built on top of it.



(a) *Total Budget Sheet before Mapping*

(b) *Total Budget Sheet after Mapping*

Figure 6.5: *WBS Planning: Mapping Result - This is the interface presented to the compilers. The two images represent the effect of the Mapping step if performed correctly. This operation adds a new input cell, for the corresponding perimeter defined in the Mapping step, in the planning tables.*

6.3.2 Budget and Revised Budget

As already mentioned, there are two types of users: controllers/process managers and compilers. The compilers have access only to the *Planning WBS* step, through *stories* developed in SAC, one for the budget and one for the revised budget. This is the core phase of the WBS Planning process, which is divided in the following three steps. The following steps are replicated for the budget and for the revised budget, the overall logic is the same.

- **Planning WBS**: As already mentioned one planning cycle consists of two phases: the initial budget planning, usually performed in June and the revision of the budget, usually performed in February. One functionality specifically requested, was the possibility to carry-over the unspent budget from the previous planning cycle, and consequently write off this amount from the past budget. This operation is specific only for the June phase. Other than this functionality, the compilers perform the same operations in June and February, however the management of the inserted data is very different. Let's see, for example, the planning of the single job order *J01IMXYZ* to see the computations performed by the system.
  Starting from the June phase:
  - Storing the planned amount is straight forward. It is simply stored a data record representing the job order *J001IMXYZ* with the corresponding characteristics and planned amount. Setting the technical InfoObject

MIDTS = *'INPUT'* allows to identify the records generated from this step. Suppose the record generated is the one presented below (some dimensions are not considered to have a simpler example).

COMPCOD	WBS	VENDOR	CATEG	FISCYEAR	AMOUNT	CURR	MIDTS
C001	J01IMXYZ	V005	BUDGET	2024	15.000	EURO	INPUT

- Instead when the user puts down the carry-over, are generated two different records. The first record simply represents the carry-over amount to store for the selected year, with MIDTS = *'CARRY OVER'*.

COMPCOD	WBS	VENDOR	CATEG	FISCYEAR	AMOUNT	CURR	MIDTS
C001	J01IMXYZ	V005	BUDGET	2024	5000	EURO	CARRY OVER

The second record represents the write off for the previous fiscal year of the carry-over, distinguished by MIDTS = *'WRITE OFF'*

COMPCOD	WBS	VENDOR	CATEG	FISCYEAR	AMOUNT	CURR	MIDTS
C001	J01IMXYZ	V005	BUDGET	2023	-5000	EURO	WRITE OFF

As mentioned in the last section of Chapter 4, planning sequences can be triggered on events, which is the approach used for the generation of the second record. The standard copy planning sequence is triggered when a record with MIDTS = *'CARRY OVER'* is stored, and it is used to generate the write off record. It is important to point out that the planned budget with the previous planning cycle is not overwritten, but given how the queries for this passage are designed, they provide an aggregated view on budget, carry-over and write off. Furthermore, this behavior is active also when importing files in the Setup step.

Suppose that during the February Phase, the budget is revised:

- the compiler is presented with an input query, pre-populated with the values inserted during the June phase, i.e the sum of initial budget and carry-over. When a new budget value is inserted, the system does not update the value intially stored, but it generates the delta between the original value and the new value. Thus, considering the example, the total budget planned for *J01IMXYZ* for the 2024 fiscal year is 20.000 €. Supposing that the compiler updates the budget to 22.000 €, the record stored in the system is

COMPCOD	WBS	VENDOR	CATEG	FISCYEAR	AMOUNT	CURR	MIDTS
C001	J01IMXYZ	V005	BUDGET	2024	2000	EURO	DELTA

This behavior is model through a query feature called *inverse formula*. The query modeller allows to define calculated key figures. So, $Rev\_Bdg = Bdg + Carry\_Over + Delta$, defined as calculated key figure. From this formula can be defined the inverse formula to derive *Delta*: $Delta = Rev\_Bdg - Bdg - Carry\_Over$. With an inverse formula, the calculated key figure became inputable. Recalling the example, the compiler inserted Rev_Bdg = 22.000€and, from the July planning phase, Bdg = 15.000€and $Carry\_Over$ = 5000 €. Plugging everything in the inverse formula, $Delta$ = 2000 €and, thus, a record with amount equal to 2000€, identified by MIDTS = *'DELTA'* is stored.

