Mendelova univerzita v Brně
Provozně ekonomická fakulta

Rozšíření firemního informačního systému elektronické výroby o komponentu implementující Ganttův diagram

Diplomová práce

Vedoucí práce:

doc. Ing. Oldřich Trenz, Ph.D. **Bc. Tomáš Vencálek**

Brno 2016

Poděkování

Tímto bych chtěl poděkovat všem, kteří mě při tvorbě diplomové práce podporovali – své rodině, přítelkyni a přátelům. Děkuji vedoucímu této diplomové práce panu doc. Ing. Oldřichu Trenzovi, Ph.D. a společnosti COMPAS automatizace, spol. s r. o. za odborné rady, připomínky a ochotu podílet se na tvorbě diplomové práce.

Čestné prohlášení

Prohlašuji, že jsem tuto práci: **Rozšíření firemního informačního systému elektronické výroby o komponentu implementující Ganttův diagram** vypracoval/a samostatně a veškeré použité prameny a informace jsou uvedeny v seznamu použité literatury. Souhlasím, aby moje práce byla zveřejněna v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů, a v souladu s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*

Jsem si vědom/a, že se na moji práci vztahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzavření licenční smlouvy a užití této práce jako školního díla podle § 60 odst. 1 Autorského zákona.

Dále se zavazuji, že před sepsáním licenční smlouvy o využití díla jinou osobou (subjektem) si vyžádám písemné stanovisko univerzity o tom, že předmětná licenční smlouva není v rozporu s oprávněnými zájmy univerzity, a zavazuji se uhradit případný příspěvek na úhradu nákladů spojených se vznikem díla, a to až do jejich skutečné výše.

V Brně dne 22. května 2016 _____

Abstract

Vencálek, T. *The extension of the company electronic manufacture information system about a component implementing Gantt chart.* Thesis. Brno: Mendel's University in Brno, 2016.
Thesis is focused on an extension of the companys information system about a component rendering the Gantt chart. The reader will be introduced to the issue of creating an extension of the company information system. It includes an analysis of technologies which are suitable for a creation of a draft and a constitutive implementaiton of a solution. The practical part will consist of a draft and a creation of the component which provides a render of the internal data through the Gannt chart, and its application within the company information system COMES.

Keywords

Thesis, information system, extension, component, Gantt chart.

Abstrakt

Vencálek, T. *Rozšíření firemního informačního systému elektronické výroby o komponentu implementující Ganttův diagram*. Diplomová práce. Brno: Mendelova univerzita v Brně, 2016.
Práce se zabývá rozšířením firemního informačního systému o komponentu vykreslující Ganttův diagram. Čtenář bude seznámen s problematikou tvorby rozšíření firemního informačního systému. Dojde k analýze technologií vhodných pro tvorbu návrhu a dílčí implementaci řešení. Praktickým výstupem práce bude návrh a tvorba komponenty, umožňující vykreslení interních dat formou Ganttova diagramu a její nasazení v rámci firemního informačního systému COMES.

Klíčová slova

Závěrečná práce, informační systém, rozšíření, komponenta, Ganttův diagram.

Obsah

Seznam obrázků

1 Úvod a cíl práce

1.1 Úvod

Lidé si usnadňují práci všemožnými způsoby, použitím dostupných technologií, jejich zlepšováním a vývojem technologií nových. Nejprve to byli nástroje uzpůsobené pro určitý typ práce, průmyslová revoluce přinesla první stroje a v současné době počítačů je využíváno především informačních a komunikačních technologií, za účelem dosažení alespoň částečné automatizace procesů.

Výpočetní technika je spojena se všemi odvětvími lidské činnosti, najít obor zejména průmyslový, kde v současné době není využito počítačů je téměř nemožné. ICT umožňují sběr a přenos dat, provádění operací nad nimi v reálném čase a možnost algoritmizace procesů. Správná funkčnost však závisí na propojení hardwarových komponent a použití vhodného software, tato kombinace například umožní odeslat email na druhý konec, nebo vypočítat analytický úkol ve zlomku času.

Podniky a firmy jsou organizovány v řídicích strukturách, které usnadňují dosažení cílů efektivním způsobem. Dnes už nezbytným pomocníkem pro dosažení podnikových cílů jsou informační systémy, umožňující propojení jednotlivých organizačních složek firmy. *„Informační systém je soubor lidí, technických prostředků a metod, zabezpečujících sběr, přenos, zpracování, uchování dat, za účelem prezentace informací pro potřeby uživatelů činných v systémech řízení“*. (Molnár, 2009)

Dodavatelé se snaží vytvářet univerzální IS, za účelem možnosti použít svůj produkt u více zákazníků. Z důvodů individuálních požadavků každého zákazníka a případného přehodnocení či kladení nových požadavků v průběhu času je vytvoření univerzálního systému velmi nákladné. IS tedy prochází evolucí, která reaguje na tyto změny úpravou své funkčnosti. Rozšíření IS může být realizováno přidáním či změnou programových komponent, nebo propojením s dalším systémem, který řeší požadovanou funkcionalitu.

Konkurenční boj v průmyslové sféře nutí firmy neustále zlepšovat interní procesy. Pro dosažení vyšší efektivnosti pracovních postupů dochází k organizovanému řízení. Činnost zvaná plánování podnikových zdrojů, poskytuje výstup pro podporu řízení projektu tak aby splnil požadavky na něj kladené a dosáhl svých cílů v čase, v nákladech i potřebné kvalitě. Jedním z důležitých nástrojů plánování je Ganttův diagram, který umožňuje grafické znázornění plánovaných činností a jejich posloupností v čase.

Zahrnutím Ganttova diagramu do informačního systému, jako nástroje umožňující plánování bude umožněno provádět plánování akcí uživatelům systému bez nutnosti používání dalších nástrojů. Samotný nástroj umožňuje široké využití i při grafické prezentaci údajů.

1.2 Cíl práce

Cílem práce je analýza problematiky nasazení Ganttova diagramu v rámci firemního informačního systému, návrh a dílčí implementace tohoto řešení. Účelem je vytvořit komponentu rozšiřující funkcionalitu stávajícího informačního systému COMES o interaktivní prvek dostupný všem uživatelům stávajícího systému online. Vytvořená komponenta bude řešit problematiku zobrazení dat, bez povědomí o jejich struktuře, ve formě Ganttova diagramu.

Při řešení úlohy dojde k seznámení s problematikou rozšíření informačního systému, tvorbou komponenty pro jeho rozšíření a jejího samotného nasazení v současném systému. Na základě zadavatelem kladených požadavků a informací získaných studiem stávající technologie dojde k návrhu vhodného řešení. Důležité je seznámení se s vybranými technologiemi, vhodnými pro použití při tvorbě návrhu a implementaci řešení. Nasazením vytvořené komponenty do systému COMES a následnou aplikací řešení na demonstrační příklad dojde k ověření funkčnosti. Na závěr proběhne celkové zhodnocení výsledků práce a poznatků při řešení problematiky.

1.3 Struktura práce

Práce je rozdělena do následujících kapitol:

- Teoretická část, ve které bude čtenář seznámen s teoretickými předpoklady ke splnění práce.
- Metodika práce, tato část bude obsahovat analýzu požadavků a uplatnění poznatků získaných v teoretické části při výběru vhodných technologií a sestavení postupu při implementaci.
- Praktická část, se zabývá samotným návrhem a implementací komponenty.
- Závěr práce poskytuje prostor pro zhodnocení řešení a shrnutí hlavních poznatků diplomové práce.

2 Teoretická část

Na počátku je nutné vysvětlit několik základních pojmů, které budou provázet celou prací. Některé pojmy mohou být již známé, zejména čtenářům z informatických oborů. Nicméně je vhodné ujednotit základní pojmy za účelem předejití různým neshodám a uvést do problematiky i případné laiky.

Znalost problematiky informačních systémů, Ganttova diagramu jako nástroje pro plánování, a rozšiřovaného systému COMES je důležitá pro zahájení prací na tvorbě návrhu řešení.

2.1 Informační systémy

Každý podnik či organizace využívá ke svému provozu informační systém. Za účelem zefektivnění poskytovaných služeb dochází v současné době k širokému využívání informačních a komunikačních technologií při budování těchto systémů.

Jako uživatelé se s různými IS setkáváme téměř každý den v běžném životě i v práci. Příkladem mohou být IS používané ve veřejném sektoru (zdravotnictví, školství, státní správa, …), ale i v soukromém sektoru jako služby zákazníkům (rezervování pokoje v hotelu, online nákup, …) nebo jako nástroje pro podporu interních procesů v organizaci. Podnikové informační systémy jsou nejrozšířenější skupinou IS, proto je pozornost zaměřena primárně na tuto skupinu. (Koch, 2010)

Jednotná definice pro IS neexistuje, každý autor používá definici vlastní, neboť je na systém nahlíženo z mnoha různých pohledů, nicméně význam je obdobný. Informační systém je definován dle Molnára (2009) jako „*soubor lidí, technických prostředků* (hardware a software) *a metodiky práce, zabezpečujících sběr, přenos, zpracování a uchování dat, za účelem prezentace informací pro potřeby uživatelů činných v systémech řízení*“.

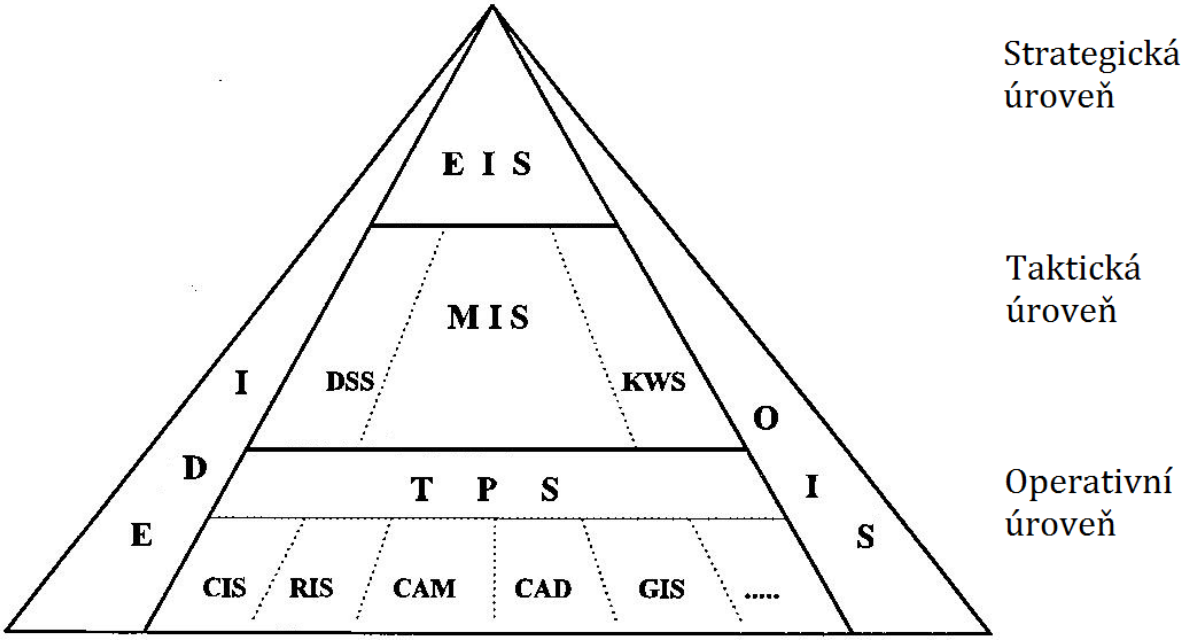
S postupem času můžeme očekávat stále vetší rozšířenost IS. Jejich význam pro firmy bude nadále narůstat, i přestože jsou již nyní nepostradatelnou součástí firemního ICT. Potřeba automatizovaného zpracování narůstajícího množství dat, složitost sběru, společně s vývojem v oblasti IT vede k neustálému vývoji v oblasti IS, které se stávají propracovanější, a také dochází k hlubší provázanosti s firemní infrastrukturou a podnikovými procesy. (Voříšek, 2007; Koch, 2010)

2.1.1 Data vs. informace

Poskytování informací je důležitou vlastností IS, jako *informace* označujeme data, kterým je přiřazen konkrétní význam, a která jsou schopna uspokojit informační potřebu svého příjemce. Za účelem získání požadovaných informací proudí do informačního systému velké množství dat vznikajících v rámci činnosti firmy nebo organizace. *Daty* nazýváme údaje popisující různé jevy nebo vlastnosti pozorovaných objektů. Úlohou IS je získaná data skladovat za účelem dalšího zpracování, zejména použitím *matematických* a *informatických* metod k získání dalších informací s přidanou hodnotou – *znalostí*. Expertní systémy (znalostní systémy) jsou schopny na základě znalostí provést důležité rozhodnutí, nebo danou znalost srozumitelně vysvětlit. (Cejpek, 2005)

2.1.2 Klasifikace IS

Z hlediska řízení můžeme strukturu organizace rozdělit na následující úrovně: *strategická*, *manažérská* a *operační*. Při výkonu firemních procesů vzniká jednotlivým vrstvám řízení potřeba po informacích, které vznikají během činností v rámci celé firmy. Informační systémy automaticky sbírají důležitá data, provedou jejich zpracování a okamžitě poskytnou informace tam, kde jsou potřeba. (Sodomka, 2006)



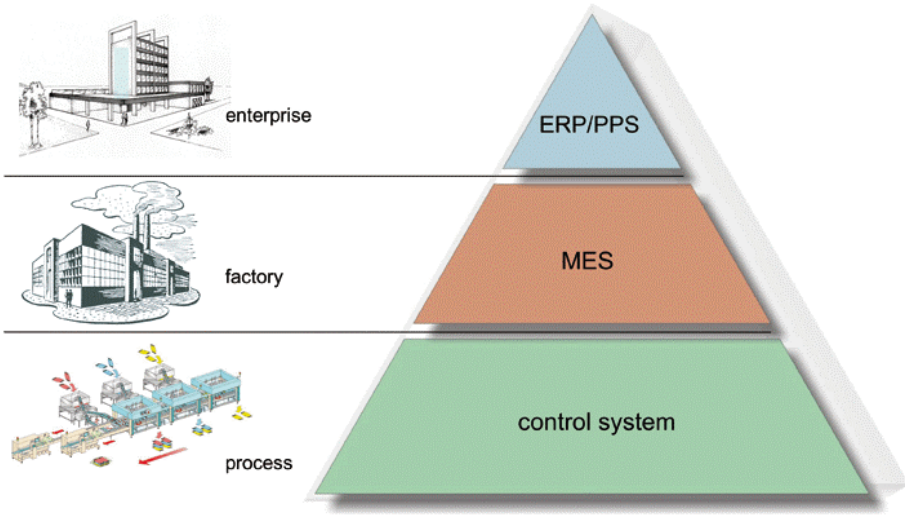
Obr. 1 Pyramidové zobrazení druhů IS vzhledem k organizační struktuře firmy (Rábová, 2006)

Z pohledu úrovní řízení firmy rozdělujeme IS následujícím způsobem podle Kocha (2010):

- **EIS** (Executive Information System) – Informační systém pro vrcholové řízení umožňují přístup k externím datům a agregují podnikové informace do nejvyšší úrovně.
- **MIS** (Management Information System) – Informační systémy pro řízení usnadňují práci řídícím pracovníkům. Zejména hodnocení výkonnosti na základě poskytnutých matematických či analytických přehledů.
- **DSS** (Decision Support Systems) – Systémy pro podporu rozhodování, které většinou pomocí analýz dat z MIS přinášejí podporu při rozhodování na taktické nebo strategické úrovni.
- **ES** (Expert System) **–** Expertní systémy mají za úkol uchovávat znalosti expertů, v případě ztráty zaměstnance a poskytovat podporu při hledání řešení.
- **TPS** (Transaction Processing System) – Transakční systém provádí automatizaci běžných operací (skladová evidence, účetnictví, docházka, …).
- **ERP** (Enterprise Resource Planing) – Systémy plánování a řízení firemních zdrojů integrují klíčové oblasti podnikání firmy (logistika, výroba, finance, …).
- **OIS** (Office IS) – automatizace kanceláře, využívá se zde elektronických dokumentů a pošty.
- **EDI** (Electronic Data Interchange) – Elektronická výměna dat slouží ke komunikaci s okolím podniku, kam patří zákazníci, banky, dodavatelé a další…

2.1.3 Podnikové IS

Cílem moderních organizací je *automatizace* firemních procesů. Tento trend vyžaduje použití moderních technologií a metodik. Vstříc těmto požadavkům vycházejí integrované systémy **ERP** pokrývající hlavní podnikové procesy. Tyto systémy právem označujeme jako *srdce firmy*, neboť pokrývají pokud možno všechny procesy ve firmě, zejména finance, logistiku, výrobu, prodej a lidské zdroje. (Koch, 2010)



Obr. 2 Provázanost systémů **ERP**, **MES** a **MCS** v rámci podniku (MES[1], 2011)

Dělení systémů je velmi podobné jako na Obr. 1. Vrstva **MCS** (Manufacturing Process Control System), která je tvořena stroji, řídicími jednotkami a lidmi, zajišťuje řízení výrobní technologie. Data vznikající na této úrovni jsou sbírána, monitorována a přenášena do vyšších úrovní systému za účelem dalšího zpracování. Na úrovni **MES** probíhá zpracování výrobních dat za účelem přeměny v informace, které jsou poskytovány příslušným pracovníkům pro okamžité řízení výrobních procesů. Vrstva MES tvoří důležitý spojovací prvek mezi automatizovanými systémy a nejvyšší úrovní informačního systému. Operativní informace je třeba integrovat i v nejvyšší vrstvě celopodnikového systému **ERP**. (Koch, 2010)

Elektronické řízení výroby je účinným nástrojem, pro dosaženo optimálního řízení výroby složitých šaržových i sériových zakázek. Umožňuje efektivně plánovat zdroje, řídit výrobu, optimalizovat kvalitu, produktivitu a náklady výroby díky poskytování přesných informací v reálném čase. Díky těmto informacím je možné efektivní rozdělování úkolů včetně přímého řízení a vyhodnocování výrobních procesů. Výrobní proces se stává závislý na funkčnosti integrovaného informačního systému, jehož nefunkčností může dojít k zastavení výroby. (Compas Automatizace, 2016)

