Masaryk University

Faculty of Economics and Administration

**Field of study: Business Informatics**



ANALYSIS OF THE IMPLEMENTATION AND USE OF PLANNING aps sw TO THE EXISTING erp SYSTEM

Bachelor's thesis

Thesis supervisor: Author:

Ing. Jaromír Skorkovský, CSc. Michal Kalinec

Brno, 2018

Name of the author: Michal Kalinec

Title of bachelor's thesis: Analysis of the implementation and use of planning APS SW to the existing ERP system

Department: Business Administration

Thesis supervisor: Ing. Jaromír Skorkovský, CSc.

Year of defense: 2019

Annotation

-TODO-

Keywords

production planning, APS, ERP, TOC, Lean, Software engineering

Anotácia

-TODO-

Kľúčové slová

-TODO-

Declaration

-TODO-

Author's signature

Acknowledgment

-TODO-

**TAble of contents**

introduction 7

Topic 7

Methodology 7

1 THEORETICAL PART 9

1.1 Lean 9

1.1.1 Definition 9

1.1.2 History 9

1.1.3 Principles of Lean 10

1.2 TOC 10

1.2.1 TOC Thinking Processes Tools 11

1.2.1.1 The Logical Thinking Process 12

1.3 ERP 15

1.3.1 Introduction 15

1.3.2 Purpose 16

1.3.3 MAX ERP 16

1.4 APS 17

1.4.1 Introduction 17

1.4.2 Definition 18

1.4.3 AHP-Leitstand 19

1.5 SW Engineering 20

1.5.1 Definition 20

1.5.2 Categories of SW products 21

1.5.3 SW development models 21

1.5.3.1 Extreme Programming 23

1.5.4 Unified Modeling Language 25

1.5.4.1 Use case diagram 25

1.5.4.2 Class diagram 25

1.5.4.3 Entity relationship diagram 26

2 PRACTICAL PART 27

2.1 Mincovňa Kremnica, š.p. 27

2.1.1 History 27

2.1.2 Current situation 27

2.2 Research topic 28

2.2.1 AHP-Leitstand 28

2.3 Definition of the previous state 29

2.3.1 LTP 29

2.3.1.1 Goal Tree 30

2.3.1.2 Current Reality Tree 32

2.3.1.3 Evaporating Cloud 34

2.3.1.4 Future Reality Tree 35

2.3.1.5 Prerequisite Tree 37

2.3.1.6 Transition Tree 37

2.4 Development and deployment of the application 40

2.4.1 Early stage of development 40

2.4.2 Programming stage 42

2.4.3 Full version of the application and its demonstration 43

2.5 Impact of the application 47

Conclusion 48

Bibliography 50

List of figures 51

List of images 51

list of abbreviations 51

list of appendices 53

# introduction

Planning is an essential part of any manufacturing organization, however, it hasn't received much attention. Planning tasks are rather challenging, taking into account many elements and oftentimes, there is no one best solution, albeit many feasible ones. APS (Advanced Planning and Scheduling) systems were introduced to address this issue in practice on a scientific and systematic level (Wiers and de Kok, 2017). Although in the present, they carry out the planning itself very well, another issue arises - integrating them to the portfolio of other information systems used by the company and subsequently utilize all of their systems together as much as possible. With the trend of using ERP (Enterprise resource planning) systems to facilitate this problem, the solution should be feasible.

## Topic

The motive for this thesis were imperfections in production caused by inadequate integration of APS into the network of company's information systems, resulting in work inefficiency and delays. The goal of this thesis is to analyze the situation, identify the problem and propose, ideally also carry out, the suggested solution. My research questions were therefore defined followingly:

1. Is APS or its integration a cause of the imperfections in the production process and if so, in what manner?
2. If the first question is answered positively, what is the suggested solution?

## Methodology

By combining employees' knowledge and experience of the company and its systems with my new perspective and different set of skills, joint efforts were to thoroughly research, analyze and eventually find the solution which would effectively improve the work efficiency and prevent further delays from occurring.

Theoretical part firstly elaborates on two of the most significant concepts of the past decades focused on the topic of production efficiency - Lean and Theory of Constraints (TOC). Topics will then narrow down on specifics of this thesis' concern. The third part describes ERP systems and specifically the one involved in this case. The fourth part is dedicated to APS systems and again, particularly the one used by the company researched. Last part introduces Software (SW) Engineering with emphasis on the parts relevant to this thesis, such as Extreme programming or Unified Modeling Language.

Practical part commences with a brief description of the involved company from a historical and current point of view. Then the focus shifts to the researched topic of production and its planning, with a more detailed description of their systems. Next chapter is where the work started to produce some output - analysis of the previous state using the TOC and its methods. Following part contains a description of how the suggested solution, resulting from previous analysis, came to life. Finally, the impact of the solution is summarized in the last chapter of the practical part.

# THEORETICAL PART

## Lean

### Definition

Lean, also often used with nouns such as production, manufacturing, philosophy, management, etc., is built on an idea to maximize customer value while minimizing waste. (Lean Enterprise Institute, 2018) Simply, lean means creating more value for customers with fewer resources, therefore perfect value without any waste at all is considered the ultimate goal. With that said, a natural question arises - what is waste? Hohmann is looking for an answer in Japanese words *muda*, *muri* and *mura* and summed up his findings in a very general definition: „waste is an outcome of problems, the result of processes not delivering what is expected but Undesirable Effects instead”. Thus in the context of lean thinking, eliminating waste can be interpreted as solving problems (Hohmann, 2015). Matthias Holwig in his article about genealogy of lean production mentions „Lean production not only successfully challenged the accepted mass production practices in the automotive industry, significantly shifting the trade-off between productivity and quality, but it also led to a rethinking of a wide range of manufacturing and service operations beyond the high-volume repetitive manufacturing environment.” (Holwig, 2006)

### History

There are instances of rigorous process thinking in manufacturing all the way back to the Arsenal in Venice in the 1450s. However, as well as e.g. the later concept of just-in-time production, the first modern efforts came in the automotive industry. The first person to truly integrate an entire production process was Henry Ford in 1913. Naturally, his system had its flaws hence Toyota, the pioneer of production efficiency, revisited Ford's original approach in the 1930s (Lean Enterprise Institute, 2018). In the modern era, this topic was thoroughly described by James P. Womack, Daniel Ross and Daniel T. Jones in their book *The Machine That Changed the World* (1990), which became one of the most widely cited references in the next few decades.

### Principles of Lean

Womack and Jones published their second book on the matter of lean production *Lean Thinking* (1996) and, among other things, introduced five lean principles. Purpose of this thought process is guidance throughout the implementation of lean techniques.

1. Specify value from the standpoint of the end customer by product family.
2. Identify all the steps in the value stream for each product family, eliminating whenever possible those steps that do not create value.
3. Make the value-creating steps occur in tight sequence so the product will flow smoothly toward the customer.
4. As flow is introduced, let customers pull value from the next upstream activity
5. As value is specified, value streams are identified, wasted steps are removed, and flow and pull are introduced, begin the process again and continue it until a state of perfection is reached in which perfect value is created with no waste

(Lean Enterprise Institute, 2018).



*(Lean Enterprise Institute, 2018)*

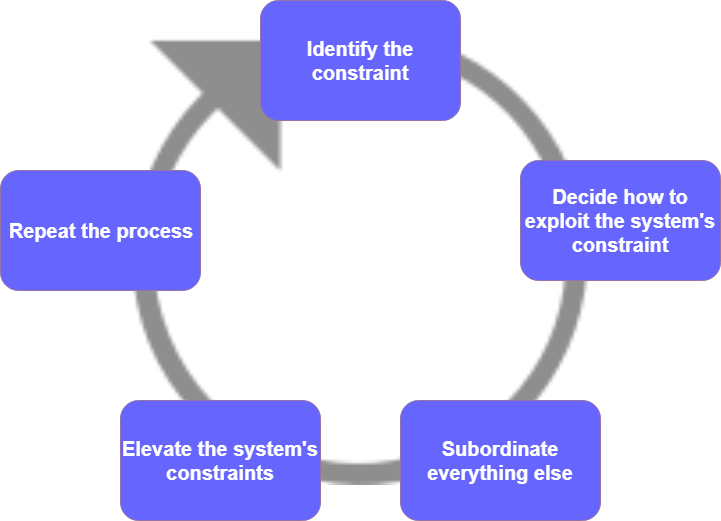
## TOC

Theory of Constraints is a popular management philosophy first introduced by Israeli businessman Eliyahu Moshe Goldratt in 1984 in his book co-authored by Jeff Cox, *The Goal: A Process of Ongoing Improvement* (Goldratt, 1984). Despite its name, it's not theoretical at all. Rather, it represents potentially big help with identifying problems and offering effective solutions. What better person to describe it than the author himself. Goldratt writes that it's not about bottlenecks, cutting batches or arranging the activities on the factory floor. His message is „We grossly underestimate our intuition. Intuitively we do know the real problems, we even know the solutions. What is unfortunately not emphasized enough, is the vast importance of verbalizing our own intuition” (Goldratt, 1990).

Systems usually consist of many different parts. The key lies in recognition of the important role of the system's constraints. It's quite straightforward what could be considered as a constraint - anything that limits the system from achieving higher performance. Obviously, every system does have at least one constraint, making this philosophy **universally applicable**.

The path to improvement is divided into five steps:

1. Identify the system's constraint
2. Decide how to exploit the system's constraint
3. Subordinate everything else to the above decision
4. Elevate the system's constraints
5. If in the previous steps a constraint has been broken, go back to step 1.



*(Author)*

As Goldratt earlier suggests, these steps can oftentimes be followed by intuition and common sense.

### TOC Thinking Processes Tools

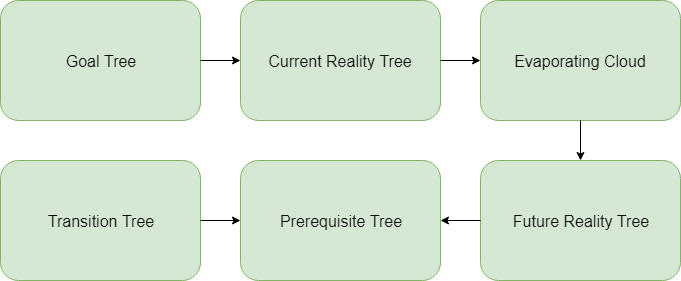
To find the root cause constraining the throughput, a collection of logical tools was established for easier analysis and picturing abstract constraints. According to Goldratt, TOC initially lacked „way which does not rely on examples or references but on the intrinsic logic of the situation itself, which is by far more convincing than the usual methods. This method of proof is called Effect-Cause-Effect and it is used extensively in all of the hard sciences.” (Goldratt, 1990). The effect-cause-effect approach was the cornerstone for creating the five basic Thinking Processes Tools (TP Tools):

* Current Reality Tree
* Evaporating Cloud
* Future Reality Tree
* Prerequisite Tree
* Transition Tree

Each of them is based on sufficient cause (if A exists then B exists) or necessary condition (in order for B to exist, A must exist), thus the adjective „logical”.

#### The Logical Thinking Process

The Logical Thinking Process (LTP) is a concept based on TOC, introduced by William Dettmer to provide system managers and executives effective technique for designing organizational strategy, planning its deployment, evaluating its effectiveness, and making corrections as needed in the shortest possible time (Dettmer, 2007). It can be divided into one to six steps, varying by necessity, choice or experience of the practitioners. The difference from the Thinking processes is fairly subtle. In addition to the five basic TP Tools, Dettmer added Goal Tree as the first step. Secondly, Dettmer considers all the tools as a single process, in contrast to the Goldratt's interpretation of each tree/cloud as a separate process.

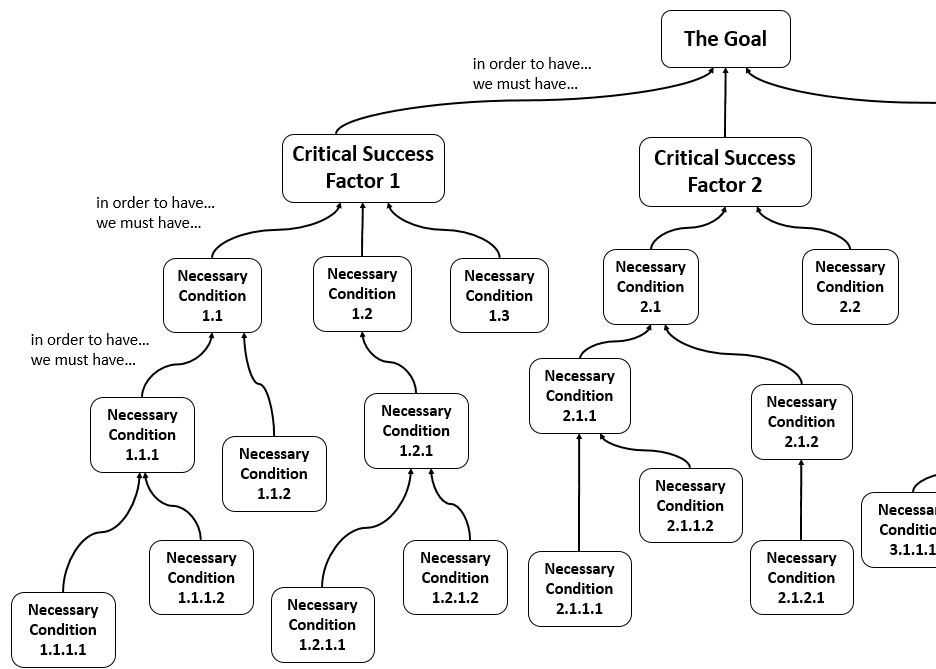


*(Author)*

Chris Hohmann, senior manager and author of few books on the topic of lean management, productivity techniques and others, explains LPT in more detail on his website.

