technische universität
dortmund

Masterthesis

A PLC based real time material flow controller/calculator in warehouse management system

Siddharth Chithadka
25 June 2018

Supervisors:
Prof. Dr-Ing. Bernd Noche
Prof. Dr. Jian Jia Chen
M.Sc Cyril Alias

Technische Universität Dortmund
Fakultät für Informätik
Lehrstuhl Informatik 12(Eigenbettete Systeme)
http://ls12-www.cs.tu-dortmund.de

Acknowledgements

The earnest efforts I have taken in my thesis couldn't have been accomplished single handedly. Therefore I would like to share the credit, with all individuals, who guided me in my thesis work.

At first I would like to sincerely thank Prof. Dr. Jian Jia Chen, Prof. Dr-Ing. Bernd Noche and Cyril Alias for their support and technical guidance during my thesis work. Besides my supervisors I would also extend my gratitude to Software lead engineer Andreas Krieger and other team members in ProLog Automation GmbH & Co. KG for offering me an opportunity to work on this topic and providing technical inputs during my thesis work.

Lastly, my thesis would be inchoate if I fail to express my profound gratitude to my parents, family and friends whose unwavering moral support and motivation throughout my Masters program helped in shaping my thesis work.

Foreword

The Master thesis work contains confidential software data model of ProLog Automation GmbH & Co KG. Any publication or duplication of this thesis or even in part is prohibited.

Some of the standard logistics terms like Material Fluss Rechner(MFR), Automatisches Kleinteillager(AKL) and Fördertechnik(FT), used in several block diagrams and texts in this thesis refers to Material Flow controller, Small warehouse and Material handling(conveyor system) respectively. Earnest efforts have been made to address most of the standard German logistics terms in in its respective form in English and few of them have also been mentioned in the section of Abbreviations.

Abstract

The thesis involves developing the software architecture for material flow calculator (Material Fluss Rechner: MFR) in small warehouse systems, which not only helps in automating the process in semi automated or manual warehouse conveyor line, but also increases the productivity of the same by automatically scanning and updating the database. It further leads to systematic representation of data in a warehouse thereby reducing the error in various stages of material in warehouse and eventually reducing the turnaround time. The proposed software model and architecture guides the material on conveyor line to the required destination in warehouse and it updates the datalog file cyclically. The main emphasis on developing the Material flow calculator software model is laid on the algorithm which guides the material to the required destination as early as possible by combining the ideas from various optimisation algorithms like Improved Djikstra's method and Disjunctive Branch and Bound. Furthermore a brief working algorithm model to synchronise the material flow with the database is also developed. In this thesis work we propose a routing algorithm by modelling the material flow as a job shop problem considering all the constraints and by having makespan as an objective function. The above proposed heuristic algorithm tends to counter problems seen like blocking or deadlocks and schedules the flow of material in a optimised manner on a conveyor line. In order to quantify the algorithm the logic is written in a PLC programming language (IEC 61131-3), similar to ANSI C language which is realised through B&R Automation Studio software ® and the data during material flow is recorded by a Bar code reader which is later synchronized cyclically with the SQL server database. To validate the algorithm, the proposed software model was simulated on a B&R PLC PC 2100, which identified MFR's performance under various scenarios.

Contents

List of Figures

List of Tables

Abbreviations

List of Abbreviations

AIDC Automatic Identification and data capture

ERP Enterprise resource planning

IEC International Electrotechnical Commission

MFR Material Fluss Rechner(German)/Material Flow Controller(English)

OPC Open Platform Communications

PLC Programmable logic controller(English)/ Speicherprogrammierbare steuerung:SPS

TCP/IP Transmission control protocol

WMS Warehouse management systems

Mathematical Notations

Notation	Meaning
$\mathbf{C_{max}}$	Makespan
$\mathbf{J_m}$	Job Shop with m machines
prec	Precedance constraint
$\mathbf{r_{ij}}$	Release time of job j in machine i
$\mathbf{p_{ij}}$	Processing time of job j in machine i
block	blocking of job j

Chapter 1

Introduction

1.1 Motivation

Warehouses presently take around 2%-5% of the cost of sales of a corporation and with the current trends such as globalization, Industry 4.0, E-commerce or same-day delivery present new challenges for logistic companies[12]. For example a typical warehouse that extends over 60,000 square meters will be employing hundreds of people working to gather and consolidate the good in orders to meet daily shipment schedules. Even though warehouses serve different purposes, mostly all share the same general pattern of material flow which can be divided as follows.

1. Receiving
2. Put-away
3. Order-Picking
4. Checking,packing and shipping

In the end it would root down to sorting and picking of goods in a warehouse and distribution centers, which is labor intensive and inefficient to the present day, which further entails to high costs. Therefore it resulted in development of complex software packages like Warehouse management system that helped to manage storage locations, Stock keeping units (SKU) and the workforce so that it orchestrates the flow of material to be timely packed and shipped to the customer. As the scope of WMS is expanding

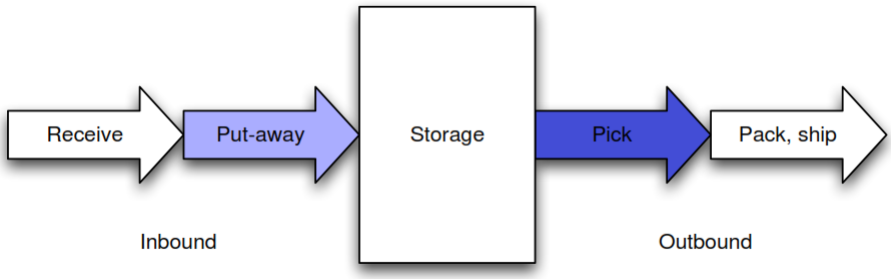


Figure 1.1: Warehouse operations [20]

each day, as it inducts new packages in its software we focus in this thesis on one of its important software components like **Material Flow Calculator** that coordinates the material flowing on conveyor systems from an input line(Einlagern) to be put away in an appropriate storage location(Auslagern). The allocation of material(put-away) usually amounts to approximately 15% of the warehouse operating expense.[20]

Warehouse management systems software component like MFR is one of the promising software solutions to impeccable demand of the logistic industry. It provides an automated solution to develop the productivity of manual or semi-automated warehouses by bridging the gap between the AIDC technologies like RFID or Barcode reader mounted in the conveyor line and the storage by automatically scanning the material flowing in the conveyor line, routing it optimally to a storage location and in parallel also synchronising with the database system by providing constant real time updates on material flow with the help of Soft PLC (Programmable logic controller) driven robust application[14]. Eventually it will speed up order turnaround time, improve inventory accuracy, provide instant order status information, manage warehouse space efficiently and enhance labour productivity.[15]

Unlike manual or semi-automatic warehouse, the automatic warehouse centers of the near future will run 24 hours per day, to better align with the distribution centers they will also operate around the clock. Automated warehouses and sorting centers will be just as effective on the last shift as they are on the first shift. Working in waves, the new supply chain will facilitate multiple shipments to end customers each day. By fully utilizing equipment across shifts, we will be capable to lower logistics costs and, by

processing multiple daily delivery waves, we will achieve faster service to end clients.[18]

An automatic small part warehouse (AKL) can store a wide mixture of articles in a space-saving and effective manner in containers or on trays, with efficient use of the room height. Smart software like MFR would monitor the automated flow of material and manages the real-time order in small warehouse plants. Transport orders that are predominantly received by a higher-level system like Enterprise resource planning (ERP) that define a particular shipment is to be transferred from a source to a terminus would be one of the extension packages that can be implemented in Material Flow calculator application but it wont be considered in our thesis.

1.2 Problem Statement

As mentioned earlier in the introduction in order to entail the high costs of warehouse by decreasing the turnaround time and to better coordinate material flow by optimising its routing and storage in a warehouse centre a smart software like MFR is one of the quintessential solutions. Some of the problems presently faced in the warehouse centre could be listed as follows.

1. **Data management:** It includes details of the inventory, order management, cyclic inventory and location management.

2. **Time management:** It encompasses the effective time required from origin to the destination.

3. **Flow of materials:** Lack of visual representation of active flow of container or a tray leads to inaccurate information in the database. However this thesis does not include developing HMI interface for this software package. [9][11]

4. **Demand variability:** Any changes in the stock levels results in the change of delivery schedule. Effective cyclic counting algorithm is one of the solutions to tackle the problem. However this algorithm is only to be implemented in ERP level. In MFR systems only the log or database will be updated so that ERP systems can use this data to effectively control the stock levels.

1.3 Objectives

The thesis aims to develop an algorithm for a material flow in an optimised fashion and later design an architecture model for it to be synchronised with the database system like SQL sever or maintain a local datalog file in .csv format. The software system like MFR designed can be implemented in a automated small warehouse enterprises with PLC and sensors using AIDC technologies like Bar code reader. The system as shown in figure 1.2 would work in a small warehouse with communication via TCP/IP. The system concept that is to be designed can be realized with Service Oriented Architecture, where in a decentralized modular approach will be adapted for seamless data exchange and integration between various module.[8] [13]

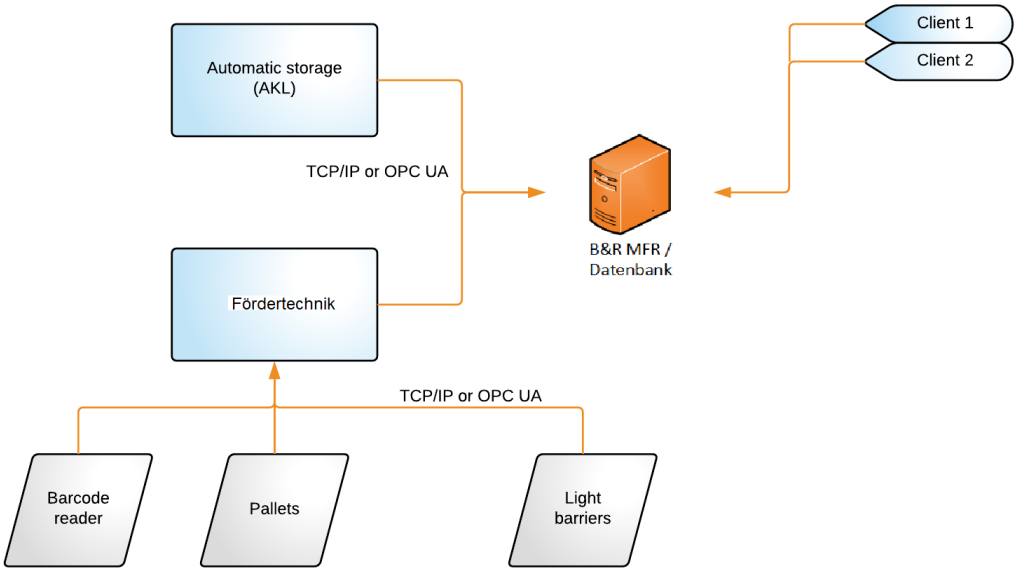


Figure 1.2: Initial proposed MFR system

1.4 Structure of thesis

The thesis is structured as follows. In the beginning the hardware and software systems that is required to develop the algorithm and the software

architecture for MFR systems are explained in detail. Later it discusses the modelling and implementation part of the problem.

Chapter 2 details out the WMS system in general and explains MFR model, its functionality and communication topology of WMS software used in warehouse industry.

Chapter 3 describes the PLC in general and explains the different styles of IEC 61131-3 programming language in detail. Furthermore it gives insight into the PLC and barcode reader considered in this project.

Chapter 4 portrays the steps involved in software development process cycle and gives a brief overview of B&R Automation Studio® environment provided by B&R Automation. At various stages the comparison between the standard and implemented method is analyzed in a detailed manner.

Chapter 5 provides a comprehensive outlook of the problem and a detailed analysis is done by modelling the problem as a scheduling job shop problem and it later discuses various algorithms ideas considered to overcome some of the problems seen like blocking, waiting time and so on.

Chapter 6 illustrates the flowchart and algorithm of the above scheduling problem defined by using the idea behind improved Dijkstra's [16] and Disjunctive Branch and Bound. Further it details out the architecture of synchronising model function blocks in PLC with SQL Server Database.

Chapter 7 does a comprehensive analysis on the results of the algorithm of the software model.

Chapter 8 summarises the work along with extensions and improvements that can be added to the software model of MFR.

Chapter 2

WMS: Material flow calculator(MFR)

2.1 Warehouse management system(WMS)

A warehouse management systems is the art of operating a warehouse and distribution system efficiently. It is a complex software package that manages inventory, storage locations, and the workforce, to ensure that customer orders are picked quickly, packed, and shipped in a timely manner. It provides additional functionality like optimising the routing of materials, maintaining data log of all materials and so on. In this chapter we will detail out the functionalities of various components of WMS systems and later cover different aspects about MFR considered for thesis study.[5]

Various components in WMS are as follows.

1. **Merchandise management systems**: They are systems used to maintain records of volume flows, which are majorly used in commerce. The major components consists of book keeping, accounting and inventory modules. Its also combined with Management Information systems, which consolidates and processes the information for management decision.[5]

2. **Production planning and control**: Manufacturing companies use these systems to optimize the orders from the customer end by utilizing

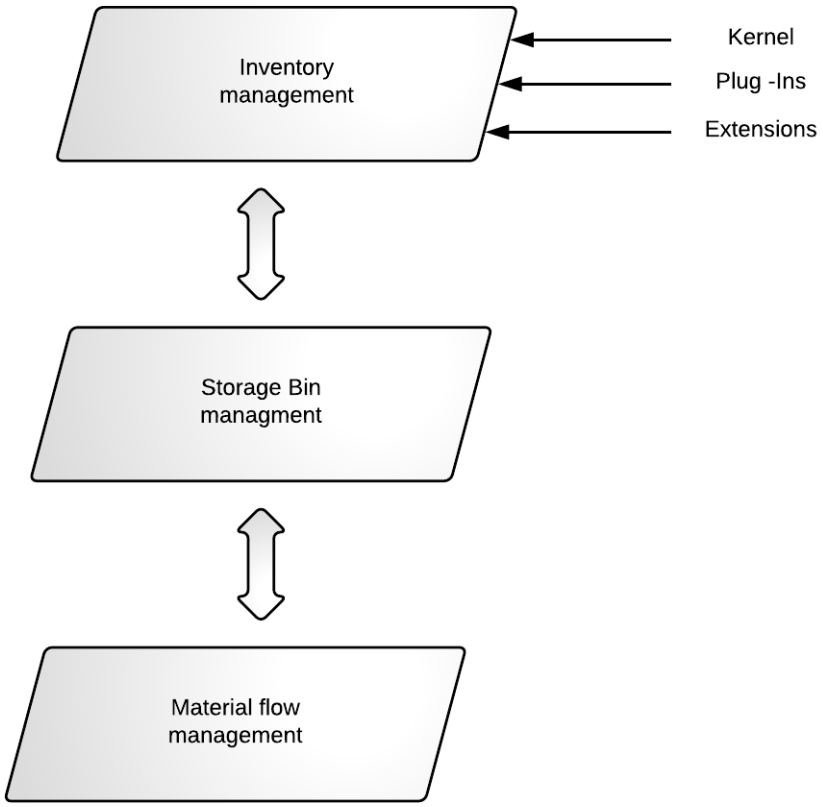


Figure 2.1: WMS system hierarchy

the present resources in order to have better throughput times while strictly adhering to delivery dates.[5]

3. **Enterprise resource planning**: In larger corporations with several manufacturing facilities it is used as intra-location production planning and control(PPC) systems.[5]

4. **Storage bin management**: It includes shelf racks containing objects of various dimensions, loading capacity and position. Status data are essential in retrieval strategies. Whenever the material is identified it has to be certain regarding the destination so that the location is not assigned twice. Ensuring this problem is resolved, one makes sure as-

signing available,blocked or reserved status for certain articles or orders. [5]

5. **Quality management systems**: It is a logical aspect which focuses on the registration of quantities and updates cyclically the inventory table in data base. It ensures the supply avoids excess quantities. It requires to keep in check the stored goods so that expiration date is not reached and it never affects the quality of other goods. Although it has similar functionality as , Quality management systems focuses majorly on Warehouse management whereas ERP has the details of customer data and prices. There are systems where in seamless exchange takes place between the two.

6. **Groupings systems**: It is usual to group unit loads in shelves to avoid labor extensive handling of elements. In order to enhance the efficiency and to optimise the space in shelves an active warehouse and distribution system is regularly reorganised to avoid wrong zone occupancy. The various methods are as follows

   (a) Rebooking - It reassigns the articles to suitable retrieval categories and storage zones.

   (b) Restorage - It restores the already existing units in times with low retrieval rates.

   (c) Densification - It densifies the partial mixed pallets, repacks the units and restores as per the requirements.

7. **Material flow management**: Most of the conveyors in warehouse are managed and controlled automatically by different systems from the manual management mode to fully automatic mode with **MFR** systems.

Kernel in the figure 2.1 refers to basic functionality independent of the type of warehouse. Plug-ins and extensions refer to additional functionalities one adds in the warehouse management systems.

2.2 Material flow calculator(MFR)

The routes of the pallets is determined via a prescribed schematic of warehouse system and it is controlled and optimised by various algorithm used

within MFR. The figure 2.2 shows the MFR software system considered for thesis, wherein it acts as an interface between procurement and retrieval systems, higher order ERP systems, databases and sensors. Few of it's functionalities envisaged can be listed as follows.