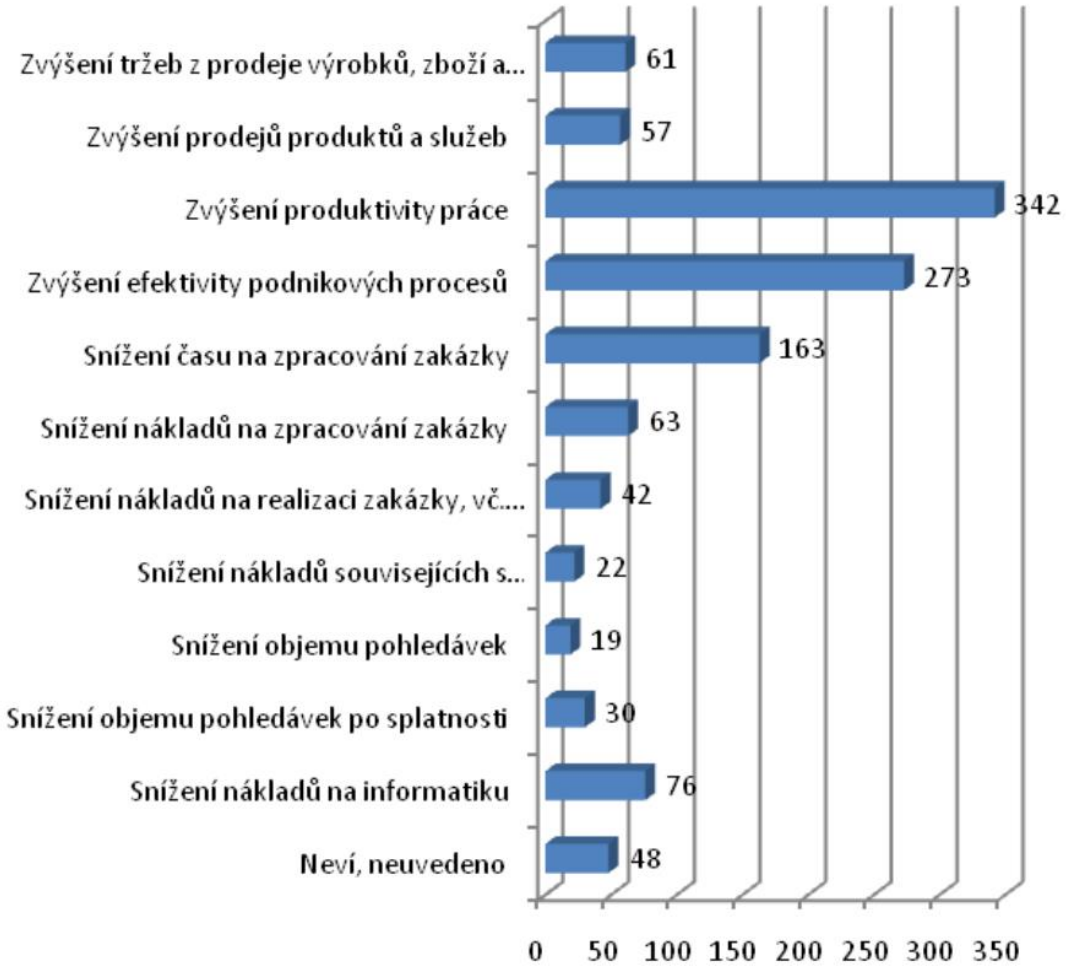
Vývojem aplikací v segmentu se zabývá řada renomovaných společností, mezi které patří Oracle, Microsoft, SAP, Helios svými řešeními oslovují zejména velké společnosti. V rámci velké poptávky je na trhu k dispozici množství produktů od dalších hráčů. (Sodomka, 2006)

[1] Manufacturing Execution System

2.1.4 Motivace zavádění IS

Moderní firmy si jsou vědomy *důležitostí informací*, které vznikají během provádění činností a jsou potřeba v rámci různých firemních procesů. Hospodářské prostředí je dynamického charakteru, neustálý konkurenční boj, změny v legislativních podmínkách či poptávka zákazníků vyžadují pružné reakce. Informace se v současném hospodářském prostředí jednou z nejcennějších komodit. (Voříšek, 2007)

Úkolem informačního systému je pokrýt informační potřeby jednotlivých úseků firmy, podporovat hlavní a podpůrné procesy firmy za účelem jejich neustálého zlepšování. IS obstarává tok dokumentů a informací v rámci celopodnikové informační strategie, automatizuje pracovní postupy, díky integraci sady organizačních pravidel vynucuje jejich dodržování. (Koch, 2010)



Obr. 3 Anketa hodnocení dosažených efektů zavedením IS/ICT respondenty (SYSTÉMOVÁ INTEGRACE, 2011)

Vlastnictví kvalitního IS je nástrojem pro zefektivnění firemních procesů, zlepšení hospodaření a řízení firmy. Jednotlivé přínosy vlastnění systému jsou vyjmenovány dále, jejich dosažení však závisí na konkrétním případě a rozsahu použitého systému. (Šmíd, nedatováno)

- Zlepšení a zrychlení komunikace se zákazníky a dodavateli.
- Zkrácení doby na vyřízení a zjednodušení standardních služeb.
- Podpora při koordinaci činností jednotlivých úseků firmy.
- Uchovávání a sdílení informací získaných jednotlivými zaměstnanci.
- Poskytovat informace o činnostech různých úseků firmy.
- Poskytnutí podpory při rozhodování managementu firmy.

- Zajištění dodržování platných pracovních postupů a legislativních pravidel vykonávaných procesů.
- Podpora pro dosažení vytyčených cílů firmy.
- Nástroj pro ušetření nákladů.
- Zavádění inovací a nových výrobků či služeb.
- Podpora celopodnikové informační strategie.

Zavedením IS získá firma mnoho přínosů, které vedou zejména ke zvýšení konkurenceschopnosti. Ušetřené zdroje může firma využít jiným efektivnějším způsobem a hospodaření se stane více transparentním. Firma v rámci provozu systému může zavádět standardizované pracovní postupy osvědčené u dalších subjektů používajících daný informační systém. (Sodomka, 2006)

Na druhé straně firma musí investovat do provozu informačního systému nemalé prostředky. Je zapotřebí kvalifikovaných specialistů, kteří zajistí provoz a údržbu ekosystému aplikací a hardwarových zařízení tvořící informační systém. Zaměstnanci musejí být vyškoleni pro schopnost správně používat přidělené prostředky a dosáhnout tak všech výhod, které jim IS poskytuje. (Koch, 2010)

2.1.5 Řešení IS

Z hlediska implementace mají firmy na výběr několik řešení. Mohou se vydat cestou vývoje vlastního systému, nebo svěřit práci externímu dodavateli.

Implementace **vlastními silami** vytvoří systém přímo podle potřeb firmy díky detailní znalosti prostředí. Zkušenosti s vývojem systému a jeho tvůrci zůstanou ve společnosti, to umožňuje provádět rychlé úpravy a další rozvoj systému. Řešení je však finančně i časově nákladné, výsledná kvalita aplikace závisí na schopnostech tvůrců. Aplikace tak nebude kvalitní jako od specializujících se firem na dodávání výrobu IS. (Koch, 2010)

Řešení na míru od renomovaného dodavatele pokryje požadavky zákazníka a přinese po technické stránce kvalitní software. Zadavatel musí správně specifikovat své požadavky, jinak hrozí riziko, že systém nebude dělat to, co očekával. Takovéto řešení je finančně velmi nákladné z důvodů implementace speciálních požadavků. Hrozí riziko, vynesení citlivých informací mimo firmu. (Koch, 2010)

Použitím **hotového řešení** zákazník získá zejména kvalitu prověřenou řadou uživatelů. Opakované používání umožňuje systém pořídit za nižší cenu. Nevýhodou je nutnost přizpůsobit firemní procesy informačnímu systému a naučit zaměstnance, jak správně zacházet se systémem. (Koch, 2010)

Firmy mohou také zvolit řešení **outsourcingovou** formou, která je vhodná pro menší firmy, bez prostředků spravovat ICT nutné k provozu systému. Nevýhodami je však dlouhodobá nenávratnost, závislost na dodavateli a uložení firemních dat mimo firmu. (Koch, 2010)

O vhodnosti řešení rozhoduje několik faktorů, v závislosti na konkrétních podmínkách (velikost firmy, aktuální stav, obor). V praxi je většinou zvoleno

zakoupení hotového řešení od externího dodavatele s následným přizpůsobením svým specifickým požadavkům. Tak bude docíleno splnění požadavků s využitím co nejnižších nákladů. (Koch, 2010)

2.1.6 Způsob vývoje IS

V rámci životního cyklu prochází IS určitými fázemi vývoje. Okamžikem rozhodnutí zákazníka pro pořízení informačního systému začíná životní cyklus specifikací zadání. Na základě analýzy požadavků dojde k návrhu, implementaci a testování řešení. K ukončení životnosti dojde po určité době provozu, kdy systém nebude dále používán. Velmi důležitým krokem je řádné provedení analýzy požadavků, jejichž změna v průběhu vývoje může vést k průtahům, a zvýšení ceny.

Podle autorky Rábové (2008) existuje několik ověřených modelů řízení návrhu a vývoje informačního systému.

Vodopádový model je levný a rychlý, oddělující jednotlivé fáze. Jeho nevýhodou špatná reakce na změnu zadání, nemožnost aplikovat na všechny projekty a výsledek je znám až po skončení poslední fáze. (Rábová, 2008)

Prototypování umožní vytvořit částečně funkční vzor k prezentaci, zákazník je pak schopen reagovat na výsledek. Zapojení zákazníka do procesu umožňuje přesné dosažení zadání a reakci na případnou změnu požadavků v časnějších fázích, prototypování však není vhodné pro rozsáhlejší projekty z důvodu vysokých nákladů. (Rybička, Čačková, 2009)

Iterativní model cyklicky opakuje fáze vodopádového modelu, kdy se může v poslední fázi zákazník vyjádřit. V další iteraci dojde ke zpracování nových požadavků, dokud není zákazník spokojen. (Rybička, Čačková, 2009)

Model **spirála** kombinuje vodopádový model s prototypováním. Výhodou je možnost paralelního řešení a rychlého vývoje prototypů, nevýhodou je vysoká závislost na komunikaci se zadavatelem a obtížné plánování. (Rábová, 2008)

Existuje řada dalších metodik, které však vycházejí zejména z vodopádového modelu. Spirálový a iterační model tvoří základ pro agilní způsob vývoje, který umožňuje pružně reagovat na změny v zadání. (Cadle, Yeates, 2008)

2.1.7 Možnosti rozšíření IS

Nezřídka kdy se stává, že stávající systém musí být rozšířen o další funkcionalitu. Na základě zkušeností se systémem, zaváděním nových procesů, nebo změnou v podnikové struktuře může zákazník vyžadovat úpravu stávajícího řešení. Nejjednodušším způsobem je kontaktovat systémového integrátora, který je schopen navrhnout řešení i požadované úpravy realizovat po technické stránce. (Sodomka, 2006)

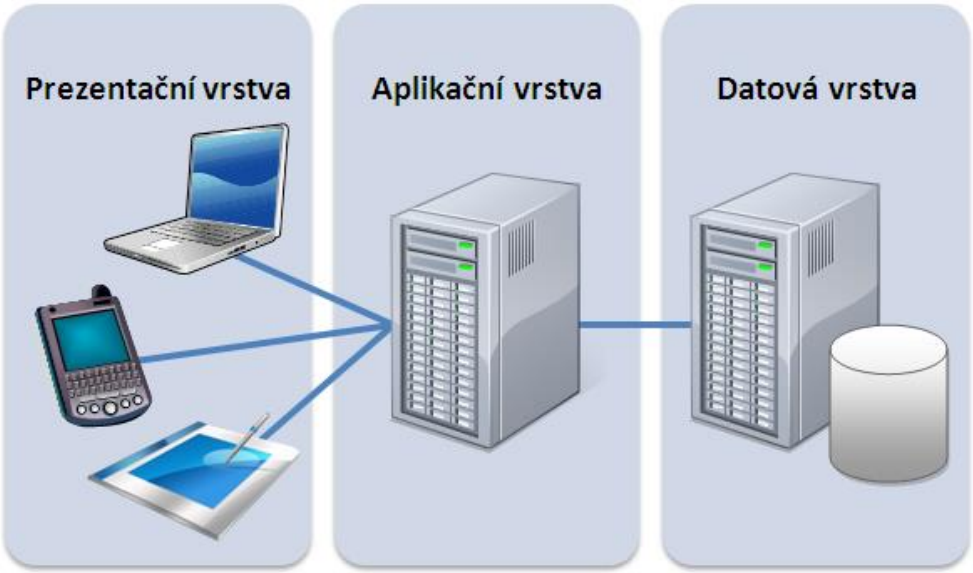
Úpravy systému mohou být realizovány následujícími způsoby:

- Nahrazení celého stávajícího systému novým, je nákladné řešení, na které je přistoupeno v případě nemožnosti řešit problém jiným způsobem, ev. pokud je zákazník přesvědčen o přínosech vydání se touto cestou.

- Použitím IS řešícího danou problematiku, firma pak provozuje oba systémy, které jsou vzájemně propojeny na datové úrovni.
- Rozšíření stávajícího IS vyžaduje dobrou znalost systému. V případě, že architektura systému umožňuje rozšíření, je nutné najít dodavatele schopného problém vyřešit.

2.1.8 Architektura

V současnosti bývá nejvíce používána pro realizaci IS **vícevrstvá** architektura **klient-server**, která umožňuje tvorbu robustních řešení. Tato architektura umožňuje oddělit datovou, aplikační a prezentační část aplikace. Aplikace běží na serveru, přistupuje se k ní pomocí webového prohlížeče, odpadá tak nutnost instalovat a spravovat další software na klientských stanicích.



Obr. 4 Schéma třívrstvé architektury (ManagementMania, 2015)

Prezentační vrstva obsahuje uživatelské rozhraní aplikace, zajišťuje vstup požadavků, komunikaci se serverem a prezentaci výsledků. Aplikační vrstva tvoří logiku aplikace, dochází zde ke zpracování uživatelských požadavků a provádění operaci za účelem poskytnutí odpovědí. Datová vrstva slouží ke správě a manipulaci s daty. (ManagementMania, 2015)

Webové aplikace, které jsou na této architektuře postaveny, přinášejí mnoho výhod oproti jiným řešením. Data jsou ukládána na jednom místě (databázový server), což zajišťuje jejich aktuálnost v rámci celého systému. Komunikace vrstev je realizována přes počítačovou síť, která umožňuje snadnou správu přístupu a distribuci dat. Práce se systémem je umožněna všem zařízením připojeným do sítě s nainstalovaným webovým prohlížečem nezávisle na typu zařízení a operačním systému. V případě provedení změn v aplikaci odpadá nutnost update softwaru klientů, kterých mohou být potencionálně až tisíce. Rozdělení na vrstvy

umožňuje vyvíjet jednotlivé vrstvy nezávisle na sobě a možnost nahrazení vrstvy za běhu aplikace. V závislosti na počtu klientů však může dojít k přetížení sítě a tím zpomalení odezvy aplikace. Další nevýhodou je závislost klientů na dostupnosti serveru, v případě přerušení spojení se serverem je klient neschopný pracovat samostatně. Z důvodů možnosti výpadku serveru bývá provozován záložní server, umístěný v jiném místě než server hlavní. (Fowler, 2003)

2.1.9 UML

Unified modeling language (unifikovaný modelovací jazyk) *„je univerzální jazyk pro vizualizaci a modelování systémů“*(ARLOW, NEUSTADT, 2007). Jedná se o způsob grafické prezentace systému tak, aby byl srozumitelný i obyčejným lidem.

Je důležité si uvědomit, že jazyk UML neposkytuje žádné přesné metodiky modelování. Nabízí pouze grafickou syntaxi, kterou je možné využít při sestavování modelů. Jazyk UML také není vázaný žádnou konkrétní metodikou nebo životním cyklem, proto ho lze využít ve spojení se všemi ostatními existujícími metodami. UML je však hojně využíváno v metodice Unified Process, která může být z tohoto pohledu označována za upřednostňovanou metodu jazyka UML, neboť je pro tento jazyk nejlépe přizpůsobena. (Arlow a Neustadt, 2007)

2.2 Ganttův diagram

Ganttův diagram se jmenuje po Henrym L. Ganttovi, který jej zpopularizoval začátkem 20. století. Může se také objevit pod názvem Harmonogram Adamieckiego podle svého objevitele, který jej poprvé použil koncem 19. století. Ganttovy diagramy se staly běžnou technikou projektového řízení díky snadné srozumitelnosti pro mnoho lidí. Využití pruhových diagramů je vhodné pro grafickou prezentaci časových harmonogramů. (Cadle, Yeates, 2008)

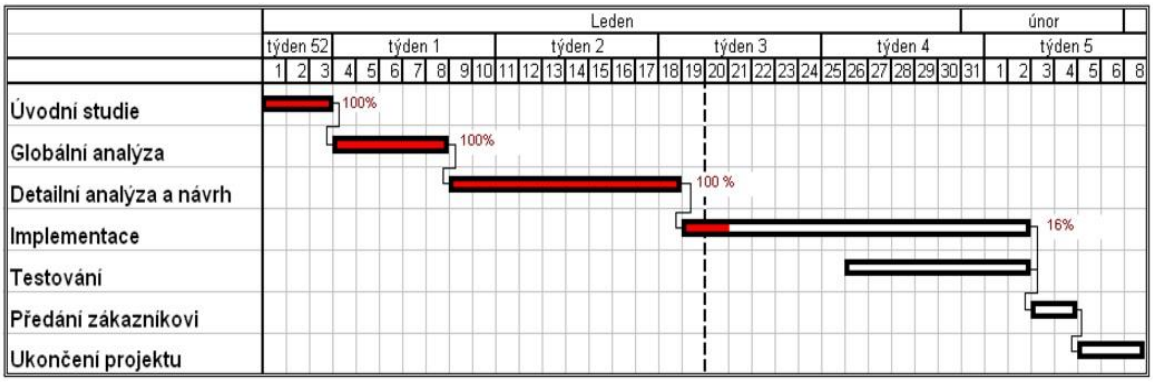
2.2.1 Projektové řízení

„*Projektem rozumíme organizované úsilí k dosažení určitého cíle*“. (Rosenau, 2010) Cílem projektového řízení je zajistit naplánování a realizaci činností vedoucí k úspěšné realizaci projektu s ohledem na dosažení požadovaných cílů. Rosenau (2010) dále tvrdí, že každý projekt má definovány tři základní parametry, a to náklady, čas a cíle projektu. Jiní autoři nahrazují cíle projektu kvalitou, rozsahem, či dostupností zdrojů. Důležité je také efektivně využít firemních zdrojů v závislosti na množství současně zpracovávaných projektů. (HRAZDILOVÁ BOČKOVÁ, 2016)

Při činnosti projektového řízení je využito dostupných znalostí, zkušeností, dovedností, nástrojů a technik. Existují softwarové aplikace, které poskytnou projektovým manažerům mocný nástroj pro plánování a řízení projektů. Tyto nástroje mohou být integrovány v rámci firemního informačního systému a poskytnout tak aktuální informace jeho uživatelům. Využití aktuálních údajů o stavu projektu umožňuje lepší kontrolu projektu a rychlou reakci na nepředvídané situace vedoucímu. Plánované činnosti mohou být automaticky předány realizačním pracovníkům požadovaných činností například formou zaplánování v kalendáři pracovních činností. (ManagementMania, 2013)

2.2.2 Ganttův diagram

Řízení projektů lze podpořit použitím určitých pomůcek a softwarových nástrojů, v závislosti na potřebách, typu projektu, zvolené metody nebo přístupu k řízení projektů. Jedním z nástrojů je Ganttův diagram, který srozumitelnou formou prezentuje posloupnost činností v čase.



