TU WIEN Informatics

Activity Based Budgeting: A conceptual model and implementation in R.

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Wirtschaftsinformatik

eingereicht von

Endi Resuli, BSc
Matrikelnummer 01429774

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.-Prof. Mag. Dr. Walter Schwaiger, MBA

Wien, 25. November 2024

Endi Resuli	Walter Schwaiger

TU WIEN Informatics

Activity Based Budgeting: A conceptual model and implementation in R

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Business informatics

by

Endi Resuli, BSc
Registration Number 01429774

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.-Prof. Mag. Dr. Walter Schwaiger, MBA

Vienna, November 25, 2024

Endi Resuli | Walter Schwaiger

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Endi Resuli, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 25. November 2024

Endi Resuli

Kurzfassung

Ein wichtiges Budgetierungssystem in Unternehmen ist das Activity-Based Budgeting (ABB). Obwohl es in der Praxis weit verbreitet ist, fehlt eine klare Definition der Activity-Based Budgetierung-Domäne und eine gemeinsame Sprache, um die Kommunikation von domänenspezifischem Wissen zwischen verschiedenen Domänen, darunter Buchhaltung, Finanzen, Managementkontrolle und Informationstechnologie (IT), zu erleichtern. Diese Arbeit schließt diese Lücke, indem sie eine konzeptionelle Definition für diese Domäne liefert und eine allgegenwärtige Sprache etabliert, die Verständnis und Integration zwischen den verschiedenen Domänen ermöglicht.

Diese Arbeit entwickelt drei Schlüsselartefakte zur Lösung dieser Probleme: ein konzeptionelles Modell, ein operatives Modell und eine Implementierung in R-tidyverse. Das konzeptionelle Modell bietet eine hochrangige Abstraktion der ABB-Domäne. Es definiert die Komponentenbeziehungen und -prozesse durch UML-Diagramme und etabliert gleichzeitig eine gemeinsame Sprache für eine effektive Zusammenarbeit. Das operative Modell übersetzt die präsentierten Abstraktionen in einen detaillierten Arbeitsablauf. Es umfasst Variablen, Beziehungen, Entscheidungsprozesse und wichtige Akteure. Die Implementierung in R demonstriert die Verwendung des konzeptionellen und operativen Modells und bietet einen Prototyp für eine zukünftige Implementierung.

Es wird eine fortgeschrittene Design Science-Forschung (DSR) verwendet, einschließlich einer Use-Case Evaluierungsmethode, um die Übereinstimmung der vorgeschlagenen Modelle mit den ABB-Domänenanforderungen zu demonstrieren. Sie wird als fortgeschrittene Design Science Forschung bezeichnet, weil sie neben der klassischen Design Science auch die wichtigsten Schritte der Domänen-Engineering-Methodik und des Domain-Driven Designs (DDD) umfasst. Die Domain-Engineering-Methodik konzentriert sich auf die Entwicklung wiederverwendbarer Artefakte und Kenntnisse für eine bestimmte Domäne. Dieser Prozess verwendet diese Methodik, indem er ihre drei wichtigsten Schritte umfasst: Analyse, Design und Implementierung einer bestimmten Domäne. Andererseits stimmt der Aufbau einer allgegenwärtigen Sprache, die der ABB zugrunde liegt, mit der DDD überein.

Diese Arbeit legt eine Grundlage für die Integration von ABB-Prozessen in verschiedene organisatorische Kontexte, indem sie eine Definition der ABB-Domäne und einen strukturellen Ansatz für ihre Anwendung liefert, das interdisziplinäre Verständnis verbessert und den Weg für weitere Forschung im akademischen und industriellen Umfeld ebnet.

Abstract

An important budgeting system in enterprises is activity-based budgeting (ABB). While it is widely applied in practice there is the lack of a clear definition of the activity-based budgeting domain and a lack of a common language to facilitate the communication of domain specific knowledge across different domains, including accounting, finance, management control, and information technology (IT). This thesis addresses this gap by providing a conceptual definition for this domain and establishing a ubiquitous language that enables understanding and integration across the different domains.
This thesis develops three key artifacts to solve these problems: a conceptual model, an operational model, and an implementation in R-tidyverse. The conceptual model provides a high-level abstraction of the ABB domain. It defines the components relationships and processes through UML diagrams, while establishing a common language for effective collaboration. The operational model translates the presented abstractions into a detailed workflow. It includes variables, relationships, decision-making processes, and key actors. The implementation in R demonstrates the use of the conceptual and operational model and provides a prototype for future implementation.
An advanced design science research (DSR) is used, including a use-case evaluation method to demonstrate the alignment of the proposed models with the ABB domain requirements. It is called an advanced design science methodology, because along with the classic design science it also covers the key steps of the domain engineering methodology and the domain-driven design (DDD). The domain engineering methodology focuses on developing reusable artifacts and knowledge for a specific domain. This process employs this methodology by including its three key steps, the analysis, design and implementation of a specific domain. On the other hand building a ubiquitous language underlying the ABB aligns with the DDD.
This thesis establishes a foundation for integrating ABB processes into diverse organizational contexts by providing a definition of the ABB domain and a structural approach for its application, enhancing interdisciplinary understanding, and paving the way for further research in both the academic and industrial setting.

Contents

CHAPTER 1

Introduction

1.1 Background

In today's fast-paced and highly competitive business environment, effective financial planning and forecasting are crucial for ensuring operational efficiency and achieving strategic objectives. The manufacturing industry faces significant challenges in optimizing production processes, managing resources, allocating costs, and enhancing profitability in fluctuating and complex market conditions. Traditional budgeting methods, such as incremental budgeting do not provide the necessary flexibility and details needed in these dynamic environments resulting in financial inefficiencies.
Activity-based budgeting (ABB) is the process of planning and controlling the expected activities of the organization to derive a cost-effective budget that meets the forecast workload and agreed strategic goals [JAB12]. It emerged as an extension of Activity Based Costing (ABC) principles and its development can be traced through several publications in the management accounting field. Activity costing was first mentioned in [Sta71]. In his book, he introduced the concept of assigning costs to specific activities rather than just to products or departments aiming to provide more accurate cost information. His emphasis on activities as core units of analysis influenced the evolution of ABC, which was formalized in the later years. Johnson & Kaplan [HTJ87] first highlighted the limitations of traditional cost accounting systems in the face of modern manufacturing needs. At the time many production companies decreased subcontracting and increased internalized processes, resulting in more complex centralized environments. The existing accounting systems were not sufficient for a clear view of centralized manufacturing. Therefore, they introduced Activity-Based Costing concepts as a solution. This method would provide more accurate and relevant information by tracing the costs of specific activities rather than allocating them based on direct labor and machine hours. Kaplan [Kap88] further developed the principles of ABC by arguing about the benefits of providing detailed and accurate cost information.

Activity-based costing for manufacturing was first described in [RC91], where he provides a detailed framework for designing cost management systems using ABC. He includes all technical aspects of its implementation, including activity identification, cost driver analysis and cost allocation methods. His work laid the groundwork for applying ABC to budgeting process.
The concept of Activity-Based Budgeting was first introduced in [Bri94]. Brimson defines ABB as a budgeting method that allocates financial resources based on the costs of activities necessary to produce outputs and deliver services. This method is an extension of ABC, which assigns costs to activities based on their consumption of resources. He shifts the focus from the traditional budgeting of departments and cost centers to specific activities and cost drivers. ABB calls for the identification of all resources required for an activity such as labor, materials, and overhead so the costs are allocated based on the consumed resources of the output. It aligns financial resources with the outputs of the organization to ensure that the budgets are directly related to the organization's objectives and performance targets. By displaying a clear connection between resource consumption and activities ABB enhances transparency and accountability in budgeting, giving decision-makers a clear view on the cost allocation. Brimson presents the steps for the implementation of ABB in a company. The process starts with the identification of activities, then resource costs are assigned to activities, cost drivers are determined, activity costs are calculated, and the budget is developed by aggregating the costs of all activities. This way the budget reflects the actual resource requirements. The last steps include monitoring and adjusting the process to ensure resources are used effectively and efficiently.
ABB was then applied to other sectors such as the hospitality industry [EK21] , the manufacturing industry [MHAK21], the public sector [MM21] and the service industry [Pie14]. ABB's application in various sectors, has demonstrated its effectiveness in improving cost control, resource allocation, and operational efficiency.Despite the wide use of ABB in different sectors, there is a lack of a clear conceptual definition and a shared understanding of the ABB process in literature.
Samonas' Financial Planning and Forecast Model, developed by Michael Samonas [Sam15], represents a comprehensive framework for long-term financial forecasting, incorporating advanced risk analysis techniques, scenario planning, and activity-based budgeting. This model emphasizes the importance of integrating uncertainty projections and sensitivity analysis into financial plans to enhance their accuracy and reliability. The starting point of this forecasting framework is the prediction of the sales volumes and not revenues like in most cases. The financial planning is rooted in the sales volumes to connect the financial costs to the operational ones, which is also the core of the activity-based approach. This gives the framework an activity based character. Currently, the framework is implemented and described as a stand-alone Excel implementation.

1.2 Problem Statement

Activity-based budgeting is a widely used budgeting system in enterprises due to its effectiveness over traditional budgeting methods. Despite the practical significance there is a lack of a clear definition of the ABB domain in literature. This absence creates challenges for integrating the system in budgeting practices in different contexts. The current challenges include the lack of a common definition of the process underlying the ABB domain, therefore making it difficult to adapt the process to different use-cases. There are also communications barriers between the different departments, the accounting, finance and management control, and the IT domain, which creates a barrier to implement this process, whether in spreadsheets or in a programming environment.
As companies progress in digitalization and adopt advanced integrated information systems like the enterprise resource planning (ERP), the integration of ABB into the systems remains challenging.
The misalignment between financial planning models and operational systems can result in budget overruns and strategic discord. For example case studies in large manufacturing firms have shown that such misalignment highlight the need for a more integrated approach. [LRM10] [SCH03].

1.3 Objectives of the study

The primary objectives of this thesis are to develop a structured conceptual definition underlying the ABB domain and to provide a common language for the process that is understood by all the involved domains and the IT domain. This work aims to address the gap in the literature and seeks to:

1. Define the ABB Domain: Develop a clear and structured conceptual definition of the ABB domain by identifying its key components, processes and relationships.
2. Establish a ubiquitous language: Create a common language to bridge the communication gaps and improve collaboration between accounting finance, management control and the IT domain.
3. Develop a conceptual model: Use UML diagrams to conceptualize the ABB domain.
4. Build an operational model: Include the operational metrics into the conceptual model and describe the process including the variables, decision-making and workflows involved in the process
5. Implement and validate the models: Implement the conceptual and operational model and demonstrate their applicability and alignment to the ABB requirements through a use-case.

These objectives aim to fill the gap in the definition of the ABB in literature, provide a common understanding, a structured approach for the implementation and a demon-

stration of a prototype of the implementation as an example for future applications and research in both academic and industrial contexts.

1.4 Research questions

To achieve the objectives defined above, this study seeks to answer the following questions:

1. Which elements must be included in a conceptual model to conceptualize the ABB system effectively?

2. Which details of the conceptual model must be operationalized to provide a complete workflow of the budgeting system based on the use-case?

3. Which architecture, entities and functions must the implementation in a programming environment use to represent the operational model accurately?

The research questions will guide the study and provide a focus on the challenges associated with the conceptual definition and the definition of a common language for the process of activity-based budgeting.

1.5 Significance of the study

This study aims to address the gap in literature regarding the definitions of the activity-based budgeting domain and the lack of a common language to describe the process underlying ABB and bridge the accounting, finance, management control and IT domains. This research contributes to the field of budgeting and management control by providing a conceptual model to serve as a definition and a common understanding, and an operational model to describe the step-by-step process including the mathematical steps as a design for the implementation. It further enhances the understanding of the ABB by offering a clear understanding of its components processes and relationships through the conceptual and operational model. By implementing and validating these models in a programming environment, it showcases the application of the models, offers an example of the practical implementation process as a blueprint and final artefact as a prototype of the implementation in a robust programming environment. This approach addresses challenges in aligning the budgeting system with the IT domain, providing an alternative solution to the current spreadsheet application. The findings of this research are significant for both the academic field and the industry. In the academic field it fills the gap in the definition of the ABB domain. For industry practitioners it offers practical insights and a blueprint to implement the ABB in diverse organizational contexts. Ultimately, this research lays a foundation for future applications and advancements in the use of activity-based budgeting.

1.6 Methodology

To guide our research, we use an extended design science methodology, the domain-driven design science methodology. The design science methodology used is based on the steps defined in [KP04] and by respecting Hevner's guidelines [Hev07] on using design science. This methodology focuses on designing, developing, and evaluating artifacts to solve real-world problems, ensuring our conceptual and operational model is theoretically grounded and practically applicable. In this thesis the design science methodology is modified into a domain-driven design science methodology. Domain-driven design (DDD) [GP11] is a software design philosophy that puts an emphasis on a deep understanding of the business domain and aligning software models closely to the real-world processes and the concepts within that domain. It is particularly useful in complex software systems where understanding and modeling the business logic accurately is critical. One of most important principles of DDD is the creation of a shared language between developers and domain experts i.e. a ubiquitous language. One of the main goal of this thesis is also establishing a ubiquitous language between stakeholders, and the specific budgeting domain and the IT domain, that is why the domain-driven design fits well to our goal. To evaluate the model we combine the domain-driven design science methodology with a use-case evaluation. Based on the design science research methodology described by Peffers et al, as also pictured in 1.1, we started with identifying the problem and motivation and showing its importance in section 1.2. We defined the objectives of solutions in 1.3 and 1.5. The design and development of the artifact include the conceptual and operational model of the activity-based budgeting process. In the next step of this research method, to demonstrate their applicability and a suitable context, we implement the financial model in a programming environment, R-tidyverse, using the conceptual and operational model as a guidelines. To evaluate the models, Samonas' Steelco use-case is first implemented in Excel, then used to instantiate the conceptual and operational model, and then the same data is used in our R implementation. The presented models and implementation are validated by achieving aligned results to the spreadsheet implementation at the end. We also evaluate the implementation process qualitatively based on metrics such as time needed and discuss on the complexity of implementing the use cases, troubleshooting and ease of data input. By developing a conceptual model, designing an operational model and implementing these models in a programming environment the steps of domain engineering, analysis, design and implementation are also covered. Using these steps, domain engineering [KCH+90] focuses on developing reusable artifacts and knowledge for a specific domain, which supports the objectives of this study.

1.7 Thesis Structure

The thesis is structured as follows:

1. Introduction: Provides the background, problem statement, objectives, research questions, significance, and methodology overview.

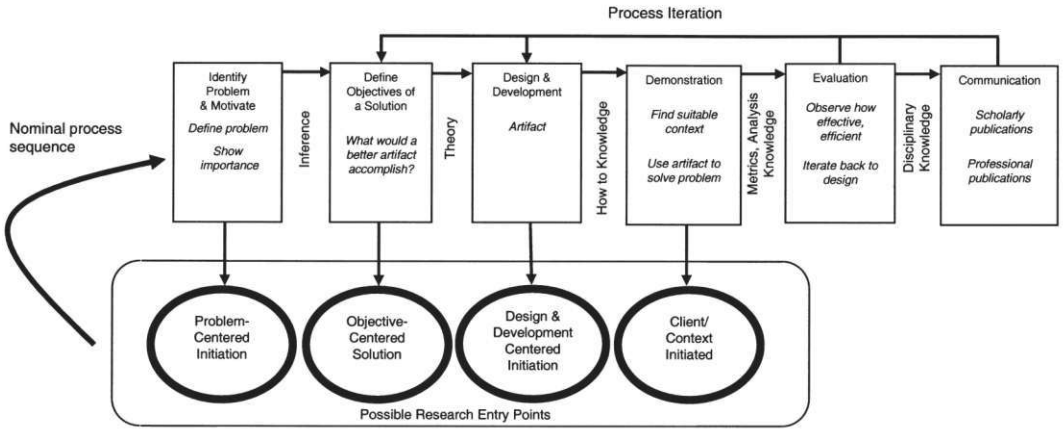


Figure 1.1: Design Science Process Model [KP04]

2. Requirement Specification: Outlines the requirements and criteria for building a conceptual model, an operational model and implementation in R that represent the process underlying the ABB domain.

3. Conceptual Modeling: Develops several UML class diagrams that abstractly represent the budgeting process in ABB, including key components and their relationships in different granularities.

4. Operational Modeling: Explains the decision making process, the parameters and mathematical calculations used in the budgeting process, using a specific use-case from the literature and detailing the functions and attributes required for integration.

5. Implementation in R tidyverse: Describes the implementation process using R and tidyverse, including validation and demonstration through a real-world use case from the literature.

6. Evaluation: Validates the models and implementation by applying the use-case.

7. Conclusion: Summarizes the findings, discusses the implications, and provides recommendations for future research.

1.8 Scope and Limitations

The scope of this study is limited to defining a theoretical conceptual definition and common language for the process underlying the activity-based budgeting. It defines a generic model that can be applied to different sectors, but its evaluation is limited to one use-case from the budgeting domain literature. The conceptual model is first defined for

an universal use, while the operational model is provided using only the specific use-case. The implementation is provided as a prototype to show the structure of the entities and functions, but it is not built to be integrated in a specific enterprise resource system. This thesis also conceptually shows the possible integration of the model in the structure of an enterprise at the level of ERP systems based on the Enterprise-control system integration standard (IEC 62264 [iec16]), but it does not include a specific and detailed guide for this integration, nor does it investigate possible specific ERP systems.

CHAPTER 2

Activity based budgeting: Requirement specification

In this section, we outline the requirements for building a conceptual and operational model, and implementing the budgeting system in a robust programming environment. The conceptual model will serve as an abstract definition of the budgeting process and as a common language to define the process. The operational model operationalizes the abstract definition by defining the workflow, decisions, calculations, and variables of the process using a specific use-case. The implementation then uses the operational model as a blueprint and provides a working example. It defines the architecture of the code, the needed entities, and functions for completing the budgeting activity. Activity-based budgeting focuses on identifying and evaluating costs based on the activities that produce them. We also consider the integration this financial model into the company's existing information system to gain insight into cost drivers. The goal is to improve the allocation of resources, and therefore enhance forecasting and budgeting capabilities.

2.1 Requirements: Conceptual Model

As described in [Rob06] the requirements of a conceptual model include validity, utility, feasibility, credibility, and simplicity. Validity is confirmed by ensuring that the model accurately represents the system it is modeling. Utility shows the usefulness or the value the model brings. Feasibility is the practicality of using the model with given resources. Credibility is the degree to which the model is trusted by stakeholders when implemented in a real-world environment. Lastly, the model should be as simple as possible and should still capture the essential elements of the system [Joh24]. The simplification reduces the complexity and makes the model easier to understand use and maintain. In [Joh24] a conceptual model is described as an abstraction used by software engineers to document and convey the essential characteristics of an application before it is developed.

It focuses on the system's structure and behavior and leaves out details on user interfaces or implementation. It is a high-level representation that helps in understanding and communicating the design's core ideas.

Based on these scientifically grounded requirements we build the requirements for our own conceptual model. Firstly, the conceptual model must be a valid, accurate and generic representation of the forecasting process underlying activity-based budgeting and must be validated by acting as a guideline for the development of an operational model and implementation. It must show that it is useful for the implementation of the budgeting process and be practical in its application to a use case. The credibility is not included as a requirement in our use case, because this paper focuses on the theoretical and operational aspects of the model rather than the practical application in a real world setting. The conceptual model must also be a simple representation of the budgeting process that includes all the essential components present, but it should also leave out details that complicate the model and reduce its simplicity. As also described in [Joh24] the conceptual model must include all the key concepts, relationships, rules and constraints, data sources, processes and operations, user roles and interfaces, and the system boundaries. The key concepts include all the entities, the financial statements, models, and actors that interact with the framework. The relationships that connect the entities and the workflow must also be included. The guidelines and restrictions that guide the behavior and interactions of the entities have to be defined. Another requirement is the definition of the sources of the data, and the data flow and process to achieve an outcome. This should be however kept simple and in a high-level view to reduce complexity. In this granularity, the users and stakeholders must also be defined. The legal requirements that apply to the accounting domain are already considered, because the data will be retrieved from the ERP system that is based on the legal requirements of the location where the business operates. Lastly, the limitations of the system must be established, by discussing on the scope of the model, what is and is not included.

2.2 Requirements: Operational Model

An operational model is described in [AK11] as a detailed representation that defines how the system operates in practice. It translates the abstract concepts from the conceptual model into actionable components, outlining the process, data flows, interactions, and behaviors that are necessary for the system to function. It is used to guide the development, implementation, and management of the system while ensuring that it meets the intended requirements and operates effectively. The requirements of an operational model are typically centered around ensuring that the model is practical, implementable, and reflects the real-world constraints and conditions under which the system will operate. This includes specifying detailed processes, workflows, data flows, interactions, and system behaviors. The model must guide the actual development and implementation of the system while ensuring that all components work together effectively to meet the goals and

requirements of the system [IRB13]. The requirements for an operational model focus on ensuring that the model can be implemented and used within the intended domain. The model must detail how the abstract concepts of the conceptual model are executed, including specifying processes, data sources, and interactions in a higher granularity. It must be practical, feasible, and capable of guiding the actual system development, ensuring that all the components align with the domain needs. Practical refers to the model being implementable with the available resources and feasible that it should be realistic and achievable to build and maintain with the given constraints.
Based on the established requirements outlined in the referenced sources, we develop requirements that apply specifically to the operational model of the chosen ABB use-case from the literature, SteelCo [Sam15]. This will ensure that our model aligns with both the theoretical foundations and practical considerations. To build the operational model of the activity-based budgeting process in Steelco, we need to ensure the model builds on the abstract concepts from the conceptual model and provides concrete instances of these concepts. It must describe all the relationships, algorithms, variables that are contained in the financial statements separately and explain the decisions on these variables, their transitions, algorithms, and the calculations that take place. The operational model must be practical to use for the implementation, serving as a guide. It must be implementable using the described criteria and environmental constraints and it must reflect constraints and conditions that are realistic and close to a real-world system.