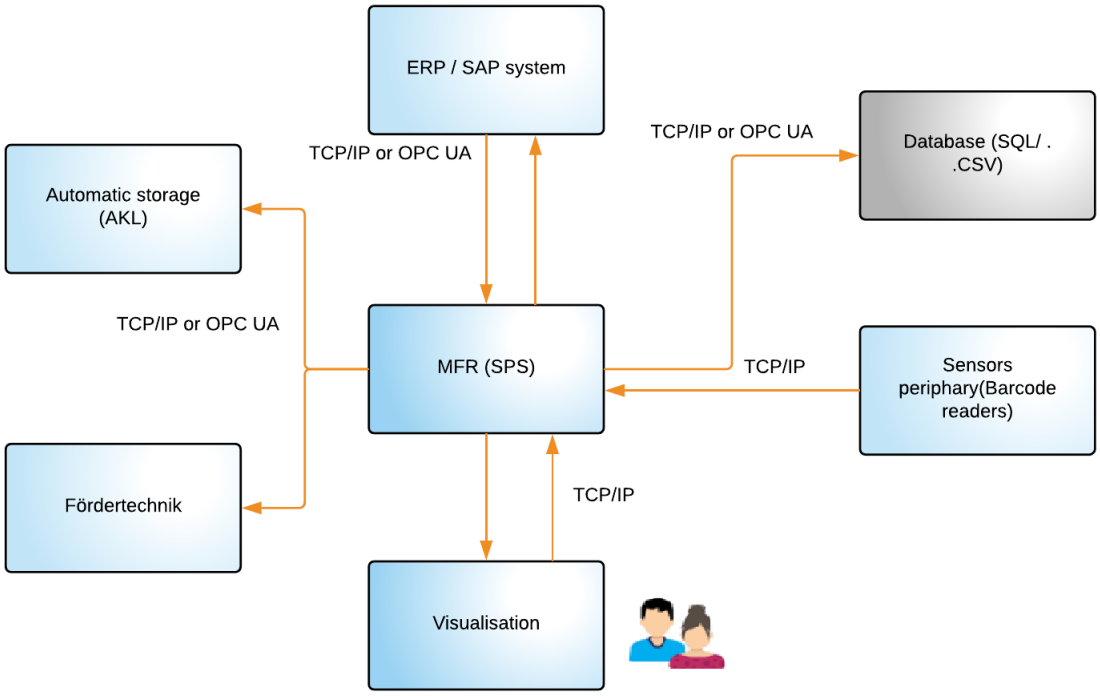


Figure 2.2: MFR System overview considered

1. To optimise the system performance by reducing empty trips, handling large volumes and higher system load.
2. It ensures flexibility for quick change in transport requirements.
3. It controls the system status (runtime of vehicles, operational costs per vehicle etc.)
4. **Dispatching**: A suitable conveyor is assigned to the current order according to various criteria and scheduling strategies. For example

while routing the next free conveyor or the nearest conveyor with the shortest connection trip would be allocated keeping in mind the system reaction time to static and dynamic environments.

5. **Scheduling sequence**: It finds an optimal sequence so that the system performance is improved. The underlying condition remains to schedule queued orders in a sequence optimally without errors.

6. **Data collection**: It involves collection of data, processing it and visualising a variety of different parameters for various purposes. For example recording the activity like documenting a shortage of stocks, picking errors, deviation of dispatch, generating cyclically bin status report, warehouse statistics (For example: Runtimes, Idle times, Maintenance etc).

7. **Online recording**: The database that is critical to generate the required data is recorded in the process and automatically synchronised with SQL server or individual .csv log file in local system. The log file is made out containing the events and occurrence of pallet flow at various stations. It has to include various fields like Order id, Barcode id, Status Id, Date and Time, Error Id, Starting point and Destination point. In addition to this, it is confirmed practice to have delivery notes and goods issue protocols for legal reasons.[5]

2.3 Communication hierarchy

As seen in the figure 2.3 the information for automatic control of an automatic warehouse is given by sensors and identification devices. The identification is via a barcode during loading aid. Later it is also applied during receipt process. Inside a warehouse each pallets are identified via scanning stations in Einlagern (Input) line and Auslagern(Output) line before the material is transferred to a traversing locomotive. Furthermore, the pallets enter the system via the input line submitted to a contour check.[5]

Scanners allow for the tracking and tracing of goods within the warehouse. The pick up stations mostly are equipped with switches that acknowledge an order and in parallel update the higher order system along with the corresponding terminals about the material.[5]

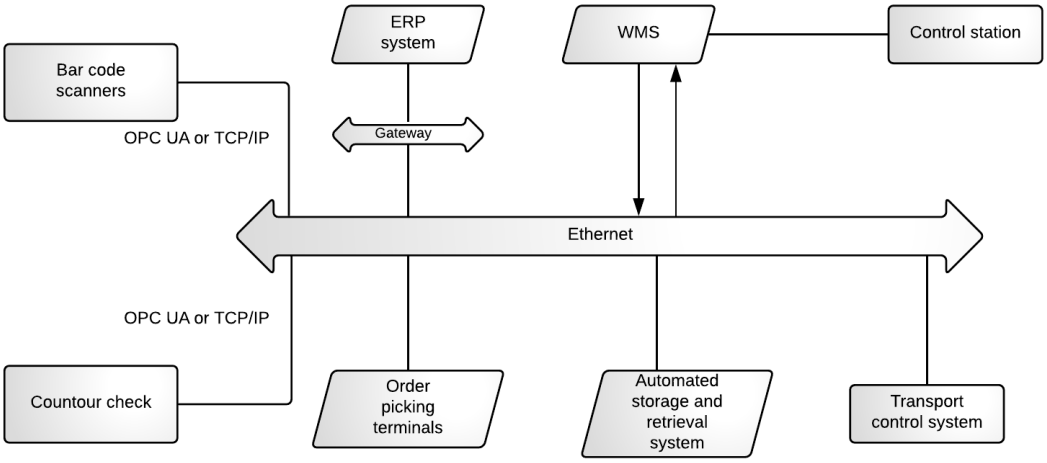


Figure 2.3: WMS communication hierarchy [5]

The barcode readers send all the information via field bus technology to super-ordinate PLC system and later communication with other PLC and other systems occurs via TCP/IP or OPC UA and Ethernet. It depends on the control philosophy defined by ware house management system or by MFR and further sub systems. Each PLC continuously receives signal from the super-ordinate system and acts accordingly. It is hierarchical and designed as client server architecture.

The warehouse is divided into various zones and each PLC is responsible for performing various functionalities defined between subordinate and super-ordinate systems.

2.4 Client Server architecture

Databases in super-ordinate systems of WMS are client server application where the actual database engine, the *Data based management systems* acts as a server for the clients. The database engine are accessed via TCP/IP, OPC UA or library modules provided by PLC library or via standard protocol such as Open database connectivity(ODBC). ODBC application links a network of servers with a database. ODBC drivers has to be installed on the client to access the application. In client server architecture of database

systems each data is exchanged between **Server- Database - Client**.

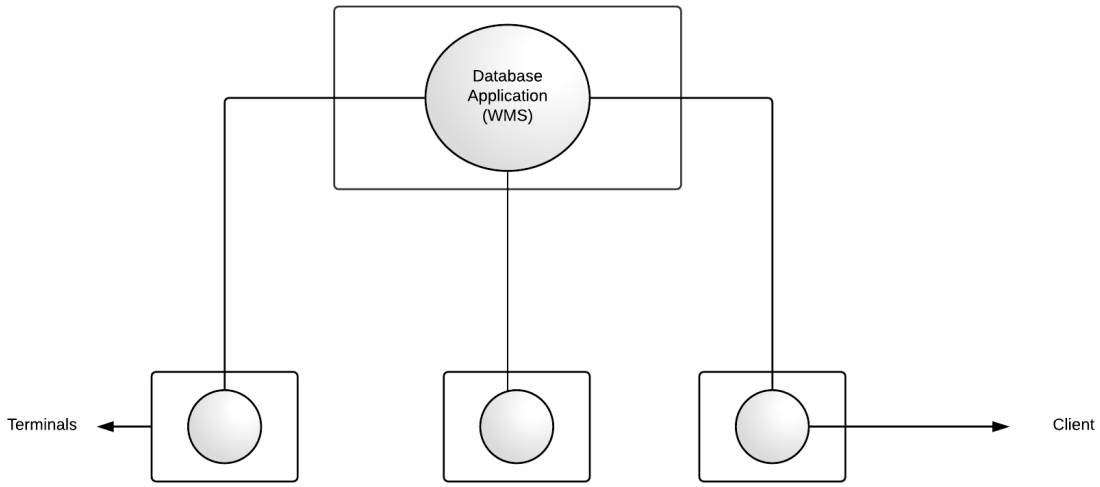


Figure 2.4: Client-Server architecture [5]

Chapter 3

PLC & Sensor

3.1 Background

PLC is an acronym for Programmable Logic Controller. PLCs are industrial digital computers which were developed in the automotive industry of the USA in the late 1960s and 1970s. They replaced hard-wired relays and timers which was a great achievement because PLCs are more robust and reliable, even in harsh real time environments. Due to available visual programming techniques, they can be used for various control tasks, as motion control, and networking. As they are modular in design PLCs are scalable, flexible and can be adjusted for a specific control application. They range from only a few connected devices with maybe ten I/Os and an included processor to large control systems with thousands of I/Os where several PLCs are networked together to control a whole production site with various robots and other automation equipment. Depending on the envisaged application a unique PLC application can be built from available components to fulfill the desired control task. [1]. A general overview of PLC systems is shown in the figure 3.1.

3.2 Operation principle

To illustrate the basic operating mode of a PLC a simple setup can be assumed consisting of a CPU (Central Processing Unit) and an I/O (Input/Output) interface as shown below. The CPU includes a processor for signal processing, a memory for storing data and a power supply to provide

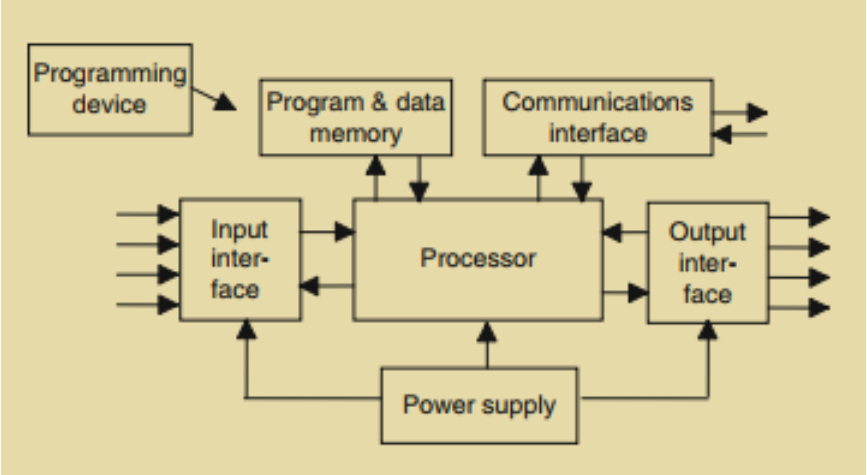


Figure 3.1: PLC system [7]

the necessary voltage for operation. In one cycle the CPU first reads data from the I/Os, in a second step executes actions depending on those I/O signals and finally writes the updated values back via the output interface. [1]

3.2.1 Modules

The available components are CPU, I/O, communication modules and condition monitoring modules. Furthermore, corresponding accessories as cables or memory cards for the technical setup are also provided.

CPU's

A CPU module fulfills the same function as a usual CPU for e.g. in a personal computer. It is mainly responsible for data and signal processing. It usually has a memory in form of an Compact flash card and several interfaces for networking (Ethernet), serial connections (COM1, COM2) and a FieldBusPlug. The CPU module considered for this thesis is PC 2100 from B&R automation.

I/O modules

Input/Output modules provide digital and analogue inputs and outputs. The I/O terminal unit allows to connect field devices as sensors, actuators

etc. The I/O modules differ in kinds and number of channels, signal voltage and field-bus connections. One can establish a local connection via the CPUs I/O bus or a peripheral connection via a field-bus if more remote devices are needed. The input modules considered for the thesis work is Barcode reader. A typical PLC with as set of IO's is shown in 3.2

Figure 3.2: PLC system (B&R X20) with IO modules from B&R automation

Communication modules and Interface modules

Communication modules offer a wide range of standards, interfaces and protocols. A list of available types is shown below.

- Ethernet
- Powerlink
- Profibus
- TCP/IP
- EtherCAT
- Profinet
- Modbus
- OPC UA

Communication interface modules are basically bus modules which allows to interconnect the other modules presented in the previous paragraphs with each other. They provide connectivity options for remote I/O modules and

are designed for communication with the distributed periphery of a plant.

Condition monitoring modules

Condition monitoring modules are used for the observation of signals and processes while the plant is operated. It enables predictive maintenance and ensures protection with regard to voltage, current and vibration and thus helps to prevent hazards and breakdowns. Condition monitoring modules usually consist of analogue inputs, a memory and a speed encoder. Such a unit automatically sends WAV files containing relevant process data to the CPU in certain time intervals. This way the status of the plant can be monitored in real-time and additionally a complete record of past processes is available afterwards for further analysis.

Accessories

To build a PLC from the presented modules some accessories are needed, others are optional. B&R, for example offers a Compact flash cards for creating memory, plugs with different poles for power supply, communication devices and I/O modules and programming cables. Furthermore, lithium batteries 4 pieces, 3 V / 950 mAh button cell are provided. They can be used to extend the PLC's memory so that process data and configurations will not be lost after the plant was shut down.

3.3 IEC 61131-3 language

The IEC 61131 defined the standards for PLC programming languages: ladder diagrams (LAD), instruction list (IL), sequential function charts (SFC), structured text (ST) and function block diagrams(FBD).

Various styles of the above mentioned programming language will be discussed below.

3.3.1 Ladder diagrams (LAD)

The simplest of all programming method in PLCs is writing logic in ladder diagrams. Writing a program consists of drawing a switching circuit. The ladder diagram has two vertical lines representing the power rails. Circuits are connected as horizontal lines, that is, the rungs of the ladder, between

these two verticals.[7]. Conventions to be followed while programming are as follows.

1. The vertical lines of the diagram depicts the power rails between which circuits are connected. The power flow is taken to be from the left-hand vertical across a rung.

2. Each rung on the ladder defines one operation in the control process.

3. Each rung must start with an input or inputs and must end with at least one output. The term input is a control action, such as closing the contacts of a switch. The term output is a device connected to the output of a PLC, such as a motor or so on.

Below symbols are used in IEC 61131-3 for input and output devices show in the figure 3.3

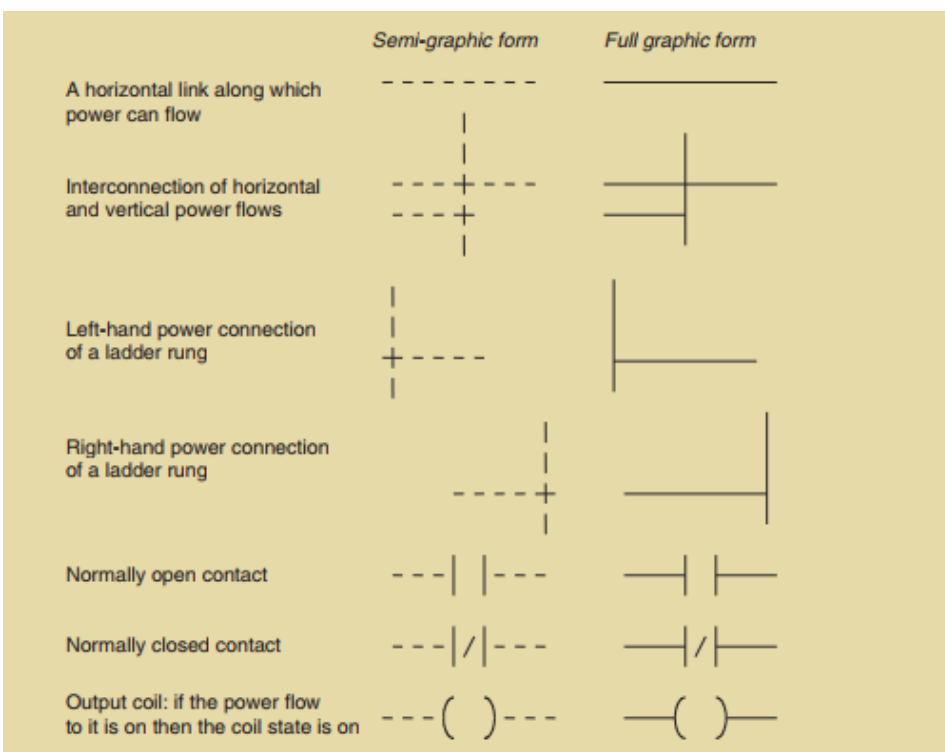


Figure 3.3: Basic symbols [7]

A typical ladder diagram with an input and an output shown in figure below

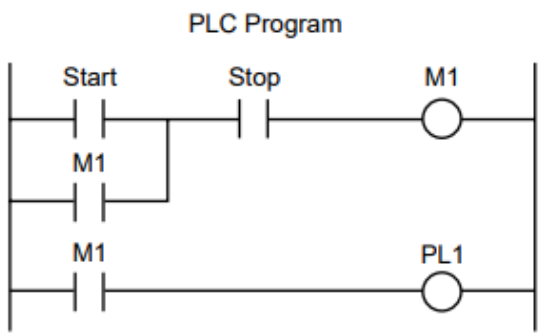


Figure 3.4: Ladder diagram [1]

3.3.2 Function blocks(FBD)

The function block diagram (FBD) in PLC programs are described in terms of graphical blocks. It is a graphical language for depicting signal and data flows through blocks, which are reusable software elements. A function block is a program instruction unit that when executed, yields one or more output values. Thus a block is represented in the manner shown in with the function name written in the box.[7]. Most of the function blocks has an inbuilt function and can be loaded or called in the program from the respective libraries of the PLC software in our case being B&R Automation Studio 4.2.