Obr. 5 Ganttův diagram je zaměřen na prezentaci činností z časového hlediska, jednotlivé pruhy prezentují činnosti (Wikipedia, 2016)

Ganttův diagram je druh pruhového diagramu, jehož svislou osu tvoří jednotlivé pruhy znázorňující procesy. Na vodorovné ose jsou zobrazeny časové jednotky, činnost je pak vykreslená v rámci plánovaného časového intervalu jako obdélníková čára v příslušném pruhu. Ve složitější formě mohou vertikální osu tvořit zdroje, kterým je zobrazeno více procesů (lidem je přiřazeno více činností). (ManagementMania, 2013)

Existují různé variace diagramu, které mohou zobrazovat doplňující informace jednotlivých úloh i celého projektu. Zobrazením těchto údajů v diagramu vznikne Ganttův graf. (Cadle, Yeates, 2008)

- Dostupnosti úlohy (Release time), doba kdy nejdříve může být úloha započata.
- Termín dokončení úlohy (Due date), doba kdy by měla úloha být dokončena.
- Nejzazší termín dokončení úlohy (Deadline) je doba, kdy musí být úloha bezpodmínečně dokončena.
- Vzájemná návaznost jednotlivých úkolů (Precedence), první úkol musí být dokončen před započetím druhého úkolu. Vazba se nazývá *Finish to Start*.
- Perioda (Period) je používaná při cyklickém rozvrhování (zároveň udává deadline), periodicky se opakuje.
- **Milníky** jsou významné události v průběhu projektu. Jednotlivé časové okamžiky se stanovují za účelem kontroly a vyhodnocení uplynulé části projektu. V digramu jsou zobrazovány pomocí svislých čar.
- Zobrazování průběhu jednotlivých činností vyplňováním jednotlivých obdélníků prezentující danou činnost.

Některé informace, které mohou být zobrazeny pomocí Ganttova diagramu lze získat pomocí analytických či algoritmických metod, mezi které patří:

- **Metoda kritické cesty** (Critical Path Method – CPM) umožňuje stanovit (odhadnout) dobu trvání projektu na základě délky kritické cesty. Posloupnost činností ležící na kritické cestě určuje minimální dobu trvání

projektu, prodloužením některé z činností ležících na cestě dojde ke zdržení celého projektu.

- **Metoda PERT** (Project Evalution and Review Technique) slouží ke stanovení odhadu délky trvání projektu. Tato metoda využívá pravděpodobnostní model pro odhad délky jednotlivých činností, které mají proměnlivé trvání v závislosti na provozních podmínkách.
- **PDM diagramy** (Precedence Diagram Method) je varianta PERT-CPM, která přináší podporu pro optimalizaci a údržbu při změnách. Doplňuje vztahy mezi reprezentovanými činnostmi o nové vazby:
    - Finish to Finish (Konec-Konec)
    - Start to Start (Začátek-Začátek)
    - Start to Finish (Začátek-Konec)

Ganttův diagram je vhodný k prezentaci plánované posloupnosti činností v čase i kontrole průběhu plnění plánu. Tím může být dosaženo lepší koordinace prací, návaznosti činností a efektivnějšího využití zdrojů. (Svozilová, 2011)

2.2.3 Software pro tvorbu Ganttových diagramů

Vývojem softwarových produktů pro podporu projektového řízení se zabývá několik firem, které se zaměřují na různé zákazníky. Lze nalézt profesionální aplikace, poskytující širokou škálu funkcí, nebo úzce specializované řešení s omezenými funkcemi. Jednotlivé produkty jsou k dispozici pod různými licencemi, od možnosti používat produkt zdarma po nutnost zakoupení licence. Některé aplikace je nutné nainstalovat, další je možné použít online pomocí webového prohlížeče.

- **Microsoft Project** patří do balíku Microsoft Office, které jsou velmi rozšířené. MS Project je využíván profesionálními projektovými manažery k plánování, řízení a vedení projektů či týmů. Umožňuje sledovat termíny, přiřazovat zdroje a sledovat jejich využití. Výstupem mohou být Ganttovy diagramy, kalendáře, přehled peněžních toků, analýzy EVA a PERT, atd. MS Project disponuje mnohem širší funkcionalitou, umožňuje výpočet kritické cesty, zobrazení různých pohledů na projekt a synchronizaci s dalšími produkty společnosti Microsoft například SharePoint nebo Exchange. (Microsoft project, 2016)
- **Gantt project** je určen primárně vývojářům software, lze využít však pro libovolné projekty. Tento nástroje je licencovaný pod GNU GPL, uživatel má možnost jej využívat zdarma pomocí webového prohlížeče. Podporuje Ganttův diagram, Pertův diagram a diagram využití zdrojů. (GanttProject, 2016)
- **Gantto (Tom's Planner)** je webová aplikace, která umožňuje vytvořit a sdílet projektové plány formou Ganttova diagramu. Nástroj je dostupný v placené verzi i zdarma. Aplikace je dostupná online pomocí internetového prohlížeče, odpadá tak nutnost instalovat klienta. (TOM'S PLANNER, 2016)

- **TeamGantt** je také realizován formou webové aplikace, avšak je nutné si tento produkt předplatit. Aplikace působí přívětivě a disponuje rozšířenějšími nástrojemi než Tom's Planner. Výhodou dostupnosti online je i možnost přizvat spolupracovníky nebo zákazníky ke spolupráci. (TeamGantt, 2016)

2.3 Informační systém COMES

„Systém COMES ® je výrobní informační systém úrovně MES pro všechna průmyslová odvětví. COMES slouží pro operativní řízení výroby a jejímu dalšímu zefektivňování ve spolupráci s celopodnikovým IT systémem (ERP) a řídicími systémy řízení technologií.“ (MES solution COMES, 2016)

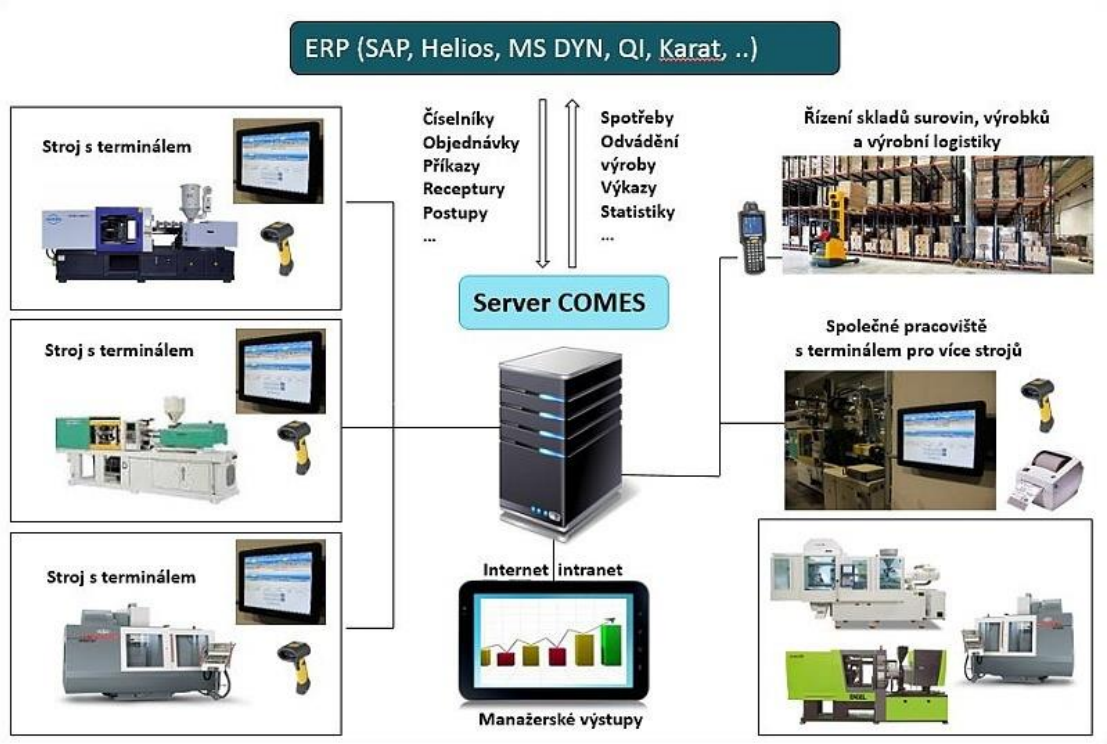
Firma Compas automatizace, spol. s r.o. se pohybuje v oboru průmyslové automatizace již více než 25 let a patří k hlavním lídrům na českém trhu. Portfolio zákazníků obsahuje velké tuzemské i zahraniční společnosti i z různých odvětví zpracovatelského průmyslu, zejména automotive, farmacie, potravinářství či strojírenství. Systém COMES umožňuje zákazníkovi detailně plánovat výrobu, zlepšovat řízení výroby, jakost i dokumentaci.

Compas je orientován na potřeby zákazníků, dodává svůj systém jako hotové modulární řešení, vycházející vstříc zákazníkům možností přizpůsobení aplikace na míru individuálním potřebám konkrétního zákazníka. (Compas Automatizace, 2016)

Více informací o firmě COMPAS a systému COMES lze nalézt na webových stránkách výrobce[2,3].

[2] *http://www.compas.cz/*

[3] *http://www.comes.eu/*



Obr. 6 Schéma použití systému COMES pro řízení lisovny (Compas Automatizace, 2016)

2.3.1 Moduly systému

Modulární řešení systému umožňuje zákazníkovi sestavit na základě svých požadavků systém disponující pouze požadovanou funkcionalitou. Každý modul je samostatně licencovaný a zákazník tak není nucen platit za něco, co nepotřebuje. Systém Comes se skládá z následujících modulů. (MES solution COMES, 2016)

- **Logon** je základní modul systému, který umožňuje přihlašování a správu uživatelů, skupin a konfiguraci uživatelských oprávnění. Modul je vstupní brána pro přístup do ostatních modulů a umožňuje jejich správu a konfiguraci. Také umožňuje vytvořit model výrobních zařízení a technologií. Umožňuje spravovat události vzniklé v systému a vytváří záznamy o veškerých změnách v konfiguraci. Modul umožňuje tvorbu záloh a obnovení systému ze zálohy, klient tak předejde ztrátě dat.
- Modul **Historian** umožňuje sběr, archivaci a prezentaci veličin získávaných z řídicích systémů a čidel výrobních technologií. Umožňuje tvorbu uživatelských pohledů formou grafů nebo tabulek a v nich zobrazovat uložené hodnoty a alarmy.
- **Modeller** slouží pro modifikaci funkčnosti systému. Uživatelům je umožněno vytvářet nebo upravovat aplikace v rámci systému. Obsahuje vývojové

prostředí pro konfiguraci systému COMES, což umožňuje flexibilní reakci na změny v pracovních postupech.

- **Batch** umožňuje řízení šaržové výroby (tvorba výrobní receptury, řízení průběhu výroby). Umožňuje nahradit papírovou výrobní dokumentaci elektronickou formou.
- Modul **Traceability** umožňuje sledovat rodokmen výrobku – použité suroviny, výrobní operace, identifikace výrobních zařízení.

2.3.2 Funkčnost sytému

Systém Comes je zákazníky využíván za účelem zdokonalení operativního řízení výroby a provázáním jednotlivých úrovní řízení. V rámci vertikální integrace tvoří systém spojovací prvek mezi automatizovanými systémy a existujícím informačním systémem úrovně ERP. (Compas Automatizace, 2016)

Systém COMES je realizován jako webová aplikace, umožňuje tak klientům přístup odkudkoli, pomocí nejrůznějších typů zařízení, které disponují webovým prohlížečem. Organizace MESA (Manufacturing Enteprise Solutions Assosiation) vydává standardy pro MES systémy, který systém COMES dodržuje. (MES solution COMES, 2016)

Výhodou systému je integrace základní funkcionality, jednotlivé funkce jsou tak lépe provázány a přinášejí lepší efekt. Zákazník může využívat funkce implementované v systému COMES: (Compas Automatizace, 2016)

- Krátkodobé kapacitní plánování výroby – týdenní, denní a směnové.
- Řízení výrobních kapacit.
- Materiálový management, označování materiálů, výrobní logistika, traceabilita, řízení skladů WMS.
- Výrobní postupy (předpisy) a receptury (parametry).
- Řízení a supervize výroby včetně přímé komunikace s automatizací technologií.
- Jištění kvality výroby, plánování testů, záznam a vyhodnocování zkoušek, protokoly a statistiky.
- Elektronické výrobní protokoly.
- Sběr, analýza, archivace technologických veličin a varovných hlášení.
- Sběr a ukládání výrobních dat.
- Operativní výpočty výrobních ukazatelů a statistik (výkonnosti, KPI, OEE, …).
- Podpora řízení údržby.

Výhodou systému COMES je jeho otevřenost, uživatelé si tak mohou funkčnost konfigurovat sami pomocí vestavěných vývojových nástrojů, které jsou součástí systému. Úpravy systému si může zákazník nechat provést v případě, že toho není schopen přímo od firmy Compas automatizace spol. s r.o., nebo dalších

spolupracujících firem. Systém je možné napojit na libovolný ERP systém a automatizované technologie.

V případě zakoupení systému COMES získá uživatel základní nástroje pro veškerou správu systému zdarma. Systém Comes je vybaven **editorem skriptů**, který umožňuje úpravu a tvorbu veškerého uživatelského obsahu. (MES solution COMES, 2016)

3 Metodika řešení

Cílem kapitoly metodika řešení je definovat postup při návrhu a tvorbě řešení této diplomové práce. Stanovení vhodné metodiky řešení vychází z poznání problematiky zobrazení dat formou Ganttova diagramu a možnostmi tvorby rozšíření informačního systému. Volbu metodiky ovlivňují také požadavky kladené zadavatelem a technologické aspekty rozšiřovaného systému.

Postup při řešení problematiky popisují následující metodické kroky:

- specifikace problému,
- analýza řešení,
- návrh řešení,
- implementace návrhu,
- zavedení do sytému.

Za účelem bližšího poznání problematiky budou probíhat *konzultační schůzky* se zadavatelem. Jednání budou probíhat v rámci analytické činnosti, při které dojde ke *stanovení požadavků* kladených na rozšíření systému, jejichž splnění je cílem řešení této práce. Formálně správně zpracovaná specifikace dělí požadavky na funkční a nefunkční, je tedy nutné získané požadavky rozdělit do uvedených skupin. Je také nutné získat bližší informace o architektuře a technologiích současného systému, které budou získány výzkumem systému, studiem technické dokumentace ev. konzultací s tvůrci systému.

Pomocí znalostí aktuálního stavu a specifikovaných požadavků již může dojít ke tvorbě abstraktního návrhu řešení problému. Takový návrh bývá také označován jako konceptuální model, který umožňuje prezentovat základní myšlenku řešení problematiky oproštěnou o technickou stránku.

Pro další rozpracování návrhu řešení je nutné stanovit **technologie** a **pracovní postupy** pro použití v dalších fázích projektu. Rozšířením prvotního konceptu o technologické prostředky vznikne logický model, který popisuje způsob řešení problému pomocí zvolených technologií.

Transformace návrhu v implementační model vyžaduje pomocí technologických prostředků realizovat navržených schémat. Dojde tedy k implementaci navržené funkcionality na fyzické úrovni. Výstupem by měl být spustitelný program.

Zavedení vytvořeného programu do informačního systému spočívá v instalaci na technická zařízení. Jedná se o překopírování zdrojových kódů, zavedení programu do softwarových zdrojů systému.

Testování funkčnosti proběhne aplikací vytvořeného programu na vhodném demonstračním příkladu. Po ověření funkčnosti již může být umožněn ostrý provoz aplikace.

Grafická vizualizace návrhů bude vytvořena pomocí standartního zápisu unifikovaného modelovacího jazyka UML. Jedná se o nástroj pro tvorbu návrhů programových systémů s podporou objektově orientovaného přístupu.

Samotný proces tvorby řešení bude využívat iterativní vývojový model, který určuje způsob přístupu k jednotlivým etapám životního cyklu. Iterativní metoda vývoje patří do skupiny nazývané agilní metodiky, které umožňují rychlý vývoj a zároveň reakci na změny požadavků během vývoje. Hlavní předností je zapojení zadavatele v celém procesu vývoje, předkládáním prototypů vzniká zpětná vazba, kterou lze využít pro zlepšení v další iteraci a celkově tak zvýšit pravděpodobnost přijetí konečné implementace uživatelem. Opakování iterací probíhá, dokud nedojde ke splnění požadavků uživatele.

Vzhledem ke tvorbě rozšíření systému typu webové aplikace, využívající vícevrstvé architektury klient-server, bude zvolena vhodná metodika pro realizaci návrhu a řešení úlohy. Různé metodiky popisuje disciplína softwarového inženýrství, zvolená technika by však měla podporovat objektově orientovaný přístup. Navržený způsob postupu je nutné předložit zadavateli ke schválení a na základě eventuálních připomínek provést patřičné úpravy.