***Goal Tree***

A Goal Tree is built on a necessity logic-based relationship. At the top is a single goal, dependent on, ideally, three to five critical success factors (CSF). More of these factors would indicate badly-stated goal making the whole effort much more likely to fail. Under each CSF are several necessary conditions (NC). The goal can only be accomplished after all NCs are met (Dettmer, 2007).



*(Hohmann, 2014)*

***Current Reality Tree***

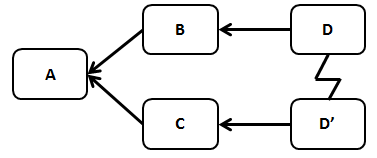
The Current Reality Tree (CRT) is the first of TP Tools. Its understanding is relatively intelligible and straightforward. As the name suggests, diagram depicts current state with focus on negative cause-and-effect relationships which end up obstructing the final goal(s) at the top. At the bottom lie root causes starting the chain reaction made from intermediary, typically undesirable effects (Hohnmann, 2014).

***Evaporating Cloud***

Next tool is the Evaporating Cloud (EC), also known as Conflict Resolution Diagram. According to Hohmann, it's based on two assumptions:

* Conflicts (opposition about objectives or opposite points of view, for instance) tend to be settled by compromise. Yet compromising requires making concessions that lead to a solution which isn’t satisfactory for neither side, hence a win-lose or lose-lose situation.
* Conflicts are often the result of false assumptions, beliefs or myths which constrain needlessly the organization. As two opposite things cannot be true at the same time, one is necessarily false. If the falseness can be debunked, the conflict disappears (evaporates) and a no-compromise, win-win solution is found

Typically, EC consists of five entities, where A represents the objective trying to be achieved. B and C are *both* necessary to reach it. D is a prerequisite to B, D' is a prerequisite to C, however D and D' cannot coexist simultaneously, therefore a conflict emerges. Cloud is then resolved by either invalidating one of the assumptions of the diagram or providing an injection which changes the situation (Hohmann, 2014).



*(Hohmann, 2014)*

***Future Reality Tree***

Future Reality Tree (FRT) visualizes the causality behind the desired change in the future, similarly to CRT, in cause-and-effect relationships. Undesirable effects from CRT are turned into desirable effects with initial injections and cascade reaction of mentioned cause-and-effect relationships, eventually resulting into the final goal. FRT represents could-be future, but doesn't necessarily show how to reach it (Hohmann, 2015).

***Prerequisite Tree***

Prerequisite Tree (PRT) is used to surface and overcome obstacles arising during achieving the intermediary goals. Related intermediary goals subsequently prove that these obstacles can indeed be neutralized. These goals form a graph of steps needed for meeting the final objective. PRT is the preparatory work for the coming implementation action plan as well as a useful communication tool and a means to overcome fear and/or resistance to change (Hohmann, 2015).

***Transition Tree***

Transition Tree (TT) is the last of both TP tools and LTP. It illustrates a step-by-step transition from the original to the desired state. Basically, the TT combines an entity of current reality, a statement of need and an action (injection) to create a new reality. This basic structure is repeated from the lowest or farthest condition to change up to the closest to the objective on top of the Transition Tree (Hohmann, 2015).

## ERP

### Introduction

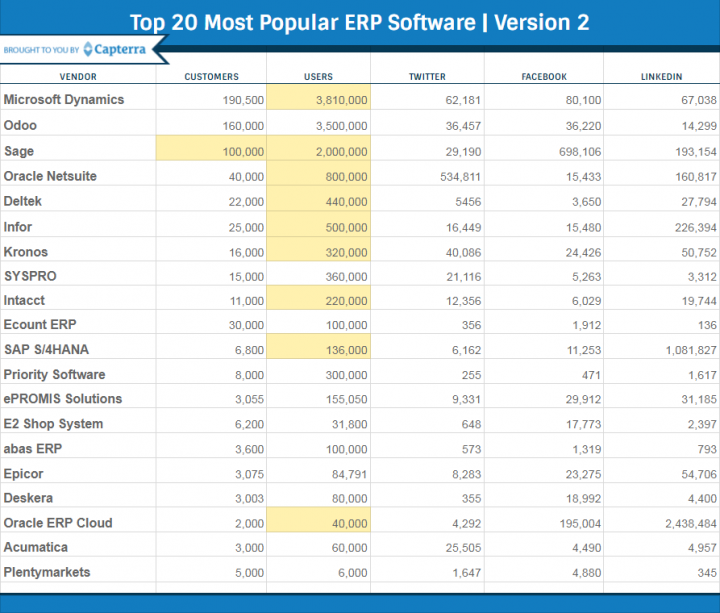
Acronym ERP refers to a business management software which covers day-to-day business activities including financials, supply chain, manufacturing, human resources, etc.

There are multiple perspectives how ERP can be looked at.

* a commodity, a product in a form of computer software
* a development objective of mapping all processes and data of an enterprise into a comprehensive integrative structure
* key element of an infrastructure that delivers a solution to business

(Klaus, Rosemann and Gable, 2000)

Roots of ERP date back to the discrete manufacturers in 1960s. Software engineers created programs to monitor raw materials and guide plant supervisors on what to order and when to replenish. Thenceforth, these programs underwent several evolution stages to ERP systems as they are known nowadays. The ERP market has grown into a billion-dollar industry with vendors targeting enterprises of all sizes from all industries (Bradford, 2015). Capterra did a research in January 2018, reaching out to software companies to collect their customer and user numbers. In case the company wasn't able to provide certain data points, Capterra generated estimates based on industry averages and additional research. Below are the results of this research (Capterra, 2018).



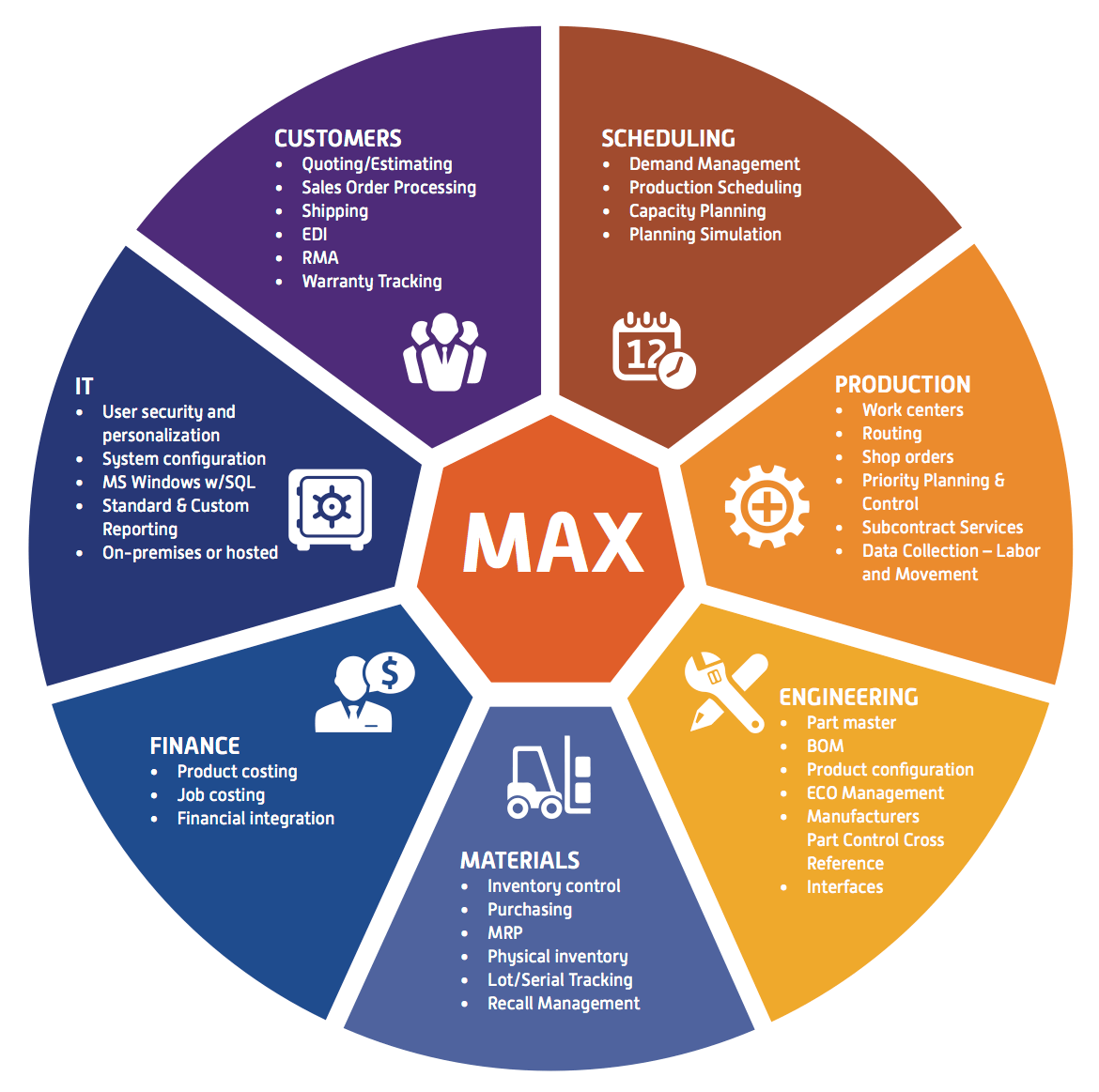
*(Capterra, 2018)*

### Purpose

Importance of software solutions has rapidly grown over the past decades. Nowadays, it's an absolute must for all competitive businesses. With an increasing number of various information systems, the issue of connecting these systems was becoming more and more severe. Oracle, one of the leading companies in the market of ERP software, define the purpose of ERP as to „tie together and define a plethora of business processes and enable the flow of data between them. By collecting an organization’s shared transactional data from multiple sources, ERP systems eliminate data duplication and provide data integrity with a single source of truth” (Oracle, 2018). Microsoft with its Microsoft Dynamics is possibly the number 1 supplier of ERP systems and seems to share Oracle's point of view „While there’s no all-up solution software for every business processes, ERP technology is getting better and better at bringing all your business processes together to improve collaboration, help your company make data-driven decisions, and advance business productivity” (Microsoft, 2018).

### MAX ERP

MAX is an ERP system from Exact, specialized for small and medium-sized manufacturers. As is standard for ERP systems, it provides tools for CRM (Customer Relationship Management) or finances, but its main focus is on tools associated with the production itself (Exact, 2018). According to MAX brochure, key manufacturing features include bill of material, inventory control, MRP (Material Requirements Planning), planning simulation, sales order processing, purchasing control, shop floor control, costing with multiple cost sets, Lot & Serial Tracking,, Warranty Tracking, Labor Tracking and Subcontract Processing (Exact, 2016).



*(Exact, 2018)*

## APS

### Introduction

For many companies, an efficient and effective supply chain can be a competitive advantage, and planning plays a crucial role in achieving this advantage. By utilizing capital-intensive resources, assigning the right skills and prioritizing customer orders, planning determines what operational performance a company will bring to its customers. However, planning has not received much attention in practice for a long time as humans responsible for this task are oftentimes not selected or trained explicitly for the job. Furthermore, the world of the planner is continuously changing, since plan made for today might not be valid for tomorrow anymore. Because planning tasks are complex, need to be performed under time pressure, and have a large impact on the operational performance of companies, a specialized type of decision support systems has emerged to support these tasks: Advanced Planning and Scheduling (Wiers and de Kok, 2017).