Logic gates are an international standard form (IEEE) that uses a rectangle with the logic function written inside it. For example CTU function block shown in the figure 3.5 does the following, it gives an output at Q when the input of pulses at the CU count has reached the set value, which is set by PV. Each time there is a pulse input, the output CV is incremented by 1. The input R is to reset the counter. The input and output labels are used to indicate the type of signal involved: BOOL for Boolean and INT for integer.

The other example shows the mathematical function blocks add and divison with inputs WEIGHT1 and WEIGHT2 along with output being AV_WEIGHT. The ladder diagram and equivalent function block is as shown below.[1]

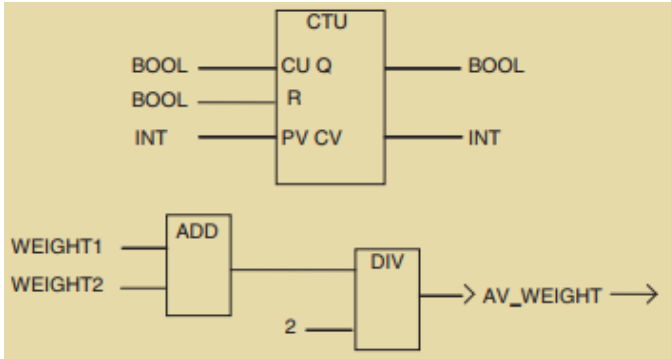


Figure 3.5: Function Block [7]

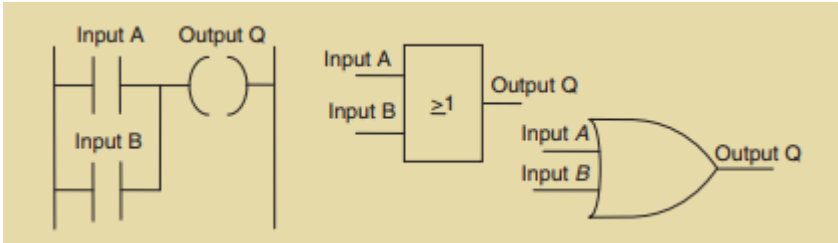


Figure 3.6: Ladder diagram and equivalent Function Block [7]

3.3.3 Instruction List(IL)

The Instructions list programming method is similar to ladder diagram. But the difference lies in the way it is executed, as this one uses the text format, each rung in ladder corresponds to a rung in Instructions list. For example we consider the following [7] .

LD A

ST B

It implicates that the operand A is loaded in the memory. It is equivalent to rung operation with open contact for input A and and the output is passed on to B. The various mnemonic standard codes used by IEC 61131-3 are as shown in table 3.1

IEC 61131-3	Operation
LD	Load the operand in result register
LDN	Load negative operand into result register
AND	Boolean AND
ANDN	Boolean AND with negative operand
OR	Boolean OR
ORN	Boolean OR with negative operand
ST	Stores the result register into operand

Table 3.1: Instruction List mnemonics

3.3.4 Sequential Flow chart(SFC)

SFC method of executing a program is used when one wants to describe a sequence of states/functions in the form of a flow chart. Following features of SFC are listed below.

1. The operations are described with individual sequentially connected states or steps that are represented by rectangular boxes, each representing a particular state of the system being controlled.

2. Each connecting line between states has a horizontal bar representing the transition condition that has to be realized before the system can move from one state to the next. Two steps can never be directly connected, they must always be separated by a transition. Two transitions can never directly follow from one to another, they must always be separated by a step.[7]

3. When the transfer conditions are met the next step in SFC are executed in a program.

4. The process thus continues from one step to the next until the complete machine cycle is completed.

5. Outputs/actions at any state are represented by horizontally linked boxes and occur when that state has been realized.

   A typical SFC example is shown in fig 3.7 with a main program and a subroutine program (Batch_Mix and Batch_Pump) being called when the start is activated. [7]

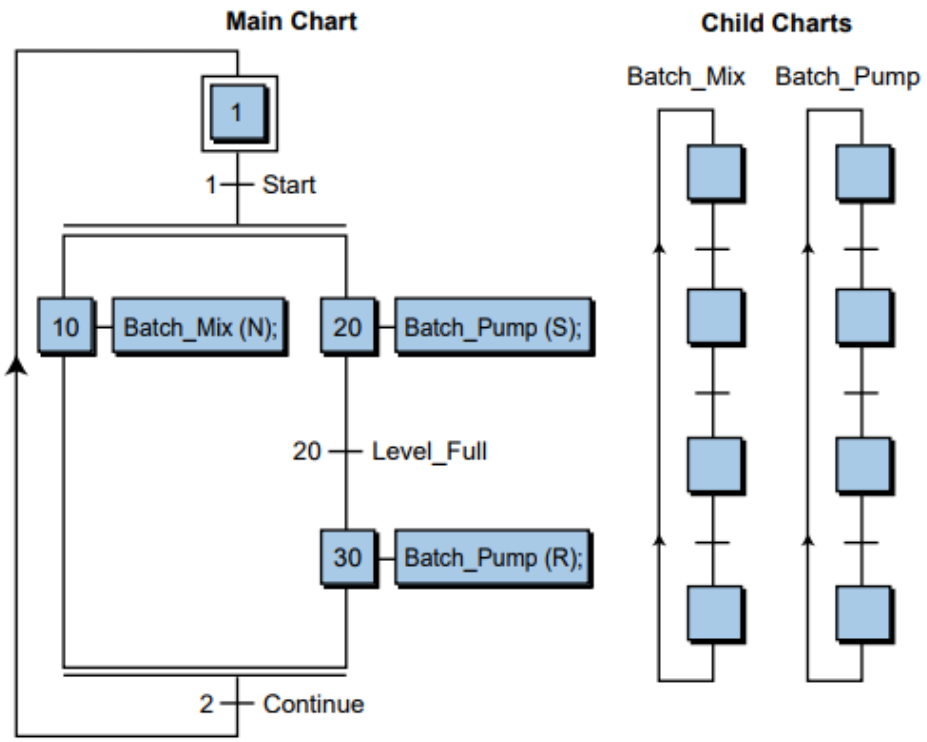


Figure 3.7: Sequential flow chart [1]

To jump to each states, there is an action or actions that have to be performed. Such actions, such as in the above example, are depicted as rectangular boxes attached to the state. The behavior of the action can be executed also using a ladder diagram, a function block diagram, an instruction list, or structured text.

3.3.5 Structured Text(ST)

It is a programming language that is similar to C or Pascal. It is written with series of statements separated by semicolons. The subroutines to change variable, calling a well defined function from the library to cut short long programs is what makes it unique, flexible and easy to use. Most of the tasks in this thesis project has been done using this style. Additionally it also offers the flexibility of using loops and conditional statements.[7]

```
FUNCTION_BLOCK TEST_VOLTAGE

VAR_INPUT
VOLTS1, VOLTS2, VOLTS3 ;
END_VAR

VAR_OUTPUT
OVERVOLTS : BOOL;
END_VAR

IF VOLTS1 > 12 THEN
    OVERVOLTS :=TRUE; RETURN;
END_IF;
IF VOLTS2 > 12 THEN
    OVERVOLTS :=TRUE; RETURN;
END_IF;
IF VOLTS3 > 12 THEN
    OVERVOLTS := TRUE;
END_IF;

END_FUNCTION_BLOCK;
```

The simple example above is a function block which can be called in many other program to check the voltage level. It just tests the VOLTS1,VOLTS2,VOLTS3 and indicate the the output OVERVOLTS to be true , the return statement terminates the execution of block.[7]

3.4 PC 2100

As industrial PC is a next generation of PLC technology suitable for scalable, rugged, powerful and remote applications like ours in Material flow calculator project which can be run remotely on modular and decentralized manner. On a hardware front it runs on Intel Atom E3800 processor capable enough for powerful graphics in SCADA or HMI system for 24 inch TFT monitor application. It comes with inbuilt Windows 7 operating system. The communication modules it supports are POWERLINK, CAN and Ethernet in fieldbus technology. It also has data storage options via CompactFlash cards

or via USB slots. It is used as a soft PLC in our project meaning the code written in Automation Studio 4.2 will be downloaded in this PC2100 and would be run throughout the project remotely.

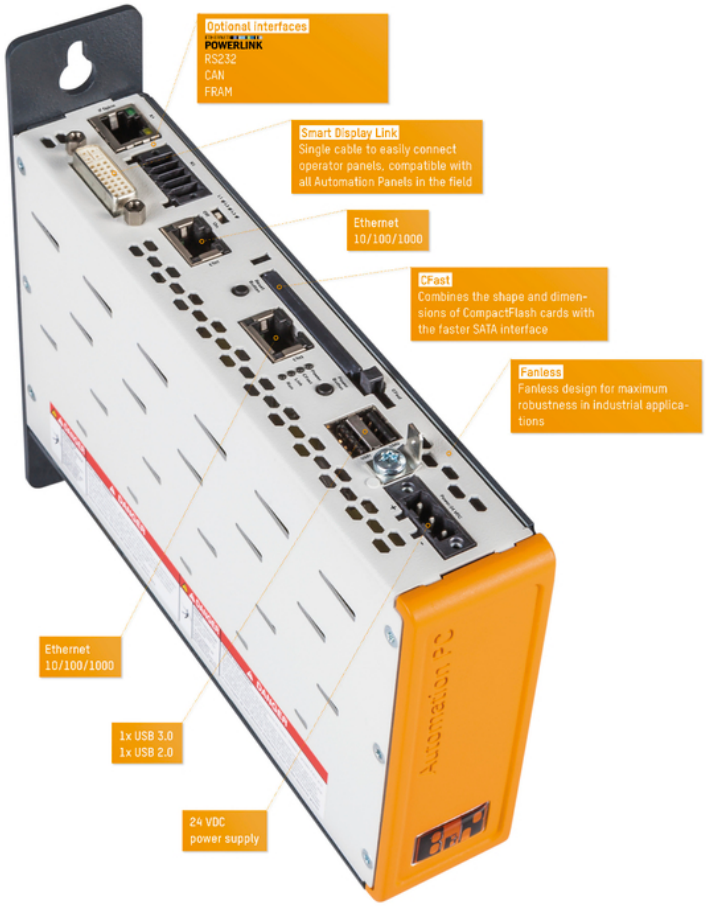


Figure 3.8: PC 2100 from B&R automation

3.5 Sensor : Barcode reader

The barcode reader is a electronic device that reads out the material barcodes and feeds the details to the PLC. It bridges the gap between the source and destination by providing all the necessary routing information for the PLC system. It is crucial in MFR to have all the details in order to execute its

routing algorithm, where in it can later decide on the optimised path to be taken.
The barcode reader considered for MFR project is CLV620-3120 , which is available for mid range application on conveyor line. It uses visible red light as a source. It has a reading distance upto 370 mm. The communication interfaces supported are via serial(RS 232 or RS 485) along with Ethernet, CAN bus, PROFIBUS DP, EtherCAT, DeviceNet. Moreover it provides data storage of 512 MB.

Figure 3.9: Barcode reader CLV620-3120 from SICK

Figure 3.10: Fieldbusmodule CDF 600-2201 from SICK

The field-bus module considered in the above project is CDF 600-2201 which provides connectivity to the above bar-code reader via Profinet or CAN bus.

Chapter 4

PLC Software development topology

Software engineering can be defined as a technical discipline that circumscribes the complete aspects of software development. The characteristic of a good software are it should provide the end user with the rated functionality and operation. Software development involves primarily consolidating the necessary requirements from the client, followed by sophisticated analysis, elucidation of the grounds where excess information is required, conceptual design of the system with detailed engineering, drafting the logic code blocks, integration of the hardware system with software codes in the background, testing the system and deployment are in the following detailed steps taken after. There are several methodologies like waterfall, agile, spiral, incremental, V-shaped architecture etc available. The necessities and budget facets are instrumental in selecting the correct type of methodology.

On a general outlook the software model developed by Prolog Automation matches with Waterfall and V shaped architecture, but its more dominant on V-shape based model shown in the figure 4.1. As it was confirmed by the software team in Prolog Automation.

4.1 Design Sequence Approach

The design sequence approach has a direct correspondence and mainly concentrates on the customer requirements specifications.

1. **Customer Requirement Specifications(CRS)** - Get the input re-

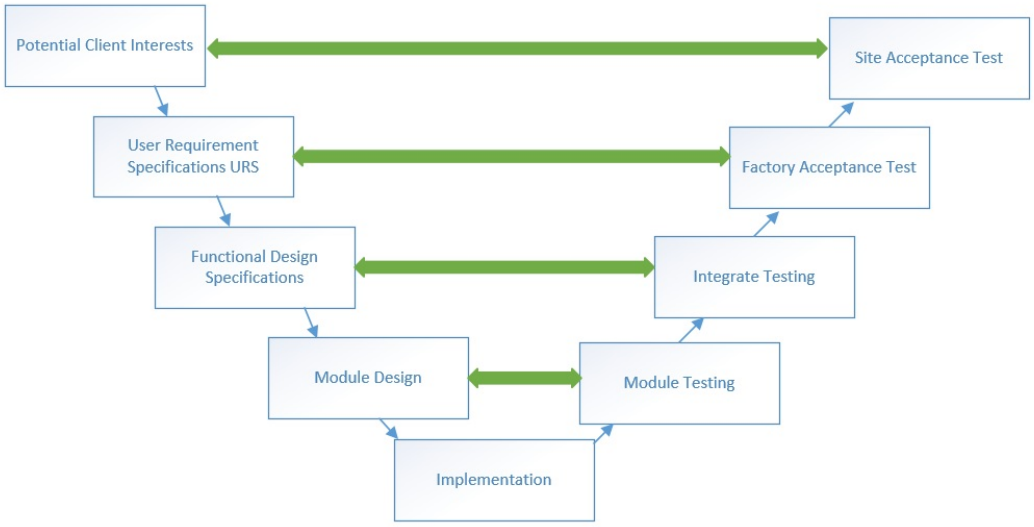


Figure 4.1: Software development cycle

quirements from the customer in signed document format.

2. **Functional Design Specifications(FDS)** - Draft a functional specification based on the requirements received. Essentially, this should say how the to-be software system would execute the requirements by the client.

3. **Module Design** - The modular design approach will include the units and sub-units independent of each other. In our case being the Barcode reader, MFR, data logging. Additionally, each of the units could be debugged and broken into a small part.[17]

   From transforming the process from CRS to FDS there is a good chance that *Translational Errors* could make way in the process . This arises, as there are possibilities that the domain language of the customer is different from the one of the developer.

$$Language_{Customer} \rightarrow Language_{Developer}$$

Secondly care was taken that information would not be missed or wrongly interpreted from the customer. It was ensured with substantial email exchange and discussion between the software team and client end. Further-

more, processing the customer requirement specification was divided in two parts

1. **Static Information** - In static information part, a structure was prepared as per the understanding by the software development team. It was later consolidated for the development and procurement stage. with the required hardware and software tools in a pdf file.

2. **Dynamic Information** - In dynamic part, the reason and the motivation behind the customers requirement was briefly questioned. " *Why does the customer need the particular module in this software and how will it benefit the customer* ". This is of significant interest to the developer as he would be able to visualize just like the customer. This critical analysis is better understood by describing the workflows and use cases.

In short typical software architecture can be summarised as shown in the figure [17].

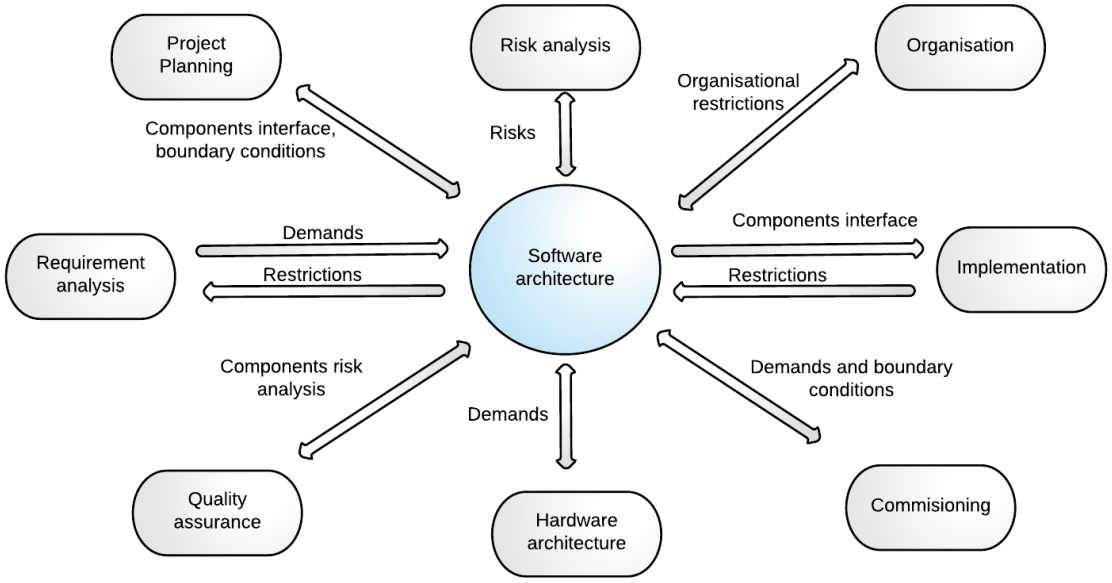


Figure 4.2: Software architecture

4.2 Alternative software models

The alternative approaches for developing a software model is briefly outline below and comparisons to our approached is covered. [2]

Waterfall is a classic conventional model also known as "*linear sequential*" life cycle model. This is followed by many corporations as its straight forward and simple to follow. This is getting out-dated with the advancement of other methodologies as it heavily depends on a fashion where each method is interdependent on each other, where one finishes, the execution goes to the next. As a result, the output is generated later in the life cycle.