4 Vlastní práce

Tato kapitola popisuje autorovi pochody při návrhu řešení, jeho realizaci a následném nasazení v cílovém informačním systému. Nejprve dojde k bližší specifikaci problematiky a rozboru technologických aspektů řešení. Následně dojde k návrhu řešení splňující požadované nároky. Navržený způsob bude realizován a nasazen za účelem ověření funkčnosti.

4.1 Specifikace problému

Úkolem této kapitoly je popsat aktuální situaci firemního informačního systému. Pozornost bude věnována použitým technologiím, vlastnostem aktuálně používaného systému a popisu požadavků kladených zadavatelem na tvorbu rozšíření. Získávání informací proběhne výzkumem systému a komunikací se zadavatelem. Z definovaných požadavků zadavatele dojde k vytvoření funkční specifikace.

4.1.1 Analýza současného stavu

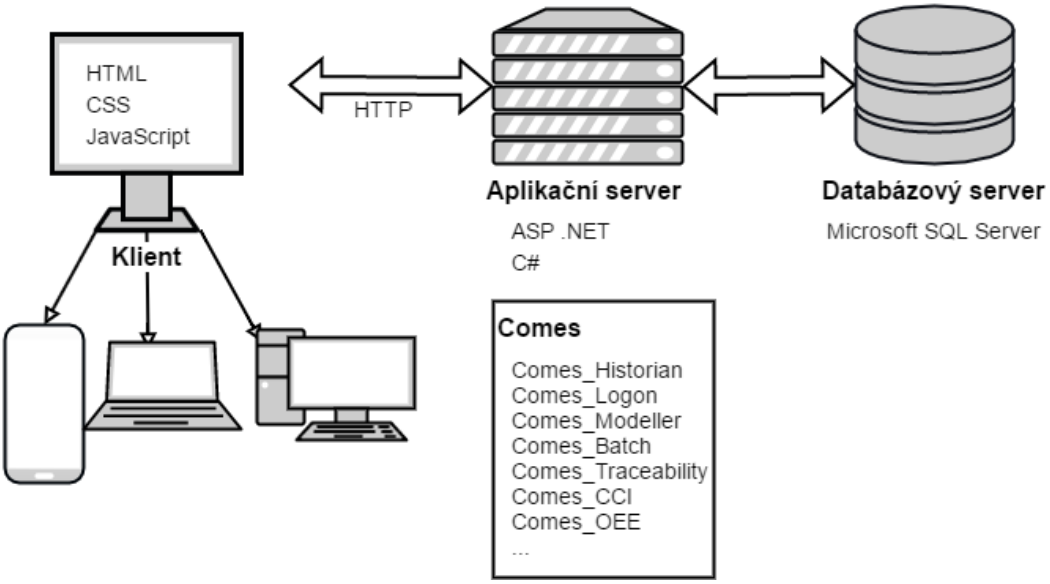
V současné situaci systém COMES disponuje omezenými nástroji pro projektové řízení. Plánování a zobrazení plánů je standardně umožněno skrze formuláře v tabulkové formě. V případě zákaznických požadavků na rozšířenější funkcionalitu systému dojde k jejich implementaci na míru. Pro dosažení požadovaného stavu může být využito softwarových nástrojů třetí strany. Například Microsoft Project – viz *Software pro tvorbu Ganttových diagramů*, které poskytnou požadované nástroje. Vzniká tak nová problematika – *propojení dvou programů*, kterou lze vyřešit vytvořením rozhraní, pomocí kterého dojde k integraci nové „komponenty“ do systému. Řešení tímto způsobem komplikuje řada omezení v uzavřenosti softwaru, funkčnosti API a samotné složitosti implementaci rozhraní pro danou funkcionalitu.

Z důvodů opakovaných poptávek by bylo vhodné uspokojit poptávku zákazníků implementováním univerzálního nástroje pro vykreslení Ganttova diagramu. Integrací komponenty do systému by bylo umožněno její opakované využití v budoucích projektech bez nutnosti opakovaného psaní kódu. Došlo by k ušetření času, který mohou pracovníci využít prací na jiných požadavcích, nebo ke snížení ceny produktu, díky které by se mohl produkt stát dostupnější pro další zákazníky. Opakovaným použitím komponenty by došlo k odhalení různých chyb, jejichž odstraněním vznikne odladěný produkt. Ušetřený čas se také může využít pro další rozšiřování funkcionality. Hlavním přínosem bude zvýšení přidané hodnoty produktu dodavatelské společnosti. Je vhodné vyvinout vlastní řešení z důvodů vyhnutí se závislosti na produktu třetí strany, zejména v případě odhalení komplikací či vzniku požadavků na rozvoj funkcionality, které poskytovatel nemusí být ochotný řešit, ev. již ukončil podporu.

4.1.2 Architektura stávajícího systému

Systém COMES je realizován jako webová aplikace, zpřístupněná uživatelům v počítačové síti. Webové aplikace využívají třívrstvé architektury klient – server, tento systém není výjimka.

V rámci spolupráce s firmou Microsoft je pro budování systému využito zejména produktů této firmy. Řešení je tedy realizováno s použitím technologií .NET a MSSQL provozovaných pod operačním systémem Windows, primárně pak jeho serverovou verzí.



Obr. 7 Architektura systému COMES

Jako hardwarových částí je využívána osvědčená výpočetní technika od renomovaných dodavatelů. Díky této kombinaci může dodavatel garantovat maximální funkčnost systému a poskytnout certifikaci pro dodávaný systém v rámci partnerství se společností Microsoft.

Systém COMES je realizován pomocí známých technologií a postupů používaných pro tvorbu webových aplikací. Ke skladování dat je využito databázového serveru, využívajícího Microsoft SQL Server. Na použití konkrétní verze nejsou kladeny požadavky, v současnosti se používá poslední verze SQL Server 2014.

Funkčnost webového serveru zajišťuje služba IIS (Internet Information Server), který umožňuje provozovat serverové aplikace využívajících služeb frameworku .NET.

Samotná aplikace je implementována s využitím technologií ASP.NET pro tvorbu dynamických stránek. Obsah systému je zobrazen pomocí serverem generovaných webových stránek, které jsou tvořeny pomocí technologií HTML, CSS a JavaScriptového kódu.

4.1.3 Analýza požadavků

Identifikace a porozumění požadavkům zadavatele je zdlouhavý proces, na jehož konci dojde ke specifikaci kladených požadavků na výslednou aplikaci. Při procesu tvorby návrhu řešení je nutné definované požadavky akceptovat a splnit.

4.1.3.1 Funkční požadavky

- Možnost časové navigace v Gantt diagramu. Uživatel se pomocí myši nebo ovládacích prvků bude schopen pohybovat po časové ose. Tento požadavek zajistí možnost sledování pouze vybraného termínu.
- Možnost zvětšovat, zmenšovat Gantt diagram. Vhodným způsobem bude uživatel schopen volně měnit měřítko časové osy, umožní mu to tak zobrazit požadovaný časový rozsah diagramu. Pro realizaci tohoto požadavku bude vhodné využít kolečko myši, nebo ovládací prvky
- Dynamické výšky řádku v diagramu. Na základě výšky celého diagramu a počtu jeho pruhů bude automaticky nastavena výška jednotlivým pruhům. Výšku pruhů bude také konfigurovat v nastavení diagramu.
- Konfigurace pruhů Ganttova diagramu. Vývojář bude mít možnost definovat a nastavit vlastní styly pro pruhy diagramu. Uživatel v rámci používání diagramu může nastavit dočasně každému pruhu diagramu některý z existujících stylů, tím dojde ke zvýšení přehlednosti barevným odlišením pruhů.
- Doplnění vertikálního gridu. V diagramu by měla existovat možnost zobrazit vertikální mřížku s definovatelnou hustotou čar. Uživatel bude mít možnost tuto volbu nastavit i sám.
- Ruční plánování v Gantt diagramu. Pomocí myši bude uživateli umožněno editovat jednotlivé pruhy diagramu. Editací rozumíme přesouvání pruhů mezi řádky diagramu, jejich posouvání po časové ose, změna trvání činnosti (časový rozsah pruhu) a přidávání nových nebo mazání existujících pruhů.
- Přichytávání položek Ganttova diagramu ke gridu. V případě aktivace této volby bude umožněno při ruční editaci diagramu přichytávat přesouvané pruhy k mřížce.
- Zobrazení více funkcí. V jednom řádku diagramu bude možné zobrazit více pruhů.
- Detailní informace pruhu. Při kliknutí myší do oblasti pruhu dojde k zobrazení detailních informací dané činnosti. Detailní informace mohou obsahovat různé prvky, a proto by měla být zachována jistá volnost pro zobrazovaný obsah.
- Volitelnost možností nastavení. Všechny zmíněné požadavky bude možné nastavit při inicializaci komponenty na základě aktuálních potřeb a způsobu použití diagramu.

- Filtrování dat. Uživateli bude při práci umožněno zvolit, které data z datové sady budou zobrazena v diagramu. Nevybrané prvky nebudou nevykresleny v rámci aktuálního zobrazení daným uživatelem.
- Interaktivita. Při provádění uživatelských operací dochází ke komunikaci. V případě zadávání vstupních dat dojde ke kontrole, při ukládání změn bude uživatel upozorněn na změny a dotázán, zdali chce danou operaci vykonat.
- Aktuální kurzor myši. Pro zlepšení navigace v časové ose diagramu bude možnost zobrazit informace uvádějící časový údaj o pozici kurzoru myši vůči časové ose diagramu.
- Zabezpečení dat. Diagram bude možné zobrazit pouze autorizovaným uživatelům.
- Panel nástrojů. Položky uživatelského nastavení manipulace v rámci diagramu bude mít možnost uživatel ovládat pomocí panelu nástrojů.

4.1.3.2 Nefunkční požadavky

- Integrovatelnost do systému. Řešení bude možné použít ve stávající webové aplikaci.
- Uživatelská přívětivost. Aplikace musí být přehledná, grafický styl bude podobný stávající grafické podobě webových stránek systému.
- Přenositelnost řešení. Řešení bude použitelné v architektonicky a technologicky odlišných systémech. Jedná se o různé verze .NET frameworku, nebo systém běžící na PHP.
- Poskytnutí kódů. Zadavateli bude dodán také zdrojový kód, který bude možné upravit v případě nutnosti, ke kterým může vést objevení chyby nebo rozšíření funkčnosti.
- Dokumentace. V rámci předání řešení bude odevzdána dokumentace obsahující návod pro práci s komponentou. Veškeré informace o programu jsou obsahem této práce, je však žádoucí vytvořit separátní dokumentaci, která poskytne návod uživatelům a vývojářům při používání softwarového řešení bez nutnosti studovat tuto práci.
- Podpora a optimalizace webových prohlížečů. Cílem je zajistit korektní zobrazení diagramu v některých webových prohlížečích. Požadována je optimalizace pro Internet Explorer verze 11. a Google Chrome verze 40. a novější.
- Minimalizování režie serveru. Je žádoucí nevyužívat více serverových prostředků, než je nezbytně nutné. V případě zobrazení rozsáhlého diagramu by prováděné operace mohli nadbytečně vytěžovat server a tím by došlo ke snížení odezvy, této situaci je nutné pokud možno předejít.

4.2 Technologické aspekty řešení

Vzhledem k tvorbě komponenty pro webovou aplikaci využívající vícevrstvé architektury, dojde ke tvorbě klientské a serverové části. Řešení bude realizováno jako tenký klient pomocí webového prohlížeče. Ve vymezené oblasti může být řešení realizováno několika různými způsoby, které budou popsány v této podkapitole.

4.2.1 Možnosti realizace klientské části

Grafická prezentace grafu bude realizována na straně klienta, může využívat různých elementů jazyka HTML v kombinaci s JavaScriptem a CSS styly. V případě zvážení uživatelských požadavků zjistíme, že použití JavaScriptu je nutné, neboť umožňuje přenést část zátěže na klienta. Pro tvorbu interaktivní a uživatelsky přívětivé aplikace přináší mnoho nepostradatelných nástrojů. V úvahu připadají tak následující možnosti zobrazování prvku.

4.2.1.1 Kreslení grafiky do elementu Canvas

Element canvas, který byl zaveden až v HTML5 slouží jako plátno pro vykreslování grafických objektů a animací na webových stránkách. Grafika je prezentována ve formě pixelové matice. Výhodou je široká podpora ve webových prohlížečích a vysoký výkon umožňující rychlé kreslení objektů. Stálá výkonost umožňuje kreslit libovolně složité obrázky se stejnou náročností, která se zvyšuje až se zvětšováním formátu obrázku. Vykreslování objektů však není možné bez použití JavaScriptu. Absence rozhraní pro animace a veškeré interaktivní prvky je možné vytvořit pomocí scriptu, ale výsledek nedosahuje takových možností jako svg.

4.2.1.2 Vykreslování vektorové grafiky pomocí SVG

Tento element umožňuje vykreslovat vektorové grafické objekty definované formátem XML. Hlavní výhoda tohoto řešení plyne z vektorové grafiky – absence ztráty kvality při přiblížení a oddálení. Možnost tvorby animace přímo pomocí atributů grafických prvků a jejich přístupnost v rámci DOMu. Dostupnost elementů umožňuje přiřazení jednotlivým grafickým elementům CSS styl a odchycení události pomocí JavaScriptu což přináší více možností pro tvorbu interaktivního prostředí. Existuje také možnost vložení textu do obrázku, který si zachová původní vlastnosti, a nadále se s ním může pracovat jako textem. V případě nutnosti můžeme jednoduše SVG konvertovat do klasických obrázkových formátů a tím umožnit například tisk nebo zobrazení v nepodporovaných prohlížečích. Složité grafické objekty jsou však výkonnostně náročné na vykreslení, proto je vhodné pro tvorbu takových obrazců zvolit jiný nástroj.

4.2.1.3 Pomocí neutrálních elementů HTML

Tyto elementy nejčastěji využíváme jako kontejnery, pro obalení obsahových elementů, jejichž síla spočívá v možnosti propojení s CSS styly, které přidá těmto prvkům význam. Pomocí elementů *div* a *span* formátovanými kaskádovými styly jsme schopni vykreslit relativně použitelné, uživatelsky přívětivé avšak po některých stránkách omezené řešení. Výhodou je jednodušší práce a dostačující nástroje pro uživatelskou přívětivost. Možnost přenést značnou část zátěže serveru na klienta v závislosti na tom, jak bude přistupováno k prezenční logice. V případě provádění formátovacích výpočtů na straně klienta bude zátěž na server minimalizována. Nevýhodou je absence některých grafických prvků, které nelze jiným způsobem nahradit a režie serveru v případě provádění formátování na straně serveru. Prováděním výpočtů formátování je myšleno generování CSS stylů jednotlivým elementům stránky. Technologie je relativně vhodná, přestože je již překonána novějšími prvky.

4.2.1.4 Zobrazení dynamicky generovaného obrázku

Řešení tímto způsobem by spočívalo v zobrazení serverem generované obrázkové bitmapy pomocí HTML. Zajistit dodržení požadavků tímto způsobem by bylo velice náročné, ve srovnání s ostatními volbami by řešení značně pokulhávalo v interaktivních možnostech prvku. Zátěž serveru a síťový tok bude v tomto případě nejnáročnější a nezanedbatelný. Výhodou by mohlo být zachování formátu při zobrazení v různých podmínkách a tisku.

4.2.2 Serverová část

Je zřejmé, že pro realizaci serverové části také existuje více způsobů. Pozornost bude věnována zejména zpracování pomocí ASP.NET, jehož technologii současný systém využívá. Při nasazování řešení do současného systému tak dojde k usnadnění integrace, která bude proveditelná bez nutnosti dodatečných úprav. Návrh řešení bude brát ohled na přenositelnost i do jiného prostředí, kterým může být například PHP s obdobnou funkčností, ale větší mírou použití.

Serverová část bude obsahovat specifikaci webového rozhraní XMLHttpRequest, pro komunikaci klienta se serverem. Tím bude zajištěna možnost univerzálního použití v dalších prostředích a další manipulace se zobrazenými informacemi. Serverová část bude disponovat funkcionalitou pro přístup a manipulaci s daty, které mohou být skladovány v databázovém systému, ale i ve vlastních formátech.

Z důvodů různorodosti způsobů uložení dat a jejich struktury, která se jistě liší v každém projektu, nelze vytvořit univerzální rozhraní. Na základě zkušeností při tvorbě řešení ve specifických podmínkách systému COMES dojde k vytvoření návodu, pro implementaci vlastního řešení rozhraní, kterého umožní získat a poskytnout data z interních zdrojů konkrétního systému klientské straně.

Komunikace mezi klientem a serverem bude probíhat pomocí technologie AJAX zasíláním zpráv ve specifikovaném formátu. Výměna dat bude probíhat pomocí technologií XML nebo JSON v předem stanovené struktuře.

4.2.3 Technologie vhodné k použití

Grafické uživatelské rozhraní ve formě webové stránky, bude realizováno formou tenkého klienta s pomocí webového prohlížeče. Zobrazované webové stránky využívají technologií a standardů definovaných mezinárodním konsorciem W3C. Pro některé nástroje neexistuje alternativa a jsou tedy jedinou možností. Jindy je na výběr z více alternativních možností, proto je nutné na základě osobních preferencí a technologických vlastností zvolit to nejvhodnější.

4.2.3.1 HTML Hypertextový značkovací jazyk

HyperText Markup Language je hlavní nástroj pro vytváření obsahu webových stránek. V průběhu let se jazyk vyvíjel až do současné verze HTML5. Jazyk disponuje sadou značek (tzv. tagů) a jejich vlastností (attributů), pro zápis elementů určitého významu. Rozlišujeme strukturální, popisné (sémantické) a stylistické značky pro definici vzhledu dokumentu. Pro zobrazení stránky se používá webový prohlížeč, který zajistí vykreslení kódu. (W3, 2013)

4.2.3.2 Kaskádové styly CSS

Cascading Style Sheet slouží pro zápis způsobu vykreslení HTML elementů ve webovém prohlížeči. Smyslem CSS je oddělit vzhled dokumentu od jeho struktury a obsahu. Pro správnou funkčnost je nutné definovat propojení HTML stránky s CSS stylem. Kaskádové styly umožňují definovat rozdílný způsob zobrazení jedné stránky pro různá zařízení ev. různé webové prohlížeče. (W3, 2016)

4.2.3.3 JavaScript

Objektově orientovaný skriptovací jazyk je používaný pro tvorby interaktivních prvků na webových stránkách. Jazyk je objektově orientovaný, multiplatformní, syntakticky blízky jazykům rodiny C. Tento nástroj má sice konkurenci, která však nedosahuje srovnatelné míry používání, s tím souvisí i menší podpora. (W3C, 2016)

JavaScriptový kód nazývaný script je spouštěn na straně klienta, umožňuje tím přenést část zátěže na klienta. Hlavním přínosem je možnost oddělit funkčnost od struktury a obsahu dokumentu.