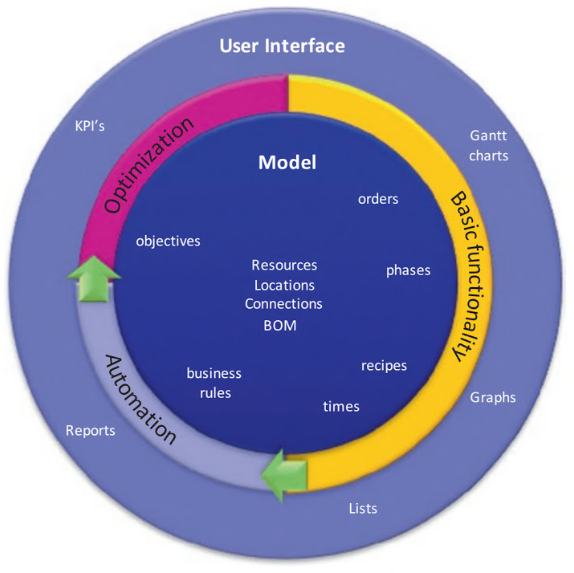
### Definition

It might be hard to distinguish APS from ERP systems, modules or even some spreadsheets which might also provide functionality to support planning or scheduling. APS systems are often deployed together with ERP systems and in that case, the distinction might be especially subtle as the two might potentially overlap in parts of functionality. Wiers and de Kok established a distinction applicable in most practical situations: „APS is an interactive planning tool, containing a *model* of a physical system, an *engine*, and an *interactive* Gantt chart”. They furthermore explained these three elements.

1. A *model* of a physical problem that needs to be planned or scheduled.
2. An engine that is able to immediately recalculate the consequences on the plan of planning actions, imported data, or other changes to the state.
3. A *graphical interactive user interface* (GUI) that depicts the consumption of resources and materials over time.

(Wiers and de Kok, 2017)

The picture below shows how these elements relate to each other.



*(Wiers and de Kok, 2017)*

It's particularly important to emphasize that initial focus should be on creating an accurate model of the real world, only then can outer circles work correctly.

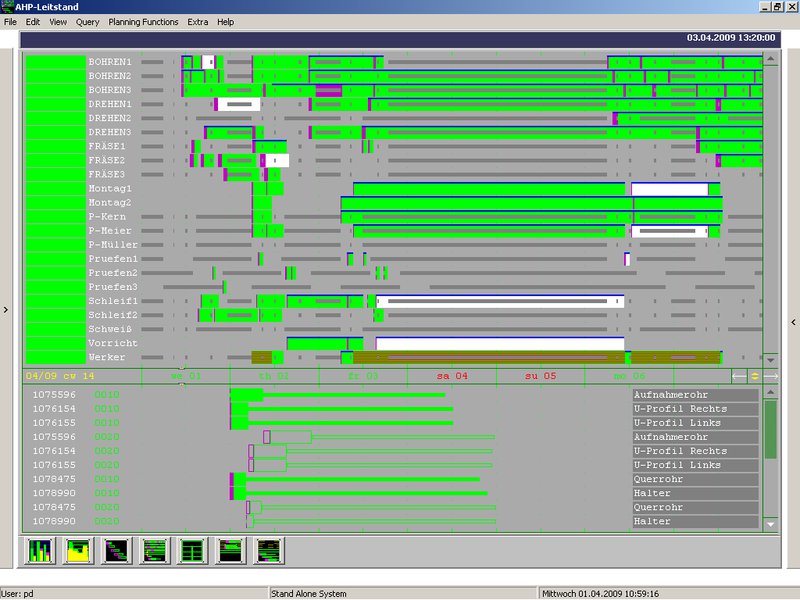
### AHP-Leitstand

This software was created in 1985 by German company Factory Solutions GmbH and is being redistributed by other companies worldwide, in this case by Czech company AXIOM PROVIS Int. s.r.o..

„The AHP-Leitstand handles resources and their allocation for production. The detailed representation of the cost locations in terms of constructions, machines and workstations shows the structure of the production. As special resources employees or tools could be included in the planning. The resources will be administrated and monitored at the AHP-Leitstand. This capacity offer is opposed to capacity demand.

Optimal sequences, maximum usage rate and the compliance to the delivery dates could be planned and monitored on the time axis. A reaction to short-term changes of the capacity or order reality could be done very fast and practically by a mouse-click. The actual planning and its progress will be visible at any time currently even over system limits” (Factory Solutions, 2018).

Below is an example of a planning screen in AHP-Leitstand.



*(Factory Solutions, 2018)*

## SW Engineering

### Definition

SW engineering is „a systematic approach to the analysis, design, assessment, implementation, test, maintenance and reengineering of software, that is, the application of engineering to software. In the software engineering approach, several models for the software life cycle are defined, and many methodologies for the definition and assessment of the different phases of a life-cycle model” (Laplante, 2001). However, if he had to give a one-word definition, it would be "modeling", if two-word, he would add "optimization". He sees modeling as a translation activity - software product concept is translated into requirements specifications, they are converted into a design, which is converted to code, further translated by compilers and assemblers to machine executable code.

Ian Sommerville prefers a much simpler definition „Software engineering is an engineering discipline that is concerned with all aspects of software production” (Sommerville, 2010). He further stresses that a good software isn't only about delivering functionality, but also maintainability, dependability and usability.

### Categories of SW products

Obviously, the purpose of SW engineering is delivering SW products. These can be divided into two basic categories:

* *Generic products* - Stand-alone systems that are produced by a development organization and distributed through the open market to any customer who buys them. The specification is therefore determined by the developing company. I.e. databases, office packages, drawing tools,...
* *Customized products* - Systems commissioned by a particular customer, tailored accordingly to his specifications. I.e. control systems for electronic devices, systems written to support a particular business process,...

However, over the course of past years, the distinction between these two types of products is becoming increasingly blurred. What's becoming modern is creating a core generic product, which is later further adapted to suit the requirements of the specific customer. This is also the case with many ERP systems (Sommerville, 2010).

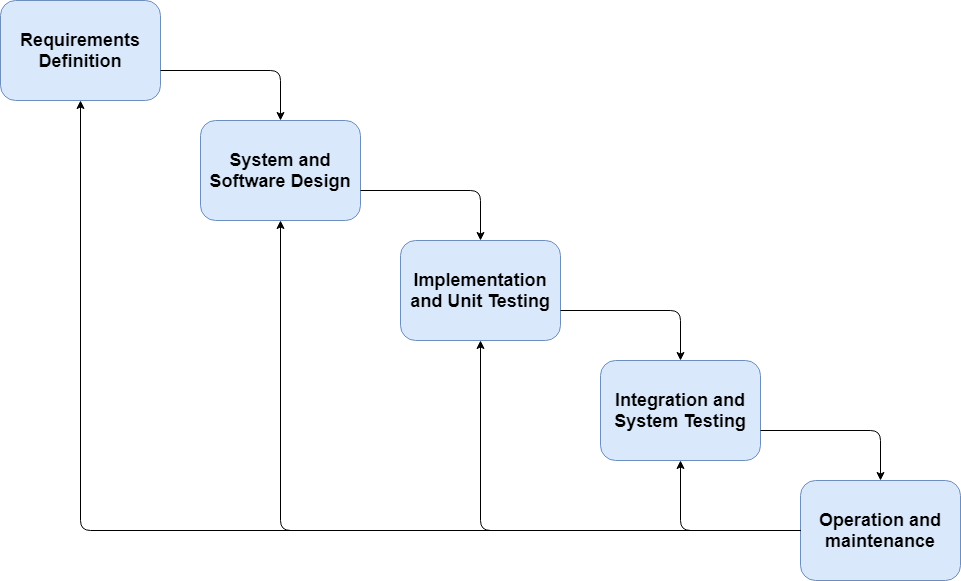
### SW development models

Sometimes also called SW development life cycle or SW processes, is a coherent series of phases for software production. There are multiple approaches how to develop a SW product, however, all must include these four activities:

* *Software specification* - The functionality of the software and constraints on its operation must be defined.
* *Software design and implementation* - The software to meet the specification must be produced.
* *Software validation* - The software must be validated to ensure that it does what the customer wants.
* *Software evolution* - The software must evolve to meet changing customer needs.

(Sommerville, 2010)

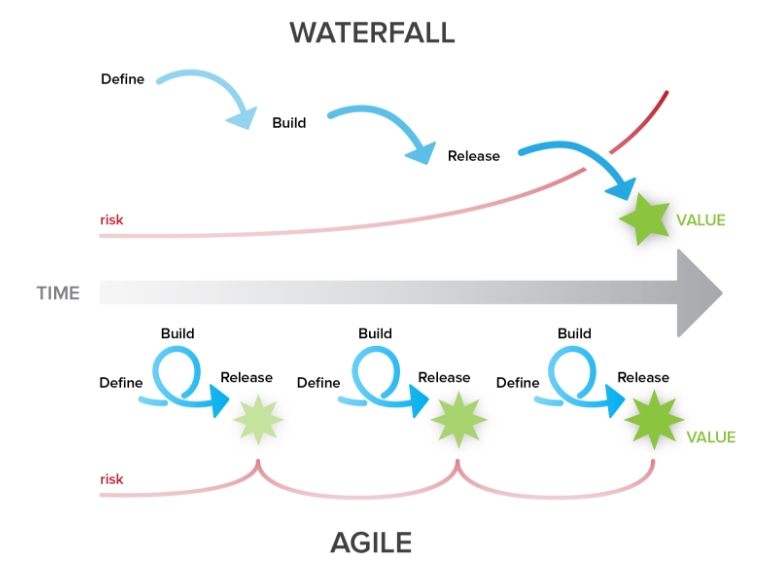
The first introduced SW development model was the **waterfall model**. Thanks to its straightforward nature, to this day, it's still fairly popular and considered as the conventional, linear and simple one. Each of the fundamental activities is done in its own, separate phase.



*(Author)*

Another popular technique is the **incremental model**. It's based on splitting the development process to increments, each providing new or improved part of the functionality, instead of the evolutionary approach of the waterfall model, thus allowing for parallel increments. In addition, the serial releases of the incremental model are planned whereas in the evolutionary model, each sequential release is a function of the experience from the previous iteration (Laplante, 2007).

As a reaction to the unpredictability of requirements, with efforts for earlier product deliveries came agile methods. They are based on incremental approach and are adaptive as opposed to rather predictive nature of rigid methodologies such as the waterfall model. This approach differs significantly from those models that emphasize planning the software in great detail over a long period of time and for which significant changes in the requirements and specifications can be problematic. Agile methods are a response to the common problem of constantly changing requirements (Laplante, 2007).

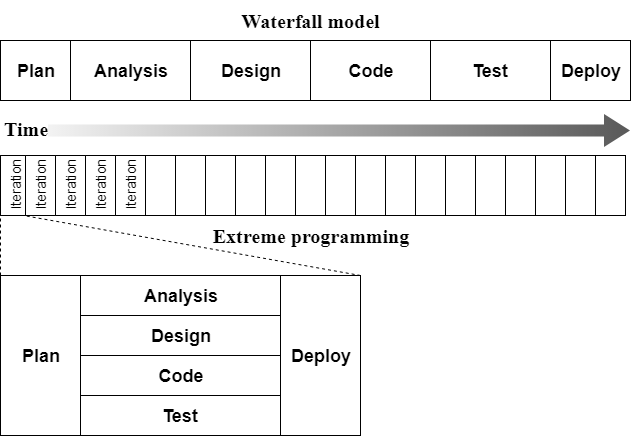


*(Cirdan Group, 2017)*

Models are not mutually exclusive and are commonly used together, especially for large systems development. Each model has its own pros and cons and it makes perfect sense, trying to combine the best features from each of them.

#### Extreme Programming

XP is one of the agile methodologies of SW development models introduced in 1999 by Kent Beck. Its original definition was „XP is a lightweight methodology for small-to-medium-sized teams developing software in the face of vague or rapidly changing requirements”, but Beck himself wrote in the second edition of *Extreme Programming Explained: Embrace Change* that „In the five years since the publication of the first edition teams have pushed XP much further than the original definition” (Beck, 2004). It took an iterative approach and pushed at the time recognized good practice to extreme levels. Customers are intimately involved in specifying and prioritizing system requirements. The requirements are not specified as lists of required system functions. Rather, the system customer is part of the development team and discusses scenarios with other team members. Together, they develop a ‘story card’ that encapsulates the customer needs. The development team then aims to implement that scenario in a future release of the software.



*(Author)*

### Unified Modeling Language

„The Unified Modeling Language is a graphical notation for drawing diagrams of software concepts. One can use it for drawing diagrams of a problem domain, a proposed software design, or an already completed software implementation” (Martin, 2002). Even though it can't be universally said that making and testing UML diagrams would be easier or cheaper to test than an actual source code (actually, not even if it could be tested at all), it does have some undisputable upsides:

* *Communicating with others* - UML is enormously convenient when it comes to communicating design concept between developers.
* *Backend documentation* - Even if the team assesses UML as unnecessary during earlier stages, it's always a good idea to document the backend at least through this channel and ease the job for the front end team.
* *What to keep, and What to throw away* - Many created diagrams won't be useful at all, however, some are very useful to save. There are diagrams that record complex protocols that are hard to see in the code, some provide roadmaps for areas of the system that aren’t touched very often, others are diagrams that record designer intent in a way that is better than code can express it. These should be easy to spot and recognize with common sense.