Iterative frequently acknowledged as "multi waterfall" model is a repetition technique with each variant of software released before the completion date. The outcome of this process is the availability of software after each cycle. Furthermore, it has tested and debugging iterations reduced. However, the major drawbacks that can be mentioned are the process in each phase is rigid. The user requirements enter in the late portion of the process, the details that are missed in starting cycle has certain influence in changing the system architecture. This results in redesigning the system in further phases.

Spiral is a slight upgradation of the iterative method. This surpasses the observed disadvantage in the former method by creating a four phase structure over and over in a spiral network, providing several rounds of optimisation. More weightage is given to risk analysis. It has a possibility of falling into never ending loop. It is implemented in large and critical process.

Big Bang model, a risky approach is started with spending long time towards planning and development. It involves huge risk as a misplaced information could widen the problem.

Agile model, a combination of iterative an incremental process, divides the process into cycles. This produces a quick working product with ongoing releases with incremental changes from earlier version. It is a highly realistic development model. Interacting with customer, developer and tester is highly helpful. The only way it can end as a negative is when the customer has partial knowledge about the requirement. It is considered for this project as it has enormous advantage of time effectiveness ans suits with adaptive

nature of customer.

Scrum, one of the agile methodology and the recently developed architecture. In the scrum, a major leap compared to the rest discussed above is that it does not need complete test plans at the starting stage like V Shaped model. As many users will not be familiar to draft a test plan, where in he proposes the testing procedures for the final model. The test plan is seen only at the end. The negative aspect of the scrum is it fails when project lacks a clear definition and makes it difficult for scrum master to organise such projects. Though this methodology is a good choice for work due its string advantages, it is not suitable for small scale projects. It is ideal for product development projects

4.3 Automation Studio ® from B&R Automation

It is a PLC programming integrated software development environment that supports all styles of programming language as mentioned in Chapter 2. Moreover interfacing controller, drives and visualisation can be done in a single environment. It reduces integration time and project costs.

4.3.1 Features of Automation Studio ®

1. **Project management:** It provides system oriented view and helps to divide a program in functional packages that can be assigned and configured for individual hardware configurations.

2. **Programming languages:** It provides support for IEC 61131-3, ANSI C and C++ language along with the libraries defined by IEC and the extended B&R libraries standard. For example the extended libraries like AsDb provides support to integrate Microsoft SQL Express®, AsTime library is used to get the current date and time of the material flow in conveyor line and furthermore function blocks are implemented in each stage of this project.

3. **Integrated visualization:** The visualisation is integrated in Automation Studio which is very effective to create displays as well as control

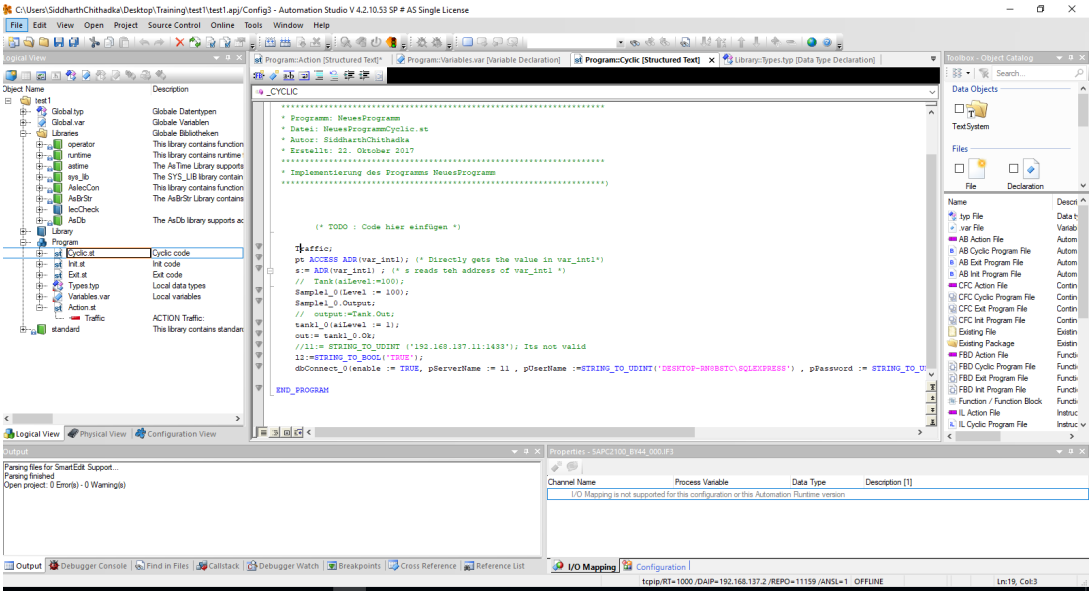


Figure 4.3: Sample program in Automation Studio 4.2 ®

remote displays with touch screen. The integration adds a flexibility of visualising material flow. This is not part of thesis. But however it is considered in later stages of the project work. Remote services such as VNC and maintenance would be easier with HMI.

4. **Diagnostics:** Automation Studio provides vivid tools to diagnose the program with usage of Breakpoints, display of graphs for various variables and System diagnostic manager that reads information about the target using the standard web access.

5. **Remote maintenance:** In a project like Warehouse management many of the system like motor control, material flow calculator, Database and maintenance functionality are diagnosed and operated remotely.

6. **Communication and field bus system:** In Automation Studio, a field bus device is added to the corresponding field-bus interface and the matching I/O device are configured. Interface from other providers can be configured by using import functions(adding GSD, ESD files). In our case from SICK barcode reader.

   As an integral part of real-time operating system software kernel that

runs application on a target system. It should always provide the functionality to scale the project applications in the future by adding individual modules. Hence the programming philosophy considered here is based on modular approach.

Chapter 5

Modelling

In the previous chapters a brief insight into hardware and software components of **MFR** was given and the kind of software development topology implemented was defined. In this chapter we are going to model our problem as a scheduling problem and then define the various algorithms approach considered in solving this scheduling problem.

5.1 Scheduling components

Modelling a problem in scheduling involves defining three important steps as defined below. [19]

$$\alpha|\boldsymbol{\beta}|\Gamma$$

1. $\boldsymbol{\alpha}$ - It indicates a machine environment of the problem. For example it can be as simple as single machine to flow shops, job shops and open shops

2. $\boldsymbol{\beta}$ - It indicates various constraints seen along with a particular processing characteristics of the machine.

3. $\boldsymbol{\Gamma}$ - It signifies the objective function that has to be minimised like makespan, tardiness, late jobs and so on.

5.2 Scheduling complexity

Furthermore the scheduling complexity of the algorithms involved in solving various scheduling problems are classified as follows.[19]

1. **Easy** - There is an algorithm that optimally solves the problem with polynomial time complexity as follows for some fixed **k**.

$$O(nlog(maxp_j)^k)$$

2. **NP hard in ordinary sense** - The problem cannot be optimally solved by an algorithm with polynomial time complexity but with an algorithm of time complexity.

$$O((n(maxp_j))^k)$$

3. **NP hard in strong sense** - The problem cannot be optimally solved by an algorithm with pseudo-polynomial time complexity.

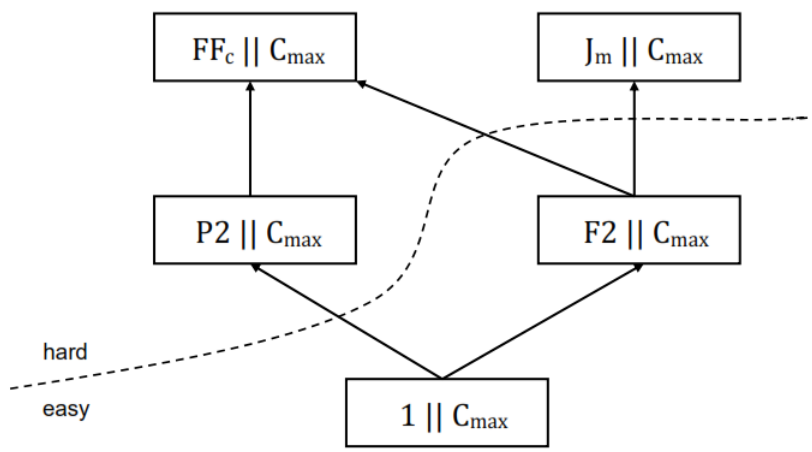


Figure 5.1: Complexity of makespan problems [19]

5.3 Problem modelling

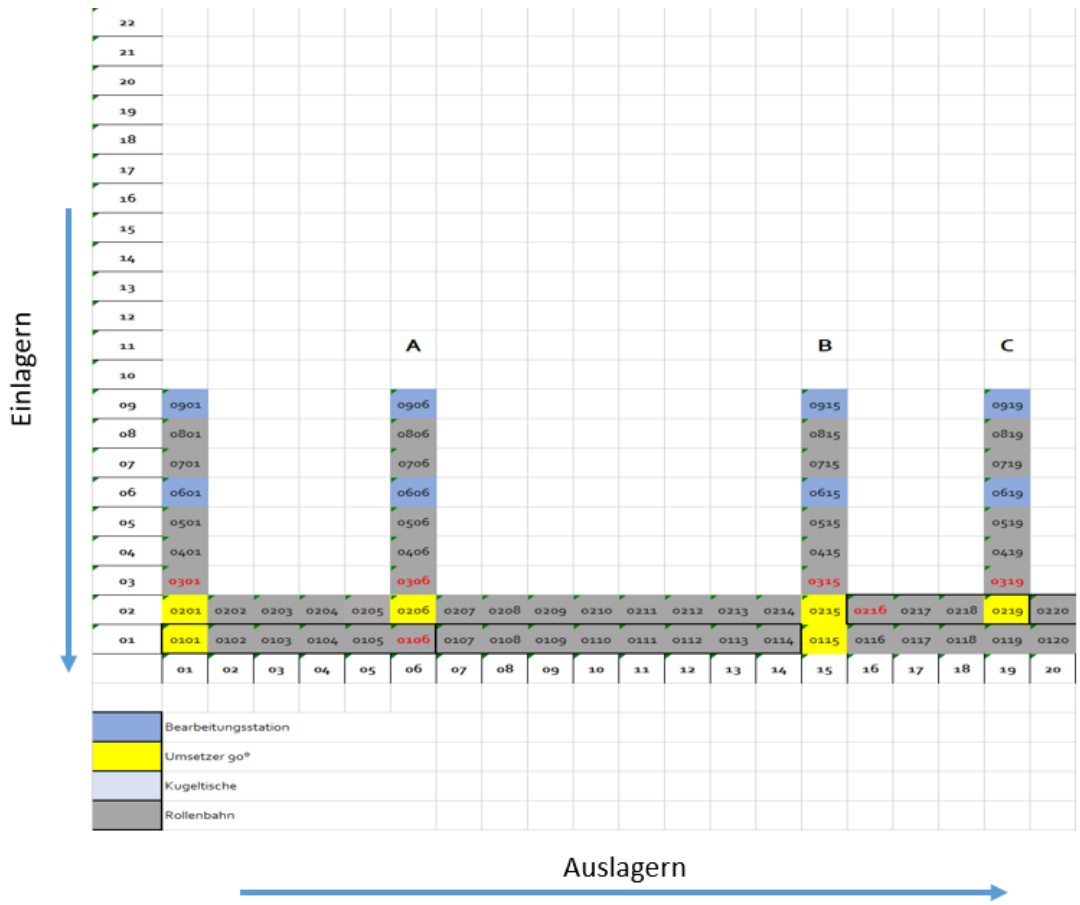


Figure 5.2: Schematic layout of one of the zones of warehouse

The figure 5.2 describes one of the zone of the warehouse system in our thesis study. The address of each element is described via x and y coordinate system. The front two digits represent the position on the y-axis (lines) and the two rear represent the position on the x-axis (columns). This zone is considered as a job shop with four machines and an algorithm is chalked out for combination of job shop and shortest path problem.

As explained earlier the above problem can be modelled to be a online Job shop scheduling problem as as the route from Einlagern to Auslagern is

fixed but its not mandatory that all jobs follow the same route due to various factors like traffic congestion, static routing and so on. In Jm, m stands for various machines in the shop environment. To be precise in our project we consider the respective motors and the readers in Einlagern and Auslagern line as the machines of the job shops.

The constraints to be considered are **precedence constraint** as the job that is already present in Einlagern(incoming) path would have to be cleared first. Assuming all the scan rates to be of same period on all the conveyor belts its also a **proportionate Job shop environment**, where in i stands for different machines and j for jobs and **blocking** as the job in Auslagern(output) line could block the successive job.

The objective function to be considered would have to be make-span as it has to be minimised so that all the material reach the destination as early as possible. Taking all the above factors in consideration the problem can be modelled as follows.

$$\mathbf{J_m|prec, block, (p_{ij} = p_j)|C_{max}} \tag{5.1}$$

As we consider C_{max} as objective function to be minimised there are some parameters to be accounted like idle time in a machine i between processing of jobs in position **k and k+1**, waiting time W_{ik} which is the amount of time job waits in position k between machine i and i+1 in Einlagern(Input) and Auslagern(Output) line. Δ_{ik} which denotes the difference between start time of job in position k+1 and completion time of job in position k on machine i leading us to the equation below. [19]

$$\Delta_{ik} = I_{ik} + p_{i(k+1)} + W_{i,k+1} = W_{ik} + p_{i+1(k)} + I_{i+1,k} \tag{5.2}$$

5.4 Ideal case

From the C_{max} scheduling complexity hierarchy it can be concluded that the problem with job shop and make-span is of NP hard stature. Moreover its online too as there are material flowing are mixed with the materials with the fixed static destination address recognised by bar-code reader and at times not recognized by bar-code reader. In short the heuristics developed must try to reduce the waiting time of the job and the idle time of machine. As

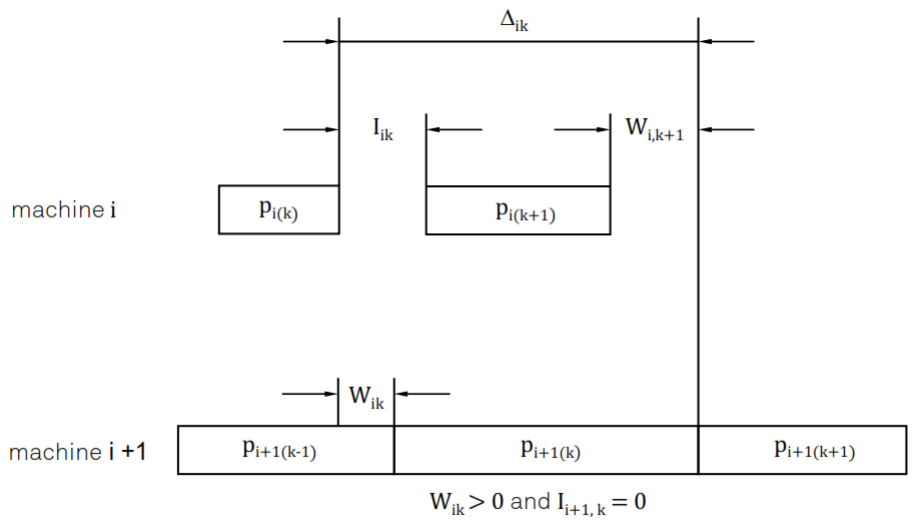


Figure 5.3: General description of Δ_{ik} [19]

considered in our previous case the machine are primarily the motors and barcode reader in each conveyor line.

The scheme proposed combines the idea from Improved Dijkstra's algorithm [16], Disjunctive Branch and Bound scheme [19] and the principle of Pseudo Adaptive XY routing [3] to avoid blocking problem by assigning priorities to Auslagern path. Further a brief outlook on each algorithm is detailed out.

5.5 Improved Dijkstra's Algorithm

As a traditional Dijkstra's algorithm divides the map into nodes in the graph and assigns weights for each node W[N]. The best route from an origin to destination point, W(i,j) is ascertained by comparison with its adjacent routes. It can be best described by the figure 5.4.[16]

However some of its shortcomings are listed as follows.

1. The algorithm to solve the shortest problem is easy to implement but comparing it each time to adjacent matrix around network nodes leads to data redundancy and waste of time.

2. It is a greedy strategy wherein it can find a local optimum at each stage and not find the global optimal path. Moreover in real time the traffic impedance has to be considered too. On encountering large scale network it becomes not suitable.

Hence **improved algorithm** which was developed [16] uses the same strategy but it doesn't go on comparing with the adjacent table rather the efficiency is raised by feeding the whole map details in advance. As in our case we have complete details of the distance of every point thereby we can reduce the cycle time of comparison with each matrix of nodes and reduce inner loop cycle time by saving distance details in a form of a heap of distances from min to max.

A **heap of distances** implemented can be seen as a priority queue which is a main key in finding minimum cost and keeping loop count low each time in comparison to traditional traverse method which is highly inefficient when it encounters large data. Effectively all the destination modes are arranged according to cost being from **low to high**.

Binary heap is seen as a pile of nodes with minimum pile sitting at the top-of pile. At first it finds the smallest cost node by traversing through all set of nodes in the map fed and adjusts the heaps accordingly. As and when the map is updated or a new node is added it adjusts binary heap. As the complexity of Dijkstra's algorithm is $O(V^2)$ and the improved Dijkstra's with the heap, each time adding node with min cost complexity will be $O(logn)$. As operation decreases the cost m times it is $O(mlogn)$ and operation of deleting and updating the minimum takes n times it finally it complexity of this algorithm is suggested as $O(m + n(logn))$.