Při použití JavaScriptu se setkáme s použitím knihovny jQuery zlepšujícím interakci mezi HTML a JavaScriptem. Knihovna obsahuje mnoho funkcí pro práci s obsahem stránky a tvorbu interaktivních prvků. (jQuery, 2016)

Efekt použití jQuery lze umocnit použitím dalšího JavaScriptového frameworku jQuery UI, který se zaměřuje na uživatelské rozhraní. JQuery UI již

implementuje mnoho pokročilých efektů a vylepšuje funkcionalitu řady HTML elementů. (jQuery UI, 2106)

Hlavní nevýhodu JavaScriptu je nemožnost komunikace se serverem, kterou nahrazuje technologie AJAX (*Asynchronous JavaScript and XML*). Prostřednictvím AJAXu může probíhat komunikace mezi klientem a serverem na pozadí. Odpovědí na zaslaný požadavek mohou být data nebo změny ve stránce, odpadá nutnost znovu-načíst celý obsah stránky. (Mozilla developer network, 2016)

4.2.3.4 ASP.NET

Pro tvorbu serverové části budou zvoleny technologie, používané v současném řešení, za účelem dosažení co nejjednodušší integrace řešení do systému.

Microsoft .NET framework je rozsáhlá softwarová platforma umožňující vývoj různých druhů aplikaci zejména pod operačním systémem Windows. Pomocí .NET frameworku můžeme vyvíjet aplikace pro desktop, webové aplikace a služby, mobilní zařízení a další. Platforma disponuje knihovnami pro práci s Windows API, COM, sítí, databází, souborovým systémem a mnoho dalšího. Jednou ze součástí frameworku je ASP.NET (Active Server Pages), která slouží pro tvorbu webových aplikací. (Microsoft developer network, 2016)

4.2.3.5 C#

Programovací jazyk C# „C Sharp“ slouží pro vytváření aplikací využívajících technologií frameworku .NET. Jazyk C# byl zvolen pro svou blízkost jazykům rodiny C, navíc je modernější než alternativa jazyku Visual Basic, umožňuje tedy využít pokročilejších technik při tvorbě aplikace. V současné době dochází k růstu využívání jazyka C#, s tím je spojena i rozsáhlejší podpora. (Microsoft developer network, 2016)

4.2.3.6 SQL

Structured Query Language, česky strukturovaný dotazovací jazyk je programovací jazyk umožňující práci s relačními databázemi. Serverová část bude zajišťovat manipulaci s daty uloženými v interní databázi. Pomocí jazyka SQL dojde k manipulaci s potřebnými daty skladovanými v databázovém systému. (Microsoft developer resource, 2016)

4.2.3.7 JSON

JavaScript Object Notation je velmi rozšířený formát pro výměnu dat na webových stránkách. Přenášené objekty v JavaScriptovém zápisu jsou převedeny do textové proměnné, díky čemuž je umožněn multiplatformní přenos dat zejména pomocí technologie AJAX. (JSON, 2016)

4.2.3.8 Vývojářské nástroje

Pro tvorbu řešení bude využito vývojového prostředí Microsoft Visual Studio 2013, které umožňuje tvorbu aplikací těžících z funkcionality technologií společnosti Microsoft. Visual Studio umožňuje vytvářet různé typy aplikací pro mnoho druhů zařízení. Výsledné aplikace mohou být provozovány na operačních systémech Microsoft Windows, Android a iOS.

Tento vývojářský nástroj disponuje širokou škálou funkcí pro zvýšení produktivity práce v různých oblastech vývoje aplikací. Některé zjednodušují psaní kódu pomocí IntelliSense a refactoringu, ladění aplikací umožňuje propracovaný debugger. K dispozici je i integrovaný designer umožňující tvorbu grafického uživatelského prostředí webových stránek, objektových tříd a databázových schémat. K dispozici jsou i nástroje pro spolupráci vývojářského týmu, verzovací systémy i další komunitní nástroje. Aplikaci je také možné rozšířit využitím různých pluginů, kterých je k dispozici velké množství. Visual Studio disponuje integrovaným webovým a SQL serverem, který umožňuje provozovat a aplikace využívající těchto technologií.

Díky oficiální podpoře pro mnoho jazyků, mezi které patří C++, C#, C, VB.NET F#, Python, Ruby, XML, HTML, CSS, JavaScript, SQL (a dalších, které lze přidat i neoficiální cestou) a zmíněné funkcionalitě vývojář disponuje mocným nástrojem pro tvorbu aplikací. Visual Studio je neustále vyvíjeno (nová verze vychází cca každé 2 roky) a nové verze přinášejí mnoho změn a zlepšení.

Dále bude využito webových prohlížečů Microsoft Internet Explorer a Google Chrome v rámci testování za účelem ověření požadavku na správné fungování aplikace.

Při vývoji aplikace bude použito webových prohlížečů Microsoft Internet Explorer verze 11 a Google Chrome verze 48. Zmíněné aplikace není potřebné popisovat, alespoň každý počítačový uživatel se s nimi nejméně jedenkrát setkal. Tyty webové prohlížeče je nutné využít při vývoji řešení jako testovací nástroj pro určení správné funkčnosti, jak je specifikováno v požadavcích projektu.

Tvorbu grafických modelů v návrhové části této práce bude použit Enterprise Architekt. Který umožňuje vytvářet grafické návrhy softwaru pomocí jazyka UML.

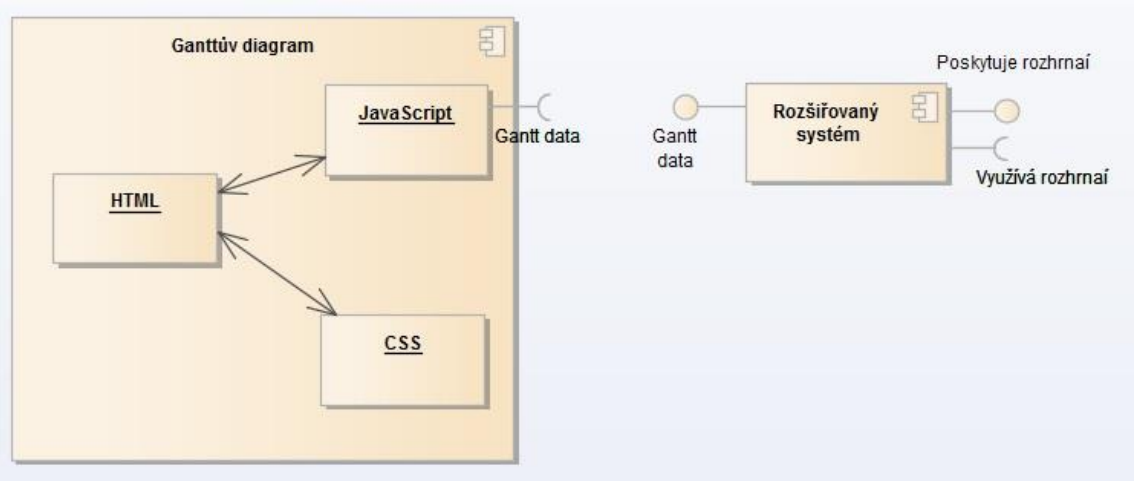
4.3 Návrh řešení

Na základě určených požadavků a vhodných technologií pro řešení problematiky je možné přistoupit k tvorbě samotného návrhu rozšíření informačního systému COMES o komponentu Ganttova diagramu. Pomocí notačního standartu UML dojde k vytvoření grafických návrhů znázorňující zamýšlené řešení problematiky.

Cílem je vytvoření návrhu splňující všechny zadavatelem kladené požadavky. Hlavní problematiku tvoří zobrazení dat předem neznámého formátu ve formě Ganttova diagramu. Z možných řešení problematiky je nutné vybrat ten nejvhodnější způsob a pro jeho realizaci zvolit správné technologie.

4.3.1 Koncept návrhu

Pro znázornění zamýšleného způsobu řešení na obecné úrovni je vytvořen následující model. V této fázi slouží k prezentaci navrženého vztahu komponenty Ganttova diagramu a současného systému, který je prezentován černou skříňkou.

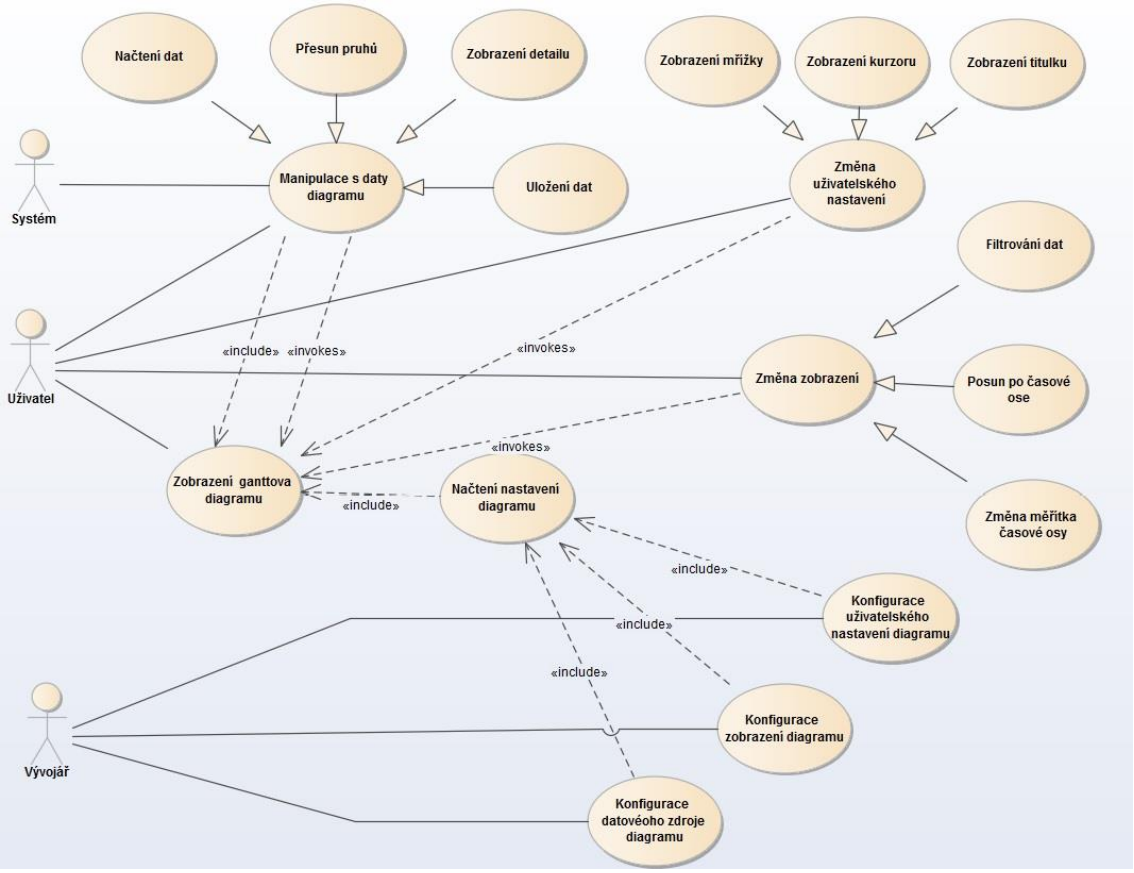


Obr. 8 Návrh komponentového diagramu

Současný systém bude rozšířen o nové rozhraní Gantt data, pro obsluhu požadavků klientské části. Tím dojde k umožnění komunikace mezi klientskou a serverovou částí komponenty. Samotná komponenta Ganttův diagram bude tvořena JavaScriptovou knihovnou zajišťující funkcionalitu, CSS styly pro definici vzhledu HTML elementů diagramu. K vykreslení diagramu dojde pomocí zobrazení webové stránky pomocí internetového prohlížeče.

4.3.2 Diagram případů užití

Cílem diagramu případů užití je grafickou formou popsat funkčnost aplikace v rámci systému.



Obr. 9 Diagram případů užití

Diagram rozlišuje mezi třemi entitami v roli aktérů, kteří využívají nebo obsluhují určitou funkcionalitu:

- uživatel systému,
- vývojář,
- systém.

V roli uživatele vystupuje entita využívající komponentu nejvíce. Hlavní činností je zobrazení diagramu. Při zobrazení diagramu musí nejprve dojít k načtení nastavení komponenty a načtení dat. V případě provedení změn uživatelem je vyvoláno nové vykreslení diagramu.

S vykresleným diagramem může uživatel různě manipulovat – přesouvat jednotlivé pruhy, měnit zobrazení a v případě aktivní volby ukládat provedené

změny v diagramu. Uživatel si může také zobrazit detailní informace o pruhu a provádět změny v nastavení diagramu týkající se zobrazovaných dat.

Vývojář nastavuje výchozí zobrazení diagramu pro uživatele a funkcionalitu, kterou uživatel může využívat při práci s diagramem. Konfigurace datového zdroje umožňuje nastavit *URL* webových služeb pro propojení manipulaci s daty.

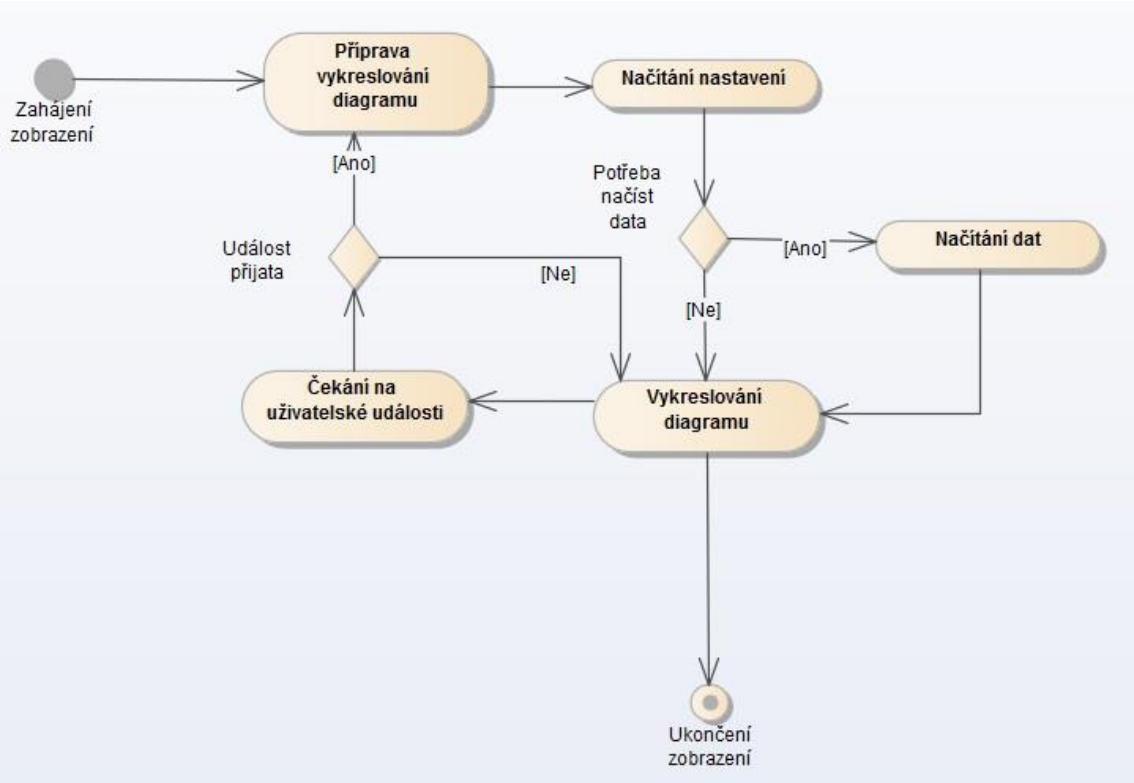
Úlohou současného systému je zejména poskytování datového propojení a obsluha klientských požadavků.

Tab. 1 Textová specifikace případu užití zobrazení Ganttova diagramu

Název	Zobrazení Ganttova diagramu		
ID	UC 01		
Popis	Cílem je vygenerovat obsah pro zobrazení Ganttova diagramu		
Primární aktéři	Uživatel		
Sekundární aktéři	Systém, vývojář		
Vstupní podmínky	Funkční propojení s databází Povolený JavaScript ve webovém prohlížeči		
Výstupní podmínky	Existující data k zobrazení		
Scénář	Číslo	Aktér	Akce
	1	Uživatel	Zobrazit Ganttův diagram
	2	Uživatel	Inicializace vykreslujícího objektu
	3	Uživatel	Načtení uživatelských nastavení
	4	Systém	Načtení dat
	5	Uživatel	Vygenerování obsahu
	6	Uživatel	Vykreslení diagramu
	7	Uživatel	Ukončit vykreslení
Alternativní scénář	6a	Uživatel	Zavření okna THEN č.7
	6b	Uživatel	Změna zobrazení THEN č. 3
	6c	Uživatel	Změna v datech THEN č. 4
Chybová hlášení	Číslo	Znění	
	1	Nefunkční spojení se serverem	

4.3.3 Stavový diagram

Stavový diagram znázorňuje životní cyklus objektu pro zobrazení Ganttova diagramu.