The MIDTS InfoObject allows to keep track of all data generated in the various steps. Furthermore, thanks to the implementation aspects just described, the query modelling becomes pretty easy: if there is the need to consider only data from the July phase (initial budget), a query can be modelled to filter the MIDTS InfoObject including the values *INPUT*, *CARRY OVER*, *WRITE OFF*. Instead, if there is the need to consider the data including the revision phase, the value *DELTA* is added to the filter just specified. This step, employs a live data connection from SAP BW/4HANA to SAC. The queries are not imported into Excel, but they are used as sources to build two SAC stories, one for the budget and one for the revised budget. The functionalities are the same of the combination of Excel and the AFO plugin, however SAC also allows to build some dashboards for a better visualization of the data. When entering the SAC story an initial prompt allow to select the scenario of interest. The stories provided to the compilers are organized in 4 pages. The first page (Figure 6.6) contains a bar chart, showing the trend by year of the budget amount compared to the actual amount, and a heatmap of the actual data, organized by vendor and fiscal year. The side filters allow to narrow the job order and year perimeter. The second and third page contain three tables, modeled through input queries. One page is

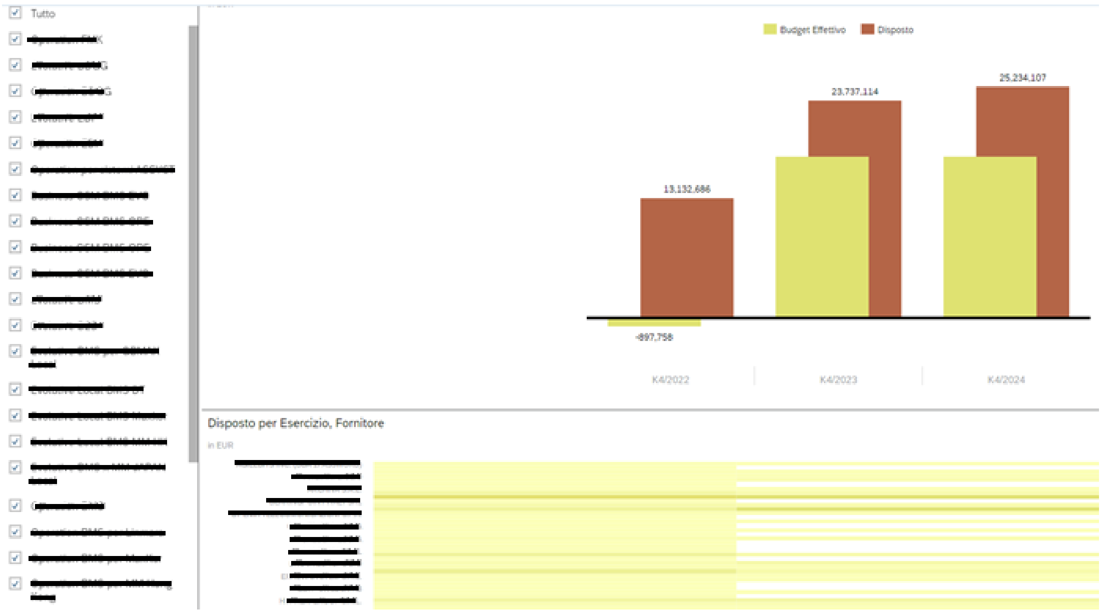


Figure 6.6: *WBS Planning: Dahboard SAC - Along with the planning tables, the SAC stories contain also various dashboards to have a view on the job order trends. For example, this dashboard displays a bar graph comparing actual and budget, and an heat-map of actual costs by Vendor and fiscal year. The lateral filter allows to select job orders to consider.*