Traffic impedance that is considered is the resistance value of material flow. Since time and distance are proportional to the relationship when material are moving on a conveyor line without traffic congestion and the material impeded on real time traffic has to be considered. The relationship function encompassing time, unit length and traffic flow can be formulated as follows where t is time required to traverse, v indicating the material speed, q is the traffic congestion on the path.[16]

$$t = (l/v) * f(q) \tag{5.3}$$

5.6 Disjunctive Branch and Bound

As it is seen Jm||Cmax to be a NP hard problem. In Step 1 of our combinatorial optimisation the map is sorted as heap of min to max cost list, in the next step the idea of Disjunctive Branch and Bound is used to solve the WMS scheduling problem. Few of its important steps can be summarised as follows.

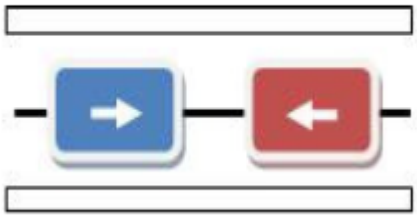
1. Set of nodes N where in each node equals to an operation (i,j) of job j on machine i.
2. Set of **conjunctive edges A** are the one (i,j) to (k,j)denotes the job that will be processed on machine k after it is processed on machine i.
3. Set of **disjunctive edges B** is a disjunctive edge from any operation (i,j) to any operation (i,h) that is between any two operators that are executed on the same machine.

There is a dummy source node which is connected to first operation of each job and a dummy sink node that is the target of last operation. In short the problem of minimizing make-span is about finding a selection of disjunctive arcs that minimizes the critical path. [19]

In our case we consider in active schedule when the job which is not recognised in Einlagern path by bar-code reader and is waiting in Einlagern path to be dynamically assigned a Virtual destination Meldepunkt. We consider its release time r_{ij} in Einlagern path and moreover we take it into account the Auslagern path that are occupied and recently used. In short we calculate the $t(\omega)$, which can be formulated as follows where i stands for machines, j for jobs, r for release date, q for time required to process a job j from Einlagern to Auslagern in particular machine i. Next step we branch and bound through the heap of min cost of Auslagern destination address distance.

$$t(\omega)_{(i,j\,)\in\omega} = min(r_{ij} + q_{ij}) \tag{5.4}$$

5.7 Deadlock



Since there is a great potential in a running warehouse that the materials run into a deadlock meaning a two or more material flowing from Einlagern wanting to go to same Auslagern line. Major conditions for deadlock are listed as follows.

1. Mutual Exclusion - Resource is unshareable.
2. Hold and wait - Process are clinged to holding a resource while its waiting for other resource. Thereby other process end up waiting and are in deadlock as they wish to use same resource.
3. No preemption - When a process can preempt then deadlock would never occur.
4. Circular wait - A circular chain occurs around two process holding resources requested by next process in loop/chain.

The second condition of hold an wait is broken by making sure when dynamically assigning a pallet that controller (MFR) assigns it to a path that is not already occupied. Hence a great attention has to be paid in chalking out the algorithm to avoid deadlock cases.[10]

As discussed above in a dynamic case where deadlock would occur would be if the following job requests for the same resource occupied by the previous job. So if the following material in Einlagern line destination address is not being recognised by barcode reader then it should be dynamically assigned to congestion free other Auslagern path. Exceptionally if the following material

in Einlagern line destination address by the barcode reader has been recognised(static scheduling) as the former material destination address, then the Einlagern line path has to be stopped till an acknowledgment is not received from that particular Auslagern line stating that the particular job has been cleared and the following job can take place without deadlock. [6]

The idea behind Pseudo Adaptive XY routing of priortising is considered in our algorithm [3][4]. It works usually in deterministic mode when it senses the network is lightly congested, the algorithm switches to adaptive mode and finds the routes that are not congested meaning when we sense the network across Auslagern paths seems to get blocked it assigns priorities by dividing the whole traffic almost equally. Nevertheless in our case we have generally taken the load in Auslagern path A and B to be higher than C or in general the priority of dynamically assigning a path to A and B would be greater as it has higher potential of taking the material/goods in comparison to the Auslagern C path.

5.8 Combinatorial optimisation

After modelling the problem as job shop scheduling problem with makespan as an objective function and detailing the various algorithm approaches above. We could summarise our combinatorial optimisation method as follows.

1. Feed the map of various zones of Warehouse including the distances details of Auslagern path from Einlagern and use Improved Dijkstra's algorithm to find and sort this data as heap of minimum cost

2. Use the principle of Disjunctive B&B in active schedule when a material with no destination point is recognised from a Barcode reader, it dynamically compares its release time and processing time in different Auslagern path ($r_{ij} + p_{ij}$) and then assigns the Virtual Destination meldepunkt.

In general we also use the principle of Pseudo Adaptive XY routing to prevent blocking of the jobs all the time by assigning priority to certain Auslagern paths (A and B in our case).

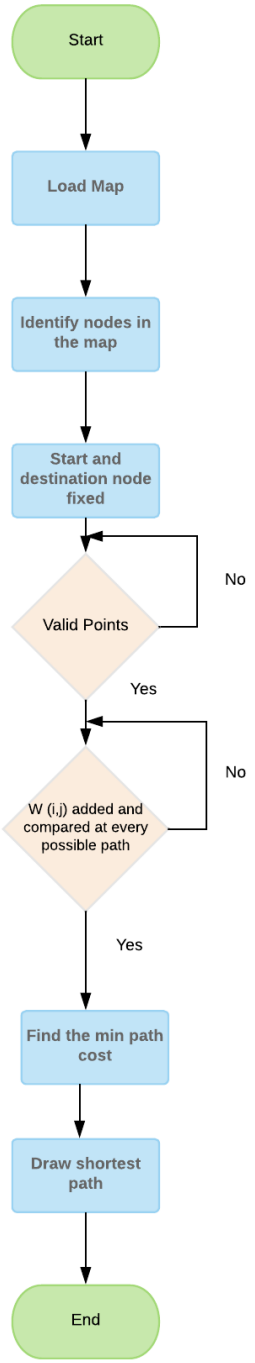


Figure 5.4: Traditional Dijkstra's Flowchart

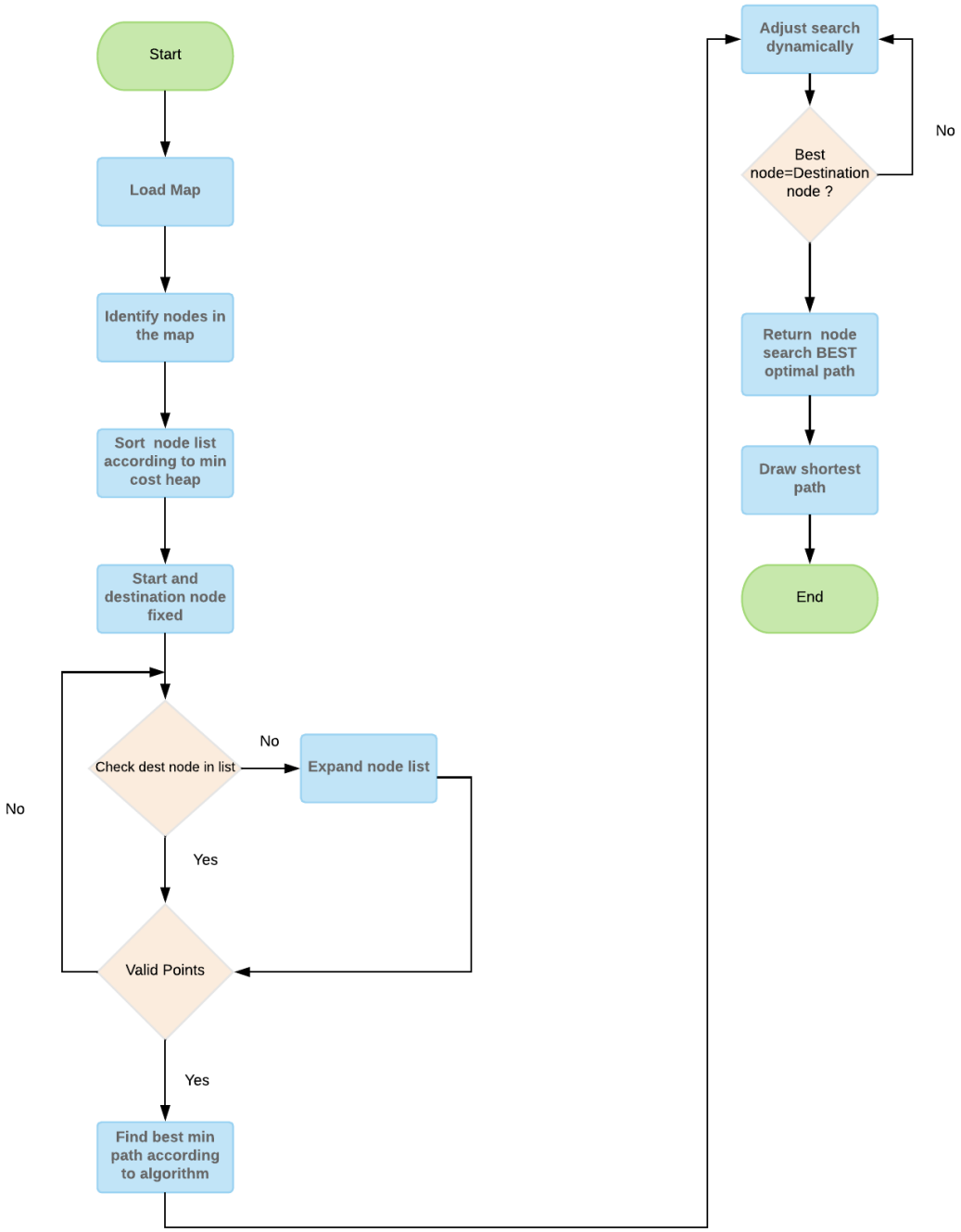


Figure 5.5: Improved Dijkstra's Flowchart

Chapter 6

Algorithm implementation

The previous chapter modelled the problem as a scheduling problem and drafted a a combinatorial optimisation algorithm. This chapter details out the implementation part through flowcharts, algorithm and block diagrams.

6.1 Barcode communication

The SICK ® barcode reader communication considered with the PLC is via TCP/IP protocol. Neverthless the barcode reader provides the additional communication functionality via Profinet, EtherCAT and so on. The communication involves following steps.

1. Set up Server and client with respective port number and IP address.
2. Call respective function blocks TCPserv, TCPsend, TCP close and TCPclient from the B&R Automation Studio library.

The TCP/IP manager issues a instruction correspondingly in the Automation Studio as and when its called which further enables the function block to connect to the Berkley sockets (BSD). Basic communication between devices on TCP/IP network happens via IP addresses. The server and the client is initialised in the subprograms. The buffer is initialised with data, and the parameters for sending, transmitting and closing the connection are specified. The programs are run in chronological and in cyclical manner. At first TCPserv() and TCPclient() functions are called to establish a connection. There is a buffer assigned to send the data in an active connection. The

length of the data is later verified with rxbuflng function. The send process is completed after status == 0 at server and the client end indicating no error. Later the connection is terminated with TCPclose() function. The flowchart of the server and client program is presented below in the figure 6.1 and 6.2.

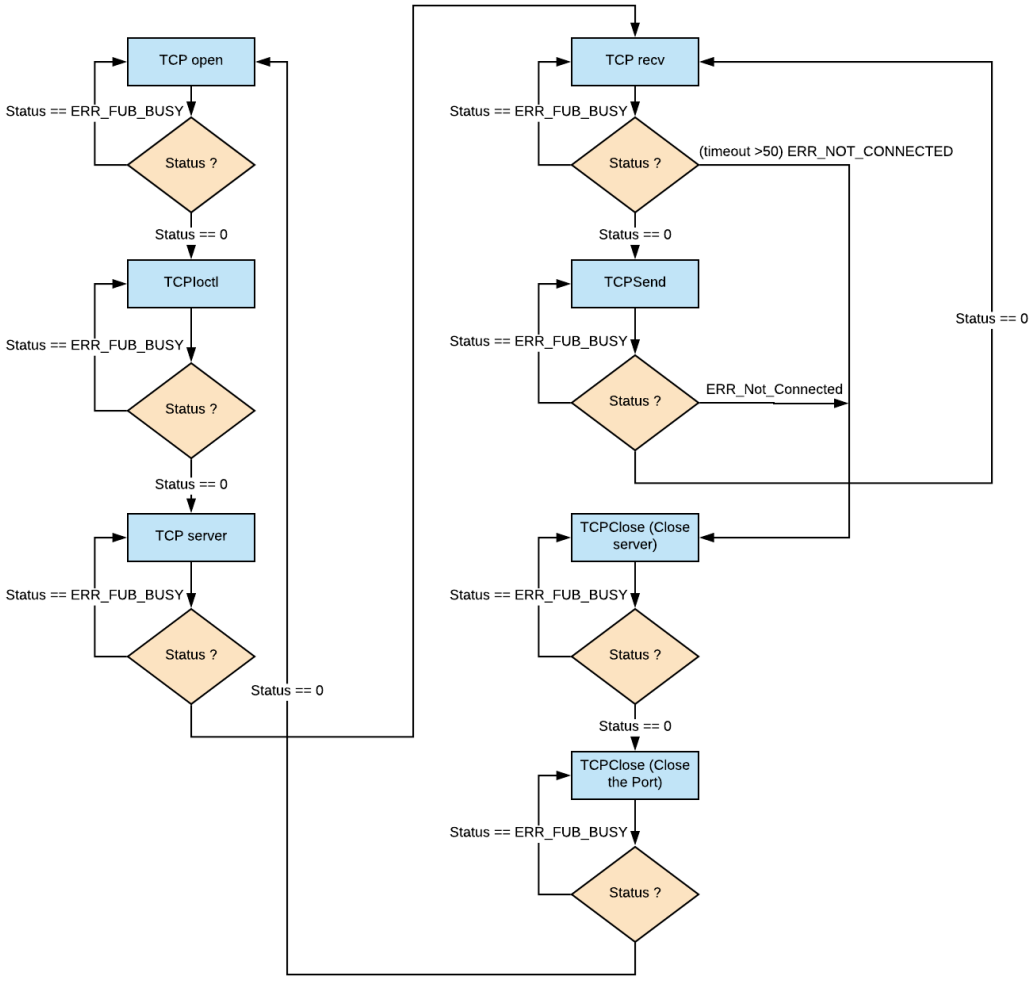


Figure 6.1: TCP/IP Server communication

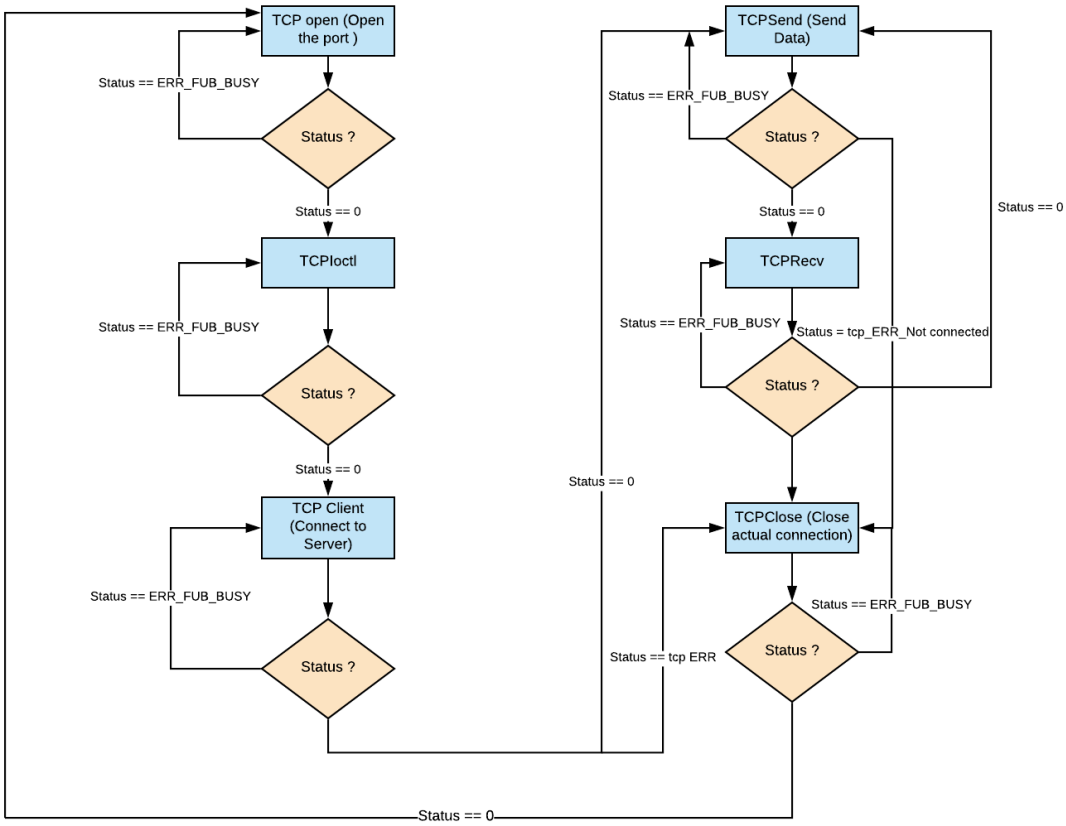


Figure 6.2: TCP/IP Client communication

6.2 Map sorting

As indicated in the earlier chapter after establishing the communication we would sort the map of warehouse in form of heap from min to max cost of distances. As we are considering one of the zones of the warehouse it is easier to sort and assign it to an array. However in order to make our algorithm scalable to the complete warehouse we resort to sort the map address in form of merge sorting with worst case complexity of **O(nlogn)**.