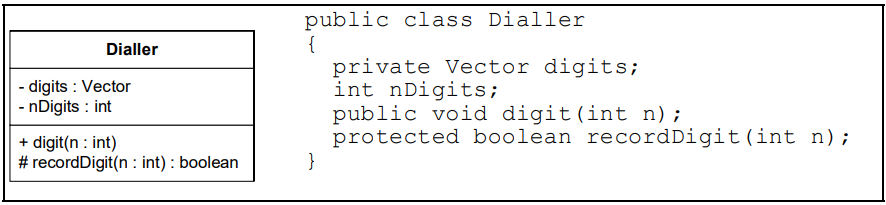
(Martin, 2002)

#### Use case diagram

Use cases are a very straightforward way to describe the behavior of the system. The description is inside ovals, written from the point of view of a user who initiated the system to do something particular. It captures the *visible* sequence of events that a system goes through in response to a *single* user stimulus (Martin, 2002).

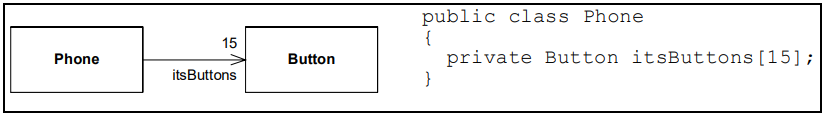
#### Class diagram

Class diagram allows us to denote the static contents of, and relationships between classes. They're especially useful for depicting all the source code dependencies. Class is represented by a rectangle subdivided into compartments with its name, variables and methods writing inside the respective compartments (example below).



*(Martin, 2002)*

The mentioned relationships between classes, such as association originating from instant variables holding references to other objects, are subsequently illustrated by arrows connecting the rectangles.

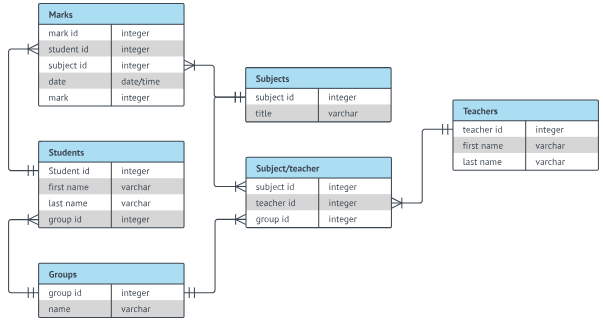
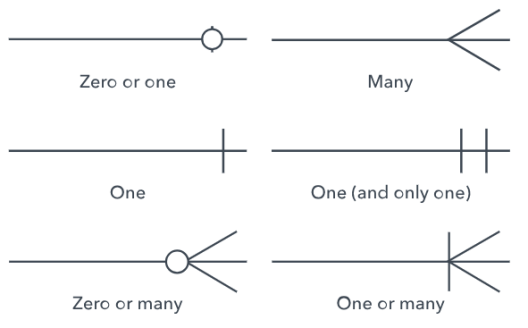


*(Martin, 2002)*

#### Entity relationship diagram

„An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases” (SmartDraw, 2018)

Entities/tables are represented by rectangles with their attributes/columns written inside. Explanation of cardinality of relationships between two entities/entity sets and an example of a simple ERD is illustrated below (Lucid Software, 2018)



*(Lucid Software, 2018) (Lucid Software, 2018)*

# PRACTICAL PART

## Mincovňa Kremnica, š.p.

### History

Mint was established in 1328 by King Charles Robert of Anjou and has been continuously producing mint articles ever since, making it one of the oldest manufacturers worldwide. Historically speaking, its relevancy was influenced strongly by nearby deposits of gold and silver as these two precious metals constituted the focal point of production. However, over the course of time, it was necessary to expand the area of expertise and, especially in past decades, adapt to the modern version of this very specific market (Mincovňa Kremnica, 2018).

### Current situation

The company currently has more than 200 employees and over the year 2017, its revenues reached over €23 million, ending the year in a profit of nearly 700 000€ (FinStat, 2018). The customers form a very diverse portfolio of requested work, ranging from small, relatively low-priced orders of trinkets and such, through limited but very expensive series of collector articles, to huge quantities of currency coins for countries all around the world, i.e. Guatemala or Sri Lanka.

Current production of the mint can be divided into these categories:

**artwork -** In case the design of the final product isn't given at all or its form isn't compatible, artists need to make it themselves. The result of this is only internal for further use as a template for manufacturing actual product.

**engraving** - Dies for all of the further metalworking operations are made here, making this fairly obvious potential bottleneck of the whole production.

**coining** - Typically production of circulating currency coins for governments in huge quantities for relatively long periods of time (weeks, sometimes even months).

**precious metals** - Typically working with gold, silver, platinum, etc. Could be considered as a logical part of the medal-making segment, but considering more sensitive nature and security risks of this work, it's represented as a separate work center in the ERP.

**stamp printing** - Excluding artwork, indifferent from the other parts of production as it doesn't have anything to do with metalworking at all.

**medal-making and others** - In a way opposite of coining - small quantities of many different products. Very diverse and more complicated than coining in regards of operations potentially needed for completion of the product. Includes plaques, tokens, badges, pendants, tie pins, labels, awards, etc.

## Research topic

The research topic at Kremnica Mint was focused on their imperfections in regards to production planning. This concerned only metalworking segments of productions, as other parts are either easy enough to plan without using APS (stamp printing) or just not suitable for attempting to include them in the APS given the bigger role of human factor and unpredictability of the work itself (artwork). So, in this context, it's safe to say that every product is manufactured accordingly to the assembly procedure. Operation represents one of multiple processes necessary, to assemble the whole product. Some products require only a few operations, others, more complex, even more than 100. In the context of this thesis, combination **order + operation** (in other words, specific operation of the specific order)represents a basic unit of production planning. For simplicity, I will call this conjunction only **operation**.

Currently, mint uses ERP MAX which tools for production planning weren't sufficient for the requirements, the main reason being lack of finite capacity planning. 8 years ago has been acquired APS software AHP Leitstand for this purpose.

### AHP-Leitstand

Although AHP has been refined continuously, latest release being version 8.0 for Windows 10 in 2018 (Factory Solutions, 2018), and the planning itself offers many options, naturally, it's obsolete at least in aspects such as interconnectivity with other systems or UI clarity. These were the biggest issues according to the IT department, master planner and workshop planners. What is very important to mention in this case is that AHP is purely planning software and as soon as the production of a certain product is finished, it disappears from the database. On the other hand, ERP system MAX only keeps track of the order descriptions (required quantity, required date of delivery, customer info, etc.), product properties (operations' work centers and remarks, product description, etc.) and, once production starts, manufactured quantities. However, its connection with AHP is only one way. The import scripts from MAX even had to be written by the mint's IT department, making the connection two-way, reportedly, wasn't possible. AHP imports orders, quantities, dates and other necessary information, but after the plan is created, it's not in any way communicated back to MAX or any other system. Because of this, there's no way to retrospectively find out whether some order was finished accordingly to the plan or not, thus making it impossible to draw any consequences.

## Definition of the previous state

The initiative which inspired this thesis came from the top management of the company, more specifically its director. Even though he, obviously, isn't directly involved in the production planning and organizing, the issues rooting in these activities were repeatedly resulting in bigger, more severe problems. In a better case „only” work inefficiency, which, apart from obvious fiscal reasons, in worse cases would result in late deliveries. These can be especially sensitive in this line of business - smaller orders such as plaques, tokens, badges, etc. are often ordered for a specific event and not being able to deliver them on time would be a big blunder. On the other hand, huge orders of currency coins from governments tend to have very strict contract conditions and violating them can get very expensive, not mentioning the reputation hit that would cause on the international scene.

In the past, information regarding production planning that AHP doesn't include was communicated through shared excel document in the company network, phone calls, messages or personal conversations. However, in a modern manufacturing company where lean manufacturing is one of the main goals, this system isn't effective at all. Not only is it too human-dependent and unnecessarily time-consuming, it also misses essential part for long-term improvement - hard data. With this system in place, upper management didn't have any data based on which it could initiate necessary changes.

### LTP

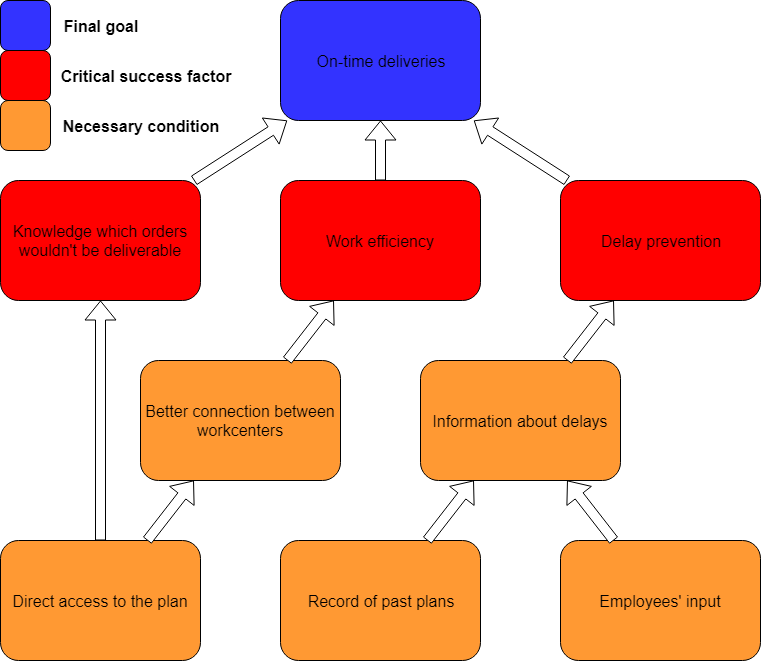
By applying Logical Thinking Process, issues mentioned before can be analyzed more deeply and presented in clear, understandable way. Additionally, some of them also represent a very simplified description of the integration of me and my work to the company. All the necessary information for creating these diagrams were collected through conversations with the director, employees of the IT department and the master planner. Second-handedly, they also interpreted to me input from the technological deputy, workshop planners and sales department.

#### Goal Tree

As shown in figure XX, the final goal is considered ensuring on-time deliveries. To consistently achieve this, three critical success factors have to be present:

1. The sales department has to have knowledge which orders wouldn't be deliverable in regard to the current state of production.
2. Work efficiency is, naturally, fundamental for any manufacturing company and therefore also a big factor for achieving the final goal.
3. Delays themselves are a very undesirable effect, however, they constitute also one of the main reasons for work inefficiency, frustration and make people more prone to further mistakes.

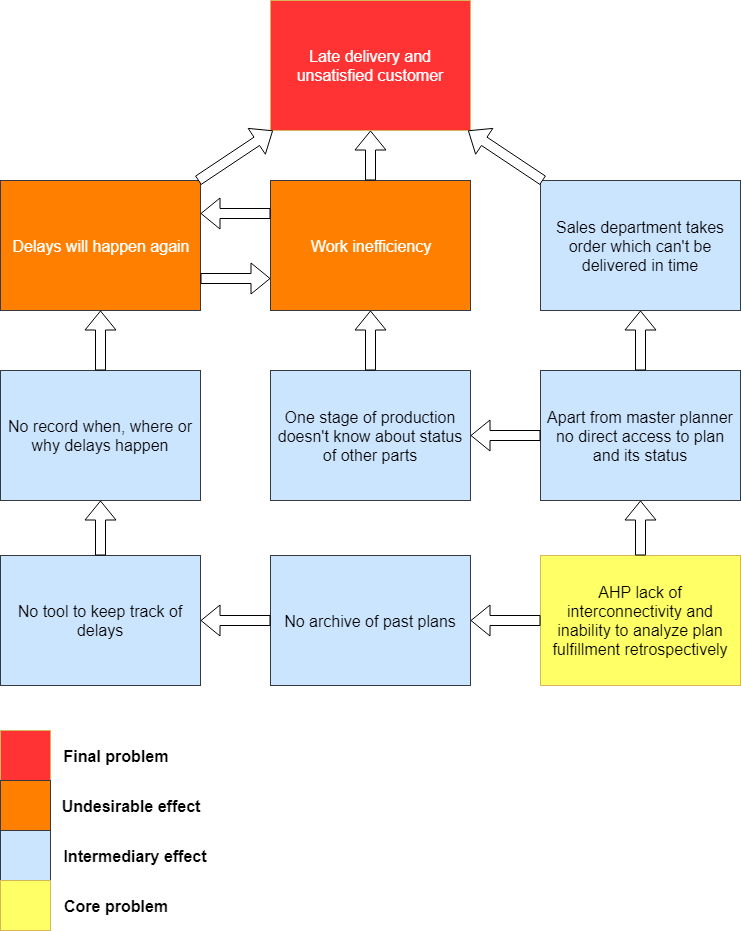
To bring these factors, couple of necessary conditions have to be met. To increase work efficiency, the information flow between work centers needs to be improved. This can be achieved by making the plan available directly, which would also prevent sales department from making undeliverable sales. For the last factor of the delay prevention, company first needs to know the nature of the delays and act upon this information. Gathering the details can be accomplished by union of having data about past plans and input from the involved employees.