specific for the *'implementations'* job orders, while the other groups job orders of type *'operations'* and *'evolutions'*. With these tables, the compilers insert the budget values of the job orders with the corresponding type. The last page contains a recap of the job orders. with their respective trends, organized by year, of budget and actual. Furthermore, the last page also implement the behavior shown in Figure 6.5, which allows the compiler to enter the overall budget for the job orders correctly mapped in the setup phase.

- **Reporting**: Two reports shaped in the same way, one specific for the budget and one specific for the revised budget, since the revision could update the budget values of the job orders. The reports contain a custom expandable structure, defined in the query designer. This structure can be expanded, to display the groupings presented in Figure 6.7. The structure categorize the type of Vendor. *Intercompany* considers all the job orders where the provider is another company within the group, *Internal Hours* considers the job orders developed internally to the reference Company, *Third Parties* considers the job orders with external providers. For example, this report details the actual for

2023 (second column) and the budget for 2024 (last column) for the split of the structure. As for the planning step, the queries for budget report and revised

Reporting BDG

	Disposto 2022	Disposto 2023	Disposto 2024	Rev Budget 2023	Forecast 2023	Budget 2024
Strutt.	EUR	EUR	EUR	EUR	EUR	EUR
[-] Totale	12.681.663	791.439	651.376	38.846.746	38.445.181,00	5.543.505
Intercompany	0			350	-179.965,00	2.800.455
Ore Interne	0				-30.000,00	1.400.000
Terzi	12.681.663	791.439	651.376	38.846.396	38.655.146,00	1.343.050

Figure 6.7: *WBS Planning: Reporting by Vendor Type - Query modeled to display the budget for the selected fiscal year, along with revised budget of the previous fiscal year, and the Actual up to the two previous fiscal years, for the selected job orders. This amount are then categorized into three groups depending on the type of Vendor. 'Intercompany' considers all the job orders where the provider is another company within the group, 'Internal Hours' considers the job orders developed internally to the reference Company, 'Third Parties' considers the job orders with external providers. Once imported into Excel, this is how the report is structured.*

budget report are defined using the same Key Figure for the Columns, however they are restricted/filtered differently, to consider only the desired data, hence the amount specific of certain data perimeters. Recalling, the example just mentioned, the major differences between the two types of restricted Key Figures are:

- One represent actual data and one budget data, so the CATEGORY InfoObject is restricted differently.
- When opening the Excel workbook, the user is presented with a prompt to select the data perimeter. The budget is filtered considering the selected year, while as requested by FG, while for the actual is presented also for previous fiscal years. This is done applying an offset to the variable.
- The actual data is managed through the data flow, while the budget data is managed manually. So the technical InfoObject representing the source of the data is filtered based on the corresponding case.

– Finally budget and actual are stored into two different ADSOs, allowing to restrict also the source InfoProvider.

- **OPEX Download**: This step allows the process managers to select which WBS to download to the OPEX application, and launch such download. It is applied the same logic of *'Input Mapping'* in the Setup step. In the left