The algorithm of **merge sorting** is as follows.

1. Step 1 : Load all the details of the map address in array of integers
2. Step 2 : Check for the number of elements in the array if its greater

than 1. Later split this array into two sub-array leftmap[p..q] and rightmap[q+1..r], where q is the mid point A[p..r].

3. Step 3 : Recursively keep splitting the two arrays like in step 2 until only an element is left

4. Step 4 : Recursively sort the two sub arrays leftmap[p..q] and rightmap[q+1..r].

5. Step 5 : Merge the two subarray elements into a sorted sequence.

6.3 Datafields

The scanned barcode provides information of the material flow in a warehouse. The data-field that is required to be indicated in the logfile in SQL server or .csv file can be classified as follows.

1. Transport Auftrag Nummer : This provides information of the every order

2. Sequenz Nummer : This is included to keep internal count of number of materials flowing on different Einlagern and Auslagern line.

3. Melde Punkt Start : This would indicate the starting point or in short the Einlagern point address.

4. Melde Punkt Ziel : Indicates the Auslagern path taken.

5. Freigabe : Indicates the path to be set free.

6. Fehlercode : Indicates the error code.

7. Start Datum und Zeit: Indicates the start date and time on Einlagern.

8. End Datum und Zeit: Indicates the end date and time after reaching the destination.

6.4 SQL Server

As the material flows the corresponding fields and the respective date and time are written not only in Automation Studio software but also synchronised with the SQL Database server. Through Open Database connectivity(ODBC) a standard-based method used for accessing data from PLC to the MSSQL server the data is synchronised cyclically. The SQL server used is Microsoft SQL Express 2014.

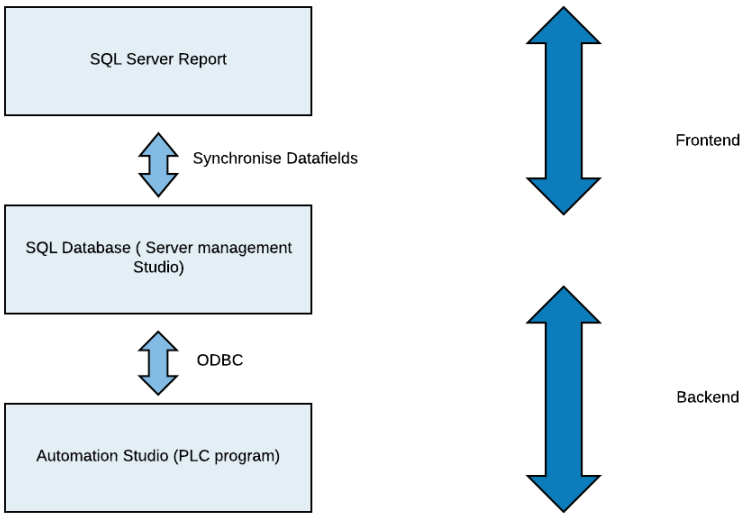


Figure 6.3: Database execution flow

6.5 Function blocks

As the library in **Automation Studio** environment provides many inbuilt functions which is quintessential in reducing the code length and in parallel to providing the required functionality. Few of the important FB's used in our program can be listed as follows.

1. **TON** : It implements a switch on delay. It has input signal (IN), delay time (PT), Output signal (Q) and elapsed time (ET). After the input is true, the elapsed time starts counting till it reaches the delay time

value. Later on output signal Q is activated and it falls only after the input signal becomes false.

2. **DTGetTime**: It is a standard function library that reads the current time. It has a enable as input signal, the output being status and DT1 gives out the date and time This is implemented in our code to read the time and date as and when the material on a conveyor line is read by the barcode reader.

3. **dbConnect**: It helps to connect to a SQL database via a network connection ISO-8859-1. The enable in the function block is executed if its not equal to 0. The server name helps to connect to particular IP address specified in MSSQL server.

4. **dbExecuteSql**: This function block executes an SQL instruction directly on SQL server database. Various key commands in SQL language can be executed like "CREATE", "INSERT INTO", "UPDATE" and so on.

5. **dbDisconnect**: It is used in the end after the respective columns in the table has been updated to disconnect with SQL database server.

6.6 Routing implementation

As discussed in the previous sections about the static and dynamic routing of various pallets on conveyor belt. The flowchart behind the code in IEC 61131-3 is detailed as below.

In the static case initially the motor in Einlagern line is checked and later if all the preliminary conditions are satisfied, and the material is clearly being read by the Einlagern line the static routing method is adapted.

If the barcode is not clearly read and the destination address has to be decided by the controller then PLC adapts the heuristics algorithm as discussed in chapter 5. In both the cases the load capacity of Auslagern line is checked continuously as and when the count in each line gets updated.

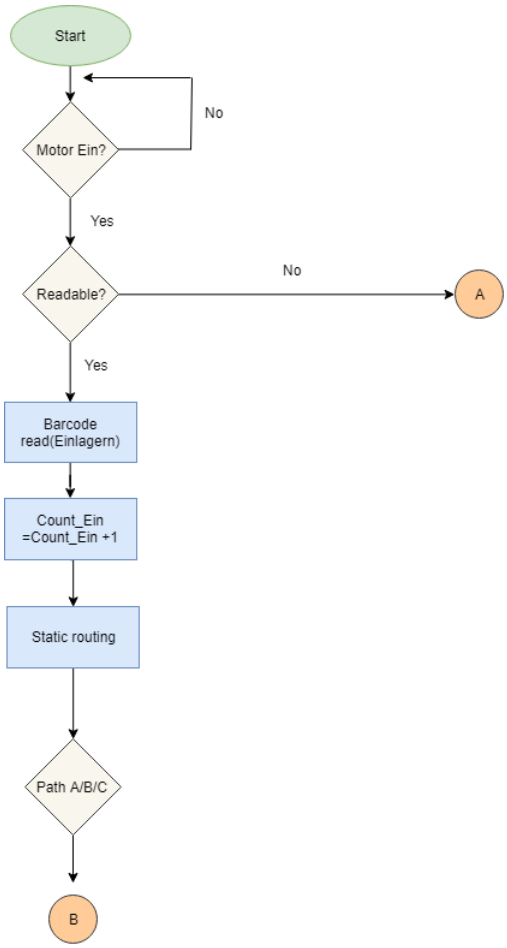


Figure 6.4: Static routing (Einlagern)

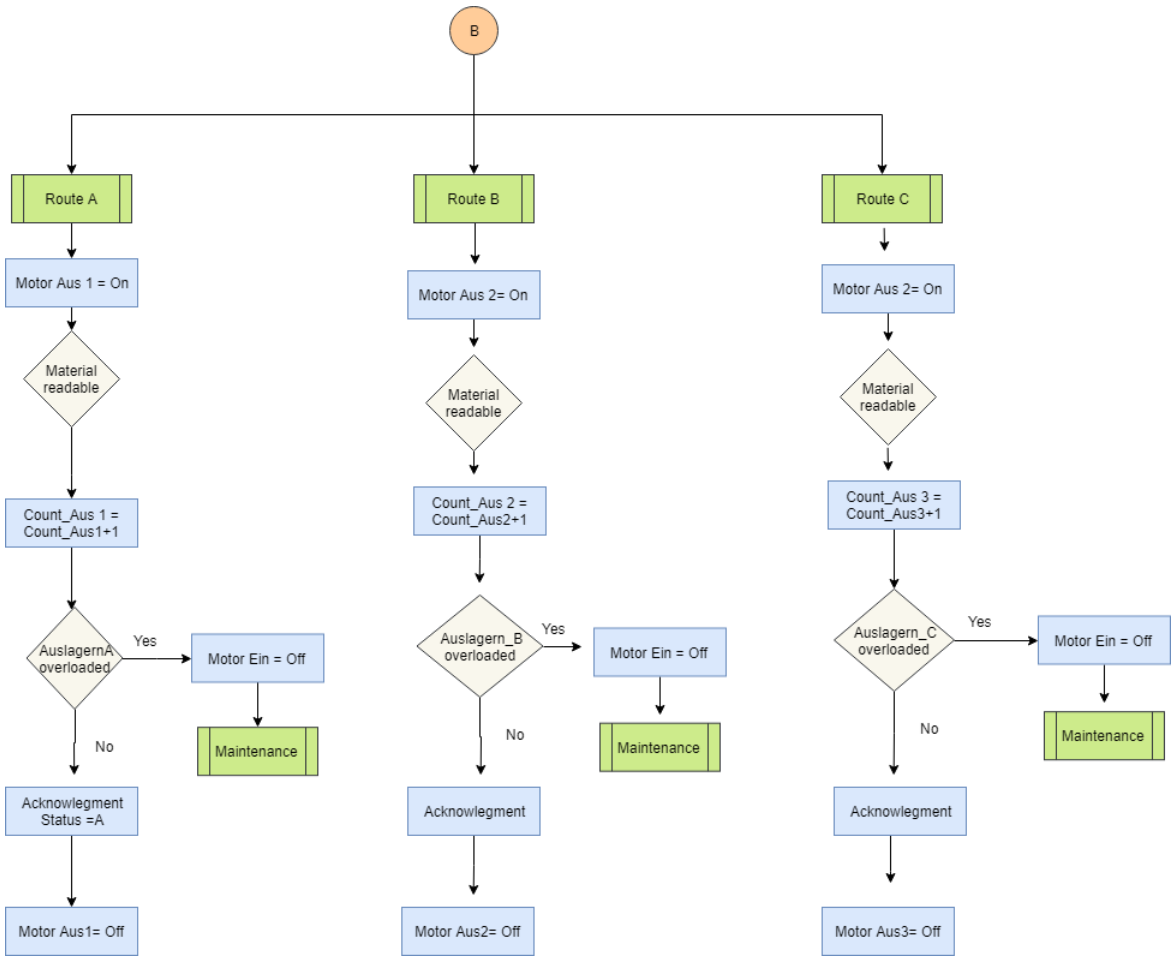


Figure 6.5: Static routing(Auslagern)

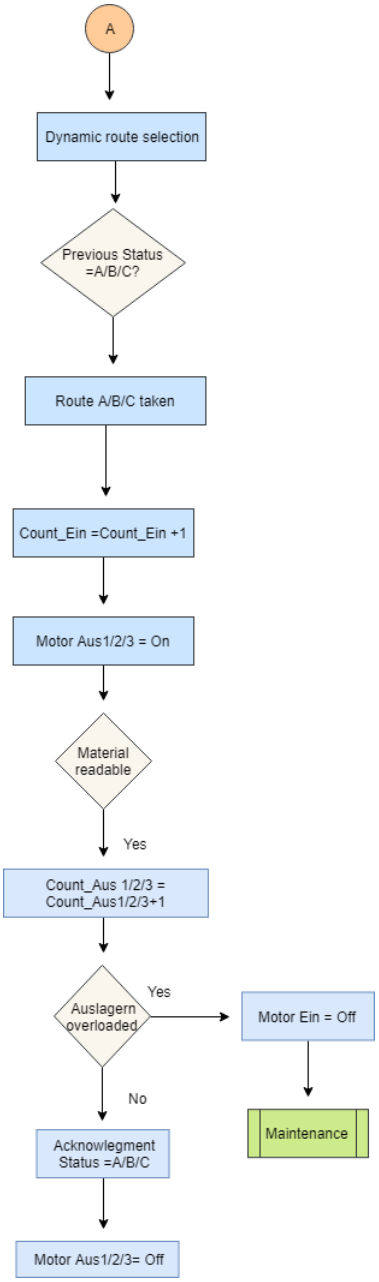


Figure 6.6: Dynamic routing (Auslagern)

Chapter 7

Algorithm analysis

This chapter gives a condensed outlook on the results that is paramaterised via Gantt chart for jobs in static routing case and later in the dynamic case.

7.1 Results of the algorithm

As outlined in previous Chapter 5 the heuristics algorithm applied for the particular zone 1 will be simulated for a particular set of materials flowing in Einlagern line and the corresponding waiting time will be calculated for particular jobs. As input provided by the customer in normal case the time taken for a material in Einlagern (901) , Auslagern path (906), Auslagern path (915), Auslagern path(919) are approximately 10 to 24 seconds respectively.

In our study we have considered barcode 128 standard and the output read as array of char(data type) in the controller(PLC) for particular barcodes as described in the figure 7.7 decides on the particular Auslagern path. In the following analysis we simulate the processing time for a particular set of barcode inputs(jobs) in following cases.

1. Static case : Destination address is specified from the barcode.
2. Dynamic case : Destination address is dynamically assigned by the controller(PLC) using the heuristics algorithm as defined.

Later by constructing Gantt chart the processing time at first in a static case and later in a dynamic cases are portrayed in the table and chart where

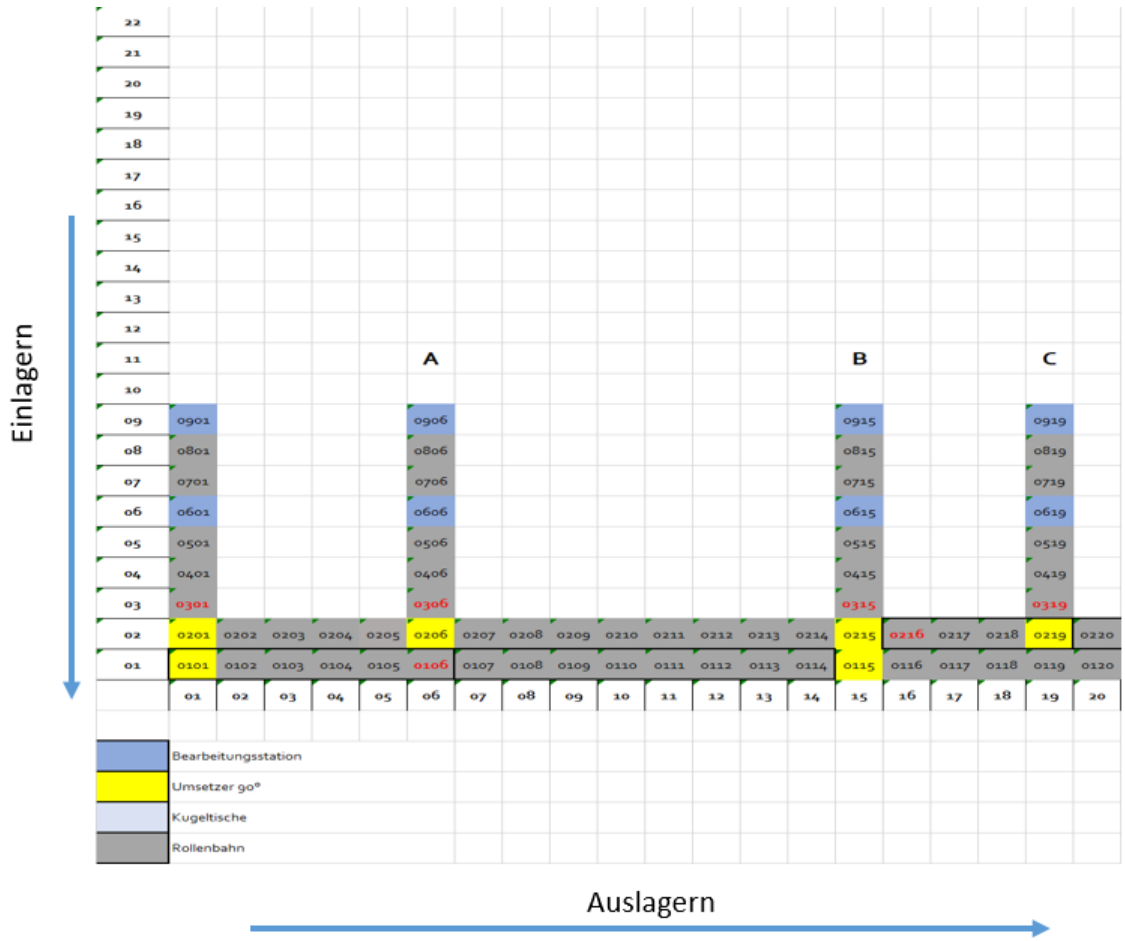


Figure 7.1: Schematic layout of one of the zones

the number of unknown pallets range from 30% to 70%. A comprehensive experiment with different set of cases when number of unknown pallets is 30% is studied and the algorithm performance is analysed. It is seen that the online algorithm performance depends on the way in which the pallets come in each conveyor lines. From the Gantt Chart it is clear that heuristics performs reasonably well for the most cases, only when the number of unknown pallets increase more than a reasonable number than the solution of dynamic assignment might always not be an optimal one as seen in the figure 7.10 and 7.11. The SQL Database screenshot shown in the figure 7.12 represents a local datalog file simulated.