*(Author)*

#### Current Reality Tree

As the tree shows (figure XX), mint considers late delivery and unsatisfied customer the final problem as these two usually imply various, more severe consequences. The root of problems can be summarized as AHP lack of interconnectivity and inability to analyze plan fulfillment retrospectively. This leads to having no archives of past plans, therefore, for example, it's not possible to determine if an operation in current plan is newly scheduled or it was for some reason rescheduled and is already in delay. Because of this, it's effectively impossible to keep track of delays, eventually leaving no record when, where, why or during the production of which product the delay occurred. Any of these pieces of information could be used to prevent the same situation from happening again, but without these data, delays will happen again. This is certainly considered a serious undesirable effect. The other branch of CRT illustrates implications of poor interconnectivity of AHP. After master planner creates a plan, usually for about a week ahead, others receive only its exported version. It's fairly obvious that even though initially plan is on point, at the first deviation from it, a chain reaction begins. For instance, in case of delay in the coining center, the medal-making center can already be waiting for engraving center to produce dies, while they are accordingly to their original plan manufacturing dies for coining center, which at the moment doesn't even need them. To find out about the whole situation would take at least a few calls. This approach just isn't in alignment with principles of lean manufacturing and causes another undesired effect in form of significant work inefficiency. And what could potentially be even worse is if sales department makes a sale which would be originally possible, but isn't in the current, real situation, thus possibly resulting into the final problem, late delivery. Additionally, the two undesired effects form a vicious circle where with more delays, deviations from plan grow and cause bigger work inefficiency, causing further delays and so on. Both of these are one of the most common reasons for the final problem, late deliveries.



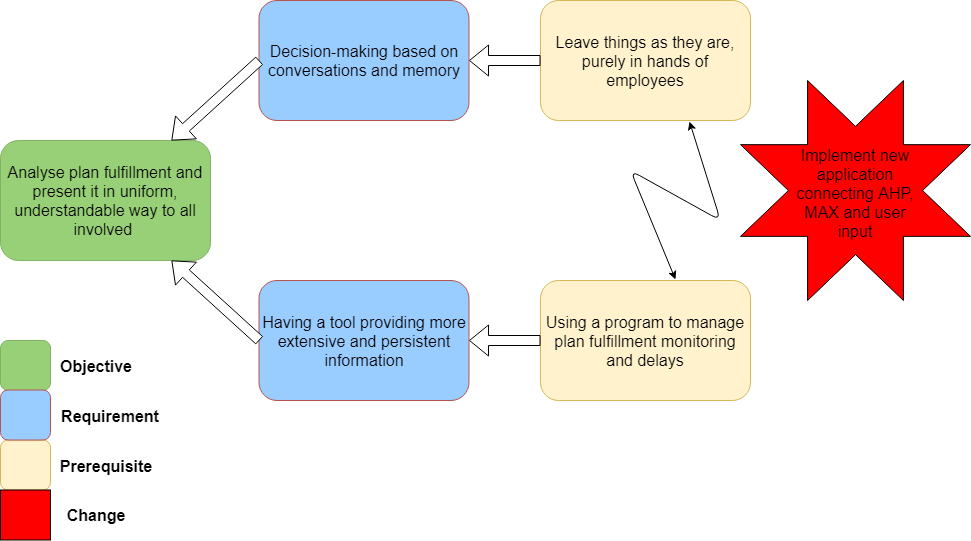
*(Author)*

#### Evaporating Cloud

During the analysis of the situation, it was agreed upon that it's necessary to add some kind of tool to compensate what AHP lacks. Given the most common feedback, that meant a way to present real-time plan fulfillment to a wider spectrum of users in an understandable fashion. After further discussion, it was determined that making a new, custom application should be a feasible and well-rounded solution for a few problems.

As pictured in figure XX, the previous state would rely purely on employees finding things out on their own initiative by contacting someone who assumably should have more information about the given matter. To meet the objective of plan fulfillment analysis and its presentation, employees would have to base their decisions solely on conversations and their memory.

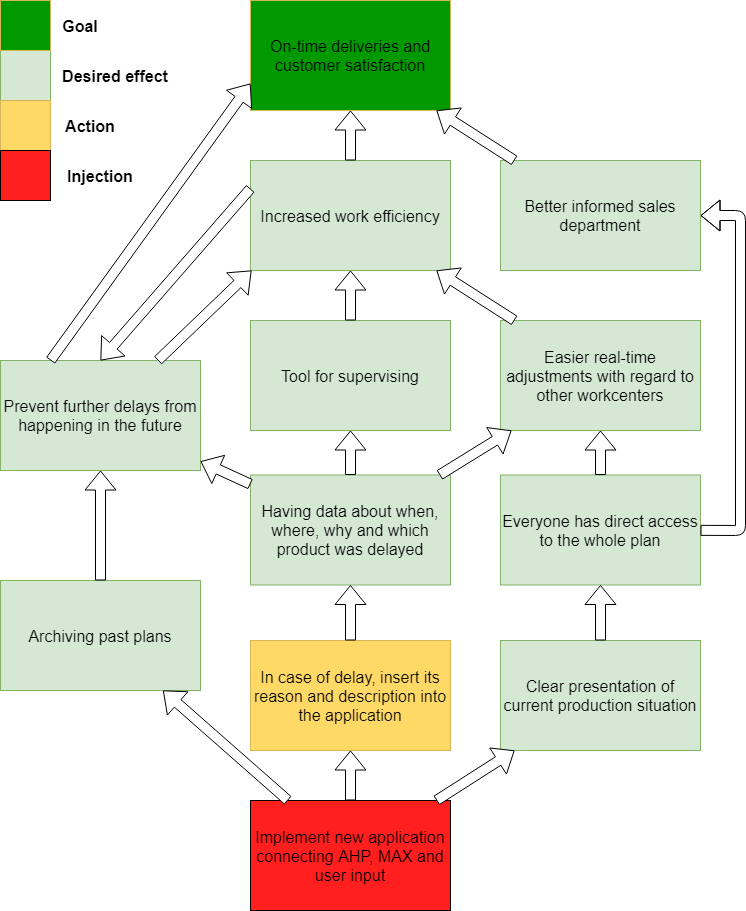
As opposed to that, using a new program for this purpose is definitely a preferable alternative. With the software assistance, decision-making can not only consider the actual current situation, but also take into account data collected throughout the past experiences. With this new tool, relevancy of the human factor would be greatly reduced thus making the final goal more systematically achievable.



*(Author)*

#### Future Reality Tree

As depicted in figure XX on the bottom is a single injection in form of a new application which would connect three pillars of production - AHP, MAX and users. Of course, the users need to provide the application with the details about the delay in case one occurs. Provided both of these conditions are met, the company will start gathering crucial data about nature of delays. This is the corner-stone of the whole process of improvement as it brings essential information for various purposes. With knowledge of the past plans, it opens the door towards diagnostics of the whole reasoning behind occurrences of the delays and with proper analysis, it should be possible to, at least in some cases, prevent the same causes from inflicting further damage. On the other hand, assuming the application is available to all parties involved, this data makes a lot easier real-time adjustments of production with the information needed for it being available instantly without having to waste time with contacting other people. Details about delays are also very valuable data even by itself, because they present another channel for supervising each part of the production. Additionally, availability of the plan is a welcomed feature for the sales department for taking the current situation in production into account. All of these desired effects are in perfect alignment with the principles of lean manufacturing and contribute to increased work efficiency. That furthermore helps to prevent more delays from happening in the future, forming a virtuous circle and together resulting in the final goal of on-time deliveries and customer satisfaction.