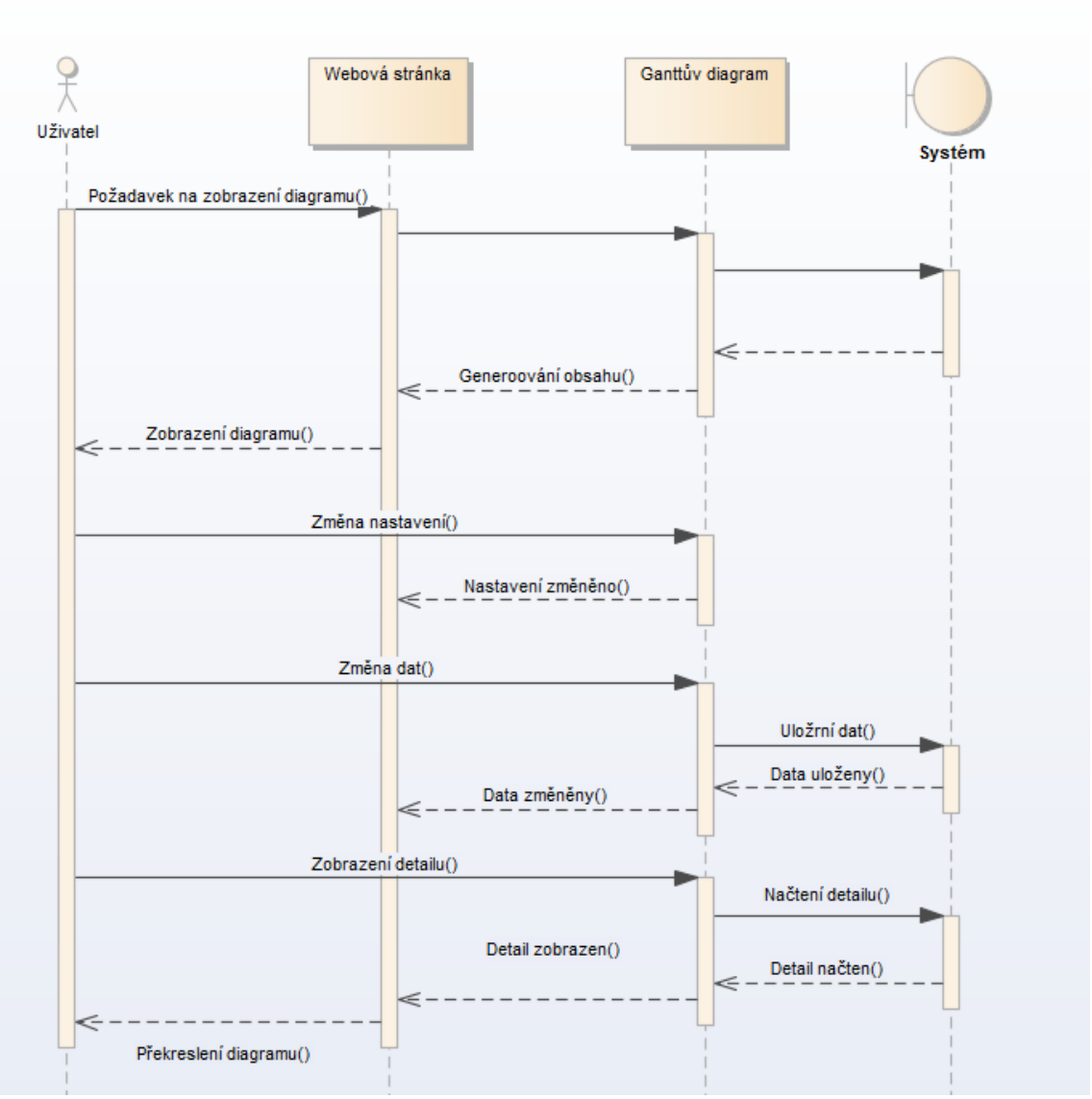


Obr. 10 Stavový diagram vykreslení Ganttova diagramu

Při zahájení zobrazení Ganttova diagramu dojde v první fázi k vytvoření vykreslovacího objektu. Vytvořený objekt nejprve načte nastavení z HTML definice příslušného elementu webové stránky. V další fázi, pokud je potřeba proběhne získání dat ze systému určených pro vykreslení. Posléze může dojít k samotnému vykreslení diagramu. Objekt, který zajišťuje vykreslování Ganttova diagramu je připraven reagovat na uživatelské akce – měnit nastavení, upravovat data a pohybovat se v diagramu. Každá událost spustí celý cyklus znovu, dojde k překreslení diagramu pro okamžitému projevení změn.

4.3.4 Sekvenční diagram

Diagram interakcí popisuje vzájemnou spolupráci skupiny objektů umožňující vykreslit Ganttův diagram.



Obr. 11 Sekvenční diagram vykreslení Ganttova diagramu

Proces je inicializován uživatelem, v okamžiku načtení webové stránky dojde k vytvoření objektu umožňující vykreslení Ganttova diagramu. První činností tohoto objektu je načtení dat ze serveru za účelem vygenerování obsahu diagramu, který bude předán do webové stránky, a tam vykreslen uživateli. Uživatel může vyvolávat události, vyvolávající příslušnou akci. Při změně dat, uživatelského

nastavení, či požadavku na zobrazení detailu dojde k překreslení diagramu, respektive provedení příslušné operace s daty.

4.3.5 Návrh uživatelského rozhraní

Cílem grafického návrhu uživatelského rozhraní je vytvořit představu o výsledném vzhledu aplikace, který bude možné prezentovat zadavateli. Výsledný vzhled uživatelského rozhraní se však může lišit, neboť v dalších fázích mohou být objeveny skutečnosti, které bude nutné zakomponovat do návrhu.

Obr. 12 Návrh uživatelského rozhraní

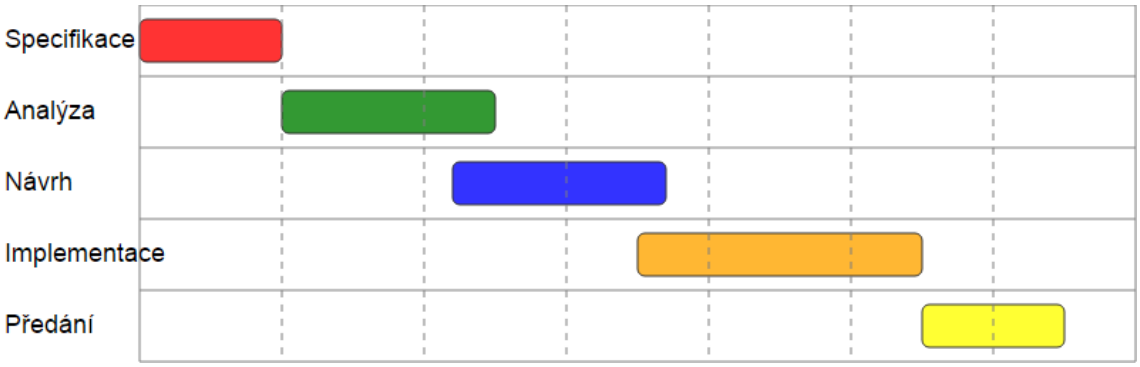
Obrázek znázorňuje uživatelské rozhraní komponenty, význam jednotlivých oblastí je následující:

- Popis diagramu bude obsahovat text informačního charakteru definovaný vývojářem.
- Panel nástrojů bude sloužit uživateli ke spuštění akcí v rámci změn v zobrazení a manipulaci s daty.
- Časová osa bude obsahovat časové údaje pro snadnější navigaci uživatele v diagramu.
- Ve vykreslovací oblasti dojde ke grafické prezentaci dat ve formě Ganttova diagramu.

4.3.6 Technika zobrazení

Po zhodnocení jednotlivých možností pro vykreslování Ganttova diagramu, se jako nejlepší a technicky nejsprávnější řešení jeví použití svg. Není plánováno vykreslovat složité grafické objekty, proto se nevýhoda vyšší náročnosti při vykreslování složitých objektů nejeví jako opodstatněná. Může dojít k zobrazení více grafů na jedné stránce zobrazující velké množství dat. Z tohoto důvodu je vhodné v dalších fázích práce otestovat chování a určit tak celkovou použitelnost. Pomocí JavaScriptu dojde k naprogramování funkcionality rozšiřující svg element o vykreslování diagramu, zajištění interaktivity a komunikaci se serverem.

Scalable Vector Graphics je standart pro kreslení vektorové grafiky na webu. Grafické objekty jsou definovány ve formátu XML a zobrazeny v DOM (Document Object Model). Tomuto elementu můžeme definovat standardní atributy, v našem případě je to pouze definice výšky a šířky a identifikátoru. Uvnitř elementu je možné používat další tagy, jejichž kompletní výčet je k dispozici na W3C[4].



Obr. 13 Grafická prezentace Ganttova diagramu pomocí HTML

Ve vykreslovací oblasti znázorněné na Obr. 12 by mohlo dojít k zobrazení Ganttova diagramu jako výstupu *SVG* elementu, jehož podoba je nastíněna ukázkovým Obr. 13.

4.3.7 Volba nástrojů a technologií

Pro tvorbu interaktivní aplikace Ganttova diagramu formou webové aplikace je nutné vytvořit klientskou a serverovou část. Ke tvorbě webové stránky bude využito konvenčních nástrojů pro tvorbu webových aplikací. Webová stránka umožňující vykreslení veškerého obsahu bude vytvořena pomocí jazyku HTML ve standardu HTML5. Grafický vzhled jednotlivých elementů bude definován pomocí kaskádových stylů CSS. Interaktivitu a funkčnost komponenty zajistí skriptovací jazyk JavaScript zejména s využitím frameworku jQuery a jQuery UI. Zvolené technologie jsou široce používané a jejich podpora je zajištěna ve všech současných webových prohlížečích. Klientská strana bude volat serverové metody

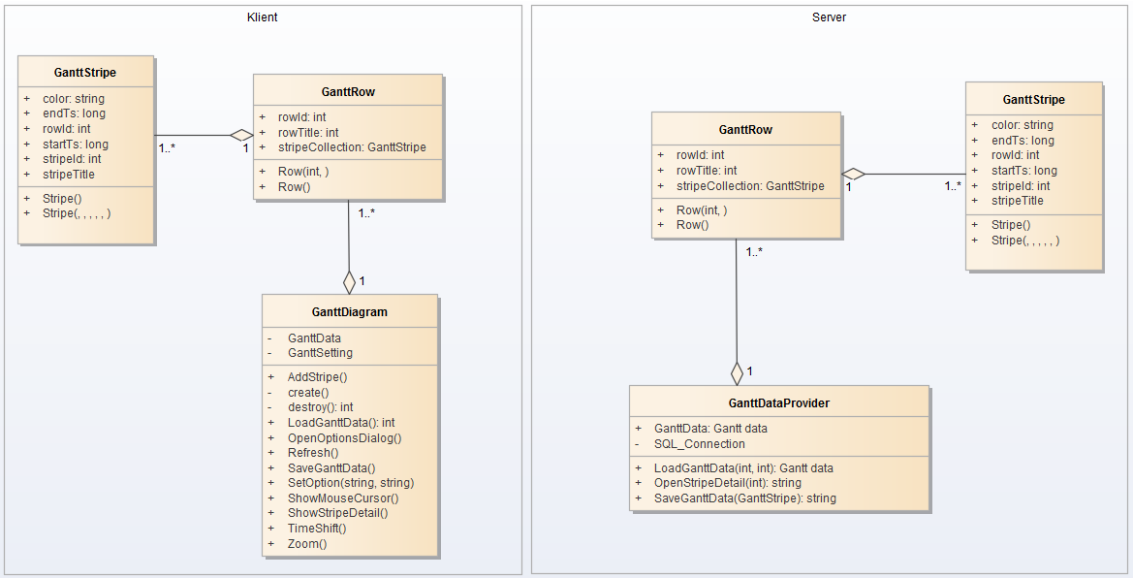
[4] *Scalable Vector Graphics, www.w3.org/TR/SVG/*

pomocí AJAXu, výměna dat mezi klientem a serverem bude probíhat ve formátu JSON.

V rámci této práce bude navrženo řešení pro ASP.NET za účelem možnosti demonstrace řešení v současném systému. Zvolený jazyk pro implementaci serverové strany v systému COMES bude C#. Přístup k datům na straně serveru bude realizován pomocí SQL (MS-SQL).

4.3.8 Diagram tříd

Model tříd slouží pro bližší popis zamýšleného řešení problematiky. Jsou zde modelovány jednotlivé třídy, jejichž budoucí implementací dojde k vytvoření programového řešení.



Obr. 14 Diagram tříd návrhu Ganttova diagramu

Diagram je rozdělený na dvě části – klient a server, v každé části jsou třídy implementovány jiným jazykem. Z důvodů nutnosti předávání dat mezi třídou *GanttDiagram* pro vykreslení a *GanttDataProvider* pro obsluhu požadavků klienta, je nutné vytvořit datovou strukturu obsahující kolekci řádků diagramu *GanttRow*. Třídy *GantStripe* a *GanttRow* slouží pro uchování dat v paměti programu a přenos zpráv mezi klientskou a serverovou částí aplikace. Předávání datových objektů bude realizováno pomocí JSON.

4.3.8.1 Třída GanttDiagram

Pro vykreslení obsahu diagramu slouží třída GanttDiagram. Tato třída slouží pro vykreslení Ganttova diagramu klientské části aplikace. Bude implementována pomocí JavaScriptu. Význam jednotlivých attributů třídy ovlivňující zobrazení Ganttova diagramu:

- *Id* – slouží k jednoznačné identifikaci elementu v rámci webové stránky.
- *Class* – umožňuje zařazení elementu do třídy, zejména příslušnost ve třídě *GanttArea* je důležitá. Znamená, že element slouží pro vykreslení Ganttova diagramu
- *Width*, *height* – definují velikost (šířku a výšku) oblasti pro vykreslení Ganttova diagramu.
- *GanttTitle* obsahuje textovou hodnotu použitou jako titulek při vykreslení diagramu.
- *ganttLoadDataUrl* URL adresa služby pro obsluhu požadavku pro načtení dat diagramu.
- *ganttSaveDataUrl* URL adresa služby pro obsluhu požadavku na uložení dat diagramu.
- *stripeDetailUrl* URL adresa pro případné zobrazení detailu pruhu.
- *viewStartTs* určuje počáteční čas pro zobrazení diagramu.
- *viewEndTs* nastavuje koncový čas pro zobrazení diagramu.
- *startTs* počáteční čas filtru pro načtení dat.
- *endTs* koncový čas filtru pro načtení dat.
- *showGrid* nastaví stav zobrazení mřížky.
- *showMouseCursor* nastaví stav zobrazení kurzoru myši.
- *showStripeTitle* nastaví stav zobrazení popisu pruhů diagramu.
- *enableDialog* umožňuje zobrazit detail činnosti prezentované pruhem.
- *enableOption* umožní provádět změny nastavení uživatelem.
- *enableDragDrop* určuje stav volby přetahování pruhů.
- *enableGridSnap* určuje stav možnosti přichycení pruhu k mřížce při aktivním přetahování.
- *enableEdit* nastavení volby ukládání změn.

Význam jednotlivých metod třídy GanttDiagram:

- *Create* slouží jako konstruktor třídy, umožní nastavení hodnot objektu, volitelné atributy jsou načteny z HTML definice elementu, v případě neexistence je nastavena výchozí hodnota.
- *Refresh* umožňuje překreslení diagramu. Logika této metody zajistí vytvoření obsahu diagramu pomocí HTML elementů. Pro správné vykreslení musí dojít k výpočtu hodnot vlastností jednotlivých obsahových elementů a nastavení příslušné interaktivní funkcionality.
- *StripeDetail* je metoda, která zajistí otevření detailu pruhu. Detail pruhu bude zobrazen uvnitř dialogu pomocí HTML.

- *AddStripe* slouží pro přidání nového pruhu. Událost bude probíhat obdobným způsobem, jako otevření detailu avšak s rozdílným účinkem.
- *OpenOptionsDialog* slouží k otevření dialogu pro změnu nastavení vlastností diagramu uživatelem.
- *SetOptionsDialog* provede samotné nastavení hodnoty příslušného klíče.
- *TimeShiftLeft* provede posun časové osy diagramu vlevo.
- *TimeShiftRight* provede posun časové osy vpravo.
- *ZoomIn* zmenšení rozsah časové osy.
- *ZoomOut* zvětší rozsah časové osy.
- *MouseMove* událost při pohybu myši slouží k získání současných souřadnic myši pro vykreslení kurzoru.
- *SaveGanttData* provede uložení změněného pruhu diagramu.
- *LoadGanttData* provede získání kolekce dat do paměti programu.
- *Destroy* slouží jako destruktor třídy, v případě jeho zavolání dojde k vymazání třídy a vykresleného HTML obsahu.

4.3.8.2 GanttDataProvider

Třída *GanttDataProvider* slouží pro obsluhu klientských požadavků, jednotlivé metody budou realizovány jako webová služba. Tato třída bude implementována jazykem, který je použit na serveru – v našem případě je to C#.
Tato třída obsahuje následující vlastnosti:

- *GanttData* slouží k uchovávání datové složky diagramu.
- *SQL_Connection* obsahuje řetězec pro vytvoření připojení k databázi.

Význam metod:

- *LoadGanttData* slouží pro obsluhu požadavku pro načtení dat.
- *SaveGanttData* obsluhuje požadavek na uložení pruhu.
- *OpenStripeDetail* poskytuje data pro otevření detailu pruhu.

4.3.8.3 GanttRow

GanttRow popisuje řádek diagramu, tato třída slouží pro tvorbu interní datové struktury prezentující data diagramu v programu. Třída nemá žádné metody, slouží pouze pro uchovávání informací, které prezentují jednotlivé atributy:

- *rowId* umožňuje identifikaci řádku, které je dále využívána v programu.
- *rowTitle* umožňuje vložit textový popis řádku.
- *stripeCollection* kolekce jednotlivých činností v řádku prezentovaných pruhy.

4.3.8.4 GanttStripe

Třída *GantStripe* prezentuje elementární činnost znázorněnou pruhem v diagramu. Tato třída obsahuje také pouze vlastnosti, jejichž význam je následující:

- *color* umožňuje přiřadit textový název CSS třídy, která určí barvu pruhu.
- *startTs* údaj určující počáteční datum a čas operace.
- *endTs* údaj určující koncový datum a čas operace.
- *rowId* identifikátor řádku do kterého přísluší pruh, umožňuje vykreslení v konkrétním řádku diagramu. Údaj je také využíván při provádění aktualizace v interním datovém uložišti serveru.
- *stripeId* identifikátor pruhu (činnosti) shodný s identifikátorem v interním uložišti na serveru pro možnost provedení změn.
- *stripeTitle* umožňuje definovat textový popis pruhu.