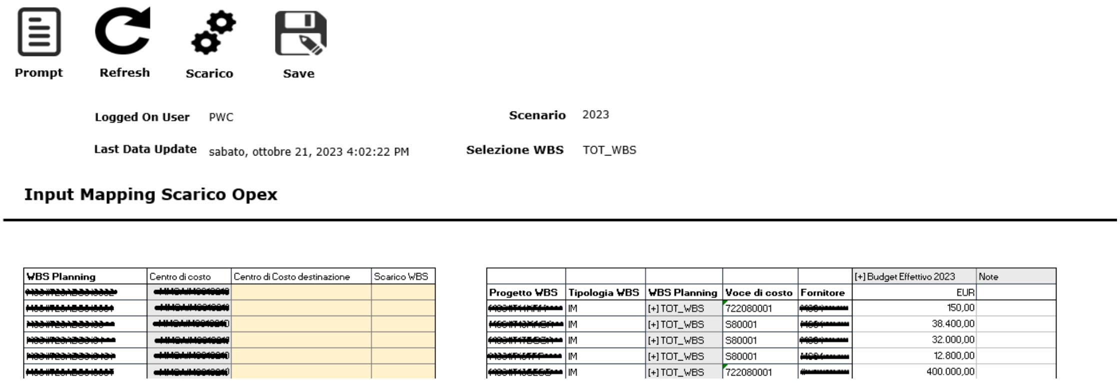


Figure 6.8: *WBS Planning: OPEX Download Report - Interface provided to the process manager to set up the download of the job orders in the OPEX application. The left table allows to select which job order to download and to map it to a new cost center if needed. The right tables provides the details of the job orders in the selected perimeter, to have a reference to the amount that it is going to be downloaded.*

table, from Figure 6.8, the user must flag with '1' which job order to download and, eventually, specifies a new destination cost center. If the destination cost center is not specified then, the source cost center is treated as the destination cost center. This detail is not relevant for the transactional data of the job orders, but it is required in the OPEX application. The right table simply provides some details on the budget data of the job orders, to check the amount that is going to be downloaded. Since, the planning cycle consists of a first planning and then on a revision of the first planning, depending on the phase, the download is performed only on the corresponding data. During the design phase, it emerged the need to change the target cost center, for accounting purposes, when downloading the job orders in the OPEX application. In addition, if a WBS is downloaded once, it will be usually downloaded also in the next planning cycles. These two remarks are the reason why this two operations are performed directly in the WBS InfoObject, where the master data is independent from the fiscal year. Thus the input mapping query is built on an aggregation level linked to a composite provider on top of the WBS InfoObject. The download to the OPEX application is executed through a custom planning sequence, since it is not a plain copy of the data. The complexity derive from the fact the data is read through one aggregation level, and written through another aggregation, thus the standard copy function is not suitable, since it works only on the same aggregation. Furthermore, the planning sequence filter is defined on the target Aggregation Level, to check the consistency of the data produced, which is going to be stored into the OPEX budget ADSO. The corresponding algorithm is developed using a mix of ABAP and SQL: ABAP to filter the input data from the source aggregation level, and SQL to derive some of the fundamental characteristics, which are not used in the WBS Planning application, but are required in the OPEX application. Basic algorithm logic:

- When opening the Excel workbook, the process manager selects company and fiscal year through a prompt, which is set up with input variables.These variables are then also used to pass the user selections to the calculation as parameters. So the fiscal year and the company selected by the user are used to filter the initial data, keeping only the *'activated'* job orders.
- Then, the filtered transactional records are sequentially joined with InfoObject internal tables, to derive missing characteristics. For example, the input data is first joined with the cost element master data (using the cost element id) to get 4LEVEL and 3LEVEL, which are accounting dimensions used to organize the financial statements in predefined structures. After that, the input transactional data is joined with WBS master data table. This operation allows to add to the transactional data the destination cost center specified by the user in the mapping table, retrieving it from the job order master data.
- The budget is planned with respect to the fiscal years. Again, joining

with the cost center InfoObject allows to get the *Timing* attribute, which is used to extract the *fall-winter* and *spring-summer* season with the corresponding months. Then the budget is evenly split into the months of the seasons.

– Finally the data is written to target OPEX ADSO, through the corresponding OPEX target aggregation level.

6.3.3 Work Status

With the following step, the user can freeze portions of data which has been validated, and verify the set up rules in a dedicated report. As for the same step in the Consolidation process, the work status mechanism is employed to freeze portions of data within the system. The InfoObjects (with corresponding hierarchies) chosen to define the data perimeter to control, for the WBS application are:

- **MIWBSELEP** (Job order);
- **MIDTS** (Datasource);
- **FISCYEAR** (Fiscal Year);
- **MICOMPCOD** (Company).