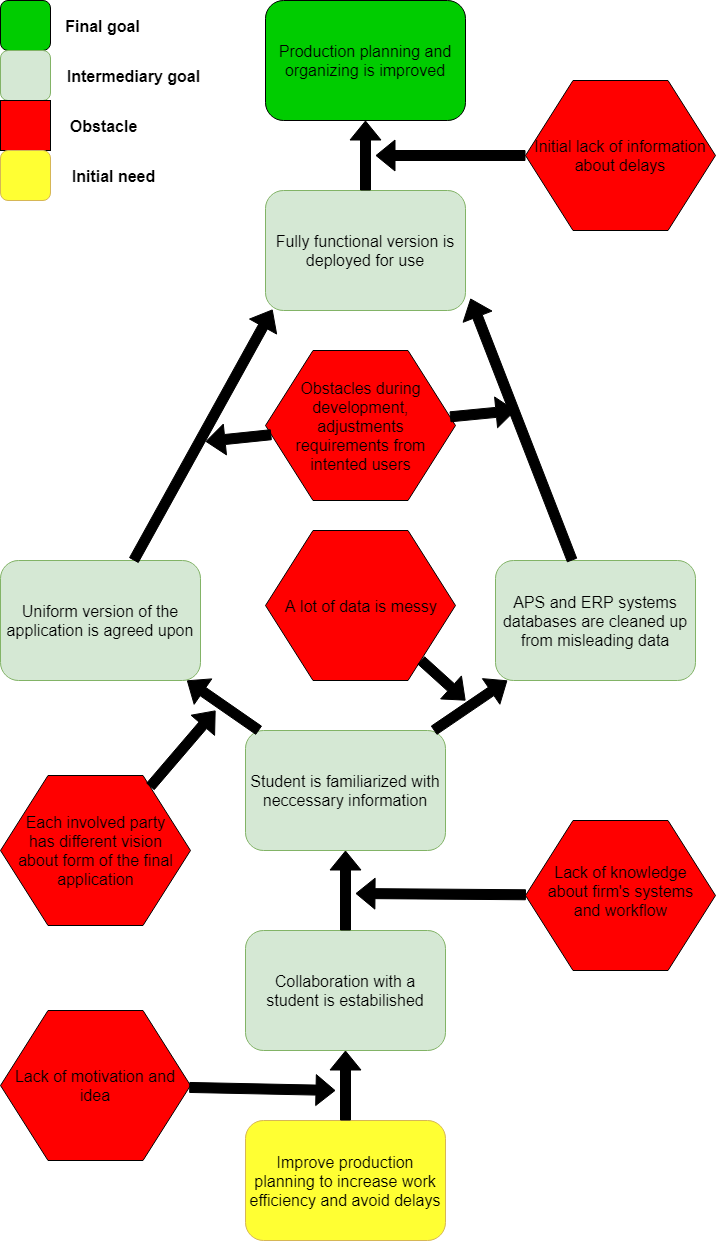
*(Author)*

#### Prerequisite Tree

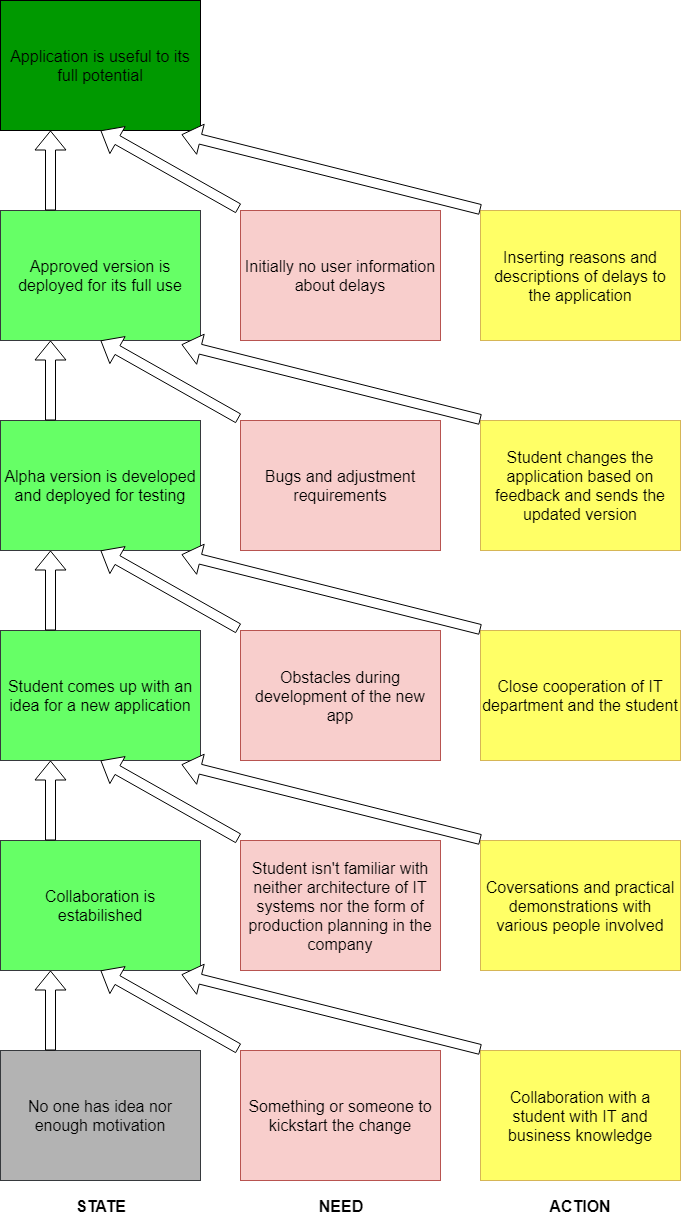
Ignition spark for initiating changes was the desire to improve work efficiency and avoid delays by changes in production planning and organizing. After initial, not so difficult, obstacles shown at the bottom part of PT were overcome and I was ready to begin working on the matter at hand, more serious ones occurred. On the human side of things, every involved party had a different point of view and therefore also a different vision of the application. After extensive conversations on this sensitive topic, a uniform version was approved. On the backend side, it was found that significant portion of MAX database is filled with bad data. I will further elaborate on this topic in latter chapter, but in cooperation with IT department, databases were prepared to suitable state for purposes of the application. Naturally, during any software development arise various issues and new ideas for minor modifications of the original design. After several iterations of debugging, adjustments and gathering feedback, the final version of the application was ready. In the early stage, its helpfulness is only limited by lack of user input about delays. However, throughout the time, its benefits should grow enough to use gathered data to draw consequences from any occurred delays thus achieve the final goal - improve planning and organization of the work.

#### Transition Tree

The final LTP tool is shown in figure XX. Original state was static for a relatively long time. It's part of human nature to rather stay content with the known, even though imperfect, system than try to initiate change. What was needed was a spark that would start the transformation. One way how to accomplish this was through collaboration with a student with both IT and business knowledge. After this step, it was needed to familiarize me with the situation and systems architecture. Few days of conversations, excursions and practical demonstrations successfully achieved this. With gained knowledge, I came up with an approximate design for the application, but came into contact with various obstacles during its development. Fortunately, with help from the IT department, it was feasible to create a prototype of the application. Next stage was debugging and consulting with the primary intended users. Based on their feedback, necessary adjustments were made to make the application in accordance with the desired functionality. After that was the case, it still lacked fundamental part needed for the final goal - user input. But over time that change and make the application useful to its full potential.



*(Author)*



*(Author)*

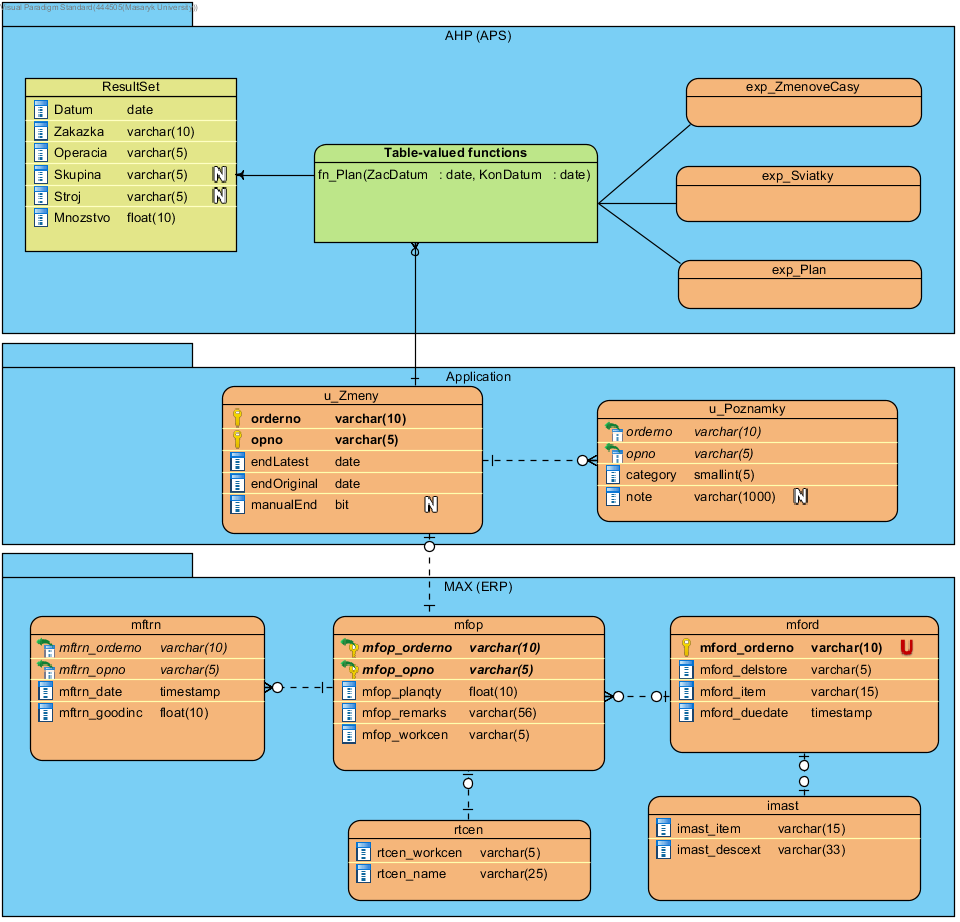
## Development and deployment of the application

The focus of this part of my thesis is on the events since discussions about the form of the application. Due to the fact that my expertise in programming, software engineering and databases was limited to several university courses mainly focused on Java, the original ambitions were fairly cautious. No one from the IT department was familiar with neither Java nor OOP (Object Oriented Programming) in general. They attempted to make a web application coded in PHP for this exact issue before, but eventually abandoned the efforts as they didn't seem to be anywhere close to the desired result. Therefore, it was clear from the beginning that if chosen to pursue the path of a desktop Java application, I would have to do all the programming by myself from scratch. They offered assistance with managing databases and, the most important part, application logic. No better alternative was suggested, so it was decided.

### Early stage of development

I received instructions on how to install database server and its requisites used in the company, Microsoft SQL Server 2013. With dump file, using Microsoft SQL Server Management Studio I restored databases with sample data on my own computer. At this point, just after brief browsing of new data, I started to notice a few cases, where the data didn't seem to make sense, looked redundant or just obvious mistakes. I addressed the IT department with this issue and after further inspection, they agreed that data are to some extent bad. As McCallum mentions, there's no precise definition for this term. Some people interpret it as a purely technical phenomenon like missing values, or malformed records. He, however, considers it as a more contextual term and calls it in short, plain language „data that gets in the way“. (Q. Ethan McCallum, 2012). In case of mint's data, this seems like a more fitting interpretation. To name just a few examples of bad data in this case: typos, duplicates, empty or nonsense values, old and redundant records, etc. Reason for this to happen was that with most of the data, they didn't work at all until this point. After noticing it, they were able to identify which of them are, indeed, wrong and sorted things out. Furthermore, they discovered and fixed one minor bug in import script which kept producing some of the bad data before.

The first necessary step was to implement a feature which would somehow enable to archive planned dates of completion on their first time of scheduling. If this information is stored, after a change of plan, operations that appear in both plans can potentially be identified as rescheduled and basically already in at least certain kind of delay. In accordance with this and other objectives, two new tables were created in the database. In figure XX you can see current Entity-Relationship diagram rid of tables and columns irrelevant for this application.



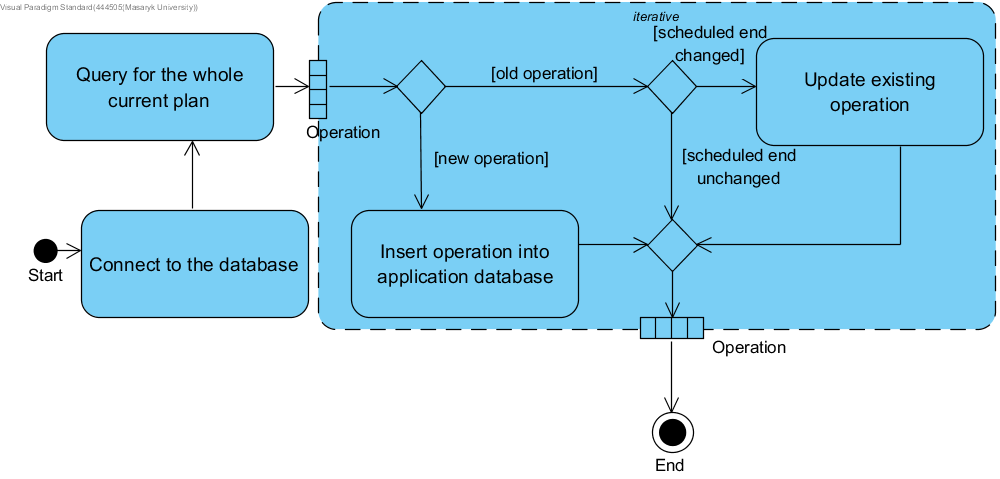
*(Author)*

As you can see, the application creates in the past missing bridge between AHP and MAX. On the AHP side, IT department programmed an SQL table-valued function which takes into account plan, holidays and shift times and returns a result with listed columns (see ResultSet). This function splits the plans into operations and further splits them to days. On the other side, MAX provides details about each operation such as description of the final product, work center where the operation is taking place or quantity already manufactured.

### Programming stage

When the databases were ready, the main part of development followed. The first logical step was to write SQL statements for earlier mentioned data archiving. In figure XX is depicted activity diagram of this process. After establishing a connection, the application queries for the whole plan from AHP. This result is compared with the current database of the application and one of three cases happens for every operation:

* if operation is new, it's simply inserted in its current state into the application database
* if operation already exists in both databases and its scheduled end has changed, this new date is updated in the endLatest column, while endOriginal remains the same
* operation's end date hasn't changed therefore nothing happens