4.3.9 Propojení klienta a serveru

Server implementuje rozhraní pro obsluhu požadavků volaných klienty. Komunikace klienta se serverem bude probíhat pomocí protokolu HTTP využitím rozhraní XmlHttpRequest. Klient bude přistupovat k rozhraní pomocí AJAXových funkcí. Samotná výměna dat bude realizována ve formátu JSON, který umožňuje převést objekty na text. Obsluhu požadavků na straně serveru bude zajišťovat webová služba naprogramovaná pomocí jazyka C# na platformě ASP.NET.

Obr. 15 Návrh komunikačního rozhraní

Obrázek blíže popisuje komunikační rozhraní klienta a serveru.

4.3.9.1 Načtení dat

Primární funkcí rozhraní je umožnit načtení dat. Dojde získání dat z datového uložiště serverového řešení a jejich převedení do specifikovaného formátu vyžadovaného pro komunikaci. Tato funkce je stěžejní, neboť by bez ní nešlo získat data pro vykreslování.

Funkce bude vyžadovat dva parametry – počáteční a koncový čas, které budou použity jako filtr pro vybírání dat z databáze. Dojde k výběru takových položek, které budou splňovat zadané podmínky. Odpovědí bude datová struktura obsahující položky pro vykreslení, které klientský program již bude schopen zobrazit.

4.3.9.2 Uložení dat

V případě aktivní volby uložení dat musí být implementována funkčnost pro uložení údaje zaslaného požadavkem. Tentokrát musí dojít k převodu z formátu využitého pro přenos dat do interní datové struktury s následnou aktualizací údaje v datovém uložišti.

Parametrem této funkce bude objekt se změněnými hodnotami, který je určený k aktualizaci. Pomocí identifikátoru bude nalezen původní záznam a provedena aktualizace jeho hodnot. Odpovědí bude výsledek operace.

4.3.9.3 Otevření detailu

Otevřením detailu pruhu dojde k zobrazení dodatečných informací činnosti. Tyto údaje však nejsou potřebné pro zobrazení Ganttova diagramu, a proto nebudou přenášeny ve funkci pro *načtení dat*. Konkrétní obsah zobrazované struktury může být neobvyklého formátu, proto bylo rozhodnuto pro vytvoření HTML stránky poskytující obsluhu činnost pro zobrazení dodatečných údajů o činnosti. Klient pak zobrazí HTML kód stránky detailních údajů pruhu uvnitř obsahu stránky vykreslující diagram. To umožní zachovat vlastní způsob prezentování dat a provádění změn. Zobrazení dat může umožňovat okamžité provedení úprav v případě implementace této funkčnosti.

4.3.9.4 Návrh komunikace

Komunikace mezi klientskou a serverovou částí bude probíhat výměnou zpráv obsahující objekty JSON předávané v textové proměnné jako parametr funkce nebo její návratová hodnota. Následující ukázka struktury objektu JSON není zcela univerzální, ale jako příklad pro ukázku je velmi výstižný.

```
{
status: „succes“,
message: „Popis …“,
data: {"1":{
        "rowId":1,
```

```
        "rowTitle":"Specifikace",
        "stripes":{
            "11":{
                "stripeId":11,
                "stripeTitle":"Specifikování požadavků",
                "startTs":1451606400000,
                "endTs":1451952000000,
                "color":"red"},
             "13":{
                 "stripeId":13,
                 "stripeTitle":"Analýza řešení 2",
                 "startTs":1451692800000,
                 "endTs":1451779200000,
                 "color":"blue"}
              }
         }
    }
}
```

V případě obdržení této zprávy zjistíme, že operace proběhla úspěšně, z datové složky dojde k získání objektu *GanttRow* s příslušnými hodnotami.

Následující kód popisuje strukturu ukázkového objektu JSON:

- *Status* zprávy může nabýt hodnoty *succes* v případě úspěchu, nebo *failed* při neúspěchu. Důležitá je pro získání informace o výsledku operace.
- Hodnota skrytá pod klíčem *message* slouží jako doplňková zpráva odpovědi. Může například blíže popisovat důvod neúspěchu, při selhání načtení dat.
- Položka *data* obsahuje datovou složku zprávy. V případě přenosu dat je zde zasílán objekt typu *GanttStripe* nebo kolekce objektů *GanttRow* v závislosti na probíhající operaci. Tyto objekty popisuje diagram tříd v sekci návrhu, pomocí JSON dojde k jejich převedení do textové podoby.

4.3.10 Vytvoření diagramu

Stěžejní myšlenkou je navržení HTML elementu, který bude sloužit jako oblast pro vykreslení Ganttova diagramu. Uvnitř elementu dojde definování vlastnosti ovlivňující způsob zobrazení a manipulaci s diagramem. Po načtení webové stránky dojde ke spuštění JavaScriptového kódu, který načte definované vlastnosti elementu a na jejich základě vytvoří programový objekt pro dynamické vykreslení obsahu diagramu. K vykreslení diagramu dojde uvnitř elementu pomocí svg. Veškerá logika problematiky zobrazení – interakce s uživatelem a komunikace se serverem bude realizována v JavaScriptové knihovně.

```
<div class="GanttArea" id="" width="" height="" GanttTitle="" showGrid=""
showMouseCursor="" enableDialog="" enableOption="" enableDragDrop=""
enableEdit=""  showStripeTitle="" enableGridSnap="" ganttLoadDataUrl=""
ganttSaveDataUrl="" stripeDetailUrl="" viewStartTs="" viewEndTs="" startTs=""
endTs="" />
```

Pro inicializaci diagramu byl zvolen blokový HTML element div. Na základě definovaných atributů tohoto elementu dojde k nastavení vlastností objektu uvnitř programu. Tyto vlastnosti jsou definovány vývojářem, a umožňují nastavit, jakou funkčnost může uživatel využívat v aplikaci.

4.4 Implementace

Hlavní problematiku řešení tvoří zobrazení dat ve formě Ganttova diagramu. V předchozí kapitole již došlo k nastínění návrhu tvorby řešení, nyní musí dojít k naprogramování jeho funkčnosti pomocí vybraných technologií a nástrojů. Tím bude umožněno ověření správnosti návrhu. Implementace rozhraní serverové části však závisí na konkrétním případě užití – realizace serverové části proběhne, až bude známo jaká data a jakým způsobem budou zobrazena.

4.4.1 Programová prezentace Ganttova diagramu

Třída pro vykreslení Ganttova diagramu bude implementována pomocí JavaScriptového kódu. Pro usnadnění implementace bude použito frameworku jQuery k vytvoření pluginu umožňující vykreslování diagramu. Programový objekt bude procházet několika fázemi, tvorba takového pluginu vyžaduje zápis mnoha řádků již existujícího kódu. Framework JQuery UI disponuje nástrojem pro usnadnění tvorby stavových pluginů jménem *Widget Factory*. Takto vytvořený plugin umožňuje obdobné použití jako u ostatních známých objektů z knihovny jQuery.

```
$(function () {
    $.widget("custom.gantt", {// jmenný prostor.název widgetu
        // inicializace proměnných
        options: {
            ...  // proměnné jsou uzavřeny v options
        },
        _create: function () {...},    // konstruktor
        _refresh: function () {...},// vykreslení diagramu
        timeShiftLeft: function () {...},
        timeShiftRight: function () {...},
        zoomIn: function () {...},
        zoomOut: function () {...},
       _destroy: function () {...}     // destruktor
        ...  // další funkce
       });
});
```

Ukázka kódu je neúplnou kostrou vlastní implementace widgetu vykreslujícího diagram. Ve skriptu je z důvodů rozsáhlosti implementace zobrazenou pouze hlavička několika nejdůležitějších metod. Pro zajištění plné funkcionality je potřeba implementovat všechny důležité funkce a vlastnosti popsané viz kapitola 4.3.84.3.8 Diagram tříd, i další pomocné funkce v návrhu nezmíněné. Jedná se zejména o podpůrnou funkcionalitu zajišťující generování opakujícího se obsahu HTML, nebo přepočet časových údajů do souřadnicového systému. Jednotlivé

atributy a metody třídy pro vykreslení byli definovány již v návrhové kapitole, zde dochází pouze k jejich implementaci. Kompletní skript (viz přílohy) již dokáže vykreslit data do podoby Ganttova diagramu nastíněného viz Obr. 13.

4.4.1.1 Vykreslování

Stěžejní funkcí komponenty je vykreslení dat do podoby Ganttova diagramu. Tuto činnost provádí metoda *_refresh* během aktualizace obsahu diagramu.

```
_refresh: function () {
    //provede se vymazání obsahu
    //vytvoření nového statického obsahu
    for (var row in GanttData)
        {//prochazení datové složky programu
        for (var stripe in row.stripes) {
            //výpočet umístění pruhu
            //generování HTML obsahu pro vykreslení a vložení do obsahu
stránky
        }
    }
    //nabindování událostí
}
```

Ukázka je pouze komentovaný výřez funkce, která je kompletní k nahlédnutí v přílohách. Funkce vytváří dynamický HTML obsah vykreslující Ganttův diagram, který je vložen do obsahu webové stránky pomocí JavaScriptu. V případě úspěšného načtení dat dojde k vykreslování diagramu postupným procházením vnitřní datové struktury. Na prvky obsahu, které disponují speciální funkcionalitou, je nutné navázat příslušné události. Každý vykreslovaný řádek diagramu je prezentován třídou *GanttRow* který obsahuje činnosti prezentované jednotlivými pruhy – objekty *RowStripe*. Při každém požadavku na překreslení je potřeba vyvolat tuto funkci pro aktualizaci obsahu.

4.4.1.2 Komunikace

Důležitou součástí programu je schopnost komunikace klienta se serverem. K tomu slouží metody *saveGanttfunData* a *loadGanttData,* které zasílají požadavky na server pomocí AJAXU.

```
loadGanttData: function () {
    var gantWidget = this;
    var param = {startDate:startTs, endDate:endTs};// parametry
    $.ajax({
        type: "POST",
        url: gantWidget.ganttLoadDataUrl, //URL webové služby
        data: JSON.stringify(param), //převod dat na JSON
        contentType: "application/json; charset=utf-8",
        dataType: "json",
```

```
        success: function (data) { //při úspěchu
            var obj = data.d;
            gantWidget.onDataObtain(obj); //provede příslušnou akci
        },
        error: function (r) { //při chybě
            gantWidget.onDataFailed(r); //provede příslušnou akci
        }
    });
},
```

Ukázkový kód metody *loadGanttData* slouží k vytvoření AJAXového požadavku pro načtení dat diagramu. Serverové službě definované v parametru *ganttLoadDataUrl* HTML elementu bude zaslána zpráva obsahující jednotlivé parametry metody. Na základě úspěchu či neúspěchu dojde k vyvolání příslušné akce a předání zprávy programu.

4.4.2 Výstup programu

Pomocí objektu *GanttDiagram* dojde k vygenerování obsahu diagramu. Grafická podoba výstupu bude popsána v následujících ukázkách. Programem vygenerovaný HTML kód poskytne po spuštění výstup v podobě vykresleného Ganttova diagramu.

4.4.2.1 HTML šablona

V programu dojde k vytvoření HTML kódu, jehož vykreslením dojde k zobrazení Ganttova diagramu. Výstupem programu je HTML kód podobný tomu v následující ukázce, kompletní kód je k nahlédnutí v příloze. Cílem je prezentovat vygenerovanou HTML strukturu elementu *SVG* sloužící pro prezentaci dat diagramu.

```
<svg id="Gantt" width="850" height="320" >
   <%-- Pruhy --%>
   <rect x="100" y="60" width="100" class="stripe red"></rect>
   <rect x="200" y="110" width="150" class="stripe green"></rect>
   <rect x="320" y="160" width="150" class="stripe blue"></rect>
   <rect x="450" y="210" width="200" class="stripe orange"></rect>
   <rect x="650" y="260" width="100" class="stripe yellow"></rect>
   <%-- Linky ohraničení --%>
   <line x1="100" y1="50" x2="100" y2="300" class="borderLine" />
   <line x1="800" y1="50" x2="800" y2="300" class="borderLine" />
   <line x1="100" y1="50" x2="800" y2="50" class="borderLine" />
   <%-- Řádky--%>
   <line x1="100" y1="100" x2="800" y2="100" class="borderLine" />
   <line x1="100" y1="150" x2="800" y2="150" class="borderLine" />
   <line x1="100" y1="200" x2="800" y2="200" class="borderLine" />
   <line x1="100" y1="250" x2="800" y2="250" class="borderLine" />
   <line x1="100" y1="300" x2="800" y2="300" class="borderLine" />
```

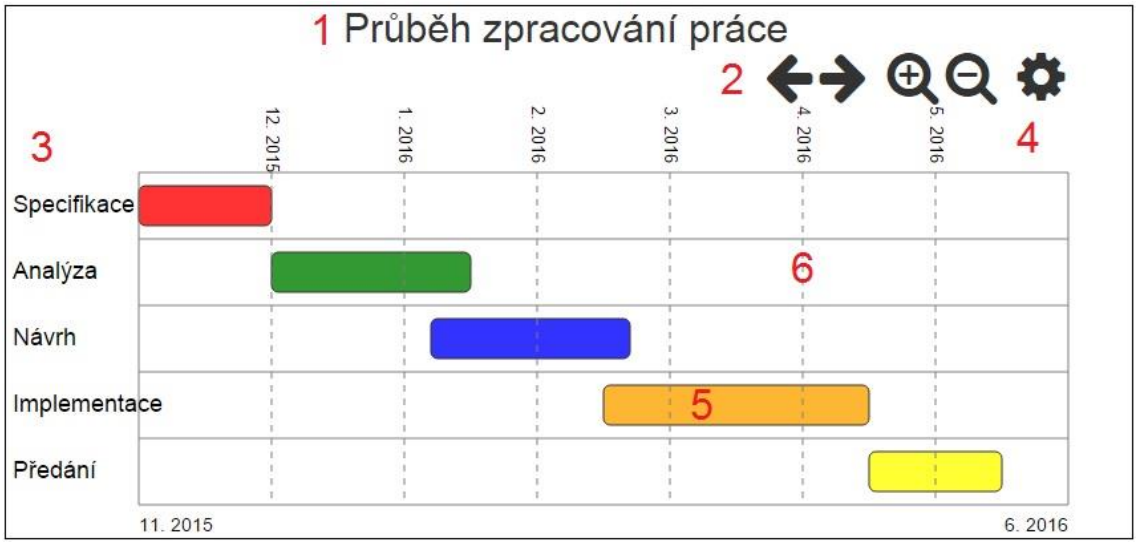
```
    <%--linky mřížky--%>
    <line x1="200" y1="50" x2="200" y2="300" class="gridLine" />
    <line x1="300" y1="50" x2="300" y2="300" class="gridLine" />
    <line x1="400" y1="50" x2="400" y2="300" class="gridLine" />
    <line x1="500" y1="50" x2="500" y2="300" class="gridLine" />
    <line x1="600" y1="50" x2="600" y2="300" class="gridLine" />
    <line x1="700" y1="50" x2="700" y2="300" class="gridLine" />
    <%-- Popis osy Y --%>
    <text x="5" y="80"  class="rowTitle">Specifikace</text>
    <text x="5" y="130" class="rowTitle">Analýza</text>
    <text x="5" y="180" class="rowTitle">Návrh</text>
    <text x="5" y="230" class="rowTitle">Implementace</text>
    <text x="5" y="280" class="rowTitle">Předání</text>
</svg>
```

Pro účely práce zde bude zobrazen zjednodušený Ganttův diagram, s popisem významu jednotlivých tagů elementu:

- *SVG* definuje oblast pro kreslení vektorové grafiky.
- *Rect* znázorňuje jednotlivé pruhy, které prezentují činnosti. Tento element se vykreslí jako obdélník, pozice jeho vykreslení závisí na vlastnostech *x* a *y*. Další vlastnost *width* určuje šířku a *height* výšku obdélníku. Jednotlivé hodnoty atributů tagu jsou vypočítány uvnitř programu. Pro příjemnější grafickou prezentaci mají obdélníky zaoblené rohy pomocí vlastnosti *rx*. Zařazením elementu do třídy *stripe* umožníme přiřadit elementu další grafické efekty pomocí kaskádových stylů CSS.
- *Text* umožňuje vykreslit textovou hodnotu uvnitř elementu svg. Slouží zejména pro popis pruhů a řádků. Pozici vykreslení textu nastavujeme opět pomocí vlastností x a y, spočítanou uvnitř programu. Dále můžeme nastavovat standardní vlastnosti jako je velikost, font či barvu textu pomocí, zde však jsou nastavené pomocí stylu CSS.
- *Line*, slouží pro vykreslení čáry využívané zejména pro oddělení jednotlivých tagů a vytvoření podpůrných zobrazovacích prostředků. Konkrétní použití bude pro vykreslení ohraničení diagramu, oddělení řádků a vertikální mřížky.

4.4.2.2 Vykreslení šablony v prohlížeči

Spuštěním HTML kódu z předchozí ukázky dojde k vykreslení Gantova diagramu v následující grafické podobě. Obsah diagramu je vykreslován do příslušné oblasti, určené k vykreslení diagramu viz Obr. 12.



Obr. 16 Grafický návrh uživatelského rozhraní

Na obrázku jsou zobrazeny hlavní viditelné prvky komponenty Ganttova diagramu, tvořící uživatelské rozhraní.

1. Nadpis diagramu slouží k možnosti popisu diagramu a také odlišení od ostatních částí stránky, ev. dalšího Ganttova diagramu.
2. Panel nástrojů umožňuje pomocí ovládacích prvků ovládání Ganttova diagramu a otevřít panel pro nastavení Ganttova diagramu. Uvnitř scriptu dojde k implementaci obslužných funkcí pro vzniklé události při kliku na příslušný prvek.
3. Popisky vertikální osy jsou údaje, pomocí kterých je umožněno uživateli rozlišovat mezi jednotlivými řádky.
4. Popisky horizontální osy jsou časové údaje, pomocí kterých je uživatel schopný se orientovat v diagramu.
5. Pruhy diagramu reprezentují jednotlivé činnosti. Každý pruh může být vykreslen pomocí určité barvy definované programátorem. Po kliknutí na pruh dojde k zobrazení detailu činnosti. Uvnitř pruhu může být zobrazen text popisující příslušnou činnost. Pruhy disponují funkčností *chyť a táhni* pro umožnění editace diagramu.
6. Mřížka umožňuje lepší orientaci v časových údajích diagramu. Při aktivní možnosti přichytávání k mřížce dojde k položení přesouvaného pruhu na patřičnou čáru mřížky.