2.3 Requirements: Implementation

The implementation of the financial forecasting framework in R tidyverse requires both requirements for ensuring a well-structured clean code and requirements that specify the connection of the implementation to the model. To produce good code, we require modularity, readability, consistency, maintainability, and testability [Hut09]. This means that the implementation must be structured to be modular and divided into self-contained units. It must be readable, i.e., clean and easy to understand. There must be a uniform structure throughout the whole code. It should be built in a structure that is easy to maintain, extend, and facilitate testing. These requirements apply primarily to the implementation structure. Additionally, thorough documentation is crucial to ensure that the code can be easily understood and maintained in the future.
In this situation, we have a model-driven design of the architecture of the implementation, where we first model our case study through an operational model and then implement the model into a programming environment based on the model. The implementation must therefore accurately represent the model it is based on, including the structure, entities, variables, algorithms, and relationships. The structure of the implementation must be consistent with the structure defined in the model. The entities and variables must be traceable to the model and named uniformly. The relationships, forecasting models, and algorithms must also represent the ones defined in the model and compute the same steps.
In this use case, because the implementation is built to demonstrate the financial

forecasting framework itself without integrating it into a larger real-world system, the data sources can be simply imported as given. The retrieval of the information will not be included in the implementation. There are further requirements that only affect the process in the programming environment. The entities must be built in tidy data structures that allow for easy manipulation. The data structures must support time series because that is the main key or index representing the financial statements. Lastly, the implementation must contain every step as the Excel implementation and produce nearly identical results to the Excel implementation.

CHAPTER 3

Activity based budgeting: Conceptual modeling

3.1 Conceptual Model: Introduction

As defined in [GP18], conceptual models are a simplified representation of reality that focuses on the essential elements and their relationship in a specific domain. They are composed of abstract concepts and serve as a framework for understanding, analyzing, and communicating complex phenomena. They help organize and structure information and guide researchers in their study by providing a basis for drawing conclusions and making decisions. They work as ontologies, representing individual entities and the organizational units they belong to. This approach aids in understanding a model and applying conclusions back to the modeled subject.
Samonas' financial forecasting model is an ABB-based budgeting model that incorporates many components to produce accurate forecasting. The model uses financial statements, like income statements, balance sheets, and cash flow statements and various forecasting models, such as regression models and time series analysis among others. All these components are integrated into one system and depend on each other for a correct implementation of the budgeting activity. Samonas [Sam15] introduces the use of these components as Excel functions and procedures, concentrating on their implementation through the available options in the Excel environment. Thus, a lot of the used logic and functions are carried out in the background, remain hidden and are only applicable in the Excel environment. Additionally Samonas' describes the process through a specific company and its needs. The conceptual definition of this activity-based budgeting process is missing in literature. In our effort to conceptually define this ABB budgeting system to allow a common understanding of all involved financial domains and the IT domain, the used logic in Excel has to be defined in a conceptual generic way. This conceptual

model serves as a common language of the ABB forecasting system. It generalizes the ABB system and makes it universally applicable.

This conceptual model of the ABB process using Samonas' Financial Forecasting model simplifies the understanding by breaking down the process into components and illustrating the relationships between them. It visualizes each of these components, the data flow, as well as the forecasting models used for building the forecasted or pro forma financial statements. The visualization of these processes will be done using Unified Modeling Language (UML) notational methodology, as defined in ISO/IEC 19501 [Int05]. The used symbols and their definitions are defined in 3.1. We will leave out the cardinalities for all the models in this use case, because they decrease readability without supporting the communication.

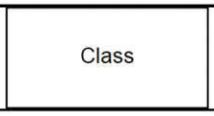
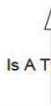
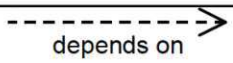
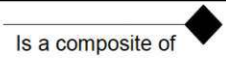
Symbol	Definition
	Defines a package, a collection of object models, state models, use cases, and other UML models. Packages are general-purpose grouping mechanisms used to organize semantically related model elements. In this document a package is used to specify an external model, such as a production rule model, or a reference to another part of the model.
	Represents a UML class of objects, each with the same types of attributes. Each object is uniquely identifiable or enumerable. No operations or methods are listed for the classes.
	An association between elements of a class and elements of another or the same class. Each association is identified. Can have the expected number or range of members of the subclass, when '*n*' indicates an indeterminate number. For example, 0..*n* means that zero or more members of the subclass can exist.
	Generalization (arrow points to the super class) shows that an element of the class is a specialized type of the super class.
	Dependence is a weak association that shows that a modeling element depends on another modeling element. The item at the tail depends on the item at the head of the relationship.
	Aggregation (made up of) shows that an element of the class is made up of elements of other classes.
	Composite shows a strong form of aggregation, which requires that a part instance be included in at most one composite at a time and that the composite object has sole responsibility for disposition of its parts.

Figure 3.1: UML notation used [Int05]

Furthermore, the conceptual model also includes the information needed for implementing this forecasting model in a real-world setting. It first presents the information sources from an ERP system based on the Enterprise Control System Integration standard [iec16], then visualizes the data flows, processes, and interactions between different system components. In addition, the conceptual model also serves as a communication

tool and ubiquitous language for stakeholders such as financial analysts, decision-makers, and also programmers to discuss, refine, and adapt the forecasting approach to their business needs. It facilitates easier communication and alignment among team members by providing a clear shared understanding of the model, therefore enhancing the overall effectiveness in the process. In this conceptual model, we will start by presenting the information sources, continue by giving a more general overview of the process, starting from the most important components, the financial statements and their forecasting, and then will move on to a more detailed view of each of these components.

3.2 Information Sources

Most enterprises use enterprise resource planning (ERP) systems for a centralized management of the information of all involved departments. The budgeting system can be used along with the information from the ERP system or integrated into it. The integration of the implementation of this ABB process in an ERP system in the company means the information can flow automatically through a query that either builds the financial statements from system data or if the financial statements are already present in the system as entities, a query that selects the financial statements of the historical years.

The ERP System is the primary data source for the Activity-Based budgeting system. According to the IEC 62264 standard [iec16], data integration and exchange between enterprise and control systems are essential for comprehensive financial planning and operational efficiency. There is key information from ERP systems that is required for the implementation of the activity-based budgeting system based on the use-case.
General ledger accounts provide a detailed financial overview. They contain all financial transactions in an organization. The general ledger accounts provide the overall financial data needed to assign costs to activities to support cost allocation and ensure that the budget reflects actual financial conditions. These accounts additionally contain the balance sheet and income statement information needed as input for the financial budgeting model. The balance sheet includes assets liabilities and equity to show the financial position at a point in time and is essential for evaluating the financial stability and liquidity. The income statement including, revenue and expenses, gives information about the profitability of the organization. Sales information also originates from the general ledger accounts. It is a crucial input in the forecasting model. They also provide insight into the financial situation and performance of the company, which is necessary for determining forecasting assumptions. In the IEC 62264 [iec16] framework this data falls under financial resource management, which integrates financial information needed by both the enterprise and production management.
Transactional data including inventory and material management data, sales and distribution and human resources data is important in tracing the origin of the costs to ensure the activity based budgeting uses an accurate cost structure. Operational management data focuses on the management of production and maintenance and the quality operations and includes production data, maintenance data and quality management data which

also support the allocation of the activity cost.
The mentioned data make out the critical data for the implementation of the financial model depending on the use case. We showed the necessary sources of data and their purpose and the context where they are defined in the standard. Considering the standard, the financial forecasting model would be integrated in the level 4, which represents the ERP system. In figure 3.2 we visualized the extension of the model in blue to display the positioning of the financial model in the operations planning entity. This shows the activity based nature of the model. It uses operating information to conclude in business numbers and then through decisions based on the results of the forecasting activity, capacities of activities and the operations plans can be determined.

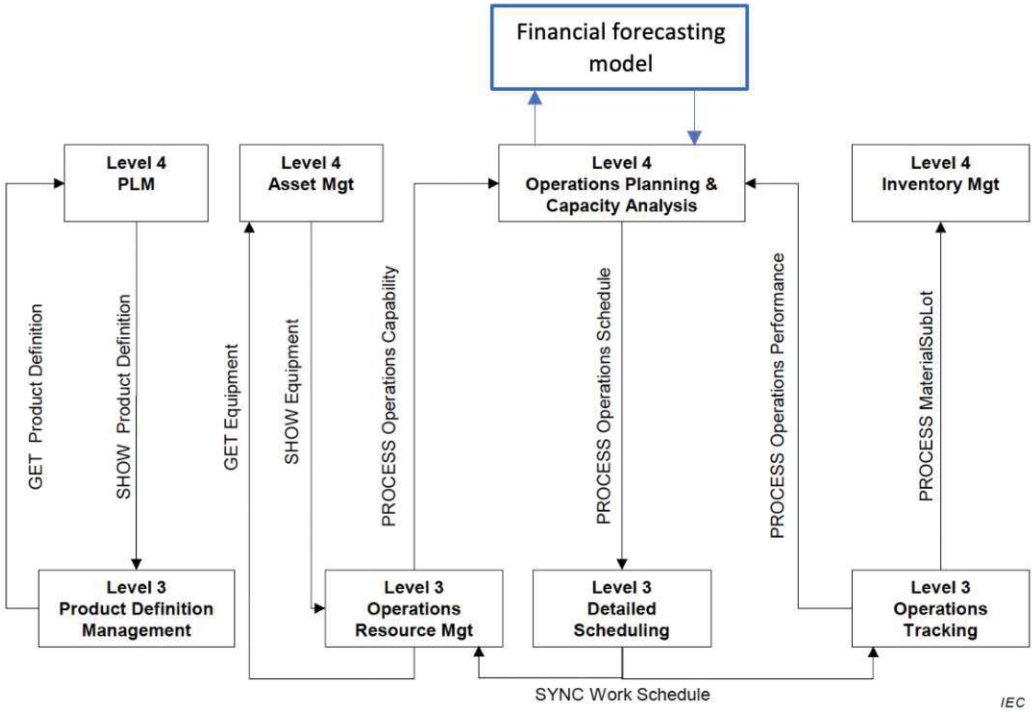


Figure 3.2: Coordination planning and operations process [iec16]

3.3 General overview: Financial Forecasting Process

We will start with a simple, high-level overview of the process to provide a clear and comprehensive understanding of the budgeting model. This approach aims to establish a solid ground knowledge of the process, to ensure understanding when adding to the complexity and granularity in the next sections. At its core, this budgeting system includes three steps, as shown in 3.3.

The process starts with the input of historical financial statements. The forecasting

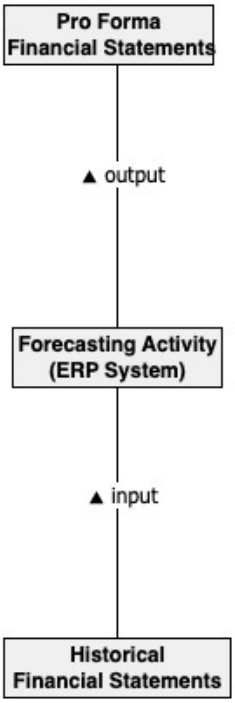


Figure 3.3: Financial Statements Forecasting model: General view

activity takes these as input and applies forecasting mechanisms to the historical data through historically based assumptions and forecasting models to generate the most probable forecast pro forma financial statements.

3.4 General overview: Financial Statement Entities

The historical financial statements are made up of the historical income statement, the historical balance sheet, and the historical cash flow statement. The historical income statement and balance sheet comprise of data that are already available in the ERP system. In the system, they are summaries of historical transactional data. Therefore we do not include the steps of building the historical statements and consider them as given. The UML diagram in 3.4 is constructed from left to right according to the chronology of the process steps. As a starting point, the given income statement is input into the forecasting model. The income statement forecasting model includes different forecasting models or functions for forecasting different variables. It includes future assumptions or parameters for the models, which are derived from internal historical transactional data, or external market information to build an accurate, reality-based pro forma income statement. The balance sheet is forecasted similarly to the income statement, with the difference that some of its variables depend on the income statement. That is why we include an input connection from the income statement in the balance

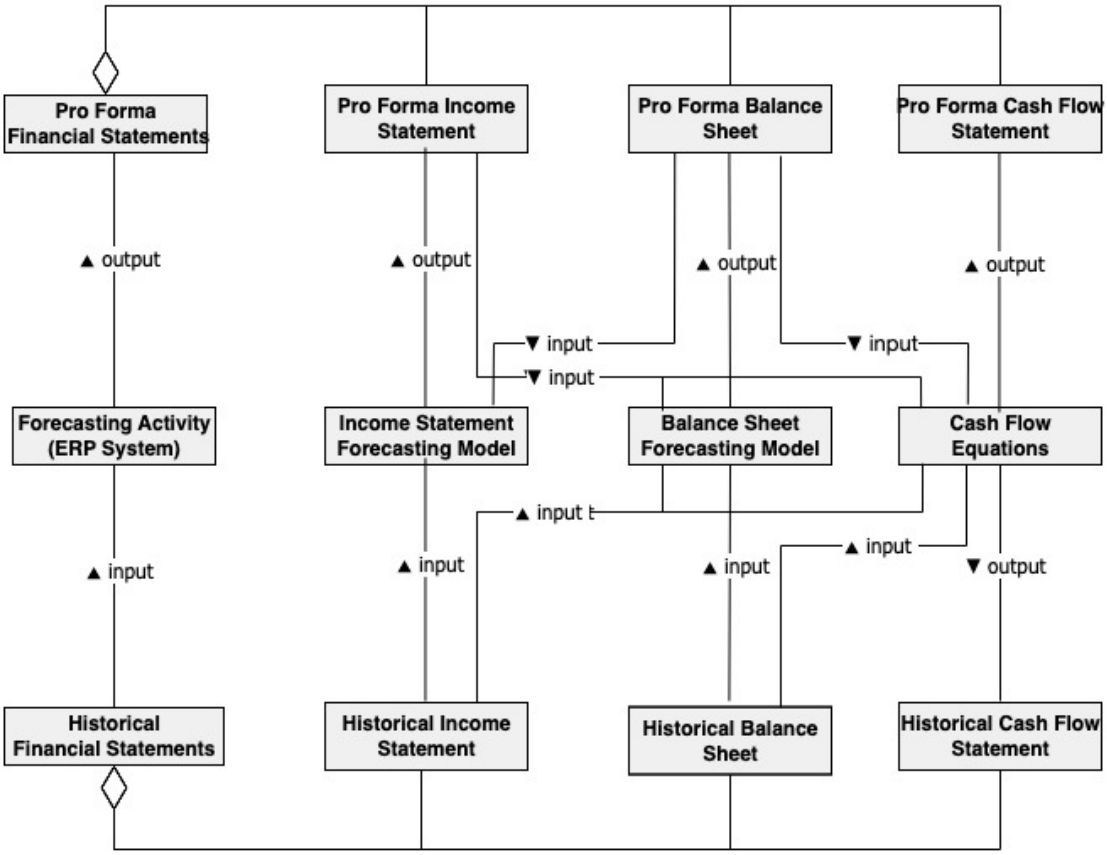


Figure 3.4: Financial Statements Forecasting: General view

sheet forecasting model to show its involvement. Analogically, the balance sheet model uses several forecasting models and appropriate assumptions or parameters for future forecasting. After completing the forecasting of all possible balance sheet variables, the results are then again input into the income statement forecasting model. This represents a circular relationship involving variables from both statements that depend on each other. Specifically it involves the relationship between the net interest expenses from the income statement and the short-term debt in the balance sheet, and the variables that depend on these two variables.

On the other hand, the cash flow statement is made up of information from both the income statement and the balance sheet. This statement could also be built within the ERP system and considered as given, but with consideration to Samonas' workflow and the benefits it provides in the implementation we build the historical cash flow statement using the other two historical statements. The benefits are related to the validation of the cash flow statement calculations. By constructing the complete cash flow statement we

can check if the historical data is calculated correctly and conclude that the future values are also correct. The pro forma cash flow construction model does not use forecasting models but is derived using the forecast values of the pro forma income statement and pro forma balance sheet using the mathematical relations between the variables.

3.5 Detailed view: Financial Statement Forecast

In the detailed view of the financial statement forecast, we discuss each of the financial statement transitions in 3.4 separately and in more detail, going through all the used models and parameters needed to forecast the pro forma financial statements from the historical financial statements. We will present a generic view of each of these components and then show how these universal models are applied to the use-case that was presented in Samonas' study.

3.5.1 Detailed view: Income Statement

Income Statement: Generic Model

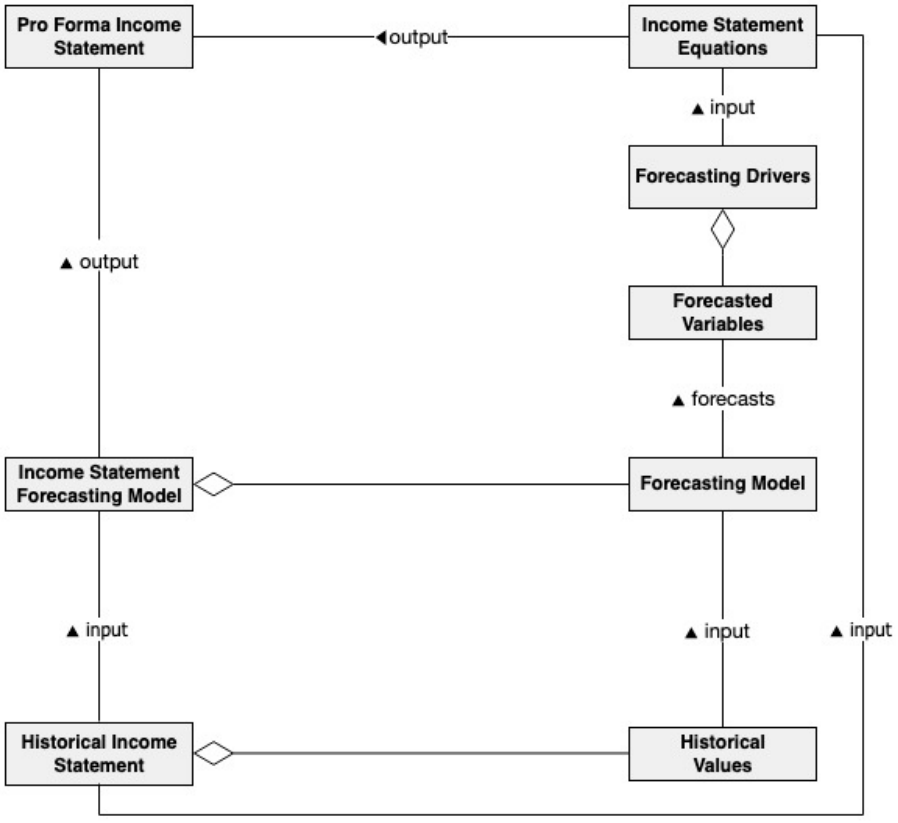


Figure 3.5: Income Statement: Generic view

The income statement is a financial statement that shows the revenue, expenses, gains and losses of a company. It is a critical component of the financial forecasting model. By examining historical income data and applying forecasting methods, the model provides insight on the future revenue, expenses and profitability. The forecasting of the income statement is done based on some key forecasting drivers. The forecasting drivers are the variables that need to get forecast while the rest of the income statement gets calculated through these drivers based on mathematical relationships.
This process is shown in a generic form in 3.5. This diagram shows an overview of the forecasting process including the forecasting entities. In this granularity the diagram can be used as a guideline to implement the income statement forecasting process in any company environment with any requirements. The process starts with the historical income statement. Several historical values are then chosen as forecasting drivers. The forecasting drivers will be the variables that will determine the full income statement. For each of these drivers the financial planner and analyst will have to determine an appropriate forecasting model and assumptions for the future progression of these variables. These assumptions will serve as parameters for calibrating the forecasting models. This step is unique to every scenario, because the assumptions and forecasting models are based on historical variable trends, internal company influences and possible external ones. The historical data progression is an appropriate indicator of the future progression, if it shows a pattern and no other forces are expected to affect the financial variables in the income statement. Internal influences include factors like management decisions for budgeting and investments, organizational philosophy and goals, historical performance, internal policies, resource availability, financial health etc. Several of the affecting factors, such as the future investment plans, are known or can be drawn by the management and can contribute to a more appropriate selection of forecasting models. External influences include changes in the economic environment where the company operates, regulatory changes, market conditions such as competition, demand and other chain factors, technological advancements, social trends that affect consumer behavior, political climate and unexpected occurrences like natural disasters. These can be even harder to predict, but financial planners, managers and other professionals in a company may have more information, or connections to professionals in different fields that make it possible to make an informed decision about these numbers. The forecasting models for the forecasting drivers should be chosen carefully, taking into account all plausible internal and external influences. Forecasting models can include, classical forecasting models such as, time series models, auto-regressive models, integrated moving average etc. [SM84], but even simpler models like the naive or average model and also hybrids of these. Once the forecast variables are calculated they can serve as forecasting drivers to build the complete income statement. These drivers make up the first income statement information. Then one by one, based on their mathematical dependencies, i.e. the structural equations, the rest of the variables in the income statement have to be calculated to build the concluding pro forma income statement.