SFS-ZO1-1906 SFS-ZO1-1915 SFS-ZO1-1919

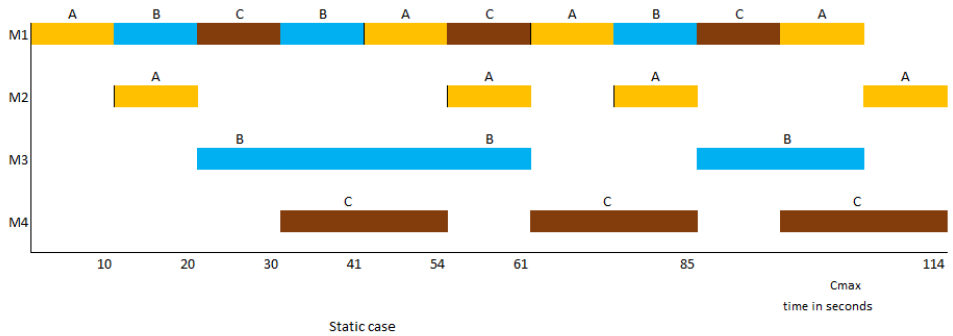
Figure 7.2: Barcodes of Zone 1

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			11
2	SFSZO1-1915		✓		21
3	SFSZO1-1919			✓	24
4	SFSZO1-1915		✓		20
5	SFSZO1-1906	✓			11
6	SFSZO1-1919			✓	24
7	SFSZO1-1906	✓			10
8	SFSZO1-1915		✓		21
9	SFSZO1-1919			✓	24
10	SFSZO1-1906	✓			10

Table 7.1: Static case

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-1919			✓	24
4	SFSZO1-1915		✓		20
5	SFSZO1-1906	✓			11
6	SFSZO1-1919			✓	24
7	SFSZO1-1906	✓			10
8	SFSZO1-1915		✓		20
9	SFSZO1-XXX	✓			10
10	SFSZO1-1906	✓			11

Table 7.2: Dynamic case (10%)

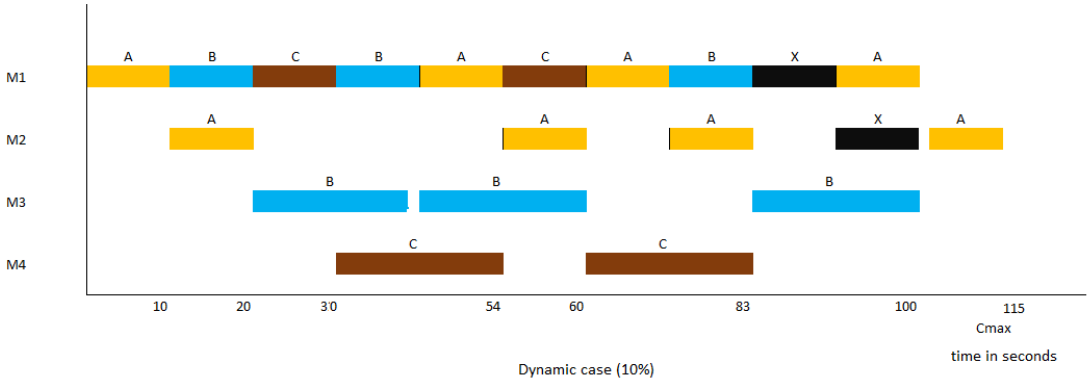


M1,M2, M3,M4 This are barcodereaders in line 0901, 0906,0915,0919

A,B,C Jobs with Barcode SFS-ZO1-1906, SFS-ZO1-1915, SFS-ZO1-1919 respectively.

X's Jobs that are dynamically assigned

Figure 7.3: Gantt Chart : Static case



M1,M2, M3,M4 This are barcodereaders in line 0901, 0906,0915,0919

A,B,C Jobs with Barcode SFS-ZO1-1906, SFS-ZO1-1915, SFS-ZO1-1919 respectively.

X's Jobs that are dynamically assigned

Figure 7.4: Gantt Chart : Dynamic case(10%)

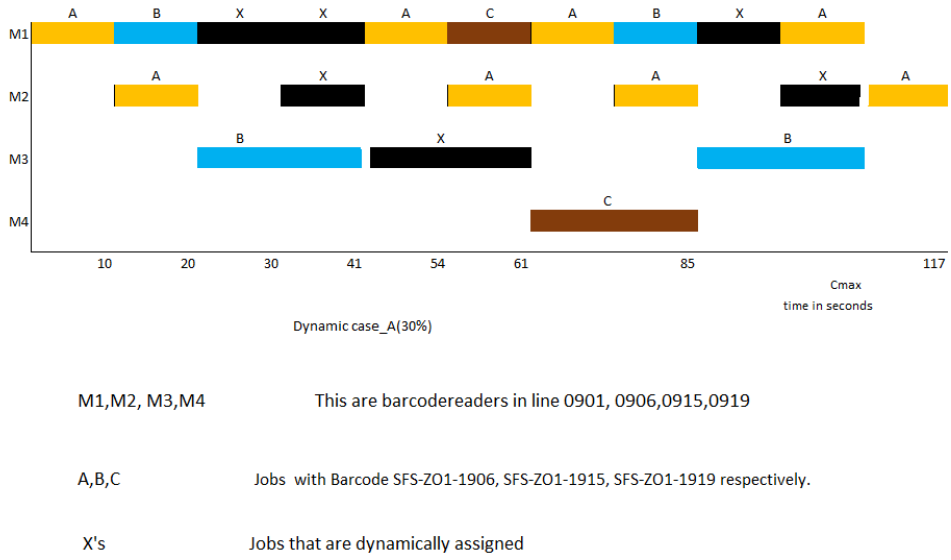
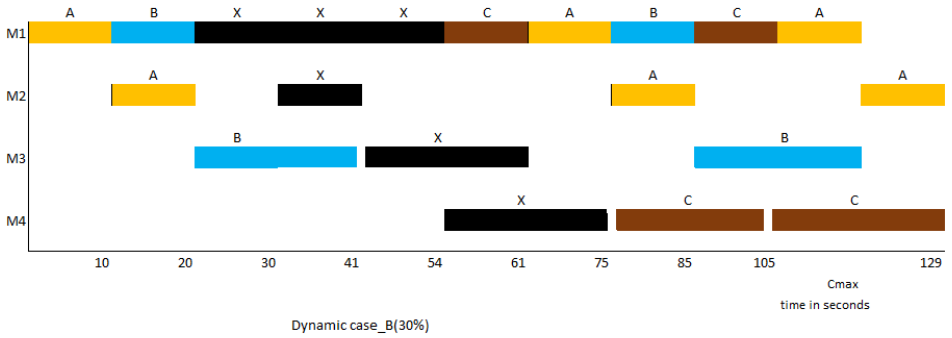


Figure 7.5: Gantt Chart : Dynamic case A(30%)

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-XXX	✓			10
4	SFSZO1-XXX		✓		21
5	SFSZO1-0906	✓			11
6	SFSZO1-1919			✓	24
7	SFSZO1-1906	✓			10
8	SFSZO1-1915		✓		20
9	SFSZO1-XXX	✓			10
10	SFSZO1-1906	✓			14

Table 7.3: Dynamic case A (30%)

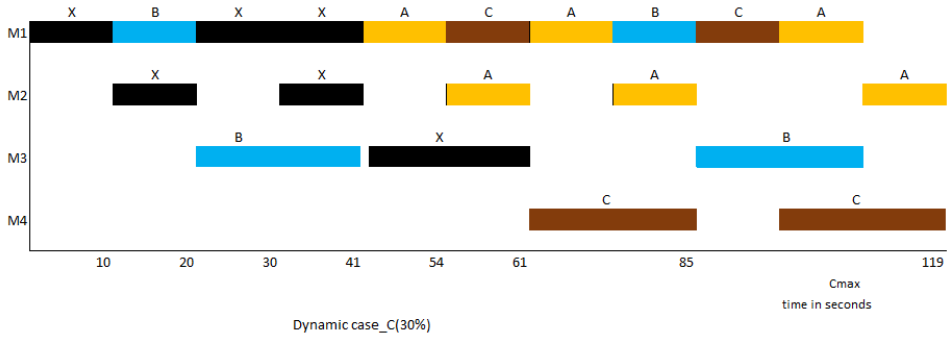


M1,M2, M3,M4 This are barcodereaders in line 0901, 0906,0915,0919

A,B,C Jobs with Barcode SFS-ZO1-1906, SFS-ZO1-1915, SFS-ZO1-1919 respectively.

X's Jobs that are dynamically assigned

Figure 7.6: Gantt Chart : Dynamic case B(30%)



M1,M2, M3,M4 This are barcodereaders in line 0901, 0906,0915,0919

A,B,C Jobs with Barcode SFS-ZO1-1906, SFS-ZO1-1915, SFS-ZO1-1919 respectively.

X's Jobs that are dynamically assigned

Figure 7.7: Gantt Chart : Dynamic case C(30%)

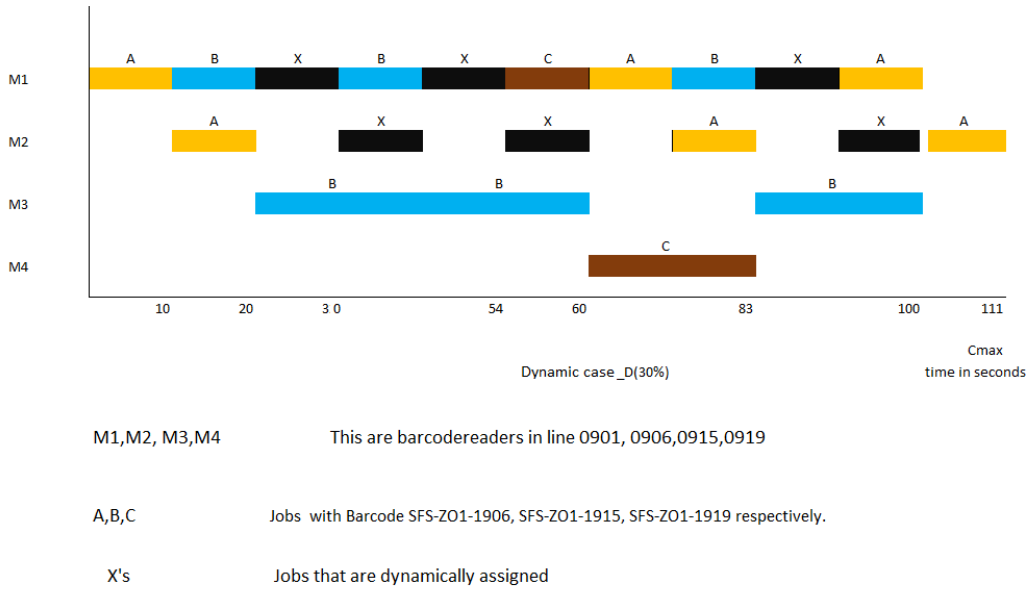


Figure 7.8: Gantt Chart : Dynamic case D(30%)

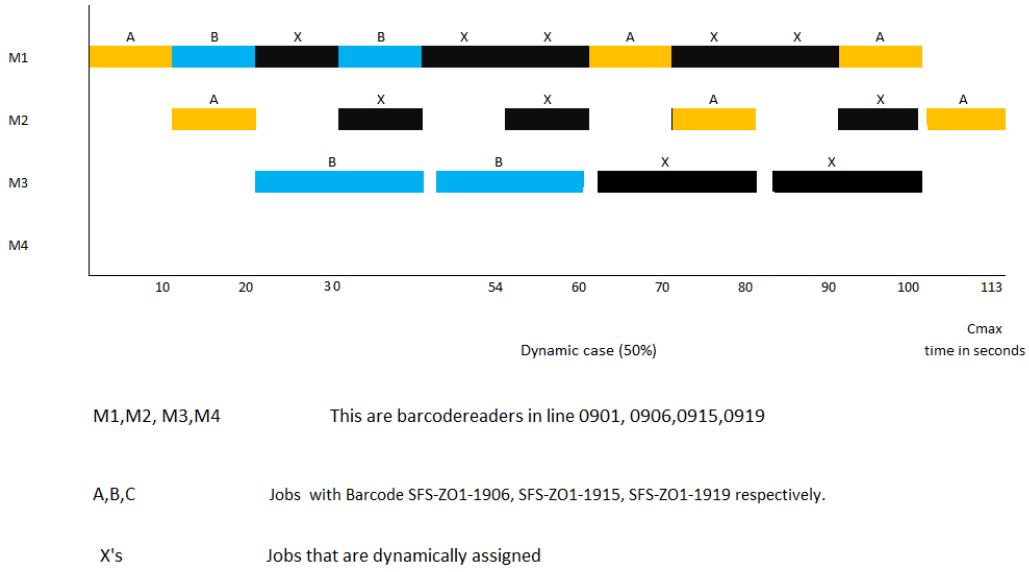


Figure 7.9: Gantt Chart : Dynamic case(50%)

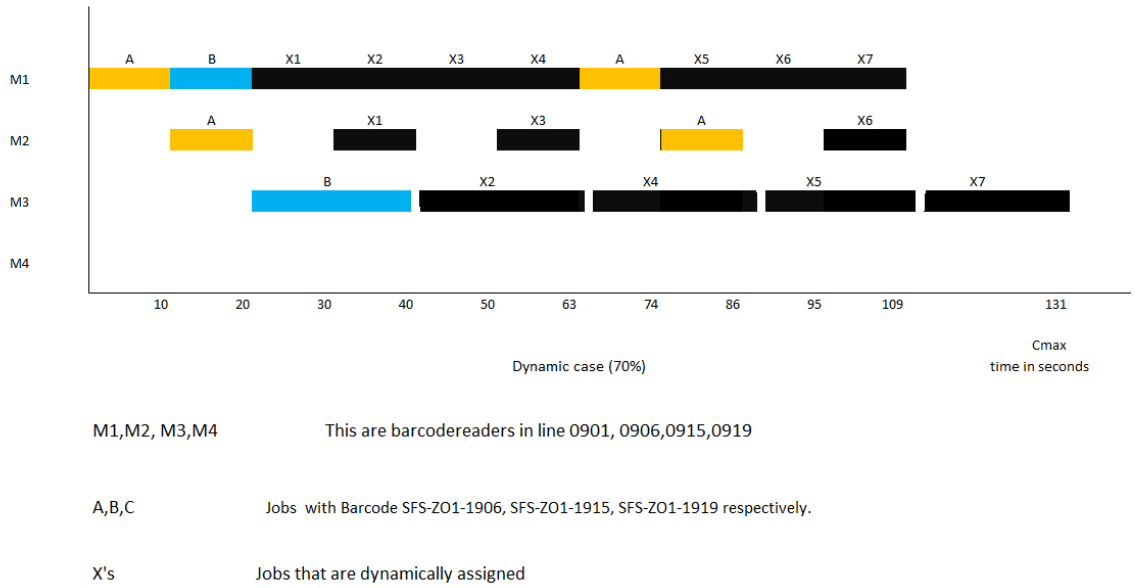


Figure 7.10: Gantt Chart : Dynamic case(70%)

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-XXX	✓			10
4	SFSZO1-XXX		✓		21
5	SFSZO1-XXX			✓	24
6	SFSZO1-1919			✓	24
7	SFSZO1-1906	✓			10
8	SFSZO1-1915		✓		20
9	SFSZO1-1919			✓	24
10	SFSZO1-1906	✓			12

Table 7.4: Dynamic case B (30%)

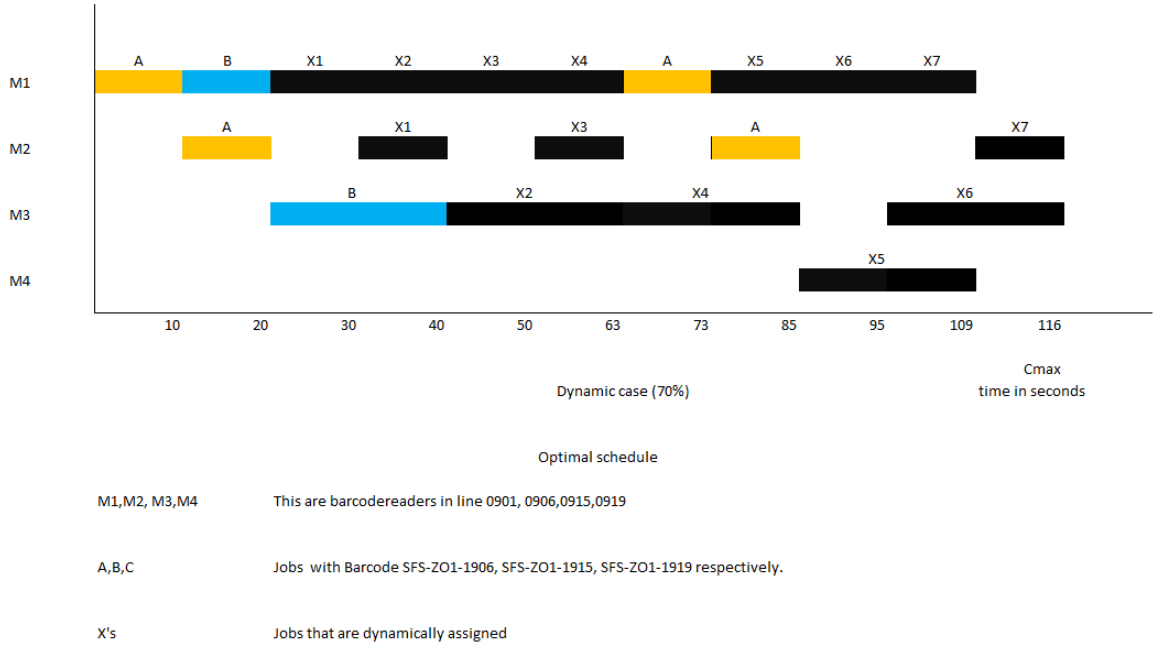


Figure 7.11: Gantt Chart : Optimal dynamic case(70%)

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-XXX	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-XXX	✓			10
4	SFSZO1-XXX		✓		21
5	SFSZO1-0906	✓			11
6	SFSZO1-1919			✓	24
7	SFSZO1-1906	✓			10
8	SFSZO1-1915		✓		20
9	SFSZO1-1919			✓	24
10	SFSZO1-1906	✓			14

Table 7.5: Dynamic case C (30%)

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-XXX	✓			10
4	SFSZO1-1915		✓		20
5	SFSZO1-XXX	✓			11
6	SFSZO1-1919			✓	24
7	SFSZO1-1906	✓			10
8	SFSZO1-1915		✓		20
9	SFSZO1-XXX	✓			10
10	SFSZO1-1906	✓			14