Suppose that the compilers of the company 'C001' are performing the budget revision of the for the fiscal year 2024. There is the need first to freeze the data coming from the initial planning (with balance carry-over), for year 2024 and for all job orders, without blocking the revised budget perimeter. The corresponding selection for the workstatus is:

- **MIWBSELEP**: 'C001'. Selecting such node freezes all the job orders belonging to projects commissioned by C001.
- **MIDTS**: 'INPUT' and 'CARRY OVER'. Need to select the values that identifies the first budget phase and the balance carry-over.
- **FISCYEAR**: '2024'.

- **MICOMPCOD**: 'C001'.

The end user is simply presented with a prompt to select the values corresponding to the InfoObjects, thus defining the perimeter of the data to freeze. This setup is embedded in SAP BPC. Finally, an Excel report allows to check how to work status has been set up. The Work Status step is not accessible by the compilers.

6.4 Securities

As already said in section 5.4, securities and data access are managed thanks to SAP BPC. In this case, the BPC model includes the ADSOs employed for the WBS Planning process. Furthermore, since the process and the overall amount of users is smaller, each user is managed independently with specific DAPs. The data perimeter controlled by each DAP is defined by WBS and Company InfoObjects.

The DAP presented in Figure 6.9 is an example of the rules defined for the WBS Planning process. In this example, the end user has writing privileges (i.e. can visualize and insert data), for the company with code *'M001'* and for the job orders grouped under *'TOT_WBS'*. *'TOT_WBS'* is the root node of the WBS hierarchy, so the option *'Subtree Below Nodes'* gives access to all the job orders.

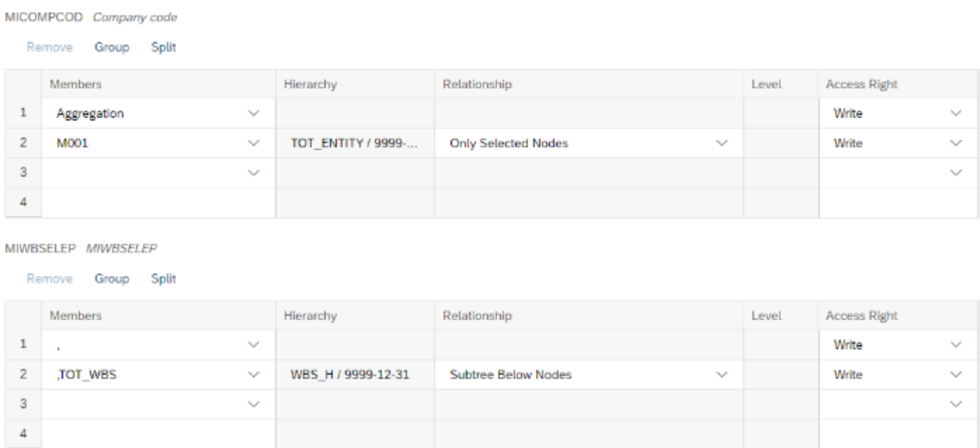


Figure 6.9: *WBS Planning: Data Access Profile - BPC Interface that allows to define the data access rules. In this example the access is defined thanks to the Company and the WBS InfoObjects. The access is granted for a specific company and the subtree of node 'TOT_WBS' for the job orders.*

Chapter 7

Conclusion

In conclusion, this work demonstrates the significance of a well designed EPM solution, and how the SAP BW/4HANA components are suitable and can be adopted to manage all aspects of the process flow, from the process design and architecture, to the data management through the ETL pipeline. A unified EPM platform, acting as centralized "single source of truth", helps achieving data consistency among the different financial processes.