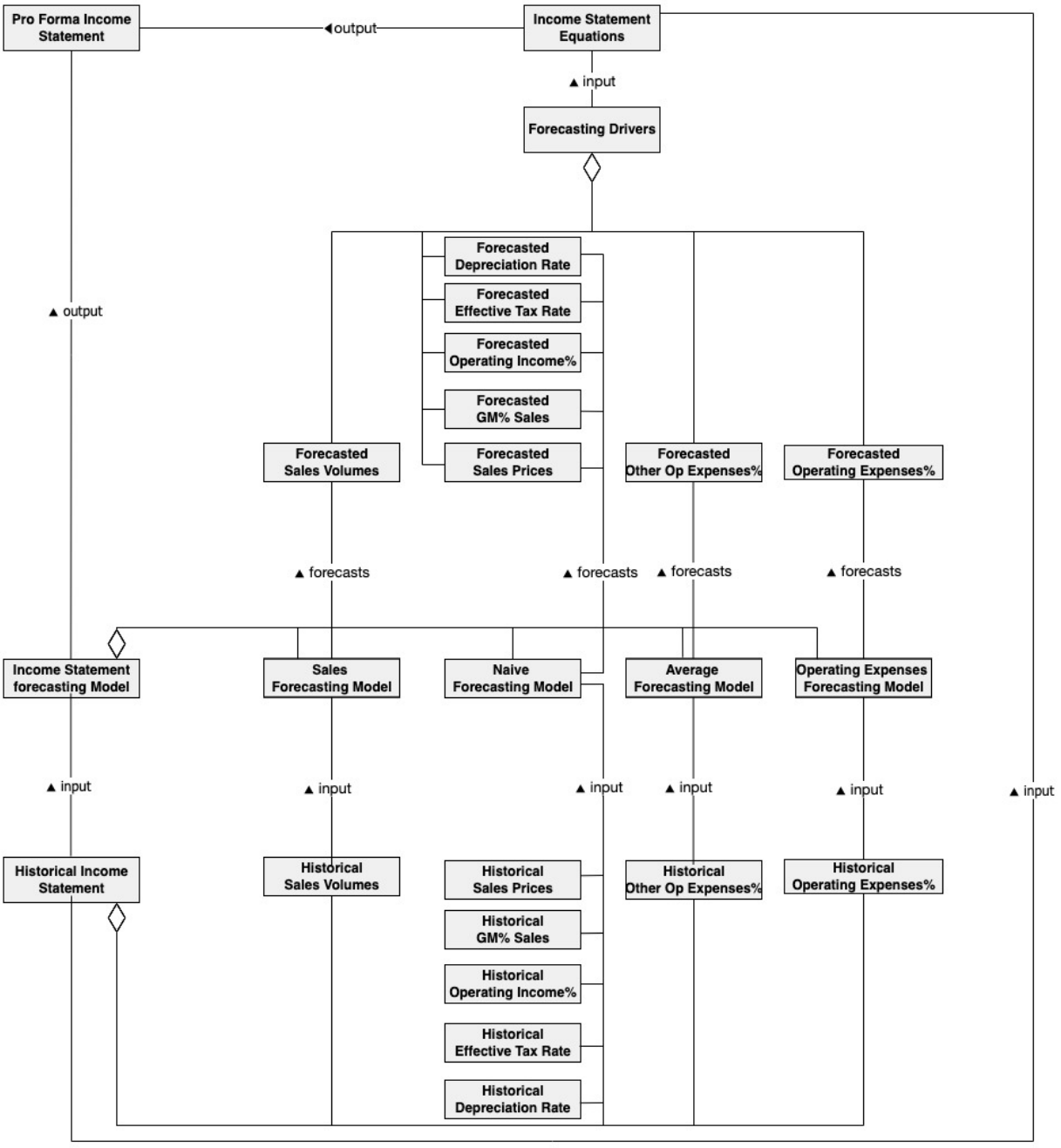


Figure 3.6: Income Statement: Detailed view

Income Statement: Specific Use-Case Model

To provide an example of how the diagram in 3.5 can be applied to a specific use case, we describe its application to the use case Samonas used in his book.
We will now go through the steps described in the generic model described in the previous section and extend it with concrete instantiations of the previous elements in the UML diagram. In the SteelCo use-case that was used to describe this financial forecasting framework in [Sam15], the financial planner and analyst analyzes historical data, collects information from management, the treasury department, accounting, commercial, procurement, technical, and HR departments to configure the model correctly. He collects information about the different product groups, sales, profitability, operating cycle, debt structure, depreciation, investment plans and macroeconomic indicators. Direct communication is important for each of these steps if the model is being implemented in an Excel spreadsheet from scratch. By using or integrating the model to an ERP system, many of these steps would be obsolete. Most of this information could be transferred directly to the forecasting framework if they are stored in the ERP system. Some of the forecasting models could also be automatically chosen by an algorithm, based on the historical data and predefined criteria, decreasing the need for human involvement. However this will not be handled in this paper. The integration with an ERP system for retrieving input data and the application of the forecasting models through the ERP system is also shown in our instance of the income statement's generic diagram in 3.5. Starting from the left, the diagram in 3.6 shows the same generic structure. The historical income statement is input in the income statement forecasting model and the pro forma income statement is produced. The forecasting drivers for the income statement include the sales volumes, sales prices, gross margin, operating income, effective tax rate and depreciation rate, operating and other operating expenses. These are the variables that will be forecasted and then used to calculate the rest of the future variables in the income statement. Most of the variables are calculated using the percentage of sales forecasting method combined with several other models. The percentage of sales forecasting model calculates each of the variables' percentage of sales value and uses forecasting functions to predict these, to then calculate the actual values of these future functions. As we can see in the diagram, there are four forecasting models used along the percentage of sales model.
The first variables to be forecast, the sales volumes, use the rate of change function combined with the linear trend function or linear regression. The future change in sales volumes is calculated as a linear continuation of the historical change in the sales volumes. The historical values for the sales prices, gross margin, operating income etc., seem to stagnate or not show significant fluctuation, therefore the naive forecasting model was chosen. Other operating expenses are calculated through the average forecasting model, because the values seem to decrease and increase again, staying close to an average value. On the other hand the operating expenses are predicted through a custom function, which takes into account a fixed part that is only affected by the inflation rate and a variable part, which changes based on the sales volume. The results of these functions build our forecasting drivers. These drivers are then input in the mathematical representations of

the relations of the variables in the income statement, the income statement equations, to derive the variables that did not serve as drivers. Once the calculations are concluded, the pro forma income statement is finalized.

3.5.2 Detailed view: Balance Sheet

Balance Sheet: Generic Model

The balance sheet is another important financial statement, which provides an overview of the company's financial situation in a certain point in time. It includes details about the company's assets, liabilities and equity, which display the company's net worth. The balance sheet is another crucial component of Samonas' financial forecasting model. The framework offers insight into the future of the assets, liabilities and equity positions by analyzing the historical balance sheet values. Similarly to the income statement, the forecast balance sheet is constructed using certain variables that serve as forecasting drivers. These drivers are forecast using forecasting models and then used as input in the relationship functions between the variables to calculate the rest of the variables in the balance sheet. Additionally in this process, after all the possible variables are forecast in the balance sheet, there are a few variables left to forecast which depend on the values of the variables in the income statement and vice versa. This forms a circular relationship. This process is presented in a generic form in 3.7. This generic UML diagram provides an overview of the forecasting process of the balance sheet that can then serve as a process definition and guideline for implementing this forecasting process based on the company's requirements.

The first step to building the balance sheet forecast is to build the forecasted pro forma income statement. This step has to happen first, because some of the variables in the balance sheet depend on several variables in the income statement for the forecast. Once all the steps in Section 3.5.1 are completed, the values of the historical and pro forma income statements are input in the balance sheet forecasting model. In this diagram we see vertically from the bottom to the top the transitions from historical entities, to forecasting model entities and at the top the forecasted entities. Horizontally from left to right, we start with the balance sheet forecasting model and in the right, using the same structure, we define the forecasting drivers and the respective models they use. The forecasting models and their parameters need to be determined based on the historical data and other internal or external influences that might apply to the variables in the future. The forecasting models are determined by the financial planner and analyst by analyzing historical data and collecting information on possible influences on the drivers in the future. The historical data is analyzed to determine patterns that might continue in the future. The information about the possible future influences can come from professionals in other department, market research, connections etc. The gathered knowledge from both the historical data and factors that might influence the balance sheet variables are used to determine the forecasting models or functions and future values. These forecast values then take the role of forecasting drivers, to calculate the rest of the variables and fill out the pro forma balance sheet through the balance sheet

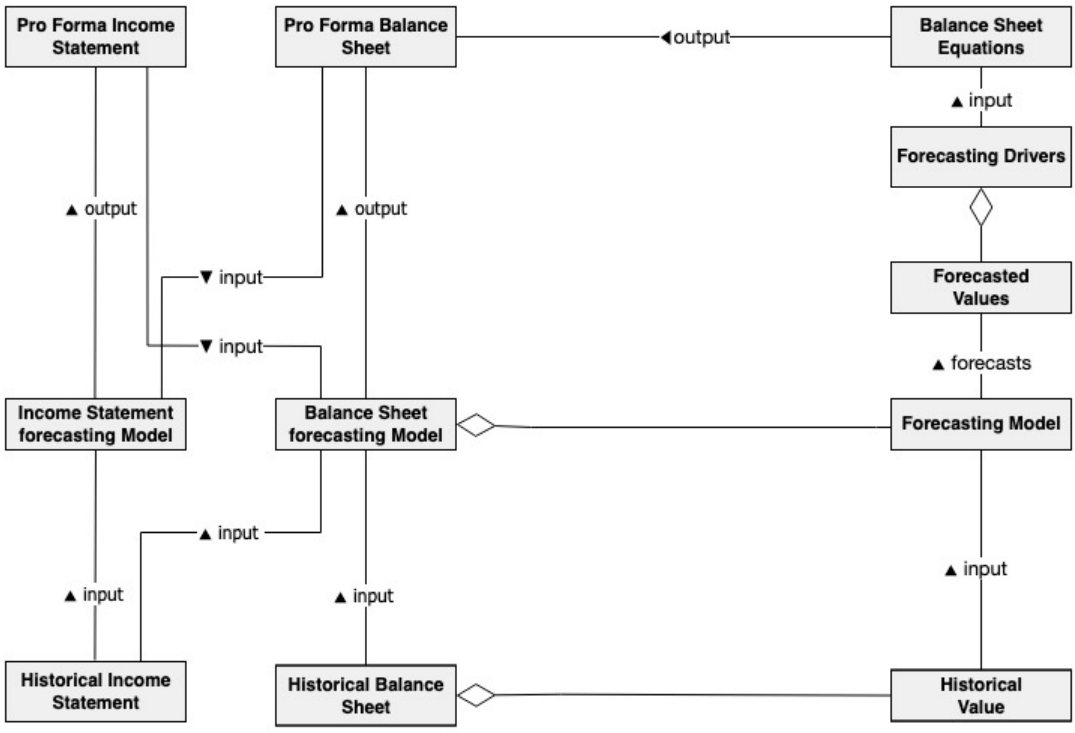


Figure 3.7: Balance Sheet: Generic view

equations. After the possible variables are forecast, the edge going from the pro forma balance sheet back to the income statement forecasting model represents the circular relationship.

Balance Sheet: Specific Use-Case Model

In this section we continue with the instantiation of our generic diagrams using the Steelco use-case with which the frameworks implementation in excel was presented in [Sam15]. This is to provide an example of how this generic balance sheet model can be applied to a specific use case. We will now go through all the steps presented in the generic version of the diagram and extend it with actual forecasting models. The detailed view of the balance sheet forecasting UML diagram begins with the forecasting of the pro forma income statement. Some of its variables are needed for the calculation of several variables in the pro forma balance sheet. The pro forma balance sheet calculation entities are shown next similarly to the generic version. The balance sheet calculation begins with the historical data. The historical data is analyzed and information is gathered on their possible progression in the future years. The forecasting models are derived from this knowledge. To produce the pro forma balance sheet, the forecasting drivers have to first get selected and forecast. For this purpose the derived forecasting models are used. In this case the forecasting drivers include the days sales outstanding (DSO), the days

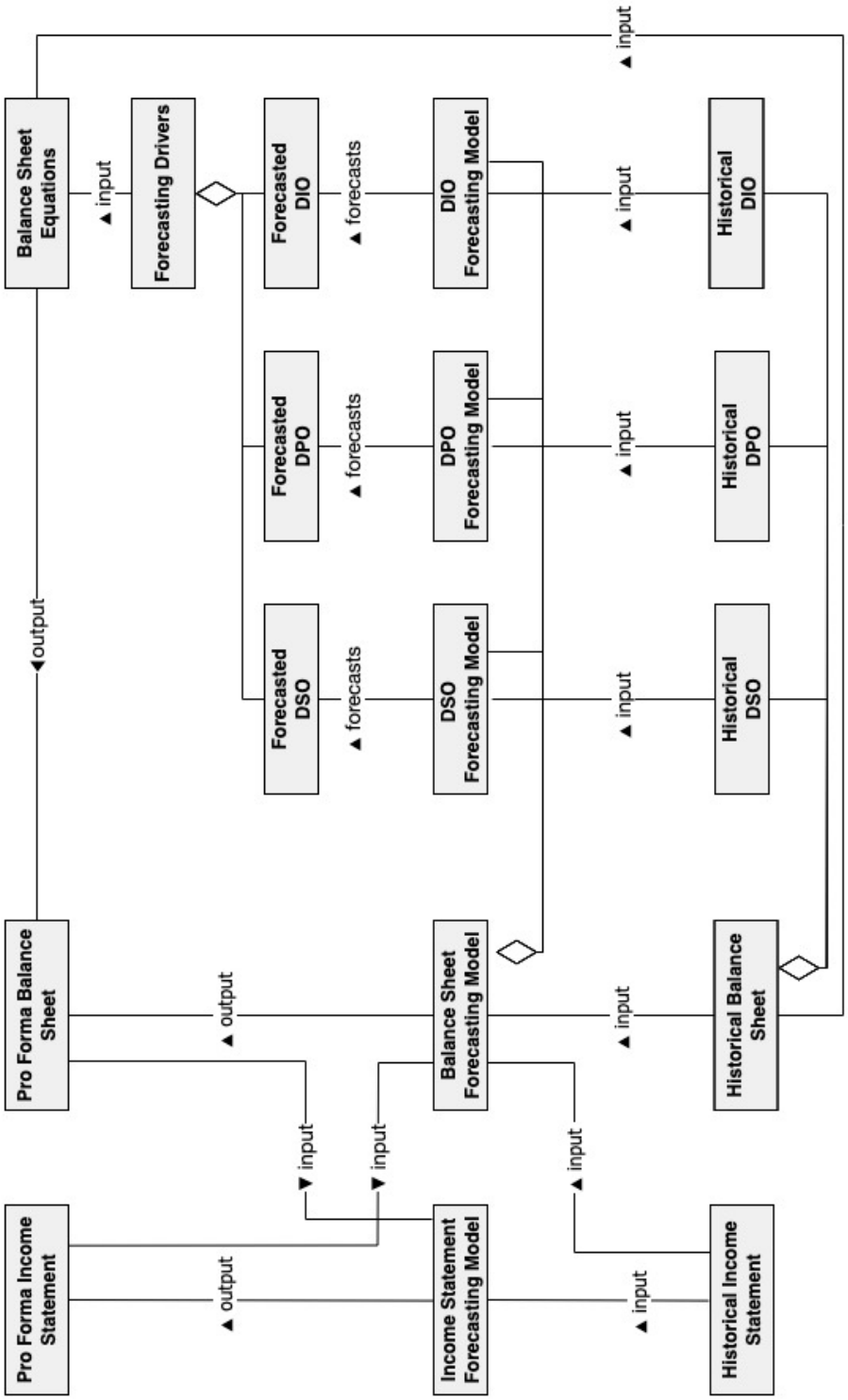


Figure 3.8: Balance Sheet: Detailed view

payable outstanding (DPO) and the days inventory outstanding (DIO).
The DSO measures the average number of days it takes for the company to collect a payment after a sale has been made. The DPO measures the average number of days it takes a company to pay its suppliers. Lastly, the DIO measures the number of days it takes a company to sell its entire inventory. These three variables are forecast based on information from the respective managers, the commercial and purchasing managers. Based on the target inventory management, credit and purchasing policy of the company the values are set for the future years. Knowing the values of the three drivers for the future years, the rest of the pro forma balance sheet can be calculated using their structural equations. After all possible variables are forecast in the balance sheet the edge leading back to the income statement forecasting model, depicts the circular dependency of the remaining variables in the income statement and balance sheet. This involves the relationship between the net interest expenses in the income statement and the short-term debt, and other variables in between.

3.5.3 Detailed view: Cash flow Statement

Once the pro forma income statement and balance sheet are calculated, the cash flow Statement can be derived. There is now a switch in perspective from the accounting domain ,where the balance sheet and income statement hold importance, to the financial domain. Just like the domain the thinking shifts. In finance it is not thought in terms of profit and loss, but in terms of cash flows. The depreciation reduces the profit but does not affect the cash flow, therefore it is not included in this statement.

To construct the pro forma cash flow statement there are no forecasting models used, because the values in the cash flow statements are all dependent on the income statement and balance sheet. The values of both the historical and pro forma income statement and balance sheet are needed to build this financial statement. In the general view of the forecasting diagram, in 3.4, we visualize the process of building the historical and forecast financial statements and then calculate the cash flow statements. In figure 3.9 we derive the illustration that only includes the process of building the cash flow statement. We do not include the forecasting of the income statement and the balance sheet in this figure, because at the time when the cash flow is constructed, the other financial statements are already forecast and the pro forma income statement and balance sheet are already built. Once the pro forma financial statements are built, they and their respective historical financial statements are then input in the cash flow model entity. The historical financial statements are necessary to build both the historical cash flow statement and the pro forma cash flow statement. The cash flow includes measurements that show change, therefore information on the measurements in the previous years is necessary. The cash flow equations entity includes all the calculations needed to derive the cash flow statement from the other two statements. The same calculations apply to both the historical and the pro forma statement and represent the dependency functions between variables in these three financial statements. The detailed explanation of this process, including the concrete economic relations and calculations will be provided in

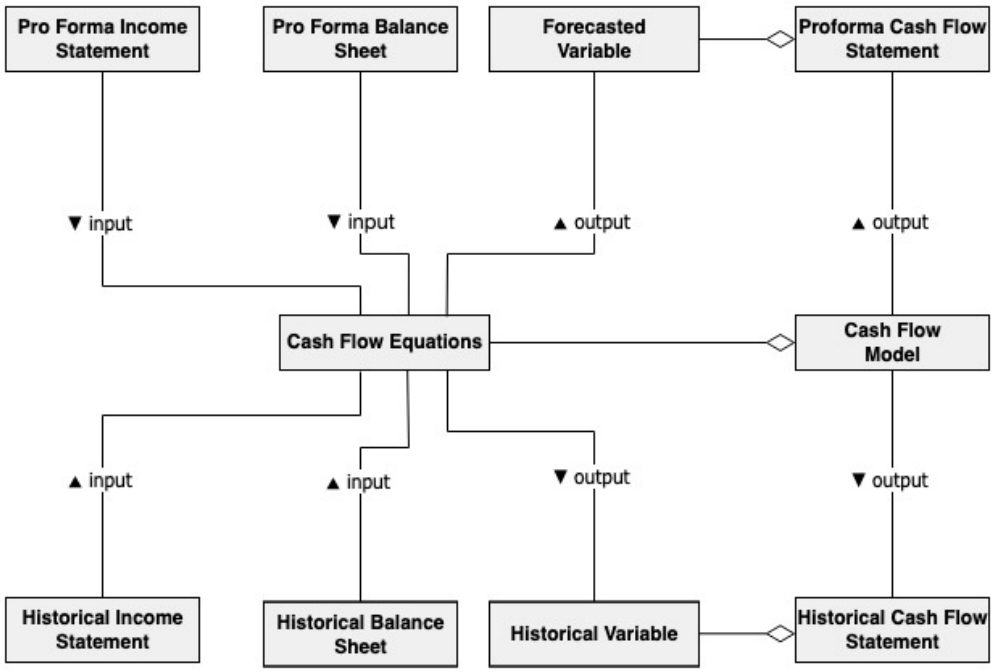


Figure 3.9: Cash Flow Statement: Detailed view

the operational model in section . Once all the historical and pro forma variables in the cash flow statements are derived, the forecasting process of the financial statements is finalized.

CHAPTER 4

Activity based budgeting: Operational modeling

In this section we present a detailed explanation of the processes, entities, forecasting models and calculations that are required for building the pro forma financial statements in the form of an operational model. This is done to understand how the different components in the financial forecasting framework cooperate, how decisions are made, which functions are used and why. Therefore the purpose of providing this operational model is to understand the mathematical side of this framework and extend the blueprint that started with the conceptual model in another dimension to aid the implementation of this framework in the IT domain. Here we will try to adapt the background logic used by Samonas in the spreadsheet implementation to the conceptual model we defined in 3. Therefore we will be using the same use-case and the defined language in the conceptual model to display this. Here we go though the same steps that we described abstractly in the conceptual model and go into more detail on the forecasting models and actual variable calculations. We will start in the beginning based on the chronological steps this framework goes through, from the historical financial statements, the decision-making on the forecasting models, we will explain the forecasting models, how they are chosen and applied, and the equations of the dependencies that are present in the three financial statements.

4.1 Income Statement

The income statement consists of information about the revenue, expenses and net income. In table 4.2 we see the complete list of measurements that are included in the income statement. The sales information that is also necessary for the calculations is included in the sales table 4.1. First we start with the sales information, the volumes, prices and turnovers of two groups of products. There can be more or less product or service groups

depending on the use case. To show it generically we named the groups as product group A and product group B. In [Sam15] the groups are split as long and flat products. In the income statement variable table 4.2 there is information on the expenses, cost of goods sold, gross margin, operating income and expenses and the different earnings.

Sales Variable	**Variable Name**
Product Group A Volume	xSA
Product Group B Volume	xSB
Total Volumes	xST
Product Group A Price	pA
Product Group B Price	pB
Product Group A Turnover	RevTA
Product Group B Turnover	RevTB

Table 4.1: Sales variables

Income Statement Variable	**Variable Name**
Total Turnover (Sales)	RevT
Cost Of Goods Sold without Depreciation	COGs_wo_Depr
Gross Margin without Depreciation	GM_wo_Depr
Other Operating Income	OpInc_o
Operating Expenses	OpEx_wo_Depr
Other Operating Expenses	OpEx_o_wo_Depr
Provision for Bad Debts	Prov_badDebt
Net Interest Expenses	IntEx_net
Depreciation	Depr
EBITDA	EBITDA
EBIT: Profit/ Loss before interest and tax	EBT
EBT: Profit / Loss before tax	EBT
EAT: Profit/ Loss after tax	EAT
Income Tax Expense	IncTaxEx

Table 4.2: Income statement variables

In order to provide a deeper overview in the process of forecasting the income statement we will assign the historical values of the SteelCo case study, also used by Samonas in his introduction to the framework in [Sam15]. The historical values in thousands can be seen in table 4.4 for the income statement and in table 4.3 for the sales variables. These values will be used to decide on the forecasting model to produce the pro forma income statement.