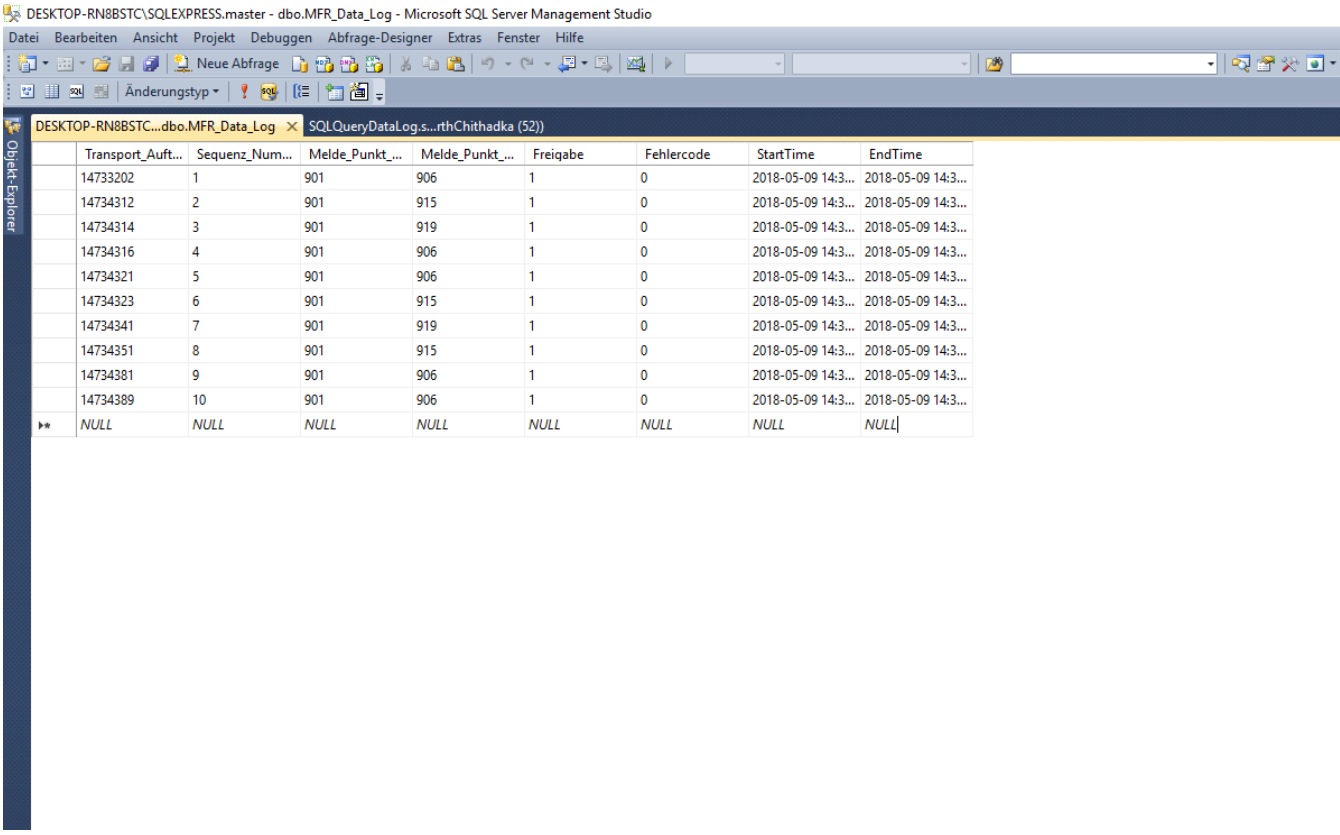
Table 7.6: Dynamic case D (30%)

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-XXX	✓			10
4	SFSZO1-1915		✓		20
5	SFSZO1-XXX	✓			11
6	SFSZO1-XXX		✓		21
7	SFSZO1-1906	✓			10
8	SFSZO1-XXX		✓		20
9	SFSZO1-XXX	✓			10
10	SFSZO1-1906	✓			14

Table 7.7: Dynamic case (50%)

Sl no	Barcode	A	B	C	Processing time(seconds)
1	SFSZO1-1906	✓			10
2	SFSZO1-1915		✓		20
3	SFSZO1-XXX	✓			10
4	SFSZO1-XXX		✓		20
5	SFSZO1-XXX	✓			11
6	SFSZO1-XXX		✓		21
7	SFSZO1-1906	✓			10
8	SFSZO1-XXX		✓		20
9	SFSZO1-XXX	✓			10
10	SFSZO1-XXX		✓		22

Table 7.8: Dynamic case (70%)



Transport_Auft...	Sequenz_Num...	Melde_Punkt_...	Melde_Punkt_...	Freigabe	Fehlercode	StartTime	EndTime
14733202	1	901	906	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734312	2	901	915	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734314	3	901	919	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734316	4	901	906	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734321	5	901	906	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734323	6	901	915	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734341	7	901	919	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734351	8	901	915	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734381	9	901	906	1	0	2018-05-09 14:3...	2018-05-09 14:3...
14734389	10	901	906	1	0	2018-05-09 14:3...	2018-05-09 14:3...
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 7.12: Datalog screenshot

Chapter 8

Summary and future work

8.1 Summary

The aim of this thesis was to primarily develop a software model of a material flow calculator(MFR:Material Fluss Rechner) in a automatic warehouse. The thesis work was divided into three parts. First step involved to identify the bottlenecks, develop the mathematical model and design an algorithm to solve the scheduling problem model. Second step was to realise the developed algorithm designed in IEC 611313-3 language. At last was to develop the database model topology sychronisation interface with the PLC (Programmable logic controller) to have regular details of material flow in a warehouse.

The thesis study conducted, designs a software model that optimises the routing of material in a warehouse in both static and dynamic cases. Thereby reducing the waiting time and makespan(Cmax) for a particular job. This was achieved by developing a heuristic algorithm combining the idea of Improved Dijkstra's model, Disjunctive Branch and Bound. Deadlocks are avoided here by prioritising the jobs when the algorithm runs in a deterministic mode [16][19][3]. The sphere of developing this software model was also broadened by adapting a agile model philosophy as described in chapter 4.

Objectives achieved in this study can be summarised as follows.

1. Service Oriented architecture software model developed.
2. Mathematically modelled routing of pallets as a scheduling problem.

3. Simulation via Automation Studio ® software was conducted to verify wait times in various scenarios.

4. Development of software model topology to synchronize the material information in Microsoft SQL Server Database and to maintain local datalog(.CSV) file.

8.2 Future Work

The algorithm and the software model applied to one of the zones of the warehouse can be further extended by loading the map of complete warehouse plant as shown in the figure 8.1.

As visualisation helps to identify the material flow it can be extended as a software package where a HMI interface can be developed with a option of providing a HTML based service, which helps to visualise the plant from a remote server. Later a dedicated software block could be developed to handle the errors generating from the routing, SQL and other function blocks.

Lastly a interactive tool could be developed that can search the whole list of active material flowing and in the storage, which would later help in procurement and optimising the storage layout.

The study and software developed was to make all the service based software plug in modules to be integrated and work heterogeneously in a existing warehouse(retrofit) independent of different automation systems.

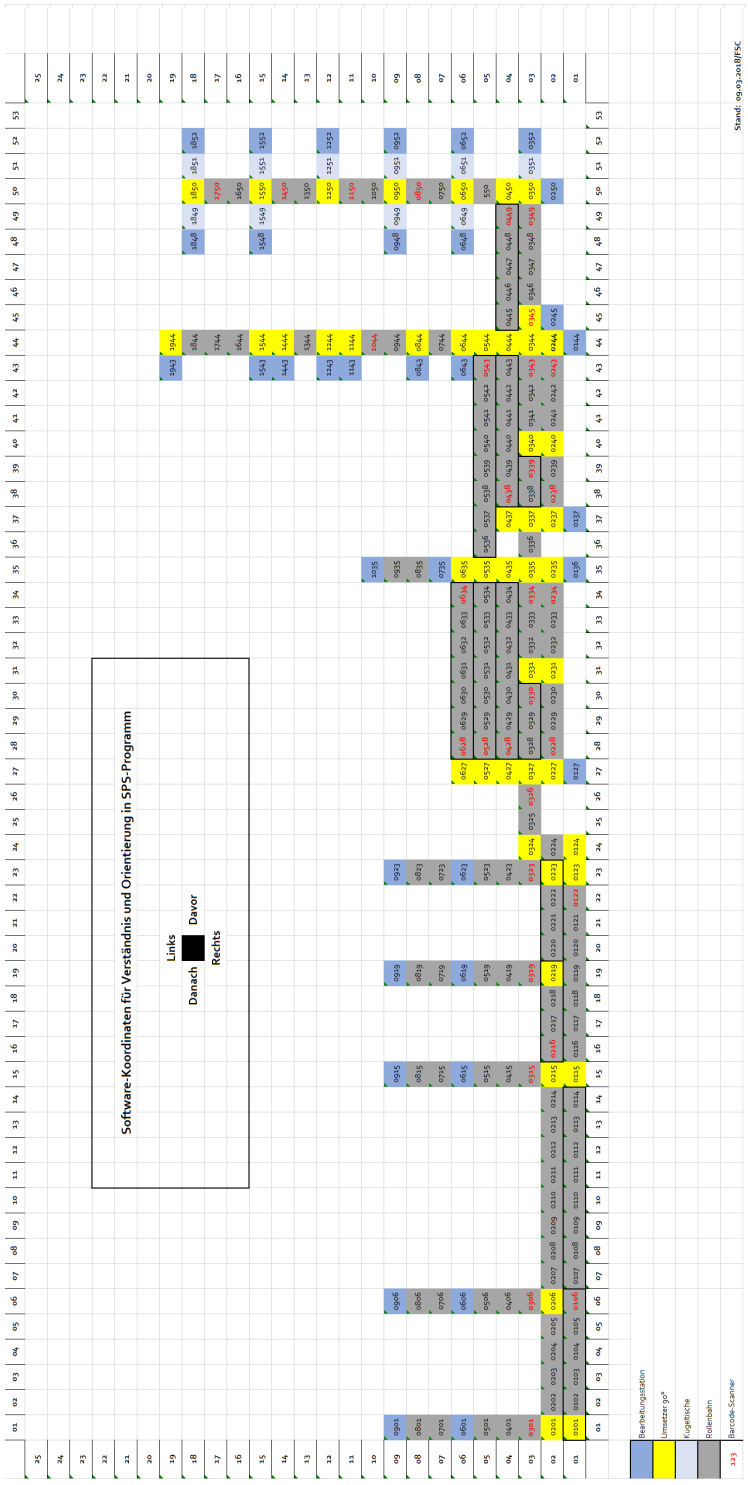


Figure 8.1: Warehouse schematic of various zones

Bibliography

[1] L. Bryan and E. Bryan, *Programmable Controllers - Theory and Implementation, 2nd Edition*. 1997, isbn: 094410732X.

[2] C. Wallin, F. Ekdahl, and S. Larsson, "Integrating business and software development models," *IEEE Software*, vol. 19, no. 6, pp. 28–33, Nov. 2002, issn: 0740-7459. doi: 10.1109/MS.2002.1049384.

[3] M. Dehyadgari, M. Nickray, A. Afzali-kusha, and Z. Navabi, "Evaluation of pseudo adaptive xy routing using an object oriented model for noc," in *2005 International Conference on Microelectronics*, Dec. 2005, 5 pp.-. doi: 10.1109/ICM.2005.1590068.

[4] M. Li, Q.-A. Zeng, and W.-B. Jone, "Dyxy - a proximity congestion-aware deadlock-free dynamic routing method for network on chip," in *2006 43rd ACM/IEEE Design Automation Conference*, 2006, pp. 849–852. doi: 10.1109/DAC.2006.229242.

[5] M. Hompel and T. Schmidt, *Warehouse Management:Automation and Organisation of Warehouse and Order Picking Systems*. Springer-Verlag Berlin Heidelberg, 2007, isbn: 9783540352181.

[6] E. Gawrilow, E. Köhler, R. H. Möhring, and B. Stenzel, "Dynamic routing of automated guided vehicles in real-time," in *Mathematics – Key Technology for the Future: Joint Projects Between Universities and Industry 2004–2007*, H.-J. Krebs and W. Jäger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 165–177, isbn: 978-3-540-77203-3. doi: 10.1007/978-3-540-77203-3_12. [Online]. Available: https://doi.org/10.1007/978-3-540-77203-3_12.

[7] W. Bolton, *Programmable Logic Controllers, 5th Edition*. 2009, isbn: 9781856177511.

[8] M. Feldhorst S.and Fiedler and ten Hompel, *TPaket Royale - Dezentrale Steuerung für das Internet der Dinge*, ser. "In Tagungsband zum 6 Fachkolloquium der Wissenschaftlichen Gesellschaft für Technische Logistik, Hannover". 2010, isbn: 9783941416710.

[9] S. Feldhorst, M. Fiedler, M. Heinemann, M. ten Hompel, and H. Krumm, "Event-based 3d-monitoring of material flow systems in real-time," in *2010 8th IEEE International Conference on Industrial Informatics*, Jul. 2010, pp. 195–200. doi: `10.1109/INDIN.2010.5549433`.

[10] K. C. T. Vivaldini, J. P. M. Galdames, T. S. Bueno, R. C. Araújo, R. M. Sobral, M. Becker, and G. A. P. Caurin, "Robotic forklifts for intelligent warehouses: Routing, path planning, and auto-localization," in *2010 IEEE International Conference on Industrial Technology*, Mar. 2010, pp. 1463–1468. doi: `10.1109/ICIT.2010.5472487`.

[11] G. Bohács and K. F. Semrau, "Automatische visuelle datensammlung aus materialflusssystemen und ihre anwendung in simulationsmodellen," *Logistics Journal: Not reviewed*, vol. 2012, no. 01, 2012, issn: 1860-5923. doi: `10.2195/lj_NotRev_bohacs_de_2012_01`. [Online]. Available: `http://nbn-resolving.de/urn:nbn:de:0009-14-32514`.

[12] K. N. Subramanya and T. M Rangaswamy, *Impact of warehouse management system in a supply chain*, Oct. 2012. [Online]. Available: `https://www.researchgate.net/publication/235432632_Impact_of_Warehouse_Management_System_in_a_Supply_Chain`, (accessed: 01.02.2018).

[13] O. Kipouridis, W. A. Günthner, M. Roidl, and M. ten Hompel, "Kollaborative planung dezentral gesteuerter materialflusssysteme in der intralogistik," *Logistics Journal*, vol. 2013, no. 10, 2013, issn: 2192-9084. doi: `10.2195/lj_Proc_kipouridis_de_201310_01`. [Online]. Available: `http://nbn-resolving.de/urn:nbn:de:0009-14-37893`.

[14] P. Kolarovszki and J. Vaculík, "Warehouse management system based on selected automatic identification technology," *Proceedings of the 13 International Conference on Reliability and Statistics in Transportation and Communication , Riga, Latvia*, pp. 257–264, 2013. [Online]. Available: `http://www.tsi.lv/sites/default/files/editor/science/Publikacii/RelStat_13/session_7_ed_kolarovszki_vaculik_final_ok.pdf`, (accessed: 01.02.2018).

[15] W. Günthner, E. Klenk, and P. Tenerowicz-Wirth, *Adaptive Logistiksysteme als Wegbereiter der Industrie 4.0*, T. Bauernhansl, M. ten Hompel, and B. Vogel-Heuser, Eds. Wiesbaden: Springer Fachmedien Wiesbaden, 2014, pp. 297–323, isbn: 978-3-658-04682-8. doi: `10.1007/978-3-658-04682-8_15`. [Online]. Available: `https://doi.org/10.1007/978-3-658-04682-8_15`.

[16] C. Ruan, J. Luo, and Y. Wu, "Map navigation system based on optimal dijkstra algorithm," in *2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems*, Nov. 2014, pp. 559–564. doi: `10.1109/CCIS.2014.7175798`.

[17] D. G. Starke, *Effektive Software Architekturen, 7th Edition*. Carl Hanser Verlag, München, 2015, isbn: 9783446443617.

[18] A. M. Atieh, H. Kayalani, Y. Al-abdallat, L. G. Abeer Qaderi, L. Jaradat, and I. Hdairis, "Performance improvement of inventory management system processes by an automated warehouse management system," *48th CIRP Conference on manufacturing systems - CIRP CMS 2015, Procedia*, vol. 41, pp. 568–572, 2016.

[19] M. L. Pinedo, *Scheduling Theory, Algorithms, and Systems*. Springer International Publishing, 2016, isbn: 978-3-319-26578-0. doi: `10.1007/978-3-319-26580-3`.

[20] J. J. Bartholdi and S. T. Hackman, *Warehouse & Distribution Science*. 2017. [Online]. Available: `https://www.warehouse-science.com/book/index.html`.

Eidesstattliche Versicherung
(Affidavit)

Chithadka, Siddharth

Name, Vorname
(Last name, first name)

189704

Matrikelnr.
(Enrollment number)

Ich versichere hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit/Masterarbeit* mit dem folgenden Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present Bachelor's/Master's* thesis with the following title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution.

Titel der Bachelor-/Masterarbeit*:
(Title of the Bachelor's/ Master's* thesis):

A PLC based real time material flow controller/calculator in Warehouse management system

*Nichtzutreffendes bitte streichen
(Please choose the appropriate)

Ort, Datum
(Place, date)

Unterschrift
(Signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird gfls. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to €50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, section 63, subsection 5 of the North Rhine-Westphalia Higher Education Act (*Hochschulgesetz*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:**

Ort, Datum
(Place, date)

Unterschrift
(Signature)

**Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.