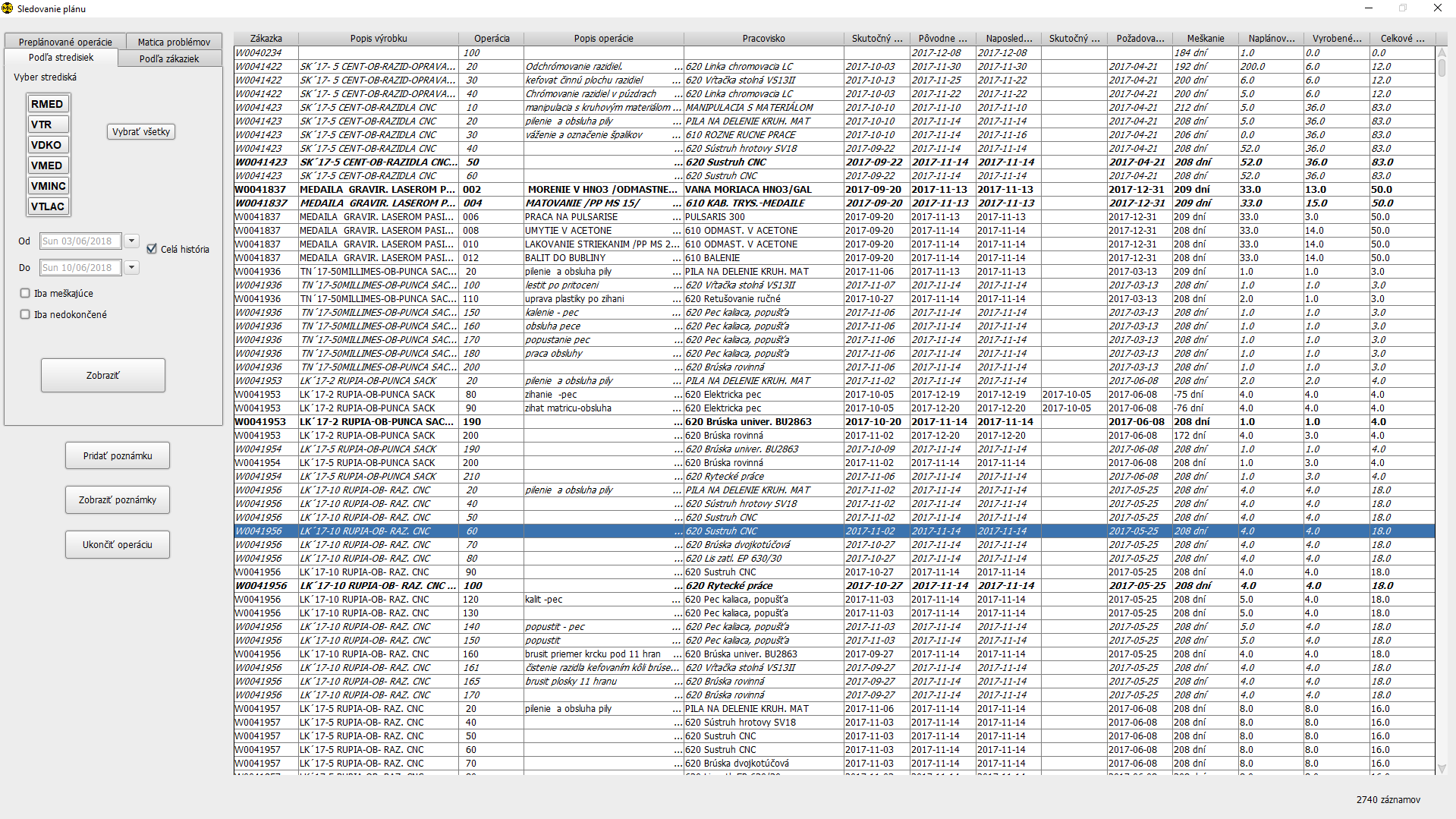


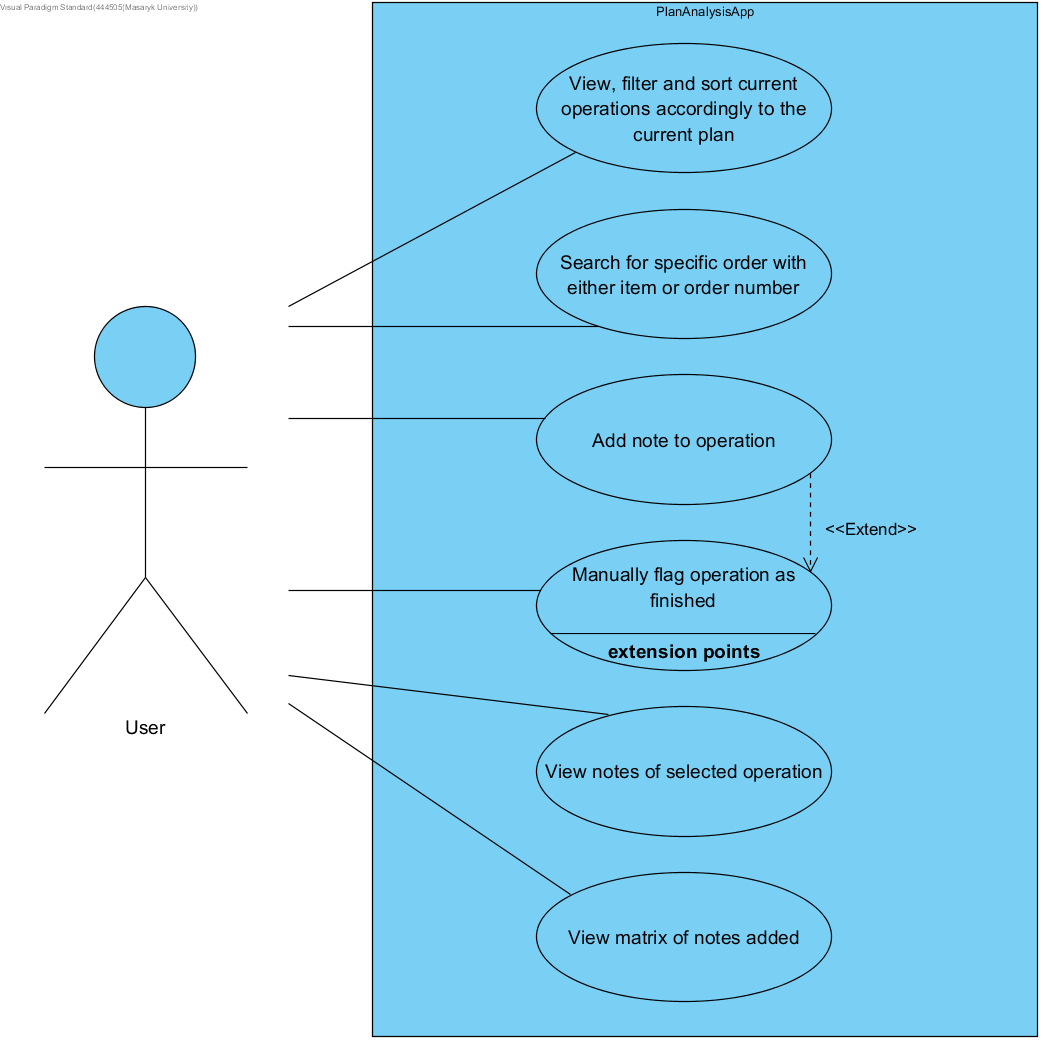
*(Author)*

With this purely backend mechanic in place, it was time to start building Graphical User Interface (GUI) and related backend. With regard to my capabilities, the fact that I was to be the sole programmer and low aesthetic requirements, I used Swing framework with couple of open-source libraries to improve clarity, controllability and looks. For easier debugging especially remotely in latter stages of development, logging is practically a necessity. I used popular log4j's more modern successor, Logback framework. When building a UI, I had to keep in mind multi-threading and keep any time-consuming operations such as SQL queries in separate SwingWorker threads. Another important aspect while working with the database was data redundancy. In order to minimize it, SQL queries consisted of relatively many JOIN statements (see appendix 1). Initially, I only cooperated with the director and master planner as testers in practice. After multiple iterations of adjustments and getting feedback, the application reached desired quality and functionality and was subsequently introduced to other users. After minor adjustments based on input from the sales department, the application reached satisfactory functionality for all involved parties, thus pausing the programming stage until any further findings.

### Full version of the application and its demonstration

This is a screenshot of the basic screen in the final application and use case diagram to describe its functionality.

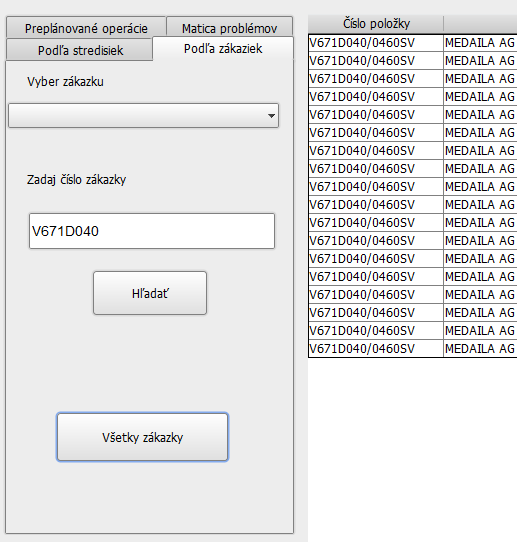
*(Author)*



*(Author)*

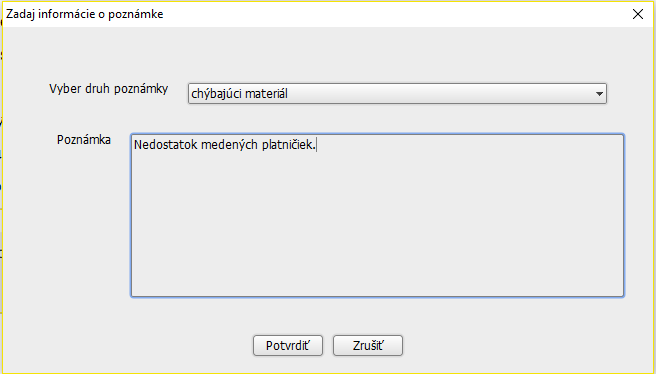
To further elaborate on the functionality currently provided by the application, it's best to describe the particular use cases from figure XX.

1. View, filter and sort operations is the most basic tool for the widest spectrum of users. It's possible to filter operations based on work centers, dates and the fact whether they are delayed or completed. Sorting works intuitively by clicking on the table headers (see screenshot).
2. Searching for a specific order was specially requested by the sales department. User can either browse the combobox filled with all orders or type at least the beginning of its number into textfield.



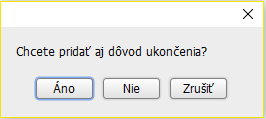
*(Author)*

1. Adding information about delays was mentioned in this thesis multiple times. In the application, it's represented by notes. With operation selected in the table, after clicking the button to add a note, dialog window pops out and user can select a category and describe the problem.



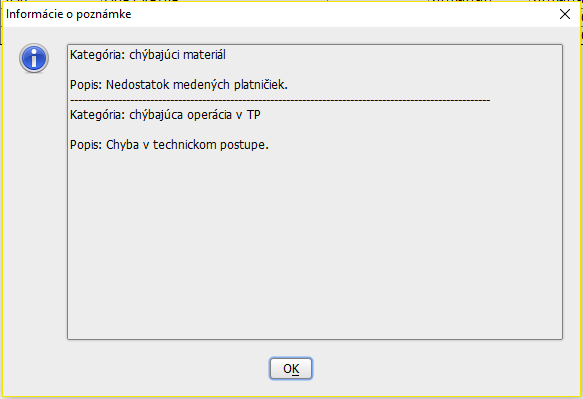
*(Author)*

1. By default, operations are flagged as finished automatically after quantity produced >= quantity scheduled. If for some reason this doesn't happen, but the user wants the operation to be considered as finished anyway, he can use another button. He is then offered an option to also add a note to describe why is the operation being ended manually. This operation will later be shown in the table in *italics* for better clarity.



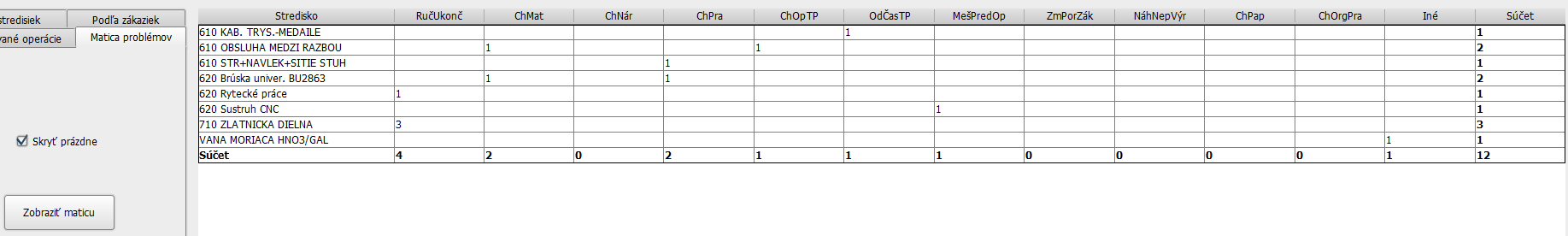
*(Author)*

1. Logical follow-up for adding notes is the option to view them. Operations with notes are indicated by **bold** rows in the table and after clicking the button for showing the notes, following dialog window will pop up listing all of the notes for the selected operation.