Income Statement	2011	2012	2013
Long Products Volume MT	200.000	170.000	150.000
Flat Products Volume MT	70.000	60.000	50.000
Total Volume MT	270.000	230.000	200.000
Long Products €/MT	700	670	650
Flat Products €/MT	650	680	700
Turnover Long Products	140.000	113.884	97.500
Turnover Flat Products	45.500	40.800	35.000

Table 4.3: Sales: SteelCo historical values

Income Statement	2011	2012	2013
Total Turnover (Sales)	185.500	154.684	132.500
Expenses (before I,T,D,A)	-176.353	-145.687	-126.218
COGS: Cost Of Goods Sold w/o D.	170.289	139.679	119.913
Gross Margin w/o Depreciation	15.211	15.004	12.588
Other Operating Income	3.555	3.493	2.539
Operating Expenses	-8.912	-8.516	-8.182
Other Operating Expenses	-707	-985	-663
EBITDA: Earnings before I,T,D,A	9.147	8.997	6.282
Provision for Bad Debts	0	0	0
Net Interest Expenses	-4.755	-4.470	-6.357
Depreciation	-2.195	-2.067	-2.235
EBT: Profit / Loss before tax	2.197	2.459	-2.310
Less: Income Tax Expense	-725	-812	
EAT: Profit/ Loss after tax	1.472	1.648	-2.310
Depreciation	-2.195	-2.067	-2.235
EBIT: Earnings before Interest and Tax	6.952	6.930	4.047
Net Interest Expenses	-4.755	-4.470	-6.357
EBT: Profit / Loss before tax	2.197	2.459	-2.310
Income Tax Expense	-725	-812	
EAT: Profit/ Loss after tax	1.472	1.648	-2.310

Table 4.4: Income Statement: SteelCo historical values

4.1.1 Income Statement: Forecasting Functions

The process of building the pro forma financial statements starts with the calculation of the pro forma income statement, because the other financial statements are dependent on it. Using the detailed diagram 3.6 as a map we will start the more detailed explanation with the bottom left entity and then move to the right and up as we progress. The first entity is the historical income statement. It represents the starting point of this process. The historical income statement can be derived from transactional data in the ERP

system, as well as the sales information. For this framework we collect the data for the income statement and the sales attributes, the volume and prices, for the last 3 years. This historical data is then used to determine forecasting models for further variables. The starting point is the prediction of the sales volumes and not revenues like in most cases. The financial planning is rooted in the sales volumes to connect the financial costs to the operational ones, which is also the core of the activity-based approach. This is because the costs of operations fall on these volumes and grow in proportion with them. The products or services the company sells can be grouped in different product groups. The financial planner should decide on future sales volumes based on the volumes in the previous years and the market trends that may affect them. These trends have to be quantified in business numbers and applied to the forecasts. There are many forecasting models that can be chosen depending on the results of the analysis. To keep it simple Samonas' chose to use the rate of change of the total sales volumes and then seeing how in the last three years the values decrease almost linearly, he uses the linear regression function.
We start with calculation of the historical rate of change of sales volumes xST_ROC in 4.1. This function applies to the first three time indexes which represent the historical years.

$$xST\_ROC_t = \frac{(xST_t - xST_{t-1})}{xST_{t-1}} \quad t \in [1,3] \tag{4.1}$$

Then continue with the calculation of the forecast rate of change of sales volumes, which is computed as the sum of last known value with the change of the last two known values in 4.2. This function applies to the future years, which in this case are the time indexes between four and seven.

$$xST\_ROC_t = xST\_ROC_{t-1} + (xST\_ROC_{t-1} - xST\_ROC_{t-2}), \quad t \in [4,7] \tag{4.2}$$

Once the rate of change is calculated, the total future sales volumes can be calculated through the rate of change and the previous years total sales volume.

The calculation of future total sales volume is then computed in 4.3.

$$xST_t = (xST_{t-1} * (1 + xST\_ROC_t)) \tag{4.3}$$

The total sales volumes are forecast, now they need to be allocated to the chosen product groups. In this case study there are two product types, long and flat products. We will label the long products with a A and the flat products with an B and the total with a T in the equations for simplicity. To assign a part of the total sales volumes to each of these product types, the last years ratio of the product types to the total sales volume is used. For a different use case another function can be used for this assignment depending on the historical trend and future expectations. The functions then can be seen in 4.4 and 4.5.

$$ratio\_xSA\_inT_t = \frac{xSA_t}{xST_t} \tag{4.4}$$

$$ratio\_xSB\_inT_t = \frac{xSB_t}{xST_t} \tag{4.5}$$

Therefore the future value of the sales volumes for each product type are calculated as the ratio times the total sales volume in 4.6 and 4.7.

$$xSA_t = ratio\_xSA\_inT_t * xST_t \tag{4.6}$$
$$xSB_t = ratio\_xSB\_inT_t * xST_t \tag{4.7}$$

To calculate the total turnover or revenue (RevT) we now need to forecast the prices and multiply them to the sales volumes. Since the prices do not change much in the last three years, the future prices are set as the price of the last historical year. Then they are multiplied to the sales volumes for each product as shown in equations 4.8 and 4.9 to derive the sales turnover for each product type. The sum of these turnovers is then the total turnover.

$$RevTA_t = pA_t * xSA_t \tag{4.8}$$
$$RevTB_t = pB_t * xSB_t \tag{4.9}$$

For the rest of the drivers the percentage of sales value is used. This transforms all variables to the part of the total turnover that they represent. This function is shown for any variable (Var) in 4.10. This allows to also take into account the changes in sales when looking at the rest of the variables. It puts them into context and shows a more realistic trend. That is why the forecasts of the variables here are made using the percentage of sales of the values.

$$Var_t = Var_t / RevT_t \tag{4.10}$$

The next key driver to be calculated is the gross margin without depreciation. Considering the historical gross margin we calculate its percentage of sales. Analyzing this, the financial planner takes the gross margin of the last year as a value for the future. The same happens for the other operating expenses as a percentage of sales and for the effective tax rate.

For the operating expense the FP&A decides to consider a part of the operating expenses as fixed, changing only based on the inflation, and the rest of the operating expenses as variable, meaning it changes based on the sales volume. In this case the value of OPEX is calculated before the percentage of sales because it makes more sense for the financial analyst. The forecasting function he chooses is shown in 4.11.

$$\begin{aligned} OPEX\_wo\_Depr_t = {} & 30\% * OPEX\_wo\_Depr_{t-1} * xST_t \\ & + 70\% * OPEX\_wo\_Depr_{t-1} * (1 + inflation_t) \end{aligned} \tag{4.11}$$

The missing drivers now are only the depreciation and the net interest expense. The depreciation will be calculated as a percentage of gross fixed assets later. The net interest expense will also be calculated later as part of a circular function with other variables in the balance sheet. For now, we use the already forecast drivers and continue to fill

out the rest of the values in the income statement. These values will then be needed to calculate the balance sheet and in return, the calculated balance sheet variables will be used to calculate the net interest expenses. In this current state, the sales table is complete as seen in 4.5 and the income statement state can be seen in 4.6.

Sales	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**
Long Products Volume MT	200.000	170.000	150.000	133.092	120.447	111.137	104.515
Flat Products Volume MT	70.000	60.000	50.000	44.364	40.149	37.046	34.838
Total Volume	270.000	230.000	200.000	177.456	160.596	148.183	139.354
Rate of change	0%	-14,8%	-13,0%	-11,3%	-9,5%	-7,7%	-6,0%
Long Products €/MT	700	670	650	650	650	650	650
Flat Products €/MT	650	680	700	700	700	700	700

Table 4.5: Sales: SteelCo historical and forecast values

Income Statement	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**
Turnover Long Products	140.000	113.884	97.500	86.510	78.291	72.239	67.935
Turnover Flat Products	45.500	40.800	35.000	31.055	28.104	25.932	24.387
Total Turnover (Sales)	185.500	154.684	132.500	117.564	106.395	98.171	92.322
Rate of change		-16,6%	-14,3%	-11,3%	-9,5%	-7,7%	-6,0%
Expenses (before I,T,D,A)	-176.353	-145.687	-126.218				
COGS: Cost Of Goods Sold w/o D.	170.289	139.679	119.913				
Gross Margin w/o Depreciation	15.211	15.004	12.588				
% of Sales	8,2%	9,7%	9,5%	9,5%	9,5%	9,5%	9,5%
Other Operating Income	3.555	3.493	2.539				
% of Sales	1,9%	2,3%	1,9%	1,9%	1,9%	1,9%	1,9%
Operating Expenses	-8.912	-8.516	-8.182	-8.020	-7.904	-7.831	-7.801
% of Sales	-4,8%	-5,5%	-6,2%				
OPEX Rate of change				-2,0%	-1,5%	-0,9%	-0,4%
Other Operating Expenses	-707	-985	-663				
% of Sales	-0,4%	-0,6%	-0,5%	-0,5%	-0,5%	-0,5%	-0,5%
EBITDA: Earnings before I,T,D,A	9.147	8.997	6.282				
Provision for Bad Debts	0	0	0				
Net Interest Expenses	-4.755	-4.470	-6.357	-4.443	-4.040	-3.778	-3.795
Depreciation	-2.195	-2.067	-2.235	-2.237	-2.248	-2.259	-2.269
EBT: Profit / Loss before tax	2.197	2.459	-2.310				
Income Tax Expense	-725	-812	0				
EAT: Profit/ Loss after tax	1.472	1.648	-2.310				

Table 4.6: Income Statement: SteelCo historical and forecast values

4.1.2 Income Statement: Structural Equations

After forecasting the forecasting drivers for the income statement, now it is time to calculate the rest of the variables to fill out this forecast financial statement for the next 3 years. The income statement equations are displayed in the UML diagram 4.1. Here we extended the diagram to show the transition of the historical income statement variable values through specific variable equations from the accounting domain to derive the pro forma income statement variable. The variable can be any of the variables that are left

to calculate. The next four variables we calculate are the gross margin in 4.12, the other operating income 4.13 and the other operating expenses 4.14. These three variables depend directly on the forecast percentage of sales drivers in the previous sections. They get calculated for each year through the future total revenue times the percentage of sales of each variable.

$$GM\_wo\_Depr_t = P\_GM\_wo\_Depr_t * RevT_t \tag{4.12}$$

$$OpInc\_o_t = P\_OpInc\_o_t * RevT_t \tag{4.13}$$

$$OpEx\_o\_wo\_Depr_t = P\_OpEx\_o\_wo\_Depr_t * RevT_t \tag{4.14}$$

With these three variables and the operating expenses that were forecast previously, the EBITDA can be derived as the incomes plus the negative expenses, as seen in equation 4.15.

$$EBITDA_t = GM\_wo\_Depr_t + OpInc\_o_t + OpE\_o\_wo\_Depr_t + OpEx\_wo\_Depr_t \tag{4.15}$$

The net interest expense is calculated next, through its relationship to the interest expense account and the interest income gained from the cash deposits in 4.16. The interest expense is the sum of interest expenses that are paid for both the short and long-term debt minus the cash deposit interest that is earned. The long term debt is known, and the minimum amount of cash needed for the company to function is also an information known by the treasury. The short-term debt is the only variable that stays unknown. This will be calculated later on, once we have calculated all the needed variables for the circular function.

$$\begin{aligned} NetIntEx_t = {} & Cost\_Debt\_lt_t * Debt\_lt_t + \\ & Cost\_Debt\_st_t * Debt\_st_t - Cash\_dep\_inc_t * Cash_t \end{aligned} \tag{4.16}$$

To calculate the depreciation we first find the depreciation rate of the historical years. This is calculated as the depreciation divided by the gross fixed assets in 4.17. The value of the depreciation rate does not change much in the last 3 years, so the forecast depreciation is set as the last known depreciation rate.

$$Depr\_rate_t = \frac{Depr_t}{Asset\_gf_t} \tag{4.17}$$

To forecast the depreciation expense account the balance sheet variable, gross fixed assets, has to be known first. This is why we make an exception and calculate the fixed assets variable in this section, before we start with the balance sheet calculation. The future fixed assets are calculated as the existing fixed assets from the previous year plus the investments in 4.18. The investments in fixed assets or the capital expenditure

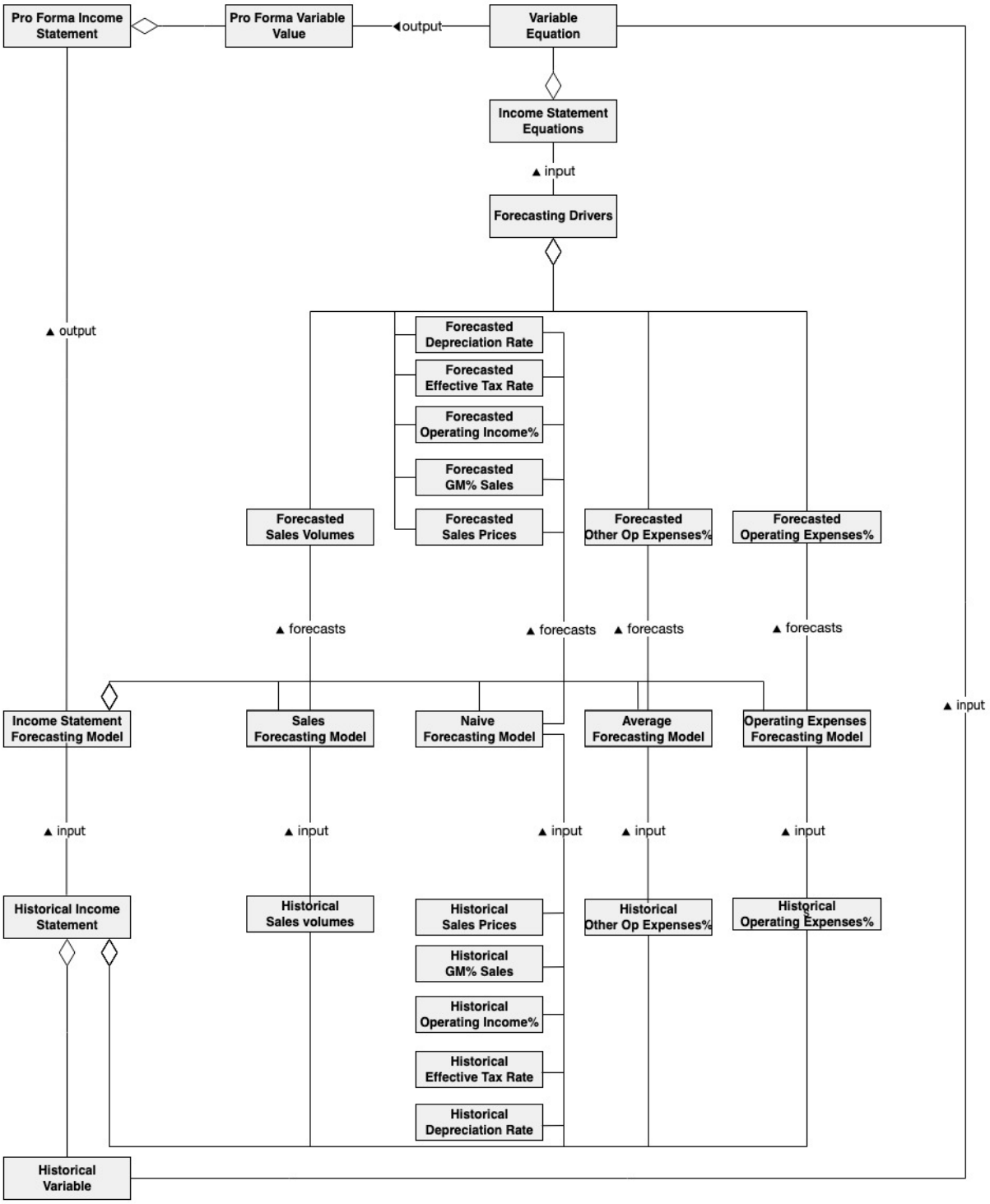


Figure 4.1: Income Statement: Detailed extended view

(CAPEX) is based on the planned investments for the following years. This is decided by the technical director. He analyzes the forecast sales volumes and because they keep declining, he decides on a minimal CAPEX for the forecast years. The gross fixed assets can now be calculated in 4.18.

$$Asset\_gf_t = Assets_{t-1} + CAPEX_{t-1} \tag{4.18}$$

Once the gross fixed assets are calculated for the future years, the depreciation expense can be calculated using formula 4.19:

$$Depr_t = Depr\_rate_t * Asset\_gf_t \tag{4.19}$$

This is the chosen method for calculating depreciation. Other forecasting models like, linear regression could also be used.
The last variable left to be calculated is the net interest expense. This will be calculated after the balance sheet future values are calculated, because it depends on several variables there.

Income Statement	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**
Long Products Volume MT	200.000	170.000	150.000	133.092	120.447	111.137	104.515
Flat Products Volume MT	70.000	60.000	50.000	44.364	40.149	37.046	34.838
Total Volume	270.000	230.000	200.000	177.456	160.596	148.183	139.354
Rate of change	0%	-14,8%	-13,0%	-11,3%	-9,5%	-7,7%	-6,0%
Long Products €/MT	700	670	650	650	650	650	650
Flat Products €/MT	650	680	700	700	700	700	700
Long Products MT/month	16.667	14.167	12.500	11.091	10.037	9.261	8.710
Flat Products MT/month	5.833	5.000	4.167	3.697	3.346	3.087	2.903
Turnover Long Products	140.000	113.884	97.500	86.510	78.291	72.239	67.935
Turnover Flat Products	45.500	40.800	35.000	31.055	28.104	25.932	24.387
Total Turnover (Sales)	185.500	154.684	132.500	117.564	106.395	98.171	92.322
Rate of change		-16,6%	-14,3%	-11,3%	-9,5%	-7,7%	-6,0%
Expenses (before I,T,D,A)	-176.353	-145.687	-126.218	-112.770	-102.702	-95.302	-90.060
COGS: Cost Of Goods Sold w/o D.	170.289	139.679	119.913	106.396	96.287	88.845	83.551
Gross Margin w/o Depreciation	15.211	15.004	12.588	11.169	10.108	9.326	8.771
% of Sales	8,2%	9,7%	9,5%	9,5%	9,5%	9,5%	9,5%
Other Operating Income	3.555	3.493	2.539	2.234	2.022	1.865	1.754
% of Sales	1,9%	2,3%	1,9%	1,9%	1,9%	1,9%	1,9%
Operating Expenses	-8.912	-8.516	-8.182	-8.020	-7.904	-7.831	-7.801
% of Sales	-4,8%	-5,5%	-6,2%				
OPEX Rate of change				-2,0%	-1,5%	-0,9%	-0,4%
Other Operating Expenses	-707	-985	-663	-588	-532	-491	-462
% of Sales	-0,4%	-0,6%	-0,5%	-0,5%	-0,5%	-0,5%	-0,5%
EBITDA: Earnings before I,T,D,A	9.147	8.997	6.282	4.794	3.693	2.869	2.262
Provision for Bad Debts	0	0	0	0	0	0	0
Net Interest Expenses	-4.755	-4.470	-6.357	-4.443	-4.040	-3.778	-3.795
Depreciation	-2.195	-2.067	-2.235	-2.237	-2.248	-2.259	-2.269
EBT: Profit / Loss before tax	2.197	2.459	-2.310	-1.885	-2.595	-3.167	-3.802
Less: Income Tax Expense	-725	-812	0	0	0	0	
EAT: Profit/ Loss after tax	1.472	1.648	-2.310	-1.885	-2.595	-3.167	-3.802

Table 4.7: Income Statement: SteelCo historical and forecast values

4.2 Balance Sheet

To build the forecast of the balance sheet we now move to the right of the of the general diagram and also provide the names of the variables included in the balance sheet in 4.8 and the values in our specific scenario in 4.9. Additionally, table 4.10 includes the historical values for the selected forecasting drivers, which are not present in the balance sheet itself.

4.2.1 Balance Sheet: Forecasting drivers

The process of building the pro forma balance sheet starts similarly to the income statement, with the forecasting of the forecasting drivers. We structure these calculations using diagram 3.8 that we defined in the conceptual model. The three measurements here, the DSO, DIO and DPO, are set through an informed decision by contacting the respective managers, the commercial and purchasing managers, although depending on the historical values other forecasting models like linear regression, or average could also be used. The forecast values for these measurements can be seen in 4.11. These three forecasting drivers will now be used to calculate the rest of the balance sheet.