After the testing phase, ended with minor fixes and adjustments, the implemented WBS Planning process received positive feedback from the users. The application is already deployed and running in the production environment, and has already been adopted for the first planning cycle. On the other hand, the implemented Closing and Consolidation processes are not fully deployed in the production environment. The new Closing process revolutionized some of the key aspects, and required a lot of testing time from the financial users. At the time of writing, the Consolidation part of the process is yet to be tested.

FG is a large enterprise which involves dealing with millions of data records. The crucial lesson learned form technical point of view, was the necessity to keep the data flow, the architecture and the custom calculations as efficient as possible. During the initial developments on the Closing process, the financial users have experienced slowdowns on the reporting phase. So, there has been the need to redesign its underlying logic. The data management is a critical point to avoid a negative impact on the system resources, and user experience. In the end, both applications have effectively managed large volumes of data, optimizing the design of process workflows.

The development of these applications underscored the crucial role of the EPM system in facilitating smoother and more efficient process flows, reducing manual efforts, and enhancing the user experience across various financial processes. Furthermore, having an efficient EPM solution, helps *FG* supporting the BI aspects, such as choosing and approving the budget for the next fiscal years. By carefully designing the process flow structure and logics, coupled with the technical capabilities of SAP BW/HANA, automation is maximized, resulting in a positive impact on business operations, improving the operational efficiency.

The next scheduled steps start from the testing of the Consolidation application with its related eventual fixes. Furthermore, *FG* already commissioned another job order, regarding the development of a reporting application performing the linked analysis of purchase requests (*RDA*) and corresponding purchase orders (*ODA*). At the time of writing, the application is at the high level design phase.

In conclusion, this work emphasizes the significance of leveraging SAP BW/4HANA, to model the financial processes. SAP BW/4HANA technical capabilities allows to improve the data management, and coupling the technical aspects with informed design choices, for large enterprises such as *FG*, can help enhance business operations.

Bibliography

Anthony, R. N., Govindarajan, V., Hartman, F. G. H., Kraus, K. and Nilsson, G., 2014. Planning and budgeting, *in* '*Management Control Systems*', first european edn, McGraw-Hill Education, chapter 9.

Chaudhuri, S. and Dayal, U., 1997. 'An overview of data warehousing and olap technology', *SIGMOD Rec.* **26**(1), pp. 65-74.
URL: *https://doi.org/10.1145/248603.248616*

Foley, E. and Guillemette, M., 2010. 'What is business intelligence?', *IJBIR* **1**, pp. 1-28.

Park, Y.-T., 2006. 'An empirical investigation of the effects of data warehousing on decision performance', *Information & Management* **43**(1), pp. 51-61.
URL: *https://www.sciencedirect.com/science/article/pii/S0378720605000194*

Pugliese, A. and Parbonetti, A., 2017. Reporting intercorporate investments and consolidation of wholly owned subsidiaries with no differential, *in* '*Advanced Financial Accounting*', 11th edn, McGraw-Hill Education, chapter 2, pp. 60-77.

SAP, 2015. '*SAP ECC Help Portal (Documentation)*', `https://help.sap.com/docs/SAP_ERP`. Accessed: 2022-11-13.

SAP, 2019. '*SAP HANA Platform Help Portal (Documentation)*', `https://help.sap.com/docs/SAP_HANA_PLATFORM`. Accessed: 2023-04-15.

SAP, 2020. '*SAP BPC Help Portal (Documentation)*', `https://help.sap.com/docs/SAP_BPC_VERSION_BW4HANA`. Accessed: 2022-09-30.

SAP, 2021. '*SAP BW/4HANA Help Portal (Documentation)*', `https://help.sap.com/docs/SAP_BW4HANA`. Accessed: 2022-09-30.

SAP, 2023. '*SAP Analytics Cloud Help Portal (Documentation)*', `https://help.sap.com/docs/SAP_ANALYTICS_CLOUD`. Accessed: 2023-02-01.