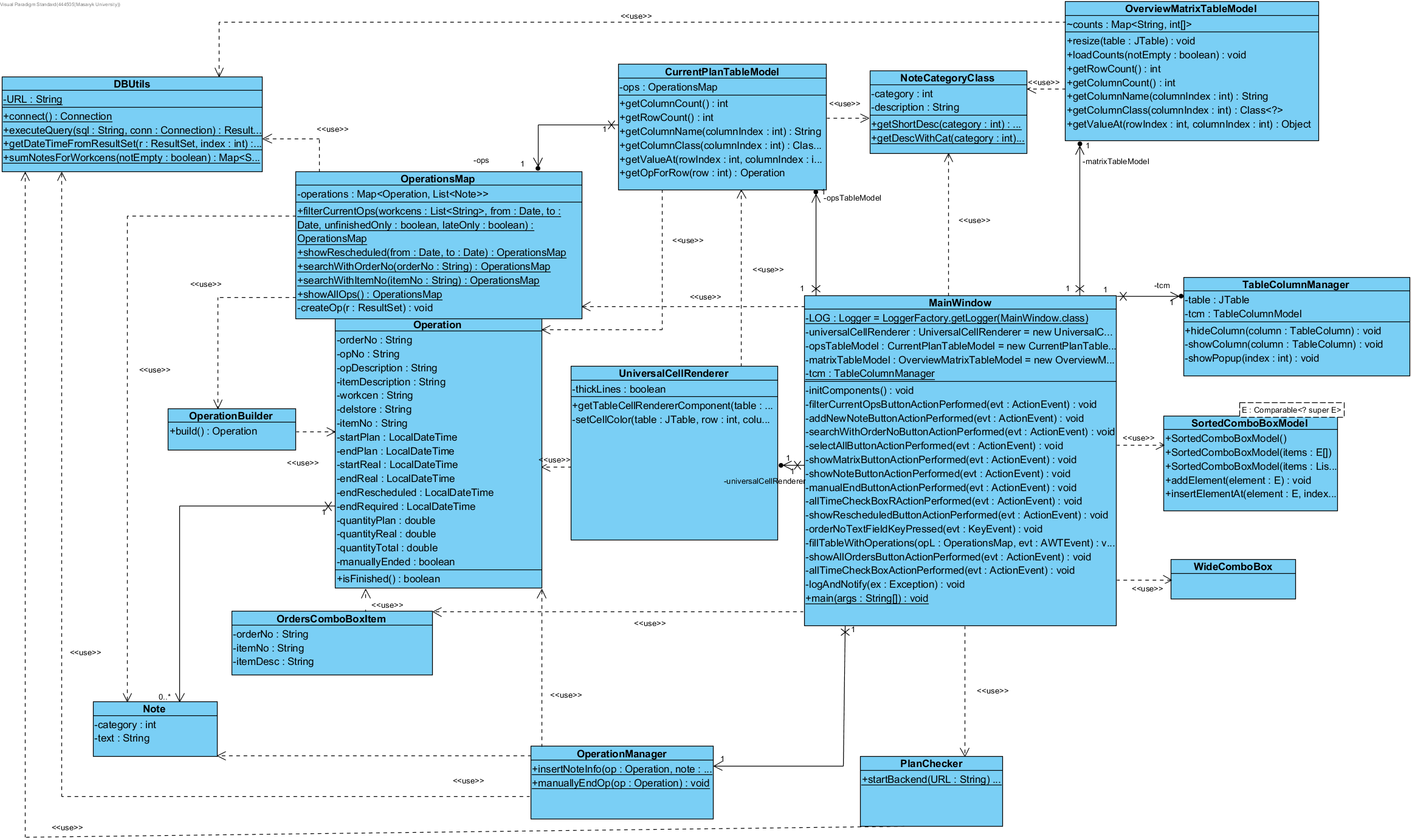
*(Author)*

1. Last part of functionality was requested by the management. Table changes from operations to a matrix where rows represent workshops and columns represent categories of delays. With an increasing number of delays uploaded, repeating mistakes will make themselves clear and easy to spot here.



*(Author)*

And finally, a simplified class diagram of the application.



*(Author)*

This is the current version of the application. All involved parties in the mint including master planner, workshop planners, sales department and top management have expressed their satisfaction with it and right now have no further suggestions for any modifications.

## Impact of the application

-TODO-

# Conclusion

In the theoretical part, I laid foundations of various thinking methods, tools and approaches used in each of the stages necessary to eventually provide the solution. In regard to the research topic, that included Lean thinking and TOC concepts. To elucidate the information systems involved in this thesis, following parts were dedicated to ERP and APS systems. Considering solution provided in this case, the last chapter of the theoretical part introduced SW Engineering, in detail the Extreme programming development model, and lastly UML as a tool to document the application.

Practical part initially unfolded Kremnica mint's past and current situation. In the next part, the emphasis shifted to the matters concerning the research topic. Logical follow-up came in form of analysis itself, using the LTP introduced in the theoretical part. Based on the analysis, a solution was suggested in form of an application. The process of implementing it is gradually described in the next part. Finally, application's impact concludes the practical part.

The first established research question sounded „*Is APS or its integration a cause of the imperfections in the production process and if so, in what manner?* “. As described in the *Definition of the previous state*, it was determined that APS is indeed the source of issues during the production process, more specifically its interconnectivity not only with other systems but also a wider spectrum of users. Therefore, the second question „*If the first question is answered positively, what is the suggested solution?*“ needed to be answered. The consensus was reached on the suggested solution of implementing a new, custom-built application which would supplement AHP and MAX systems

Purpose of this thesis was to analyze the situation, identify the problem and propose, ideally also carry out, the suggested solution. All parts of this goal, including the last, in the beginning uncertain one, were achieved. Results of this thesis were in alignment with its purpose and company's interests. Although it's impossible to quantify the improvement from the previous state as there are no data from that period of time, involved parties agree that the application exceeded their expectations from collaboration with me. The final program is documented using Javadoc and UML diagrams, in compliance with to me known conventions of Java programming and with implemented logging, therefore any potential changes by other programmers should be facilitated at least to some extent. Apart from coding the application itself, I would say my contribution was as an objective, impartial mediator between all of the involved parties. When I first came into the company, it was instantly obvious that each employee has a different opinion and after years spent in the job, they lost the ability to see things from a different point of view. After hearing every angle, I tried to suggest a compromise which was further tailored throughout the discussions until consensus was reached. Being able to see things from not only the real-life perspective of planners and management but also from the technical standpoint as a programmer was absolutely essential for this solution.

# Bibliography

1. Wiers, Vincent C. S., de Kok, A. (Ton) G., 2017. *Designing, selecting, implementing and using aps systems,* Springer; 1st ed., 2018, ISBN 978-3319650548
2. Goldratt, Eliyahu M. and Jeff Cox., 1984. *The goal: a process of ongoing improvement*., North River Press, 2004. ISBN 978-0884271789
3. Goldratt, Eliyahu M., 1990. *What is this thing called Theory of Constraints and how should it be implemented?* North River Press, 1990. ISBN 978-0884271666
4. Dettmer, William H., 2007. *The Logical Thinking Process: A Systems Approach to Complex Problem Solving (2nd Edition),* Amer Society for Quality, 2007, ISBN 978-0873897235
5. https://hohmannchris.wordpress.com/2014/11/06/thinking-processes-current-reality-tree/
6. https://hohmannchris.wordpress.com/2014/11/17/conflict-resolution-diagram-evaporating-cloud/
7. https://hohmannchris.wordpress.com/2015/04/06/thinking-processes-future-reality-tree/
8. https://hohmannchris.wordpress.com/2015/07/11/thinking-processes-prerequisite-tree/
9. https://hohmannchris.wordpress.com/2015/07/13/thinking-processes-transition-tree/
10. https://www.lean.org/WhatsLean/
11. https://hohmannchris.wordpress.com/2015/09/08/muda-muri-mura/
12. https://www.lean.org/WhatsLean/History.cfm
13. https://www.sciencedirect.com/science/article/pii/S0272696306000313?via%3Dihub
14. https://www.lean.org/WhatsLean/Principles.cfm
15. https://link.springer.com/content/pdf/10.1023%2FA%3A1026543906354.pdf
16. Bradford, Marianne, 2015. *Modern ERP: Select, Implement, and Use Today's Advanced Business Systems,* lulu.com, 2015, ISBN 978-1312665989
17. https://blog.capterra.com/most-popular-erp-software/
18. https://www.oracle.com/applications/erp/what-is-erp.html
19. https://dynamics.microsoft.com/en-us/erp/what-is-erp/
20. https://www.max4erp.com/max-erp
21. http://cdn2.hubspot.net/hubfs/142655/Literature/Brochure/MAX\_\_Brochure\_2016.pdf?t=1528818013675
22. Factory Solutions, 2018. AHP-Leitstand Product Description [online] . cited [2018-06-08]. http://www.factory-solutions.com/english/ahp-leitstand/description/
23. Laplante, Phillip A., 2001, *Dictionary of Computer Science, Engineering, and Technology*, CRC Press, 2001. ISBN 0-849326915
24. Sommerville, Ian., 2010, *Software Engineering (9th Edition)*, Pearson, 2010. ISBN 978-0137035151
25. Laplante, Phillip A., 2007, *What Every Engineer Should Know about Software Engineering*, CRC Press, 2007. ISBN 978-0849372285
26. Beck, Kent, 2004, *Extreme Programming Explained: Embrace Change (2nd Edition)*, Addison-Wesley, 2004. ISBN 978-0321278654
27. Martin, Robert C., 2002, *UML for Java Programmers*, Prentice Hall, 2003. ISBN 978-0131428485
28. https://www.smartdraw.com/entity-relationship-diagram/
29. https://www.lucidchart.com/pages/er-diagrams
30. https://finstat.sk/00010448
31. Mincovňa Kremnica, 2018. History [online]. cited [2018-06-08]. https://www.mint.sk/onas\_historia.php
32. Q. Ethan McCallum, 2012. *Bad Data Handbook: Cleaning Up The Data So You Can Get Back To Work.* O'Reilly Media, Inc., 2012. ISBN 978-1-449-32188-8

# List of figures

-TODO-

# List of images

-TODO-

https://blog.capterra.com/wp-content/uploads/2017/01/ERP-Pop-Chart-V2-720x613.png

# list of abbreviations

**Abbreviation Explanation**

APS Advanced Planning and Scheduling

CRT Current Reality Tree

EC Evaporating Cloud

ERP Enterprise Resource Planning

FRT Future Reality Tree

GUI Graphical User Interface

IT Information Technology

LTP Logical Thinking Process

OOP Object Oriented Programming

SQL Structured Query Language

SW Software

TOC Theory of Constraints

TT Transition Tree

UML Unified Modeling Language

XP Extreme Programming

# list of appendices

Appendix 1 SELECT part of SQL queries

SELECT A.orderno,   
       A.opno,   
       A.endoriginal,   
       A.endlatest,   
       A.manualend,   
       B.mfop\_planqty,   
       B.rtcen\_name,   
       B.imast\_descext,   
       B.mfop\_remarks,   
       B.mford\_delstore,   
       B.mford\_item,   
       B.mford\_duedate   
       C.futureplanned,   
       D.pastplanned,   
       E.category,   
       E.note,   
       F.goodinc,   
       F.startreal,   
       F.endreal,   
FROM   (SELECT orderno,   
               opno,   
               endlatest,   
               endoriginal,   
               manualend   
        FROM   ahp.dbo.u\_zmeny) AS A   
       LEFT JOIN (SELECT imast\_descext,   
                         mfop\_remarks,   
                         mford\_delstore,   
                         mfop\_orderno,   
                         mfop\_opno,   
                         mfop\_planqty,   
                         rtcen\_name,   
                         mford\_item,   
                         mford\_duedate   
                  FROM   max2ostr.maxmast.mfop   
                         JOIN max2ostr.maxmast.rtcen   
                           ON mfop\_workcen = rtcen\_workcen   
                         JOIN max2ostr.maxmast.mford   
                           ON mford\_orderno = mfop\_orderno   
                         JOIN max2ostr.maxmast.imast   
                           ON imast\_item = mford\_item   
                  GROUP  BY mfop\_orderno,   
                            mfop\_opno,   
                            imast\_descext,   
                            mfop\_remarks,   
                            mford\_delstore,   
                            rtcen\_name,   
                            mford\_item,   
                            mfop\_planqty,   
                            mford\_duedate) AS B   
              ON A.orderno = B.mfop\_orderno   
                 AND A.opno = B.mfop\_opno   
       LEFT JOIN (SELECT zakazka,   
                         operacia,   
                         *Sum*(mnozstvo) AS futurePlanned   
                  FROM   ahp.dbo.**Fn\_plan**(CURRENT\_TIMESTAMP, '2200-01-01')   
                  GROUP  BY zakazka,   
                            operacia) AS C   
              ON C.zakazka = A.orderno   
                 AND C.operacia = A.opno   
       LEFT JOIN (SELECT zakazka,   
                         operacia,   
                         *Sum*(mnozstvo) AS pastPlanned   
                  FROM   ahp.dbo.**Fn\_plan**('2017-01-01',   
                         *Dateadd*(day, -1, CURRENT\_TIMESTAMP))   
                  GROUP  BY zakazka,   
                            operacia) AS D   
              ON D.zakazka = A.orderno   
                 AND D.operacia = A.opno   
       LEFT JOIN ahp.dbo.u\_poznamky AS E   
              ON A.orderno = E.orderno   
                 AND A.opno = E.opno   
       LEFT JOIN (SELECT mftrn\_orderno,   
                         mftrn\_opno,   
                         *Min*(mftrn\_date)    AS startreal,   
                         *Max*(mftrn\_date)    AS endreal,   
                         *Sum*(mftrn\_goodinc) AS goodinc   
                  FROM   (SELECT mftrn\_orderno,   
                                 mftrn\_opno,   
                                 mftrn\_date,   
                                 mftrn\_goodinc   
                          FROM   max2ostr.maxmast.mftrn   
                          UNION ALL   
                          SELECT uvop\_orderno,   
                                 uvop\_opno,   
                                 uvop\_datestrt,   
                                 uvop\_qtygood   
                          FROM   max2ostr.maxmast.uvop   
                          WHERE  uvop\_status < 2) AS odpisane   
                  GROUP  BY mftrn\_orderno,   
                            mftrn\_opno) AS F   
              ON A.orderno = F.mftrn\_orderno   
                 AND A.opno = F.mftrn\_opno