Balance Sheet Variables	**Variable Names**
Gross Fixed Assets	Asset_nc_gf
Accumulated Depreciation	Depr_acc
Other Non Current Assets	Asset_nc_o
Non Current Assets	Asset_nc
Account Receivables	Acc_rec
Inventory	Inv
Cash	Cash
Other Current Assets	Asset_c_o
Current Assets	Asset_c
Total Assets	Asset
Share Capital	Sharecap
Retained Earnings	RetEarn
Owners Equity	Equity
Debt - Long Term	Debt_lt
Non Current Liabilities	Liability_c_o
Debt - Short Term	Debt_st
Account Payables	AccPay
Other Liabilities	Liability_o
Current Liabilities	Liability_c
Total Liabilities	Liability
Owners Equity & Liabilities	EquAndLiab

Table 4.8: Balance Sheet: SteelCo Variables

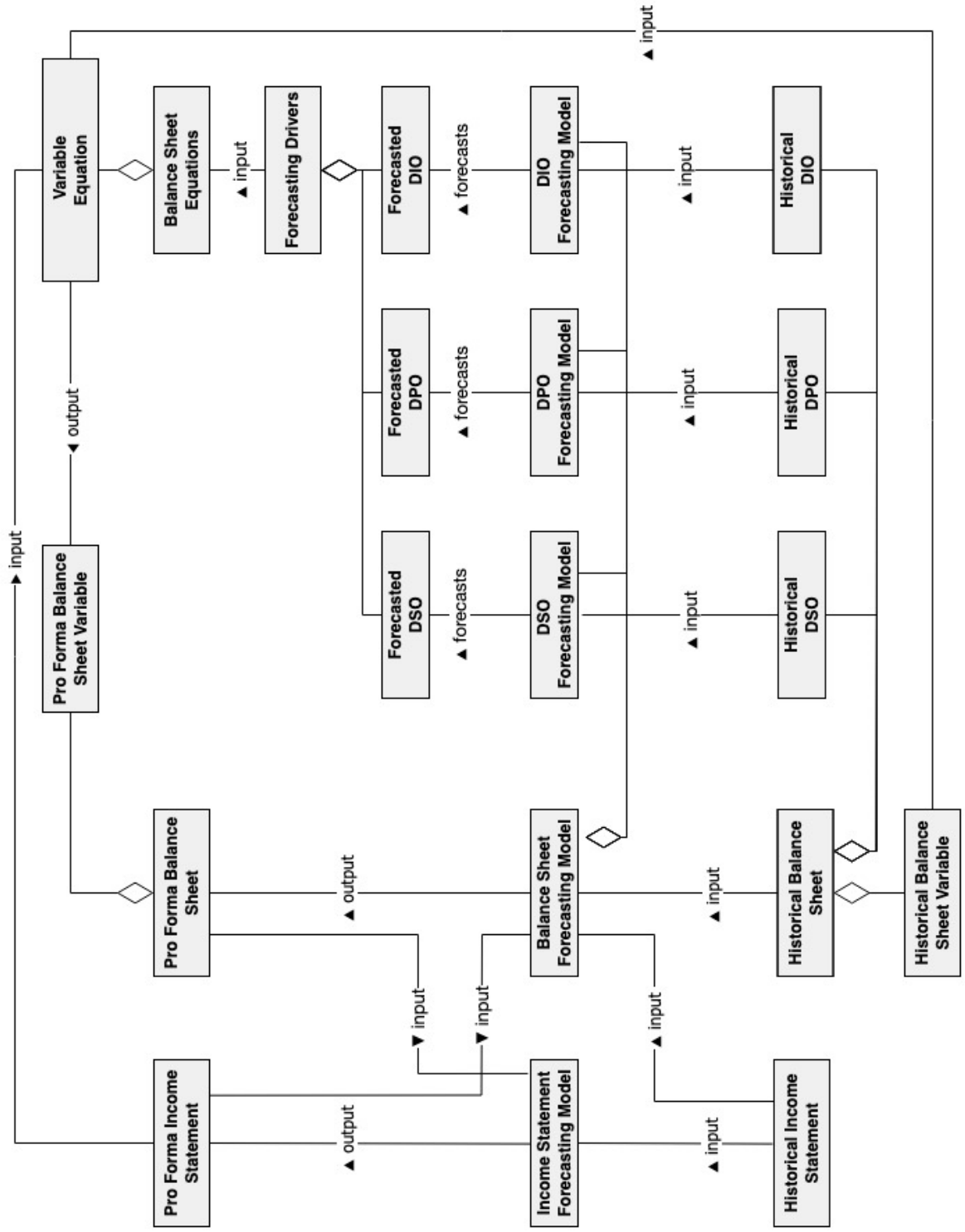


Figure 4.2: Balance Sheet: Detailed extended view

Balance Sheet Variables	2011	2012	2013
Gross Fixed Assets	58.850	61.440	61.840
Accumulated Depreciation	-15.000	-17.067	-19.302
Other Non Current Assets	18.840	18.840	18.840
Non Current Assets	62.690	63.213	61.378
Account Receivables	69.969	57.466	47.918
Inventory	39.984	34.962	31.210
Cash	15.579	12.290	7.000
Other Current Assets	3.736	2.483	1.230
Current Assets	129.267	107.201	87.358
Total Assets	191.958	170.414	148.736
Share Capital	31.260	31.260	31.260
Retained Earnings	1.472	3.120	810
Owners Equity	32.732	34.380	32.070
Debt - Long Term	59.383	56.350	53.317
Non Current Liabilities	59.383	56.350	53.317
Debt - Short Term	76.167	61.795	51.346
Account Payables	19.129	14.542	9.856
Other Liabilities	4.546	3.346	2.147
Current Liabilities	99.842	79.684	63.349
Total Liabilities	159.225	136.034	116.666
Owners Equity & Liabilities	191.958	170.414	148.736

Table 4.9: Balance Sheet: SteelCo historical values

Working Capital Policy	2011	2012	2013
Stock Days - DIO	86	91	95
Customer Days - DSO	115	113	110
Supplier Days - DPO	41	38	30

Table 4.10: Working Capital Policy: SteelCo historical values

Working Capital Policy	2011	2012	2013	2014	2015	2016	2017
Stock Days - DIO	86	91	95	85	80	80	80
Customer Days - DSO	115	113	110	85	80	75	75
Supplier Days - DPO	41	38	30	50	50	55	55

Table 4.11: Working Capital Policy: SteelCo historical and forecast values

4.2.2 Balance Sheet: Structural Equations

Once we calculated the forecasting drivers we can move on with the calculation of the balance sheet, the assets, liabilities and equity. This is also represented with the balance

sheet variable equation entity in the diagram 4.2. Each of these equations represents an instance of the balance sheet equations. This shows the flow from the historical balance sheet variables through the equations to result in the pro forma balance sheet variable that populated the pro forma balance sheet. We only show a generic entity for this process in the diagram to not overpopulate it.
We first start with the calculation of the non-current assets. The first value is the the accumulated depreciation, because we already calculated the gross fixed assets when we needed the values to calculate the depreciation in the income statement.
The accumulated depreciation is now calculated as the sum of the depreciation that occurred in the previous years and the new depreciation that falls in the current year. Since the last historical value for the accumulated depreciation already contains the previous depreciation, we calculate the accumulated depreciation as the sum of the one from the last known year with the current depreciation amount as seen in 4.20.

$$Depr\_acc_t = Depr\_acc_{t-1} + Depr_t \tag{4.20}$$

For the non current assets, it depends which non-current assets the company possesses. In this case we only use the gross fixed assets and the rest is classified as other non current assets. The value of other non-current assets includes the long term investments which was a constant in the three historical years, so it is assumed that it does not change in the future either. The complete non-current assets are then calculated using the equation 4.21.

$$Asset\_nc_t = Asset\_nc\_gf_t - Depr\_Acc_t + Asset\_nc\_o_t \tag{4.21}$$

In the current assets, in this scenario, are included the accounts receivable, the inventory, cash, and the rest is grouped in the other current assets. We first start with the calculation of the accounts receivable. The accounts receivable depend on one of our forecasting drivers, the DSO. Accounts receivable is calculated as shown in 4.22. As seen in the equation, accounts receivable is calculated with the DSO, the total sales and the value added tax. The value added tax can also be checked in the ERP system, in the items or services that are included in the sales value. The value added tax (VAT) is also added to show that the customer owes the company the price of the product and its value added tax.

$$AccRec_t = RevT_t * \frac{DSO_t}{365} * VAT_t \tag{4.22}$$

The next current asset is the inventory, which also depends on one of the forecast forecasting drivers. The inventory is based on the DIO and the cost of goods sold. The cost of goods sold on the other hand is the revenue minus the gross margin in 4.24. These future values were previously calculated in the pro forma income statement calculation. Their relation is shown in 4.23.

$$Inv_t = CoGs_t * \frac{DSO_t}{365} * VAT \tag{4.23}$$

$$CoGs = RevT_t - GM\_wo\_Depr_t \tag{4.24}$$

Then looking at the declining values of cash in the last years, the cash is set as the minimum cash deposit the company needs for a smooth operation. The other current assets are declining yearly in the historical values, but since the value is so small it is almost insignificant, it is kept the same as the last years. These variables are then summed to result in the current assets as seen in 4.25.

$$Asset\_c_t = AccRec_t + Inv_t + Cash_t + Asset\_c\_o_t \tag{4.25}$$

Finally, the total assets can be calculated as the sum of the non current and current assets in the formula 4.26.

$$Asset_t = Asset\_c_t + Asset\_nc_t \tag{4.26}$$

Now we calculate the equity, which in this case is made up of the share capital and retained earnings. Share capital is a constant through the years so it is kept the same value in the future years as well. The retained earnings will also be calculated later as they depend on the income statement variable, net interest expenses. Moving on to the liabilities, they consist of the current and non current liabilities. The non-current liabilities only contain the long term debt in this example. Knowing the debt repayment schedule and the amounts to be paid, the value is set for the future years. So the remaining long term debt is calculated as the original amount minus the repaid debt in 4.27 .

$$Debt\_lt_t = Debt\_lt_{t-1} + Debt\_repayment_{t-1} \tag{4.27}$$

Then in the current liabilities are included the following variables: short-term debt, the accounts payable and other liabilities. The short term debt is used to balance the balance sheet so it will be calculated later. The accounts payable account is calculated through the forecasting driver DPO and the cost of goods sold as seen in 4.28, similarly to the accounts receivable. The other current liabilities are also declining like the other current assets, but they have a relatively small value that would not affect the calculations much, so it is kept as the last known value for the future years.

$$AccPay_t = COGs\_wo\_Depr_t * \frac{DPO_t}{365} \tag{4.28}$$

4.2.3 Circular References: Balancing financial statements

To finish with the forecasting of the balance sheet and income statement we have to calculate the remaining variables that were dependent on each other and could not be forecast previously in the income statement and balance sheet sections. These variables include the short term debt, retained earnings, the net interest expenses and variables that depend on them like the equity, earnings before tax and earnings after tax. There is a circular reference between these variables, which means their equations are related to

Balance Sheet	2011	2012	2013	2014	2015	2016	2017
Gross Fixed Assets	58.850	61.440	61.840	62.140	62.440	62.740	63.040
Accumulated Depreciation	-15.000	-17.067	-19.302	-21.539	-23.787	-26.046	-28.315
Other Non Current Assets	18.840	18.840	18.840	18.840	18.840	18.840	18.840
Non Current Assets	62.690	63.213	61.378	59.441	57.493	55.534	53.565
Account Receivables	69.969	57.466	47.918	32.854	27.983	24.207	22.764
Inventory	39.984	34.962	31.210	24.777	21.104	19.473	18.313
Cash	15.579	12.290	7.000	3.500	3.500	3.500	3.500
Other Current Assets	3.736	2.483	1.230	1.230	1.230	1.230	1.230
Current Assets	129.267	107.201	87.358	62.361	53.817	48.410	45.807
Total Assets	191.958	170.414	148.736	121.801	111.310	103.944	99.372
Share Capital	31.260	31.260	31.260	31.260	31.260	31.260	31.260
Retained Earnings	1.472	3.120	810	-1.076	-3.670	-6.837	-10.640
Owners Equity	32.732	34.380	32.070	30.184	27.590	24.423	20.620
Debt - Long Term	59.383	56.350	53.317	50.283	47.250	44.217	42.700
Non Current Liabilities	59.383	56.350	53.317	50.283	47.250	44.217	42.700
Debt - Short Term	76.167	61.795	51.346	24.612	21.133	19.770	21.314
Account Payables	19.129	14.542	9.856	14.575	13.190	13.388	12.590
Other Liabilities	4.546	3.346	2.147	2.147	2.147	2.147	2.147
Current Liabilities	99.842	79.684	63.349	41.334	36.471	35.304	36.051
Total Liabilities	159.225	136.034	116.666	91.617	83.721	79.521	78.751
Owners Equity & Liabilities	191.958	170.414	148.736	121.801	111.310	103.944	99.372

Table 4.12: Balance Sheet: SteelCo historical and forecast values

each other in a closed circular dependency. To calculate these variables we will allow the calculations to iterate a few times so that the final value is reached and the difference of the results from one iteration to the other is minimal. In this case minimal is defined as a maximum change of 0,01 of the results between iterations. The first variable that we will look into is the retained earnings. The retained earnings depend on the earnings after tax as seen in 4.29.

$$RetEarn_t = RetEarn_{t-1} - EAT_t \tag{4.29}$$

The earnings after tax (EAT) on the other hand, are calculated using the earnings before tax and the income tax expenses in the equation 4.30.

$$EAT_t = EBT_t + IncTaxEx_t \tag{4.30}$$

The earnings before tax (EBT) depends on other variables that get calculated here, which are the net interest expenses, and the earnings before interest and tax. The formula for EBT is shown in 4.31.

$$EBT_t = EBIT_t + IntEx\_net_t \tag{4.31}$$

The earnings before tax are based on the earnings before interest, tax and depreciation (EBITDA) and the depreciation as shown in 4.32.

$$EBIT_t = EBITDA_t - Depr_t \tag{4.32}$$

Finally the earnings before interest, tax, depreciation and amortization stay the same in each iteration, because they do not depend on any of the variables that are calculated within the circular function. They are needed for the calculation of the earnings before interest and tax (EBIT), which in turn is used to calculate the earnings before tax (EBT), which depends on the net interest expenses that are calculated here.

$$EBITDA_t = GM\_wo\_Depr_t + OpInc\_o_t + OpEx\_wo\_Depr_t + OpEx\_o\_wo\_Depr_t \quad (4.33)$$

Now that the EAT can be calculated and therefore the retained earnings can be calculated in 4.29, the equity can be calculated as seen in 4.34.

$$Equity_t = Share\_Cap_t + Ret\_Earn_t \quad (4.34)$$

The short term debt only depends on the equity within this circular function, while the rest of the variables are already forecast, so it can be calculated using 4.35.

$$Debt\_st_t = Asset_t - Equity_t - Debt\_lt_t - AccPay_t - Liability\_o_t \quad (4.35)$$

Lastly, the net interest expenses, which depends on the short term debt, can be calculated using the formula 4.36.

$$IntEx\_net_t = Cash\_t * P\_Deposit\_rate_t + Debt\_lt_t * P\_fin\_cost\_lt_t + Debt\_st * P\_fin\_cost\_st_t \quad (4.36)$$

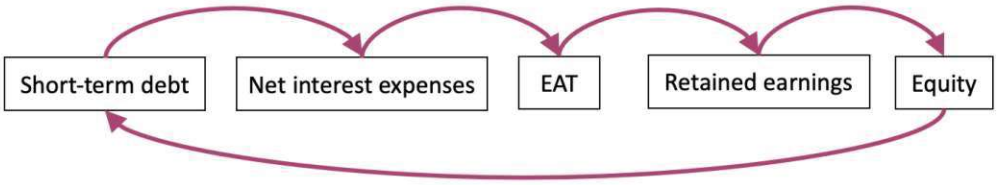


Figure 4.3: Circular dependency

Figure 4.3 visualises this circular dependency in a simple diagram. The arrows show which variable that needs to be calculated is used in the calculation of which. This process has to be repeated a few times until all the variables are calculated and the results

converge to a value. This means that the calculation stops when the last calculated value does not differ much from the next to last result. Through this calculation, the pro forma income statement and the pro forma balance sheet are completed, concluding the forecasting process. All these calculations were performed in a spreadsheet to provide the results and also to be used later for the evaluation of our implementation. The results are summed up in 4.7 and 4.12 .

4.3 Cash Flow Statement

The cash flow statement is the last financial statement we need to build, and in this case we need to construct both the historical and the pro forma cash flow statement. The cash flow statement shows the changes in the balance sheet and income statement. It derives the company's cash inflows and outflows in operations, investing and financing activities. The cash flow statement includes the variables as seen in 4.13. The "D" in the variable names represents a delta symbol or change.

Cash Flow Statement Variables	**Variable Names**
Operating Activities	
Profit / Loss Before Tax (EBT)	EBT
Depreciation & Amortization	DeprAmor
Decrease/(Increase) in Receivables	DRecv
Decrease/(Increase) in Inventories	DInv
(Decrease)/Increase in Payables	DPay
Decrease/(Increase) in Other Receivables & Payables	DRecPay
Total Inflows / (Outflows) from Operating Activities (a)	InOutOp
Operating Activities	
Purchase or Disposals of Assets	DAssets
Total Inflows / (Outflows) from Investing Activities (b)	InOutInv
Financing Activities	
Share Capital Increase	DShareCap
New Bank Loans Raised & Repayments of Loans	DLoans
Total Inflows / (Outflows) from Financing Activities (c)	InOutFin
Increase/(Decrease) in Cash (equivalents) (a)+(b)+(c) minus Tax	DCash
Cash and Cash Equivalents at the beginning of the period	DCashBeg
Cash and Cash Equivalents at the end of the period	DCashEnd

Table 4.13: Cash Flow Statement: SteelCo Variables

The variables are split into three sections: operating activities, investing activities and financing activities. For each of these sections the respective changes in the balance sheet and income statements and the total changes for the category are calculated. In the fourth and last section are included the total of all three types of activities together and the cash balance in the start and end of each period. In this case a period represents a year.
Similarly to the other two financial statements we will use the cash flow UML diagram 3.9 we built in the conceptual model as a guide and now go into the cash flow statement

construction. Each of the cash flow equations can be presented as an instance of the cash flow construction model. Starting with the operating activities, the first variable to derive is the Profit or loss before tax, or the earnings before tax. This variable was already calculated in the income statement therefore it is simply copied from the income statement table. The adjustments for depreciation and amortization are represented by the depreciation and amortization in the income statement. The changes in working capital are made up of the changes in receivables, inventories, payables, and in other receivables and payables.

The decrease or increase in receivables is calculated as the difference of the accounts receivable balance from a year to the preceding year. The accounts receivable balance is recorded in the income statement. This is also shown in 4.37

$$\Delta AccRec_t = AccRec_t - AccRec_{t-1} \tag{4.37}$$

The same logic is used for the rest of the changes in working capital. The decrease or increase in inventories calculates the changes in the inventory account in 4.38.

$$\Delta Inv_t = Inv_t - Inv_{t-1} \tag{4.38}$$

The decrease or increase in payables calculates the changes in the accounts payables in 4.39.

$$\Delta AccPay_t = AccPay_t - AccPay_{t-1} \tag{4.39}$$

The decrease or increase in other receivables and payables in 4.40 is the difference in other current assets minus the difference of the other liabilities between two years.

$$\Delta RecPay\_o_t = (Liability\_o_t - Liability\_o_{t-1}) - (Asset\_c\_o_t - Asset\_c\_o_{t-1}) \tag{4.40}$$

Then the total for the operating activities is calculated as the sum of all its components as seen in 4.41 and labeled in table 4.13 with an (a).

$$InOutOp = EBT_t + Depr_t + \Delta AccRec_t + \Delta Inv_t + \Delta AccPay_t + \Delta RecPay_t \tag{4.41}$$

Next there are the investing activities. In this case we only have the purchase or disposal of assets, which also represents the total inflows or outflows from investing activities. The investment or capital expenditure is calculated as the change in fixed assets from one year to the other. Therefore the total investing activities and the purchase or disposal of assets is calculated as seen in 4.42.

$$\Delta Inv = \Delta Assets = Asset\_gf_t - Asset\_gf_{t-1} \tag{4.42}$$

The third activity group, the financing activities, include the share capital increase and the new bank loans and the repayment of loans. The share capital increase here is not

Cash Flow Statement	2011	2012	2013	2014	2015	2016	2017
Operating Activities							
EBT : Profit/Loss Before Tax	2.197	2.459	-2.310	-1.885	-2.595	-3.167	-3.802
Adjustments for:							
Depreciation & Amortization	2.195	2.067	2.235	2.237	2.248	2.259	2.269
Changes in Working Capital							
Decrease/(Increase) in Receivables	13.001	12.503	9.548	15.064	4.870	3.777	1.442
Decrease/(Increase) in Inventories	3.842	5.022	3.752	6.433	3.673	1.631	1.160
(Decrease)/Increase in Payables	-5.982	-4.587	-4.686	4.719	-1.385	198	-798
Decrease/(Increase) in Other Receivables & Payables	173	54	54	-	-	-	-
Total Inflows / (Outflows) from Operating Activities (a)	15.427	17.517	8.592	26.568	6.812	4.697	272
Investing Activities							
Purchase or Disposals of Assets	-1.427	-2.590	-400	-300	-300	-300	-300
Total Inflows / (Outflows) from Investing Activities (b)	-1.427	-2.590	-400	-300	-300	-300	-300
Financing Activities							
Share Capital Increase							
New Bank Loans Raised & Repayments of Loans	-16.274	-17.405	-13.482	-29.768	-6.512	-4.397	28
Total Inflows / (Outflows) from Financing Activities (c)	-16.274	-17.405	-13.482	-29.768	-6.512	-4.397	28
Net Increase/Decrease in Cash and Cash Equivalents a+b+c	-2.274	-3.289	-5.290	-3.500	0	0	0
Cash and Cash Equivalents at the beginning of the period	17.853	15.579	12.290	7.000	3.500	3.500	3.500
Cash and Cash Equivalents at the end of the period	15.579	12.290	7.000	3.500	3.500	3.500	3.500

Table 4.14: Cash Flow Statement: SteelCo historical and forecast values

defined as there are no share capitals applied. The loans and repayment are defined as the sum of the differences in long term debt and short term debt. Therefore the total inflows and outflow of the financing activity are calculated in 4.43

$$InOutFin\_total_t = \Delta Loans_t = Debt\_lt_t - Debt\_lt_{t-1} + Debt\_st_t - Debt\_st_{t-1} \tag{4.43}$$

Finally the last section includes a summary of the activities. The net increase or decrease in cash is calculated in 4.44 as the sum of the total changes in all three activities, operating, investing and financing.

$$\Delta Cash_t = InOutOp_t + InoutInv_t + InOutFin_t \tag{4.44}$$

The cash and cash equivalents in the beginning of the period is the cash that was recorded in the end of the previous period. This information is recorded in the income statement. The equation that represents this relation is 4.45.

$$CashBeg_t = Cash_{t-1} \tag{4.45}$$

The cash equivalents in the end of the period in 4.46 is the sum of the cash in the beginning of the period with the net change in cash in each period.

$$CashEnd_t = CashBeg_t + DCash_t \tag{4.46}$$

The results of the calculations of the cash flow statements for the 7 years, using the financial information from SteelCo and the previously forecast income statement in 4.1

and balance sheet in 4.2, can be seen in table 4.14. In the functions that calculate change the first year can not be calculated, because the previous year is not known. Nevertheless, these values can be retrieved from the ERP system and set. In the table these values are filled based on the information provided in [Sam15] to provide a complete financial statement.

CHAPTER 5

Activity based budgeting: Implementation in R tidyverse

In order to provide a demonstration of a prototype of the activity based budgeting system based on the specified conceptual and operational model we will now go through the last step, the implementation. The implementation serves as an example of a concrete instance of the framework in the IT domain and also allows us to further evaluate the applicability of the conceptual model and the operational model that we previously introduced in chapters 3 and 4. From the beginning, we used the same use-case that Samonas used in [Sam15], because in this paper we are not presenting the decision-making and assumption selection of the framework, but rather defining a ubiquitous language that supports the understanding of the complex AB-budgeting domain. That is why using the same use-case ensures a correct and realistic choice of models and use of the framework, allowing us to focus on the transition. The results of the spreadsheet implementation of the use-case will serve as a ground truth, allowing us to come to a conclusion about the validity of our programmed artifact. The framework was built in a spreadsheet during the operational model, where the results Samonas' achieved in his book [Sam15] were replicated. The results are shown in the result tables of the operational model 4.7 and 4.12.

On the other hand, the implementation process uses the previous two models that we introduced as blueprints. This allows us to validate the models and evaluate their usage in the transition process from the accounting and financial domains to the information systems domain.

The framework is implemented in R tidyverse. We chose R, because it is a simple tool to visualize the implementation and its results. Although it is a functional programming language, it allows us to use an object oriented structure that can be adapted universally to other popular programming languages. The used language depends on the ERP system where the framework should be integrated, therefore there is no specific language that

would benefit all systems universally.
In this section we will start by defining the data sources, the input variables and the chosen data structure. Then we will describe the process chronologically based on the previously introduced models. The first step is to build the historical financial statements data structures. Then we will describe the forecasting models in this environment and produce the pro forma financial statements. In the end we will build the cash flow statement through calculations using the forecast statements.

5.1 Programming Environment

We chose to implement this framework in R [RMa] using the tidyverse package [tid]. R is an open source language and environment for statistical computing and graphics. It includes effective data handling and storage facility, operators for calculations on arrays and matrices, a large collection of tools for data analysis and their visualization and a well-developed, simple and effective programming language, which includes conditionals, loops, user-defined functions and input and output facilities [R]. Tidyverse [tid] is an R package that we will be using along the base R. Tidyverse is a system of packages for data manipulation, exploration and visualization that share a design philosophy and serve as tools to increase productivity in data science tasks. From the tidyverse package we will be using the packages dplyr, lubridate, tidyr, ggplot2. The 'dplyr' package is used for data manipulation, such as filtering, joining and mutating data. 'lubirnate' simplifies working with dates by providing functions for parsing and manipulating date data. The 'tidyr' package is used for reshaping data, building data structures and extracting data. The main data structure we will be using is a tsibble, which is a separate package that extends the tidyverse tibble data structure to temporal data. In the context of financial statements over the years, the key attribute is the year in which the financial statement was recorded. That makes a tsibble an appropriate data structure, because it uses years as keys. Furthermore the package 'ggplot2' is used for data visualization.

5.2 Input Data

The first step to starting the implementation is the import of the historical data and the forecasting assumptions. In an integrated system the historical data of the income statement and balance sheet could be input directly through a query that filters through and groups the transactional data from our statements in the ERP system or its database. In our simple demonstration of the implementation we built the input data as simple data frames, input the historical financial statements data of the last three years of SteelCo as provided in [Sam15] and saved the workspace in an .RData file to save the initial data in a file that we can then reuse when necessary. The assumptions table which contains all the forecasting parameters and decisions is also structured in a data frame and saved in the workspace. The parameters of the forecasting models are defined by the forecasting planner and analyst by consulting experts from different departments in the company, such as finance, accounting or management. These parameters are defined

based on expected internal and external influences, historical data, business vision and goals and other factors. In an integrated information system this data could be input in a forecasting model calibration page in the user interface before the forecasting model runs.

5.3 Historical Financial Statements

The financial statements are modeled through the tsibble data structure. A tsibble is part of the package tsibble, which extends the tibble, a tidy data structure contained in the tidyverse package. A tsibble is a package that provides a data class to represent tidy temporal data. It consists of a time index, keys and other measured variables [tsi]. The index represents the financial statement variables and values within a year. We can then create a historical and forecast tsibble for each financial statement, which respectively contain the historical years and the future years to be forecast. In our approach we group the historical financial statements and future financial statements into one to perform the calculations, because most calculations are based on both historical and forecast years. Keeping all the data in one tsibble benefits the code and allows for more streamlined calculations. The tsibbles are then split again to reflect the structure that was defined in the conceptual model and the operational model. The years are defined as global variables from the beginning and used throughout the implementation. These variables include *numHistYears*, *numFutureYears* and *numYears*, which define the number of historical, forecast and total number of years we are considering. Out of these variables the years vectors are also built globally because of the frequent use and to ensure uniformity. The year vectors include *HistYears*, *FutureYears* and *AllYears*.

5.3.1 Income Statement

After building the conceptual and operational model, we now implement the framework in R tidyverse based on these two models. As seen in diagram 3.4 the first step to build the financial forecasting framework is to initialize the historical financial statements, starting with the income statement. The income statement is imported as a data frame from our storage. The income statement data frame is saved as *IncomeStatInput*. Then we used a help function to convert the year column into a date data type so that we can then use it as an index in the tsibble data structure *IncomeStatHist*. The year is set to the first of January of the year to complete the requirements of a date data type.

```
IncomeStatInput <- IncomeStat

IncomeStatInput$Year <- convert_custom_dates_to_year(IncomeStatInput$Year)

#Convert the dataframe to a tsibble
IncomeStatHist <- as_tsibble(IncomeStatInput, index = Year)
```

As discussed in the conceptual and operational model, we employ different methods of forecast to generate the pro forma financial statements. To calibrate the models and to calculate future values we need certain parameters and assumptions that are not present in the income statement. These parameters will be saved for each of our financial statements in separate tsibbles called assumptions. For the income statement assumption tsibble we start off with a few values that should be provided by either different experts or decision makers, or have to be sourced from the ERP system, because they are needed for the forecasts, but are not present in the income statement. The assumptions list contains the quantified measurements of the decisions made by the financial planner for the forecasting models and other supporting variables that are necessary for the calculations. The historical assumptions tsibble for the income statement, *AssumptionsHistIS*, contains the variables displayed in table 5.1.

Assumptions Income Statement	**AssumptionsIS Names**
Deposit Rate	P_Deposit_rate
Financial cost of long term debt	P_fin_cost_lt
Financial cost of short term debt	P_fin_cost_st
Effective tax rate	EffTaxRate
Cost of sales	CostSales
Distribution	Distribution
Consumer price index	CPI
Operating expenses percentage of volume	P_Vol_OPEX
Provision bad debt	provision

Table 5.1: Income Statement assumptions list

Then the historical assumptions list is constructed using these variables and their values of the last three years. We first build the tsibble for the three historical years and then include each of the variables and populate them with the values of the three historical years using the mutate function.

```
AssumptionsHistIS <- tibble(Year = HistYears) %>%
        as_tsibble(index = Year)
AssumptionsHistIS <- AssumptionsHistIS %>%
     mutate(
        P_Deposit_rate = c(0.05, 0.03, 0.01),
        P_fin_cost_lt = c(-0.025, -0.025, -0.0565),
        P_fin_cost_st = c(-0.0531623, -0.055509, -0.0665),
        EffTaxRate = 0.33,
        CostSales =c(995000, 945000, 1028000),
        Distribution=        c(1200000,1122000,1207000),
        CPI = c(0.02 , 0.018 , 0.017),
        P_Vol_OPEX = 0.3,
        provision=0
        )
```

There are further parameters necessary for the forecasting of the income statement. These

parameters can be derived from the historical financial data. In this case, the parameters that need to be calculated support the percentage of sales forecasting method. These variables include the percentage of sales of the gross margin, operating income, operating expenses and other operating expenses, and the depreciation rate. The variables for these parameters will be named as seen in 5.2.

Assumptions Income Statement	**AssumptionsIS Names**
Depreciation Rate	DepreciationRate
Percentage of sales of gross margin without depreciation	P_GM_wo_Depr
Percentage of sales of operating income	P_OpInc_o
Percentage of sales of operating expenses without depreciation	P_OpEx_wo_Depr
Percentage of sales of other operating expenses without depreciation	P_OpEx_o_wo_Depr

Table 5.2: Income Statement calculated parameter list.

All the variables are calculated through the same function, which finds the ratio of one variable in another. We defined the function as follows.

```
calcVarPInVar <- function(variable, inVariable){
  return(variable/inVariable)
}
```

Then we used this function to calculate each of the ratios, depreciation rate as a percentage gross fixed assets and the rest of the variables as the percentage of sales. We added these variables and their values to the historical income statement assumptions list tsibble, *AssumptionsHistIS* that we previously built.

```
AssumptionsHistIS <- AssumptionsHistIS %>% mutate(
 DepreciationRate= calcVarPInVar(IncomeStatHist$Depr, BalanceSheetHist$Asset_nc_gf),
 P_GM_wo_Depr= calcVarPInVar(IncomeStatHist$GM_wo_Depr, IncomeStatHist$RevT),
 P_OpInc_o= calcVarPInVar(IncomeStatHist$OpInc_o, IncomeStatHist$RevT),
 P_OpEx_wo_Depr= calcVarPInVar(IncomeStatHist$OpEx_wo_Depr, IncomeStatHist$RevT),
 P_OpEx_o_wo_Depr= calcVarPInVar(IncomeStatHist$OpEx_o_wo_Depr, IncomeStatHist$RevT)
 )
```

The historical assumption and parameter table is complete with all the necessary variables. Its current state in the implementation can be seen in 5.3.

Another tsibble we will need for the forecasting of the income statement, which we should already initialise is the sales tsibble. In table 4.1 we already defined the names of the variables we will use in the sales tsibble and their definition. The sales tsibble is then built similarly to the income statement. The years are transformed in the date format, then the tsibble is built using the imported historical sales data frame and the year column as the index.

Variable	2011-01-01	2012-01-01	2013-01-01
P_Deposit_rate	0,05	0,03	0,01
P_fin_cost_lt	-0,0250	-0,0250	-0,0565
P_fin_cost_st	-0,0532	-0,0555	-0,0665
EffTaxRate	0,33	0,33	0,33
CostSales	995000	945000	1028000
Distribution	1200000	1122000	1207000
CPI	0,020	0,018	0,017
P_Vol_OPEX	0,3	0,3	0,3
provision	0	0	0
DepreciationRate	0,0373	0,0336	0,0361
P_GM_wo_Depr	0,082	0,097	0,095
P_OpInc_o	0,0192	0,0226	0,0192
P_OpEx_wo_Depr	0,0480	0,0551	0,0618
P_OpEx_o_wo_Depr	0,0038	0,0064	0,0050

Table 5.3: Transposed AssumptionsHistIS Table

```
SalesInput <- Sales
SalesInput$Year <- convert_custom_dates_to_year(SalesInput$Year)

# Convert the dataframe to a tsibble
SalesHist <- as_tsibble(SalesInput, index = Year)
```

The historical sales table is now built and contains the structure and data as seen in 5.4 table.

SalesHist	2011-01-01	2012-01-01	2013-01-01
xSA	200000	170000	150000
xSB	70000	60000	50000
xST	270000	230000	200000
xST_ROC	0	-0,1481	-0,1304
pA	700	669,9042	650
pB	650	680	700

Table 5.4: Transposed Sales Data Table

5.3.2 Balance Sheet

The historical balance sheet is built using the tsibble data structure like all the financial statements. We will be using the same names for the variables as defined in the operational model in 4.8.

```
BalanceSheetInput<- BalanceSheet

BalanceSheetInput$Year <- convert_custom_dates_to_year(BalanceSheetInput$Year)

# Convert the dataframe to a tsibble
BalanceSheetTsbl <- as_tsibble(BalanceSheetInput, index = Year)
```

In order to forecast the balance sheet later we will also need a few parameters that are not included in the balance sheet itself. The historical values of these parameters are included in a separate tsibble called *AssumptionsHistBS*. This tsibble contains the variables as listed in the table 5.5.

Assumptions Balance Sheet	**AssumptionsBS**
Days inventory outstanding	DIO
Days sales outstanding	DSO
Days payables outstanding	DPO
redemption	redemption
Capital expenditure	CAPEX
Cost of sales	CostSales
Distribution	Distribution
Value after tax	VAT
Minimum cash amount for regular operations	MinCash

Table 5.5: Balance sheet parameter list.

The historical assumptions for the balance sheet are then initialized as a tsibble with the length of the historical years and the values of each parameter for each year are filled in.

```
AssumptionsHistBS <- tibble(Year = HistYears) %>%
        as_tsibble(index = Year)
AssumptionsHistBS <- AssumptionsHistBS %>%
        mutate(
                DIO = c(86, 91, 95),
                DSO = c(115, 113, 110) ,
                DPO = c(41, 38, 30) ,
                redemption = c(0,3033000, 3033000),
                CAPEX = c(1425000,2590000,400000),
                CostSales =c(995000, 945000, 1028000),
                Distribution=        c(1200000,1122000,1207000),
                VAT = 1.2,
                MinCash = 350000
        )
AssumptionsHistBS <- as_tsibble(AssumptionsHistBS, index = Year)
```

The historical values of the balance sheet assumptions are also displayed in table 5.6.

AssumptionsHistBS	2011-01-01	2012-01-01	2013-01-01
DIO	86	91	95
DSO	115	113	110
DPO	41	38	30
CAPEX	1.425.000	2.590.000	400.000
Redemption	0	3.033.000	3.033.000
CostSales	995.000	945.000	1.028.000
Distribution	1.200.000	1.122.000	1.207.000
VAT	1,2	1,2	1,2
MinCash	350.000	350.000	350.000

Table 5.6: Transposed AssumptionsHistBS Table

5.4 Pro Forma Financial Statements

After building the historical data structures needed for forecasting the financial statements we can now proceed with the forecasting process. Following the description in the conceptual and operational model, to forecast the income statement and balance sheet we need to first forecast the forecasting drivers and then apply the respective structural equations to derive the rest of the variables. Once both the income statement and balance sheet are forecast we can then calculate the cash flow statement.

5.4.1 Pro Forma Income Statement

The forecast of the pro forma income statement uses information from both the future forecast sales and assumptions table. Therefore we start the process with forecasting the sales variables. We first initialize the sales forecast tsibble for the future years, *SalesForecast*, and then, to make the calculations easier, we bind the historical sales tsibble to the forecast one to create a tsibble for all years, called *SalesAll*, where the forecasts are empty. We will be using the same approach for all the statements.

```
#Build empty tsibble for forcasted values of Sales
SalesForecast <- tibble(Year = FutureYears) %>%
        as_tsibble(index = Year)

#Merge historical and future sales tsibble, to make calculation easier.
SalesAll <- bind_rows(SalesHist, SalesForecast)
```

Then we build forecasting functions for each sales variable based on the functions we decided on in the operational model and apply them. To calculate the prices of each product group we use the last known value as the forecast value, because the historical values did not change significantly. The function in R looks as follows. It takes as input the measure and the number of historical years to consider and applies the last value to all years that come after our historical years.

```
# Use the last historical year value for future years
forecast_w_last_value <- function(measure, num_hist_years) {
  measure = if_else(row_number() > num_hist_years,
                    nth(measure, num_hist_years),
                    measure)
  return(measure)
}
```

Then we calculate the rate of change of the total revenue, *RevT*, for the historical years and then forecast the future values. In R this is implemented as seen in the code below. We first calculate the difference of the total revenue from one year to another in percentage, then use the rates of change of the previous two years for each year in the future to forecast them. This forecast function calculates the difference in rates of changes in the past two years and then adds this change to the last rate of change. These two calculations for the historical and future rate of change reflect functions 4.1 and 4.2 in the operational model.

```
calculate_RevT_ROC <- function(xST) {
  roc <- numeric(numYears)
  #Historical rate of change of the total revenue
  for (i in 2:numHistYears){
     roc[i] <- xST[1] / xST[i-1]-1
  }
 #Future rate of change
  for (i in (numHistYears+1):numYears) {
    roc[i] <- (roc[i-1] - roc[i-2]) + roc[i-1]
  }
  return(roc)
}
```

Using the forecast rate of change of the total revenue we can now calculate the future total revenue by multiplying the year's rate of change with the total revenue of the previous year. In R it is implemented in the function below.

```
calculate_xST <- function(xST, RevT_ROC) {
  # Calculate future total sales based on Rate of Change
  for (i in (numHistYears+1):length(xST)) {
    xST[i] <- xST[i-1] * (1 + RevT_ROC[i])
  }
  xST
}
```

The last step is to calculate the ratio of each product group in the last known total revenue as seen in 4.4 and 4.5 and apply this ratio to the future to derive the respective future volumes as seen in 4.6 and 4.7.

```
calculate_xS <- function(xST, xSA, xSB) {
  # Calculate future volumes of products A and B based on total sales
  # and historical proportions
```

```
  xAinT <- calculate_AinB (xSA[numHistYears],xST[numHistYears])
  xBinT <- calculate_AinB (xSB[numHistYears],xST[numHistYears])
  list(
    xSA = sapply((numHistYears+1):length(xST), function(i) xST[i] * xAinT),
    xSB = sapply((numHistYears+1):length(xST), function(i) xST[i] * xBinT)
  )
}
```

The ratio is calculated through a simple function, which will be also reused in the future.

```
calculate_AinB <- function(A,B){
  return(A /B)
}
```

All these functions are applied with the mutate function to the sales tsibble to store the results.

```
SalesAll <- SalesAll %>%
        mutate(
                pA = forecast_w_last_value(pA, nrow(SalesHist)),
                pB = forecast_w_last_value(pB, nrow(SalesHist)),
                xST_ROC = calculate_RevT_ROC(xST),
                xST = calculate_xST(xST, xST_ROC)
        )
# Apply calculate_xS function and update xSA and xSB future values
results <- calculate_xS(SalesAll$xST, SalesAll$xSA, SalesAll$xSB)
SalesAll$xSA <- c(SalesAll$xSA[1:numHistYears], results$xSA)
SalesAll$xSB <- c(SalesAll$xSB[1:numHistYears], results$xSB)
```

The sales variables are now calculated and contain the values as seen in 5.7.

Year	xSA	xSB	xST	xST_ROC	pA	pB
2011-01-01	200000	70000	270000	0,0000	700	650
2012-01-01	170000	60000	230000	-0,1481	669,9042	680
2013-01-01	150000	50000	200000	-0,1304	650	700
2014-01-01	133091,8	44363,93	177455,7	-0,1127	650	700
2015-01-01	120447	40149	160596	-0,0950	650	700
2016-01-01	111137,1	37045,69	148182,8	-0,0773	650	700
2017-01-01	104515,4	34838,46	139353,9	-0,0596	650	700

Table 5.7: Forecast Sales Values

Now that the sales are fully forecast we can move on to the parameters and the other forecasting drivers. The historical variables are stored in the *AssumptionsHistIS* in the implementation. Now we build the *AssumptionsIS* tsibble that contains all the historical values and placeholders for the future values. In order to calculate the future depreciation for the income statement we need the future gross fixed assets. Knowing that the future

capital expenditure will be a constant 300.000 we can already calculate the forecast gross fixed assets. Therefore we calculate this balance sheet variable here using function 4.26.

```
calculate_gross_fixed_assets <- function(asset_nc_gf, CAPEX) {
  for (i in 2:length(asset_nc_gf)) {
    asset_nc_gf[i] <- asset_nc_gf[i-1] + CAPEX[i]
  }
  return(asset_nc_gf)
}
```

Based on the forecasting methods that were decided on in the operational model, we now include the forecast values in the income statement assumptions table *AssumptionsIS*. As we can see in the lines below, most of the variables are forecast using the last historical value. Therefore we reuse function *forecast_w_last_value*.

```
AssumptionsIS <- AssumptionsIS %>%
mutate(
  P_Deposit_rate = forecast_w_last_value(P_Deposit_rate, numHistYears),
  P_fin_cost_lt = forecast_w_last_value(P_fin_cost_lt, numHistYears),
  P_fin_cost_st = forecast_w_last_value(P_fin_cost_st, numHistYears),
  EffTaxRate = forecast_w_last_value(EffTaxRate, numHistYears),
  DepreciationRate = forecast_w_last_value(DepreciationRate, numHistYears),
  CoSinDepr= forecast_w_last_value(calculate_AinB(CostSales,CostSales+Distribution),
               numHistYears),
  CostSales = forecast_costSales(DepreciationRate, BalanceSheetAll$Asset_nc_gf,
               CoSinDepr),
  Distribution = forecast_distribution(DepreciationRate, BalanceSheetAll$Asset_nc_gf,
               CostSales),
  P_GM_wo_Depr = forecast_w_last_value(P_GM_wo_Depr,numHistYears),
  P_OpInc_o= forecast_w_last_value(P_OpInc_o,numHistYears),
  P_OpEx_o_wo_Depr= forecast_w_last_value(P_OpEx_o_wo_Depr,numHistYears),
  CPI= set_future_value(CPI, 0.02, numHistYears),
  P_Vol_OPEX = forecast_w_last_value(P_Vol_OPEX, numHistYears),
  P_OpEx_wo_Depr = forecast_OPEX(P_OpEx_wo_Depr, CPI, P_Vol_OPEX, SalesAll$xST),
  provision = forecast_w_last_value(provision, numHistYears)
    )
```

For calculating the cost of sales and distribution we first use the ratio function to calculate the percentage of each variable in the depreciation. Then each of these variables is multiplied with the already forecast depreciation rate and gross fixes assets.

```
forecast_costSales <- function(DepreciationRate, Assets_gf , CoSinDepr){
    Depreciation = DepreciationRate * Assets_gf
    return(Depreciation * CoSinDepr )
}
```

At last the operating expenses forecasting formula 4.10 is built in R as follows.

```
forecast_OPEX <- function(P_OpEx_wo_Depr, CPI, P_Vol_OPEX, xST) {
  t <- numHistYears+1
  while (t <= numYears) {
```

```
    P_OpEx_wo_Depr[t] = P_Vol_OPEX[t]*xST[t]/xST[t-1]+
                (1 - P_Vol_OPEX[t])*(1 + CPI[t]) -1
    t <- t + 1
  }
  return(P_OpEx_wo_Depr)
}
```

The forecast assumptions for the income statement are displayed in 5.8.

Variables	2011	2012	2013	2014	2015	2016	2017
P_Deposit_rate	0,05	0,03	0,01	0,01	0,01	0,01	0,01
P_fin_cost_lt	-0,025	-0,025	-0,0565	-0,0565	-0,0565	-0,0565	-0,0565
P_fin_cost_st	-0,0532	-0,0555	-0,0665	-0,0665	-0,0665	-0,0665	-0,0665
EffTaxRate	0,3300	0,33	0,33	0,33	0,33	0,33	0,33
DepreciationRate	0,0373	0,0336	0,0361	0,0360	0,0360	0,0360	0,0360
CostSales	995.000	945.000	1.028.000	1.029.000	1.034.000	1.039.000	1.044.000
Distribution	1.200.000	1.122.000	1.207.000	1.208.000	1.214.000	1.220.000	1.225.000
P_GM_wo_Depr	0,082	0,097	0,095	0,095	0,095	0,095	0,095
P_OpInc_o	0,0192	0,0226	0,0192	0,019	0,019	0,019	0,019
P_OpEx_wo_Depr	-0,0480	-0,0551	-0,0618	-0,0198	-0,0145	-0,0092	-0,0039
P_OpEx_o_wo_Depr	-0,0038	-0,0064	-0,005	-0,005	-0,005	-0,005	-0,005

Table 5.8: Forecast AssumptionsIS

Now that we have all the necessary data we move on to building the pro forma income statement. Similarly to the sales data structure we build an empty forecast tsibble with the future years for the income statement and then bind both the historical and forecast tsibble into *IncomeStatAll* to have all the years in one data structure to aid the calculations, because several calculations rely on both historical and future data. In this case, we also join the tsibbles of the assumptions and sales, because most of their values are needed for the calculations and then remove them again once finished. Each variable from the income statement is now calculated using the forecasting drivers that we previously forecast. The calculations are included in functions for each variable and called for each of the future years. This is done by using a condition in each line that skips the rows of the historical years and only applies the calculations to the values of the variable in the future years. This condition is not included in the code below for better visibility. In this step of the calculation we leave the calculation of the net interest expenses for later as described in the operational model, because it depends on variables in the balance sheet. The full income statement is implemented as follows.

```
IncomeStatAll <- IncomeStatAll %>%
 left_join(SalesAll, by = "Year") %>%
 left_join(AssumptionsIS, by = "Year") %>%
 mutate(

  RevT = calculate_RevT(xSA, pA, xSB, pB),
  GM_wo_Depr = calculate_GM_wo_Depr(RevT, P_GM_wo_Depr),
  OpInc_o = calculate_VarPSales(P_OpInc_o, RevT),
  OpEx_wo_Depr = calculate_VarPSales(P_OpEx_wo_Depr,OpEx_wo_Depr, num_hist_years),
```

```
 OpEx_o_wo_Depr = calculate_VarPSales(P_OpEx_o_wo_Depr, RevT),
 EBITDA = calculate_EBITDA(GM_wo_Depr, OpInc_o, OpEx_wo_Depr,OpEx_o_wo_Depr),
 Depr = forcast_Depr(BalanceSheetAll$Asset_nc_gf, DepreciationRate),
 EBIT = calculate_EBIT(EBITDA, Depr),
 Prov_BadDebts = provision,
 Exp_before_ITDA = calculate_Exp_before_ITDA(EBITDA, RevT),
 COGS_wo_Depr = calculate_COGS_wo_Depr(GM_wo_Depr, RevT)
)%>% select(-colnames(SalesAll))%>% select(-colnames(AssumptionsIS))
```

Each of these functions represents the functions we presented step by step in the operational model. We will now show how these functions are implemented in R.
The variables for which we forecast the percentage of sales are now calculated using the same function which multiplies the sales with their percentage.

```
calculate_VarPSales<- function(Var, P_Sales) {
  return(P_Sales *  Var)
}
```

The EBITDA is calculated as the sum of the four input variables as previously presented in the formula 4.15.

```
calculate_EBITDA <- function(GM_wo_Depr, OpInc_o, OpEx_wo_Depr, OpEx_o_wo_Depr) {
  return(GM_wo_Depr + OpInc_o + OpEx_wo_Depr + OpEx_o_wo_Depr)
}
```

The *EBIT* is calculated using the formula in 4.32. The other two earnings variables', *EAT*'s and *EBT*'s, future values and the income tax expense are calculated later on, because they depend on the net interest expenses. The last variable we can calculate now is the cost of goods sold, which is the total revenue minus the gross margin and is equivalently written in R as follows:

```
calculate_COGS_wo_Depr <- function(GM_wo_Depr, RevT){
  return(RevT-GM_wo_Depr)
}
```

The pro forma income statement calculations are now complete for all the variables that are not a part of the circular function.

5.4.2 Pro Forma Balance Sheet

We now move on to the implementation of the balance sheet as seen in the general view 3.4 and in the more detailed view in 3.8. As described in the detailed UML model and the operational model, we first forecast the forecasting drivers and then use the relations between the variables to derive the remaining ones. In the end, some of these variables in combination with the remaining income statement variables form a circular relation, which we iterate through to derive the final values.

The first step to building the pro forma balance sheet, as also seen in the operational model, is to forecast the forecasting drivers, which we already initialized in section 5.3.2. These variables get forecast in the code snippet below.

```
AssumptionsForecastBS <- tibble(Year = FutureYears) %>%
        as_tsibble(index = Year)
AssumptionsBS <- bind_rows(AssumptionsHistBS, AssumptionsForecastBS)
AssumptionsBS <- AssumptionsBS %>%
    mutate(
        DIO = set_future_vec(DIO, Decisions$DIO, numHistYears),
        DSO = set_future_vec(DSO, Decisions$DSO, numHistYears) ,
        DPO = set_future_vec(DPO, Decisions$DPO, numHistYears) ,
        redemption =forecast_redemption(redemption, numHistYears),
        CAPEX = set_future_vec(CAPEX, Decisions$CAPEX, numHistYears),
        VAT = forecast_w_last_value(VAT, numHistYears),
        MinCash = forecast_w_last_value(MinCash, numHistYears)
          )
```

The forecasting of three days outstanding variables and CAPEX were determined by the financial planner after consulting different stakeholders. These values are therefore saved as vectors in a tsibble called *Decisions*. The decision tsibble contains the fixed future values that were set for the variables. These values could be set in the user interface and imported, but in this case we simply set them in the tsibble based on our scenario. The other three variables use the function that sets the last value as forecast to fill in the future values. Redemption calls a different function which also uses the last value as forecast function, but it sets the value of the last forecast year to half of the value of the last historic year as decided by the experts. The functions that set a future vector and forecast redemption are implemented as follows.

```
forecast_redemption <-function(redemption,num_hist_years){

  redemption <- forecast_w_last_value(redemption, num_hist_years )
  redemption[numYears] <- redemption[2]/2
  return(redemption)
}

set_future_vec <- function(measurement , vector, num_hist_years){
  measurement[(num_hist_years+1):numYears] <- vector
  return (measurement)

}
```

The forecast balance sheet assumptions table is displayed in 5.9.

Having forecast all the forecasting drivers and the income statement, we can now build the pro forma balance sheet and then also fill in the missing variables in the income statement. The pro forma balance sheet tsibble is built with its variables and then bound with the historical balance sheet to form a balance sheet tsibble that contains all

Variables	2011	2012	2013	2014	2015	2016	2017
DIO	86	91	95	85	80	80	80
DSO	115	113	110	85	80	75	75
DPO	41	38	30	50	50	55	55
CAPEX	1.425.000	2.590.000	400.000	300.000	300.000	300.000	300.000
Redemption	0	3.033.000	3.033.000	3.033.000	3.033.000	3.033.000	1.516.500
CostSales	995.000	945.000	1.028.000	1.029.000	1.034.000	1.039.000	1.044.000
Distribution	1.200.000	1.122.000	1.207.000	1.208.000	1.214.000	1.220.000	1.225.000
VAT	1,2	1,2	1,2	1,2	1,2	1,2	1,2
MinCash	350.000	350.000	350.000	350.000	350.000	350.000	350.000

Table 5.9: Forecast AssumptionsBS

years called *BalanceSheetAll*. This is done similarly to the other tsibbles to simplify the calculations. In the first part of the forecasting process we calculate the future variables of the balance sheet that are not part of the circular function. This is shown in the code below. The functions represent the variable relations.

```
BalanceSheetAll <- BalanceSheetAll %>%
 left_join(IncomeStatAll, by= "Year") %>%
 left_join(AssumptionsBS, by = "Year") %>%
 mutate(
   Asset_nc_o = forecast_w_last_value(Asset_nc_o, numHistYears),
   Depr_acc = calculate_accumulated_depreciation(Depr_acc,Depr),
   Asset_nc = calculate_non_current_assets(Asset_nc_gf, Depr_acc, Asset_nc_o),
   ShareCap = forecast_w_last_value(ShareCap, num_hist_years),
   AccRec = calculate_accounts_receivable(AccRec, IncomeStatAll$RevT, DSO, VAT),
   Inv = calculate_inventory(Inv, RevT, GM_wo_Depr, DIO),
   Cash = set_future_vec(Cash,MinCash[numHistYears+1:numYears],numHistYears),
   Asset_c_o = calculate_other_current_assets(Asset_c_o),
   Asset_c = calculate_total_current_assets(AccRec, Inv, Cash, Asset_c_o),
   Asset = calculate_total_assets(Asset_c, Asset_nc),
   Debt_lt = calculate_long_term_debt(Debt_lt, redemption),
   AccPay = calculate_account_payables(RevT, GM_wo_Depr, DPO),
   Liability_c_o = forecast_w_last_value(Liability_c_o, num_hist_years))
```

The functions used here are the representation in R of the calculation functions that we presented in the operational model. For some of the variables, the non current assets, the share capital, and the other current liabilities, the function that sets the last historical value is used. The rest of the variables each use their own calculation function using the forecasting drivers and other already calculated variables as input. For the accumulated depreciation the function 4.20 is translated into R as follows.

```
calculate_accumulated_depreciation <- function(Depr_acc, Depr) {
  for (i in 2:length(Depr_acc)) {
    Depr_acc[i] <- Depr_acc[i-1] +Depr[i]
  }
  return(Depr_acc)
}
```

The functions to calculate the asset variables also use the respective functions as presented in the operational model in their calculation functions 4.21, 4.25, 4.26. The total assets variable is made up of the current and non current assets and the function is written in R as follows.

```
calculate_total_assets <- function(Asset_c, Asset_nc) {
  Asset <- Asset_c + Asset_nc
  return(Asset)
}
```

The functions for the non-current and current assets are implemented equivalently to 4.21 and 4.25.

```
calculate_total_current_assets <- function(AccRec, Inv, Cash, Asset_c_o) {
  Asset_c <- AccRec + Inv + Cash + Asset_c_o
  return(Asset_c)
}

calculate_accounts_receivable <- function(AccRec, Sales, DSO, VAT) {
    AccRec <- Sales * DSO / 365 * VAT
  return(AccRec)
}
```

To calculate the accounts receivable, payable and inventory we use their relation to the forecast drivers DSO, DIO and DSO from 4.22, 4.28, 4.23.

```
calculate_accounts_receivable <- function(sales_volume, DSO) {
  accounts_receivable <- sales_volume * DSO / 365
  return(accounts_receivable)
}

calculate_inventory <- function(Inv, Sales, GM, DIO) {
    CostOfGoodsSold <- Sales - GM
    Inv<- CostOfGoodsSold * DIO / 365
    return(Inv)
}

calculate_account_payables <- function(Sales, GM, DPO) {
  AccPay <- (Sales - GM) * DPO / 365
  return(AccPay)
}
```

Finally, for the long-term debt equation 4.27 we use a for each loop to access the previous years long term debt and substract this years redemtion.

```
calculate_long_term_debt <- function(Debt_lt, redemption) {
  for (i in 2:length(Debt_lt)) {
    Debt_lt[i] <- Debt_lt[i-1] - redemption[i]
  }
  return(Debt_lt)
}
```

The balance sheet variables that do not depend on the income statement are now forecasted. To calculate the rest of the variables we need to apply the circular function that we described in 4.3. The circular function is of composed of variables whose functions depend on each other. For this reason we need to iterate a few times until the resulting values of the variables stabilize. We will derive the retained earnings from the circular function and then apply this variable again to all the other functions to . The dependency of the variables is visualized in 4.3. In R this calculation is done through the circular dependency function as follows.

```
calculate_circular_updates <- function(Asset, Equity, Debt_lt,Debt_st, ShareCap,
 AccPay, Liability_c_o, Cash, GM_wo_Depr, OpInc_o, OpEx_wo_Depr, OpEx_o_wo_Depr,
 Depr, IncTaxEx,IntEx_net,EAT,EBT, P_Deposit_rate, P_fin_cost_lt, P_fin_cost_st,
 RetEarn, num_hist_years, EffTaxRate) {

  max_iterations <- 4
  max_change <- 0.001

  old_retearn <- RetEarn
  new_retearn<- num(numYears)

  iteration<-0
  while (iteration< max_iterations && abs(old_retearn[1] - new_retearn[1])>max_change){
    old_retearn<-new_retearn

    Equity <- calculate_equity(ShareCap, old_retearn)
    Debt_st <- calculate_short_term_debt(Asset, Equity, Debt_lt,AccPay,Liability_c_o),
    IntEx_net <- calculate_net_intrest_expense(Cash,P_Deposit_rate,Debt_lt,P_fin_cost_lt,
                  Debt_st,P_fin_cost_st)
    EBT <-calculate_EBT(calculate_EBIT(calculate_EBITDA(GM_wo_Depr,
                  OpInc_o,OpEx_wo_Depr, OpEx_o_wo_Depr),Depr),IntEx_net),
    EAT <- calculate_EAT(EBT,IncTaxEx),
    new_retearn[i] <-calculate_retained_earnings(new_retearn, EAT)

    iteration<- iteration +1
  }
  return (new_retearn)
}
```

We allow for 4 iterations, because they should suffice to stabilize the values and use the retained earnings variables to check if the difference of this iteration to the last one is smaller than the threshold of 0,001. That means the calculations will iterate until the retained earnings stop changing more than 0,001 from one iteration to the other or the maximum iteration number is reached. The called functions for each variable represent the equations that we described in the operational model. All of these functions only contain the equation to calculate the variables as presented in 4.34, 4.35, 4.36, 4.31, 4.30 and 4.29. We will not be visualising these functions as they are built equivalently to the previous functions based on their equations in the operational model. When the conditions are fulfilled we have the final value of the retained earnings and can go back to our balance sheet tsibble to fill in the retained earnings and update the rest of the

variables that depend on this variable as shown below.

```
BalanceSheetAll <- BalanceSheetAll %>%
mutate(
  RetEarn = calculate_circular_updates(Asset, Equity, Debt_lt,Debt_st,
            ShareCap, AccPay, Liability_c_o, Cash, GM_wo_Depr, OpInc_o,
            OpEx_wo_Depr, OpEx_o_wo_Depr, Depr, IncTaxEx,IntEx_net,EAT,
            EBT, AssumptionsIS$P_Deposit_rate, AssumptionsIS$P_fin_cost_lt,
            AssumptionsIS$P_fin_cost_st,RetEarn, num_hist_years,EffTaxRate)
  Equity = calculate_equity(ShareCap, RetEarn),
  Debt_st = calculate_short_term_debt(Asset, Equity, Debt_lt, AccPay, Liability_c_o),
  Liability_nc = calculate_non_current_liabilities(Debt_lt),
  Liability_c = calculate_total_current_liabilities(Debt_st, AccPay, Liability_c_o),
  Liability = calculate_total_liabilities(Liability_c, Liability_nc),
  EquAndLiab = calculate_equity_and_liabilities(Equity, Liability),
  Debt_lt = calculate_long_term_debt(Debt_lt, redemption),
      )%>% select(-colnames(IncomeStatAll)) %>% select(-colnames(AssumptionsBS))
```

The balance sheet is now fully forecast. The income statement is still missing some values which depend on the balance sheet, which we can now also calculate.

```
IncomeStatAll <- IncomeStatAll %>%
left_join(BalanceSheetAll, by= "Year") %>%
left_join(AssumptionsIS, by = "Year") %>%
 mutate(
  IntEx_net = calculate_net_intrest_expense(Cash, P_Deposit_rate,
     Debt_lt, P_fin_cost_lt, Debt_st, P_fin_cost_st)
  EBT = calculate_EBT(EBIT, IntEx_net),
  IncTaxEx = calculateIncomeTaxExpenses(IncTaxEx, EffTaxRate, EBT)
  EAT = if_else(row_number() > num_hist_years, calculate_EAT(EBT, IncTaxEx),
   )%>% select(-colnames(BalanceSheetAll)) %>% select(-colnames(AssumptionsIS))
```

The results of the calculations of the income statement and balance sheet can be now seen in table 5.11 and 5.10.

Balance Sheet	2011	2012	2013	2014	2015	2016	2017
Asset_nc_gf	58.849.955	61.439.955	61.839.955	62.139.955	62.439.955	62.739.955	63.039.955
Depr_acc	15.000.000	17.067.000	19.302.000	21.539.000	23.787.000	26.046.000	28.315.000
Asset_nc_o	18.840.127	18.840.127	18.840.127	18.840.127	18.840.127	18.840.127	18.840.127
Asset_nc	62.690.081	63.213.081	61.378.081	59.441.081	57.493.081	55.534.081	53.565.081
AccRec	69.968.958	57.466.059	47.917.808	32.853.617	27.983.302	24.206.570	22.764.311
Inv	39.983.640	34.961.959	31.210.103	24.777.102	21.104.074	19.472.841	18.312.623
Cash	15.578.713	12.289.760	7.000.000	3.500.000	3.500.000	3.500.000	3.500.000
Asset_c_o	3.736.108	2.483.105	1.230.102	1.230.102	1.230.102	1.230.102	1.230.102
Asset_c	129.267.419	107.200.883	87.358.013	62.360.821	53.817.478	48.409.513	45.807.036
Asset	191.957.500	170.413.964	148.736.095	121.801.903	111.310.559	103.943.595	99.372.118
ShareCap	31.259.993	31.259.993	31.259.993	31.259.993	31.259.993	31.259.993	31.259.993
RetEarn	1.472.155	3.119.932	809.693,2	-1.070.932,5	-3.644.581,4	-6.760.462,8	-10.464.360,8
Equity	32.732.148	34.379.925	32.069.686	30.189.060	27.615.412	24.499.530	20.795.632
Debt_lt	59.383.334	56.350.334	53.317.334	50.284.334	47.251.334	44.218.334	42.701.834
Liability_nc	59.383.334	56.350.334	53.317.334	50.284.334	47.251.334	44.218.334	42.701.834
Debt_st	76.167.787	61.795.252	51.346.153	24.606.643	21.106.669	19.691.053	21.137.624
AccPay	19.128.353	14.541.964	9.855.822	14.574.766	13.190.046	13.387.578	12.589.929
Liability_c_o	4.545.877	3.346.488	2.147.099	2.147.099	2.147.099	2.147.099	2.147.099
Liability_c	99.842.018	79.683.704	63.349.074	41.328.508	36.443.813	35.225.730	35.874.651
Liability	159.225.352	136.034.039	116.666.408	91.612.842	83.695.148	79.444.064	78.576.485
EquAndLiab	191.957.500	170.413.964	148.736.095	121.801.903	111.310.559	103.943.595	99.372.118

Table 5.10: Balance Sheet: Tsibble Results

Income Statement	2011	2012	2013	2014	2015	2016	2017
RevT	185.500.000	154.683.713	132.500.000	117.564.412	106.394.847	98.171.090	92.321.927
Exp_before_ITDA	176.352.895	145.687.096	126.218.209	112.770.034	102.701.639	95.301.649	90.059.703
COGS_wo_Depr	170.289.000	139.679.392	119.912.500	106.395.793	96.287.336	88.844.837	83.551.344
GM_wo_Depr	15.211.000	15.004.320	12.587.500	11.168.619	10.107.510	9.326.254	8.770.583
OpInc_o	3.554.707	3.492.877	2.539.076	2.233.724	2.021.502	1.865.251	1.754.117
OpEx_wo_Depr	-8.911.568	-8.515.509	-8.182.286	-8.020.142	-7.903.831	-7.831.207	-7.800.866
OpEx_o_wo_Depr	-707.033	-985.072	-662.500	-587.822	-531.974	-490.855	-461.609
EBITDA	9.147.105	8.996.617	6.281.791	4.794.379	3.693.207	2.869.441	2.262.224
Depr	-2.195.000	-2.067.000	-2.235.000	-2.237.038	-2.247.838	-2.258.638	-2.269.438
EBIT	6.952.105	6.929.616	4.046.791	2.557.340	1.445.369	610.803	-7.215
Prov_BadDebts	0	0	0	0	0	0	0
IntEx_net	-4.754.902	-4.470.258	-6.356.949	-4.442.407	-4.038.294	-3.772.791	-3.783.306
EBT	2.197.261	2.459.359	-2.310.173	-1.885.066	-2.592.925	-3.161.988	-3.790.520
IncTaxEx	-725.096	-811.589	0	0	0	0	0
EAT	1.472.165	1.647.771	-2.310.173	-1.885.066	-2.592.925	-3.161.988	-3.790.520

Table 5.11: Income Statement: Tsibble results

5.4.3 Pro Forma Cash Flow Statement

The forecasting activity is now complete. The income statement and the balance sheet contain the historical and future variables. Considering the steps in the general UML diagram 3.4 we now can input the historical and pro forma financial statements into the cash flow equations to derive the cash flow statement with both the historical and future values. This cash flow equations include all the equations that calculate the cash flow variables using balance sheet and income statement variables as input. For this purpose we initialize the cash flow tsibble for all the years and join the two financial statement tsibbles to easily access their variables for the calculations. When the calculations are done we remove the variables that we do not need at the end again.

```
CashFlow <- tibble(Year = AllYears,) %>%
                      as_tsibble(index = Year)

  CashFlow <- CashFlow %>%
    left_join(IncomeStatAll, by= "Year") %>%
    left_join(BalanceSheetAll, by= "Year") %>%
    mutate(
      DeprAmor = Depr,
      DRecv = calculate_differences(AccRec),
      DInv = calculate_differences(Inv),
      DPay = -calculate_differences(AccPay),
      DRecPay = calculate_differences(Asset_c_o) - calculate_differences(Liability_c_o),
      InOutOp = calculate_inoutop(EBT, DeprAmor, DRecv, DInv, DPay, DRecPay),
      DAssets = calculate_differences(Asset_nc_gf),
      InOutInv = DAssets,
      DShareCap = calculate_differences(ShareCap),
      DLoans = calculate_differences(Debt_lt) + calculate_differences(Debt_st),
      InOutFin = calculate_inoutfin(DShareCap, DLoans) ,
      DCashBeg = calculate_dcash_beg(Cash, Decisions$Cash_beg[1]),
      DCash = calculate_dcash(InOutOp, InOutInv, InOutFin, IncTaxEx),
      DCashEnd = calculate_dcash_end (DCash, DCashBeg),
       DCash[1] <- DCashEnd[1] - DCashBeg[1]
    ) %>% select(-colnames(BalanceSheetAll))
      %>% select(-one_of(setdiff(names(IncomeStatAll), "EBT")) )
```

The variables in the cash flow statement show differences in accounts, i.e. the inflows and outflows for different activities, therefore most variables are calculated as the difference of the balance from one period to another and the rest are the sum of these differences for the different kinds of activities: operating, financing and investing activities. The function that is mostly used is the *calculate_differences* function, which computes the difference between the value of this year and the last ones using a for each clause.

```
calculate_differences <- function(values) {
  diffs <- numeric(length(values))
  for (i in 2:length(values)) {
    diffs[i] <- values[i] - values[i-1]
  }
  return(diffs)
}
```

Then there are also three other functions. The first calculates the sum of the inflows and outflows for the financial activities.

```
calculate_inoutfin <- function(DShareCap, DLoans) {
  InOutFin <- DShareCap + DLoans
  return(InOutFin)
}
```

To calculate the cash in the beginning of the period we set its value to the value of the last year. For this the value for the first year is missing in the data of the last three years,

so we retrieve the value and set it for the first year. In an integrated information system this value would be available to source through a query.

```
calculate_dcash_beg<- function(Cash, cashbeg){
  DCashBeg <- numeric(length(Cash))

  DCashBeg[1]<- cashbeg
  DCashBeg[2:length(Cash)] <- Cash[1:(length(Cash)-1)]
  return (DCashBeg)

}
```

Then we calculate the cash difference by summing up the total in- or outflows of investing, operating and financing activities and the income tax expenses. The first value is set to zero, because the differences for the first year are not computed due to the missing value of the precursive year.

```
calculate_dcash <- function(InOutOp, InOutInv, InOutFin, IncTaxEx){
  DCash <- numeric(length(InOutInv))
  DCash[1] <- 0
  DCash[2:length(InOutInv)] <- InOutInv[2:length(InOutInv)] +
    InOutOp[2:length(InOutInv)] - InOutFin[2:length(InOutInv)] +
    IncTaxEx[2:length(InOutInv)]
  return(DCash)
}
```

In the end, by adding the cash change to the cash in the beginning of the period we calculate the cash in the end of the period.

```
calculate_dcash_end<- function(DCash,DCashBeg ){
  DCashEnd <- numeric(length(DCash))
  DCashEnd[1] = DCashBeg[2]
  DCashEnd[2:length(DCash)] <- DCash[2:length(DCash)] + DCashBeg[2:length(DCash)]
  return(DCashEnd)
}
```

The results of the cash flow statement are shown in the table 5.12

This completes the implementation of the historical and pro forma financial statements.

Cash Flow	2011	2012	2013	2014	2015	2016	2017
EBT	2.197.261	2.459.359	-2.310.173	-1.885.066	-2.592.925	-3.161.988	-3.790.520
DeprAmor	2.195.000	2.067.000	2.235.000	2.237.038	2.247.838	2.258.638	2.269.438
DRecv	-	12.502.899,2	9.548.250,5	15.064.191,6	4.870.314,5	3.776.731,9	1.442.259,4
DInv	-	5.021.681,3	3.751.856,2	6.433.000,2	3.673.028,8	1.631.232,7	1.160.217,6
DPay	-	-4.586.389,3	-4.686.142,2	4.718.944,3	-1.384.720,1	197.532,1	-797.649,6
DRecPay	-	53.613,27	53.613,27	0	0	0	0
InOutOp	4.392.260,9	17.518.163,7	8.592.405	26.568.108,1	6.813.536,6	4.702.147,2	283.745,6
DAssets	-	-2.590.000	-400.000	-300.000	-300.000	-300.000	-300.000
InOutInv	-	-2.590.000	-400.000	-300.000	-300.000	-300.000	-300.000
InOutFin	-	17.405.534,84	13.482.098,93	29.772.510,43	6.532.974,32	4.448.615,33	69.929,44
DShareCap	0	0	0	0	0	0	0
DLoans	-	17.405.534,84	13.482.098,93	29.772.510,43	6.532.974,32	4.448.615,33	69.929,44
DCash	-	-3.288.959,68	-5.289.693,93	-3.504.402,30	-19.437,72	-46.468,13	-86.183,81
DCashBeg	17.853.000	15.578.713	12.289.760	7.000.000	3.500.000	3.500.000	3.500.000
DCashEnd	15.578.713	12.289.754	7.000.066	3.495.598	3.480.562	3.453.532	3.413.816

Table 5.12: Cash Flow Statement: Tsibble results

5.5 Evaluation

The evaluation of this thesis is done using a use-case evaluation and a short qualitative evaluation of the implementation process. We applied the model to a use-case from literature, which uses real data from a metal processing company, SteelCo. More concretely the evaluation is done through the models, implementation in R and in Excel. The first step we took to evaluate the work was to identify the gaps in literature concerning the activity based budgeting domain, and determine the usefulness of building a common understanding through a ubiquitous language. We then followed Samonas' instructions to implement the forecasting model in Excel using the data from SteelCo. This was done to get a general understanding and idea of how the process works, to determine how time consuming and how complicated it is to implement, and how error prone and difficult to diagnose it is in order to provide a simple qualitative evaluation on the process and have a ground truth for the our implementation. The Excel spreadsheet results are also shown in 4.7 and 4.12.

The conceptual model is an abstract view of the process which shows the structure of the forecasting, entities, relationships and workflow. It defines a set of concepts to be used for a common understanding in the process. It presents the forecasting model from its most simple and generic form to the most detailed and specific use-case form by fulfilling all requirements stemming from the accounting, finance, management control and specifically the ABB domain. This is shown by applying the ABB use-case to the model. The operational model operationalizes the conceptual model using the data and decisions from the use-case. It uses the entities as described in the conceptual model, but adds much more detail to the process, by presenting all the steps in a higher granularity. It does not only show the statements and their relationships, but all the variables in the statements and how they are computed, it goes into detail about the forecasting models, defining the specific algorithms and the calculations that are used to derive each variable of the pro forma statements. This presents a design of the implementation of the activity

based budgeting process. By defining and fulfilling requirements based on the relevant domains and correctly applying a use-case from literature to the models we already validate the conceptual and operational model and demonstrate their functionality and application.
This detailed model is then used for the implementation of this use-case in R. The implementation process follows the operational model step by step and transitions the variables, forecasting models and calculations into the R programming language. The results of the implementation are then compared to the results that we achieved in the spreadsheet. Our results align with the results that Samonas' achieved in his book [Sam15]. For the comparison we used a percentage difference formula as shown in 5.1. In tables 5.13 and 5.14 we see the difference in percentage of the variables in the financial statements where we computed the forecasting activity.

$$\Delta var = (var_{excel} - var_R)/var_R * 100 \tag{5.1}$$

Balance Sheet	2011	2012	2013	2014	2015	2016	2017
Asset_nc_gf	0	-0,0006	-0,0006	-0,0006	-0,0006	-0,0005	-0,0005
Depr_acc	0	0	0,0005	0,0006	-0,0002	-0,0016	0
Asset_nc_o	0	0	0	0	0	0	0
Asset_nc	0	-0,0005	-0,0007	-0,0008	-0,0005	0,0001	-0,0006
AccRec	0	0	0	0	0	0	0
Inv	0	0	0	0	0	0	0
Cash	0	0	0	0	0	0	0
Asset_c_o	0	0	0	0	0	0	0
Asset_c	0	0	0	0	0	0	0
Asset	0	-0,0002	-0,0003	-0,0004	-0,0003	0,0001	-0,0004
ShareCap	0	0	0	0	0	0	0
RetEarn	0	0	0	0,4412	0,705	1,1381	1,6744
Equity	0	0	0	-0,0156	-0,093	-0,314	-0,8426
Debt_lt	0	-0,0006	-0,0012	-0,002	-0,0028	-0,0038	-0,0043
Liability_nc	0	-0,0006	-0,0012	-0,002	-0,0028	-0,0038	-0,0043
Debt_st	-0,0014	0	0,0004	0,0213	0,1266	0,3996	0,836
AccPay	0,0057	0	0	0	0	0	0
Liability_c_o	0	0	0	0	0	0	0
Liability_c	0	0	0,0003	0,0127	0,0733	0,2234	0,4926
Liability	0	-0,0003	-0,0004	0,0046	0,0303	0,0969	0,2225
EquAndLiab	0	-0,0002	-0,0003	-0,0004	-0,0003	0,0001	-0,0004

Table 5.13: Balance Sheet Results Difference in %

By achieving aligned results to the values in Excel we can prove that the implementation is valid. The validity of the implementation additionally shows the validity of the operational model it was built on, and the validity of the operational model shows the validity of the conceptual model it operationalized. The results we achieved show an average difference of 0,0167% for the income statement, an average difference of 0,0532% for the balance sheet and an average difference of 0,128% for the cash flow statement. We allow this slight tolerance because of several factors. The implementation uses full numbers, while the excel implementation uses the values in thousands to make the spreadsheets more

Income Statement	2011	2012	2013	2014	2015	2016	2017
RevT	0	0	0	0	0	0	0
Exp_before_ITDA	0	0	0	0	0	0	0
COGS_wo_Depr	0	0	0	0	0	0	0
GM_wo_Depr	0	0	0	0	0	0	0
OpInc_o	0	0	0	0	0	0	0
OpEx_wo_Depr	0	0	0	0	0	0,0001	0
OpEx_o_wo_Depr	0,0001	0,0003	0	0	0	-0,0001	0,0001
EBITDA	-0,0002	0	-0,0001	-0,0001	-0,0001	-0,0001	-0,0001
Depr	0	0	0,0045	-0,0005	-0,0005	-0,0005	-0,0005
EBIT	-0,0002	0	-0,0026	0,0003	0,0006	0,0015	-0,1319
Prov_BadDebts	0	0	0	0	0	0	0
IntEx_net	-0,0012	-0,0002	-0,0004	0,0066	0,0422	0,1364	0,3084
EBT	-0,0007	0,0004	0,0029	0,0151	0,0654	0,1624	0,3076
IncTaxEx	-0,0007	0,0004	0	0	0	0	0
EAT	-0,0007	0,0004	0,0029	0,0151	0,0654	0,1624	0,3076

Table 5.14: Income Statement Results Difference in %

readable. In Excel the percentages are rounded while in the implementation this step is not needed as many calculations happen in the background and rounding before the calculations are computed is redundant. This allows for a higher precision of the results in the R implementation. In the spreadsheet solution it was more time consuming and harder to diagnose the process or make changes, because most functions are hidden, small mistakes are harder to find and each cell or row has to be checked and updated manually. In R all the equations and algorithms are written in separate functions that are called where necessary. Most functions in R are reused for different variables, while in the spreadsheet the functions need to be rewritten in every cell they apply to. This makes it easier to make changes in R. When something is changed or extended in a function, the change is automatically applied to all the variables where the function is called. In the spreadsheet implementation this change has to be done manually in each cell, increasing error proneness and the time needed to apply changes. The data also took more time to arrange in Excel sheets, while in R it was simply done by one import from our data storage. All these factors show the perceived benefits of the implementation in a programming environment compared to Excel.

CHAPTER 6

Conclusion

The purpose of this thesis was to define the activity based budgeting domain and build a common understanding for the accounting, finance, management control and IT domain using a ubiquitous language. It aimed to fulfill this purpose by providing a conceptual model to serve as a common language, operationalize it using a use-case from literature and then demonstrate its practical applicability by building a prototype of the domain driven implementation in a robust programming environment, specifically R-tidyverse.

6.1 Summary of conclusions

This thesis made significant contributions to the understanding and application of the ABB system and its implementation and integration in ERP systems in several ways. The development of the conceptual model represents an abstract view of the forecasting process, by visualizing the structure, entities, relationships and workflows involved, which then serves as a theoretical foundation for the budgeting process. It developed a common language and understanding on the process based on requirements stemming from the accounting, finance and management control domains. This model laid the groundwork for the operational model which went deeper into the specifics of the financial forecasting process. The operational model detailed the steps necessary for the forecasting process, including the variables, algorithms and calculations required to produce accurate pro forma statements. It provides a valid design for the implementation. The following implementation in R represents a practical use of these theoretical models. The transition from the abstract conceptual model to a detailed operational model and finally to a working implementation demonstrates the applicability of our models. The R implementation not only replicated the results achieved in Excel but achieved this with a higher precision and efficiency, displaying the advantages of using a programming environment for such tasks. Furthermore the developed models and implementation take into consideration the application of financial models in existing enterprise information

systems, specifically the widely used ERP systems, to provide an integrated approach to forecasting and budgeting.

6.2 Key findings

This thesis fills a gap in literature by providing a conceptual definition for the activity based budgeting domain and a common language that facilitates communication between the involved financial management and control domains and the IT domain. The presented conceptual model, its operationalization and demonstration through a use case align with the requirements of the domains they operate in and provide a valid definition of the process. On the other hand the implementation in R-tidyverse demonstrates a valid applicability in the IT-domain. The comparison between the R implementation and the Excel model proves that the results of our implementation align closely with the results documented by Samonas' and therefore validate it and show the accuracy and reliability of the R-based approach. The slight difference in results between the R and Excel implementation, despite the differences in rounding and numerical precision proves the effectiveness of the R implementation. The small variance is due to the differences of how the calculations are handled in R comparing to Excel, with R offering a higher level of precision. The implications of these findings are significant. For practitioners, the R implementation provides a more efficient, less error-prone, more automated and future oriented alternative to the current spreadsheet-based financial forecasting. The ability to import data seamlessly, integrate financial models into current information systems, reuse functions across multiple variables and easily apply changes across the entire model makes a programming environment a superior choice for implementing complex forecasting frameworks. For academics and researchers, this work offers a conceptual definition and a common language of the activity based budgeting domain for understanding, extending or transitioning the budgeting model from the traditional spreadsheet implementation to a robust data science programming environment.

6.3 Challenges and Limitations

The development of the definition and common language required a deep understanding of the complex budgeting process in the activity based budgeting and the unrelated IT domain. The definition of the process through the UML diagrams was not without challenges. It required multiple iterations in order to provide a well-structured, clear and intuitive representation that accurately captures the relationships, workflows, and entities involved, while remaining accessible to stakeholders from diverse domains.
While the R implementation proved successful the process also had its challenges. Implementing the model in Excel highlighted the limitations of spreadsheet-based approaches, including the difficulty in diagnosing errors, the time-consuming arrangement of the input data and formulas and the need to manually update functions in each applicable cell. On the other hand, the R implementation, required a careful translation of the operational model into code. Debugging and optimizing the R code, particularly for more

complex calculations, presented its own set of challenges. The slight differences between the results in R and Excel are regarded as acceptable given the higher precision in R and the context of the implementation.
This thesis focuses on defining the activity-based budgeting (ABB) domain and developing a ubiquitous language, with a theoretical approach that lacks a real-world application. Empirical validation of the models through practical implementation remains a task for future research. The use-case evaluation, while demonstrating alignment with ABB processes, is based on a single example from the literature. Although the application of the model was done using instances of the generic entities further applications would further prove its generalizability.

6.4 Future Research

Building upon the work presented on this paper offers a base for future work in several directions. This paper theoretically defines the process underlying the ABB domain and establishes a ubiquitous language to bridge the accounting, finance, management control, and IT domains. Further work could be done to show the practical application of this ubiquitous language to integrate financial models in specific information systems, which would aid the forecasting and budgeting activities connecting directly to the operational activities in an enterprise. Additionally, expanding the implementation to apply it to different financial markets and scenarios would provide further insight into the versatility and robustness of this definition. This would also aid in assessing its effectiveness in improving communication and the clarity of budgeting processes across industries with industry specific variations of the budgeting process to provide a deeper insight into its applicability. Another possibility is to explore the extension of the model to include identified patterns and rules in decision making to further automate the budgeting workflows and integrating artificial intelligence and machine learning to analyze historical data and external factors to generate intelligent suggestions.

6.5 Final Remarks

This thesis has presented the development and evaluation of conceptual and operational models aimed at defining the process underlying activity based budgeting domain and a ubiquitous language to bridge the communication between the accounting, finance, management control and IT domains. Through a use-case evaluation from the literature, these models were demonstrated to align with key requirements from the involved domains, and showcase their practical applicability. This approach underscores the value of the proposed models in bridging the conceptual understanding with the operational execution. While the scope of this thesis is limited to a specific use-case, it establishes a foundation for future applications and further refinement in different organizational contexts, therefore ensuring its relevance in both the academic field and the industry.

Overview of Generative AI Tools Used

The assistance of generative AI tools, specifically ChatGPT , was only used for building structured result tables of the financial statements in Latex swiftly based on excel spreadsheet tables or R result tables.

List of Figures

List of Tables

Bibliography

[AK11] Samuel J. Seymour Steven M. Biemer Alexander Kossiakoff, William N. Sweet. *Systems engineering : principles and practice—2nd ed.* Springer Berlin, Heidelberg, 2011.

[Bri94] James Brimson. *Activity-based management : for service industries, government entities, and nonprofit organizations.* Wiley, 1994.

[EK21] AfşarAslı Afşar Erdem Korkmaz. Activity-based budgeting for hotel food and beverage services: A case study. *International Journal of Arts and Commerce*, 10(9), 2021.

[GP11] Cesar Gonzalez-Perez. *Domain-Driven Design.* Addison-Wesley, 2011.

[GP18] Cesar Gonzalez-Perez. *Information Modelling for Archaeology and Anthropology.* Springer Cham, 2018.

[Hev07] A. R. Hevner. Design science research methodology for information systems research. *Journal of Management Information Systems*, 24:3:45–77, 2007.

[HTJ87] Robert S. Kaplan H. Thomas Johnson. *Relevance Lost: The Rise and Fall of Management Accounting.* Boston, Mass.: Harvard Business School Pr., 1987.

[Hut09] D.M. Hutton. Clean code: A handbook of agile software craftsmanship. *Kybernetes*, 38 (6):1035–1035, 2009.

[iec16] Enterprise-control system integration – part 5: Business to manufacturing transactions, 2016. English version EN 62264-5:2016.

[Int05] International Organization for Standardization. ISO/IEC 19501: Information technology – Open Distributed Processing – Unified Modeling Language (UML) Version 1.4.2, 2005. International Standard.

[IRB13] Tony Clark Sholom Cohen Jorn Bettin Iris Reinhartz-Berger, Arnon Sturm. *Domain Engineering.* Springer Berlin, Heidelberg, 2013.

[JAB12] Edward Mendlowitz James A. Brimson, John J. Antos. *Handbook of Budgeting, Sixth Edition.* John Wiley Sons, 2012.

[Joh24] Henderson A. Johnson, J. *Conceptual Models. Synthesis Lectures on Human-Centered Informatics.* Springer Cham, 2024.

[Kap88] Robert S. Kaplan. One cost system isn't enough. *Harvard Business Review*, 66:61–66, 1988.

[KCH+90] Kyo Kang, Sholom Cohen, James Hess, William Novak, and A. Peterson. Feature-oriented domain analysis (foda) feasibility study. Technical Report CMU/SEI-90-TR-021, Carnegie Mellon University, Nov 1990. Accessed: 2024-Sep-25.

[KP04] Marcus A. Rothenberger Samir Chatterjee Ken Peffers, Tuure Tuunanen. Design science in information systems research. *MIS Quarterly*, 28:1:75—-105, 2004.

[LRM10] Libby Theresa Lindsay R. Murray. Beyond budgeting or budgeting reconsidered? a survey of north american budgeting practice. *Management Accounting Research*, 21:56–75, 2010.

[MHAK21] Zina Adnan Kareem Maytham Hazim Azeez Khzaee, Donya jasim sahib. The strategic impact of activity-based budgeting technology to improve profitability and increase competitiveness. *Himalayan Journals*, 4(1), 2021.

[MM21] M. Oberholzer M.J. Maluleke. *Considerations of Activity-Based Costing in the Public Sector: The Case of SASSA*. Palgrave Macmillan, 2021.

[Pie14] Zaneta Pietrzak. Traditional versus activity-based budgeting in non-manufacturing companies. *Social Sciences*, 82(4), 2014.

[R] https://www.r-project.org/about.html.

[RC91] Robert S. Kaplan Robin Cooper. *The Design of Cost Management Systems.* Prentice Hall, 1991.

[RMa] The r manuals.

[Rob06] S. Robinson. Conceptual modelling for simulation part i: definition and requirements. *Journal of the Operational Research Society*, 59(3):278–290, 2006.

[Sam15] Michael Samonas. *Financial forecasting, analysis, and modelling : a framework for long-term forecasting.* Wiley, 2015.

[SCH03] Wim A. Van der Stede Stephen C. Hansen, David T. Otley. Practice developments in budgeting: An overview and research perspective. *Journal of Management Accounting Research*, 15:95–116, 2003.

[SM84] Rob J Hyndman Spyros Makridakis, C. Wheelwright. Forecasting: Methods and applications. *Journal of the Operational Research Society*, 35(1):345–346, 1984.

[Sta71] G. Staubus. *Activity Costing and Input-Output Accounting*. R.D. Irwin, Homewood, Ill., 1971.

[tid] https://www.tidyverse.org/.

[tsi] https://cran.r-project.org/web/packages/tsibble/index.html.