4.4.2.3 CSS styly

Každý element může být zařazen do třídy, pro kterou bude uvnitř souboru CSS definován styl zobrazení. Některé vlastnosti nemohou být zapsány přímo uvnitř CSS, neboť v případě zobrazení v Internet Exploreru nemusí být vykreslen. Tato skutečnost byla zjištěna u vlastnosti *rx*, která nastavuje zaoblení hrany.

Následující ukázka CSS stylu popisuje některé (zejména společné) vlastnosti zobrazení jednotlivých elementů. V originálním souboru jsou vlastnosti zobrazení definovány pro každou třídu zvlášť viz příloha.

```
fill: Black; /*Vyplnění černou barvou*/
stroke:black; /*Barva ohraničení*/
stroke-width:1; /*Šířka ohraničení*/
opacity:0.8; /*Průhlednost*/
height:30px; /*Výška*/
width:30px; /*Šířka*/
```

4.4.3 Implementace uživatelských požadavků

Při implementaci funkcionality komponenty Ganttova diagramu, dojde k naplnění zadavatelem definovaných požadavků. Programové metody jsou vytvořeny za účelem splnění požadovaných možností zobrazení diagramu v klientské části aplikace. Tato sekce popisuje, jakým způsobem byli splněny hlavní uživatelské požadavky.

4.4.3.1 Možnost zvětšovat, zmenšovat Gantt diagram

Zvětšování měřítka časové osy je realizováno pomocí funkcí *ZoomIn* a *ZoomOut*. Funkce je volána událostí vzniklou při kliku na příslušnou ikonu v panelu nástrojů ev. vertikálním pohybem kolečka myši v oblasti diagramu, obdobným způsobem jako u většiny aplikací. V případě spuštění této funkce dojde ke změně rozsahu časové osy v závislosti na operaci (přiblížení/oddálení). Celková doba časové osy bude zvětšena respektive zmenšena o jednu desetinu svého rozsahu.

```
toolBar.find("zoomIn").bind('click',function(event) {gantt.zoomIn()});
```

Událost je spuštěna klikem na tlačítko pro přiblížení, respektive oddálení zobrazení.

```
element.find(".gantt-draw-area").bind('mousewheel', function (event) {
        if (event.originalEvent.wheelDelta == 120)
             gantt.zoomIn();
        else if(event.originalEvent.wheelDelta == -120)
             gantt.zoomOut();
});
```

Předchozí ukázka byla svázána s ovládacím tlačítkem diagramu, tato však nastavuje událost při pohybu kolečka myši.

4.4.3.2 Možnost časové navigace v Gantt diagramu

Posun časové osy je zajištěn pomocí funkcí *ShiftLeft* a *ShiftRight.* Při posunu osy dojde ke změně časové osy určitým směrem, zobrazovaná doba bude stejná. Posun bude probíhat směrem vlevo nebo vpravo o jednu desetinu zobrazované doby. Funkci vyvolá událost vzniklá při kliku na příslušné tlačítko v panelu nástrojů nebo horizontální posun kolečka myši.

```
toolBar.find("#shiftRight").bind('click',function(){
    gantt.timeShiftRight()
});
toolBar.find("#shiftLeft").bind('click',function(){
    gantt.timeShiftLeft()
});
```

4.4.3.3 Dynamické výšky řádku v diagramu

Na základě nastavené výšky diagramu a počtu pruhů dojde k výpočtu výšky vykreslovaných řádků. To platí v případě, že není implicitně definována výška řádku, pak takový výpočet bude přeskočen. Tento požadavek zajišťuje vykreslovací funkce s využitím vlastnosti objekt *rowHeight* a *height*.

4.4.3.4 Konfigurace pruhů Ganttova diagramu

Tato schopnost umožňuje vývojáři definovat vlastností způsob zobrazení jednotlivých pruhů. Vývojář může nastavit barvu pruhu, a zdali vůbec dojde k jeho vykreslení. Barvu prezentuje atribut *color,* hodnotou je textový název třídy *CSS*. Samotný způsob vykreslení je definován uvnitř CSS třídy, ev. může být vytvořen kdykoliv. Vlastnost pro zapnutí – vypnutí vykreslení se nastavuje vlastností *visible* objektu *GantRow*. Klient se postará o vykreslení na základě přiřazených hodnot.

4.4.3.5 Zobrazení vertikální mřížky

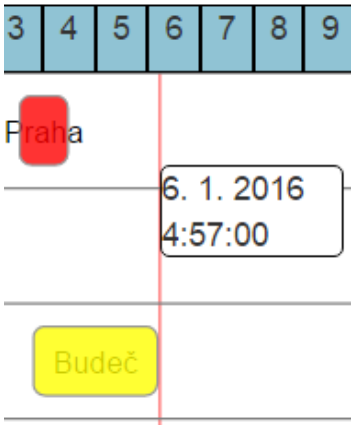
Při aktivním volbě nastavení zobrazení mřížky dojde k vykreslení vertikálních čar v oblasti grafu. Čáry budou vygenerovány v závislosti na zobrazované časové jednotce, kdy každá čára bude vykreslena na přelomu časových jednotek. V případě že zobrazujeme dny, dojde k zobrazení čáry na konci každého dne, při hodinových jednotkách na konci každé hodiny. Tato funkce usnadňuje uživateli orientaci na časové ose diagramu a v případě aktivní volby přichycení přesouvaného pruhu zpřesňuje ruční plánování.

```
if (options.showGrid)
{
    var gridLine = createLine(options.offsetLeft + aggrCX,
            options.offsetVertical, options.offsetLeft + aggrCX,
            options.rowsNum * options.rowHeight, "gridLine");
    gridLine.appendTo(drawArea);
}
```

Tento blok kódu uvnitř metody pro vykreslení obsahu Ganttova diagramu umožňuje vytvoření jednotlivých čar mřížky.

4.4.3.6 Kurzor myši

Funkce umožňuje vykreslit informace o současné poloze kurzoru myši vůči časové ose diagramu. V případě aktivní volby dojde k vykreslení údajů pomocí HTML kódu do obsahu diagramu, který je generován pomocí JavaScriptu uvnitř události vzniklé pohybem myši. Tato funkce je implementována v rámci požadavku na přívětivé uživatelské prostředí za účelem větší přehlednosti o pozici kurzoru myši v časové ose.



Obr. 17 Kurzor myši v Gantt diagramu

Funkce vykreslí pomocí HTML svislou červenou čáru, znázorňující aktuální pozici kurzoru myši na časové ose a pomocí textového okna rozšiřující informace o datu. Na základě souřadnice kurzoru myši dojde k výpočtu ekvivalentního data a vykreslení této hodnoty do příslušné oblasti.

4.4.3.7 Ruční plánování

Ruční plánování bude realizováno využitím přetahování pruhů pomocí myši. Při implementaci funkce je třeba dát si pozor, aby nebylo možné překrýt jednotlivé pruhy – nesmí dojít k zaplánování dvou činností ve stejném řádku, jejichž časový průběh se bude překrývat.

V případě stisknutí tlačítka myši nad pruhem dojde k jeho chycení, objekt bude možné přesouvat, dokud nebude tlačítku uvolněno. Uvolněním tlačítka myši bude položen přetahovaný objekt na aktuální pozici myši. V případě aktivního uživatelského nastavení ukládání dat dojde k aktualizaci pruhem reprezentované činnosti ve zdrojových datech. Tato funkce také implementuje chování, kdy dojde k přichycení k čáře mřížky v případě aktivní volby.

```
$(".stripe").draggable({
  start: function (event, ui) {},
  drag: function (event, ui) {},
  stop: function (event, ui) {}
  });
```

Přetahování pruhů je realizováno pomocí funkce z knihovny jQuery UI draggable. Tato funkce umožňuje využít událostí, které jsou nutné pro realizaci potřebného chování.

4.4.3.8 Detailní informace pruhu

Cílem požadavku je možnost zobrazení detailních informací pruhu. Spuštění funkce provádí událost při kliku na pruh. Následně dojde k otevření dialogu s obsahem stránky zobrazující detailní informace. Dialog zobrazuje HTML stránky

detailu uvnitř elementu *iframe*, tím si vývojář ponechá kontrolu nad způsobem zobrazení detailních informací.

```
    $(".stripe").bind("click", this.stripeDetail);
```

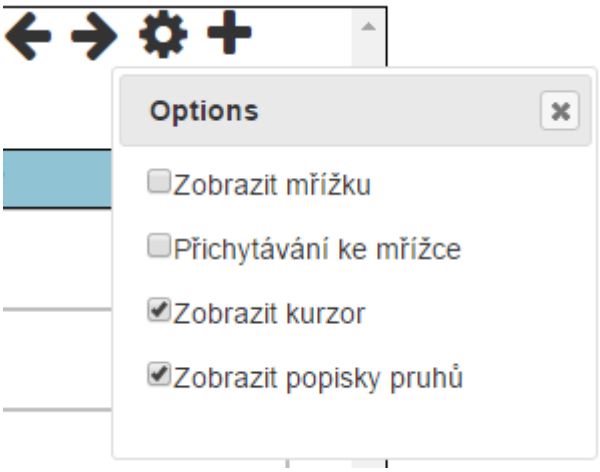
Přiřazení události pro otevření detailu při kliknutí na pruh.

```
stripeDetail: function (event) {
    var stripeId= $(this).attr("stripeId"));
    $("#detailUrl").attr("src", "StripeDetail?id=" +stripeId;
    $("#detailDialog").dialog("open");
}
```

Ukázkový blok kódu vytvoří odkaz na stránku zobrazující detailní údaje a následně otevře dialog.

4.4.3.9 Volitelnost možností nastavení

Tento požadavek je splněn vygenerováním ikony pro otevření okna nastavení v panelu nástrojů. V případě aktivní volby provádění uživatelských nastavení, dojde při kliknutí na ikonu nastavení k otevření dialogového okna, kde bude uživatel schopen provádět změny v zobrazení.



Obr. 18 Dialog uživatelského nastavení diagramu

Při spuštění některé z popisovaných činností dojde k překreslení diagramu. Z důvodů ovlivnění vzhledu zobrazení a chování diagramu, dojde k překreslení diagramu. Popisované funkce mají vliv na způsob zobrazení a chování vykreslovaného diagramu, proto je žádoucí aby se změny projevili okamžitě, a proto musí dojít k novému vykreslení.

4.5 Nasazení řešení

V této části práce bude popsán postup, pomocí kterého dojde k zavedení komponenty pro vykreslování Ganttova diagramu do současného systému. Hotové řešení se skládá z HTML stránky, JavaScriptové knihovny a CSS stylu, které je potřeba nahrát na server. Pro celkové ověření funkčnosti bude nutné vytvořit serverové rozhraní pro obsluhu požadavků klienta.

4.5.1 Zavedení

Zavedení vytvořeného řešení do systému COMES by nemělo být náročné. Zvolené technologie umožňují rozumnou přenositelnost. Ke zprovoznění klientské části bude stačit nahrát JavaScriptové knihovny a vytvořený kaskádový styl do systému COMES.

Aplikace využívá funkcionalitu knihovny jQuery a jejího rozšíření jQeury UI. Pro zajištění funkčnosti aplikace je nutné umožnit přístup programu také k těmto zdrojům. V případě existence knihoven jQuery (verze 1.6 a novější) a jQuery UI (verze 1.10 a novější) stačí pouze nastavit cestu k požadovaným souborům. V opačném případě je nutné nahrát chybějící knihovny do softwarových zdrojů serveru. V aplikaci je také použit styl *font-awesome*, díky kterému může aplikace využívat přehlednější ikony.

Obr. 19 Soubory pro nahrání (levá část obrázku), předpřipravená souborová struktura na serveru (pravá část obrázku)

Obrázek znázorňuje soubory pro nahrání (levá část obrázku) do vytvořeného souborového systému na serveru (pravá část obrázku).

Po nahrání souborů do systému COMES může dojít v libovolné webové stránce systému k vykreslení Ganttova diagramu, stačí pouze zapsat stanovenou HTML definici diagramu uvnitř obsahu stránky. Stránka pro vykreslování musí být

však propojena s příslušným JS a CSS souborem a JavaScriptovými knihovnami jinak nedojde k zajištění vyžadované funkcionality.

```
<script src="/Scripts/jquery.js" type="text/javascript"></script>
<script src="/Scripts/jquery-ui.js" type="text/javascript"></script>
<script src="/Scripts/GanttScript.js" type="text/javascript"></script>
<script src="/Scripts/JSTst.js" type="text/javascript"></script>
```

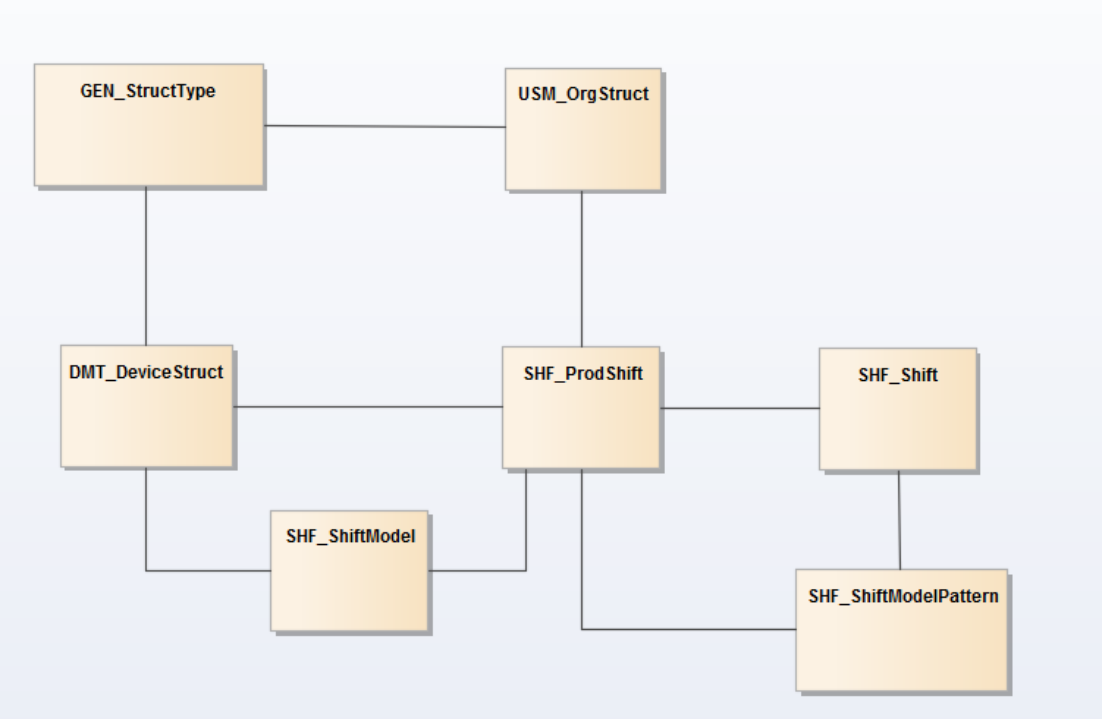
Nastavení cesty ke knihovnám je realizováno obdobně, jako přidání jakéhokoliv JavaScriptu v externím souboru do HTML stránky. Uvnitř HTML kódu stránky určené pro zobrazení Ganttova diagramu dojde ke vložení tagu *script*, jeho atribut *src* odkazuje na relativní umístění souboru. Pro správné fungování aplikace je nutné zachovat pořadí zápisu jednotlivých knihoven. Obdobným způsobem musí dojít také k nalinkování stylů CSS.

S využitím funkčnosti nahraných knihoven by již mohlo dojít k vykreslení Ganttova diagramu. Za tímto účelem je nutné definovat HTML element pro vykreslení Ganttova diagramu viz kapitola 4.3.10 Vytvoření diagramu. To zatím není možné, neboť není vytvořena přístupová stránka a také chybí služby poskytující data komponentě.

4.5.2 Výběr oblasti aplikování

Předem nebyla stanovena konkrétní oblast pro aplikaci řešení, neboť mělo být možné univerzálně vykreslit libovolná data. Zavedené řešení je třeba za účelem ověření funkčnosti otestovat a proto musí dojít k výběru konkrétní oblasti pro aplikaci Ganttova diagramu. Na základě bližší specifikace vykreslovaných dat dojde k implementaci webových služeb pro obsluhu požadavků klienta.

Ve stávajícím informačním systému dojde k aplikaci komponenty Ganttova diagramu na údajích o výrobních směnách. Data prezentují plánované obsazení strojů směnovými týmy v čase. Znalost datové struktury umožňuje transformovat zpracovávaná data z interního formátu na objekty umožňující přenos do klientské části.



Obr. 20 Databázová struktura tabulek z oblasti evidence výrobních směn

Obrázek znázorňuje část interní databázové struktury systému COMES (bližší specifikaci struktury včetně sloupců tabulky a datových typů lze nalézt v příloze), která je určena pro vykreslení formou Ganttova diagramu. Pro vytvoření diagramu je však možné použít libovolná data, neboť komponenta umožňuje vykreslení univerzálních datových struktur. Před zahájením vykreslování je nutné pro danou oblast vytvořit webovou službu umožňující transformaci a přenos dat do klientské části.

4.5.3 Implementace webových služeb

Pro uvedení aplikace do života schází poslední krok – implementace serverové služby pro obsluhu volání klientských požadavků. Při vytváření rozhraní vycházíme z již známé struktury a místa uložení dat v interním uložišti informačního systému. Účelem rozhraní je přemapovat data z databázového uložiště do vnitřní paměti programu, který již předá data v požadovaném formátu klientské aplikaci.

Pro zprovoznění vykreslování Ganttova diagramu je nutné implementovat rozhraní webové služby umožňující manipulaci s daty vybranými k tvorbě obsahu diagramu.

4.5.3.1 Načtení dat diagramu

Implementací metody *LoadGanttData* jako webové služby dojde k poskytnutí funkcionality pro načtení požadovaných dat a jejich převedení do podoby srozumitelné klientu.

Nejprve je nutné vytvořit SQL dotaz pro získání dat, která budou dále zpracována uvnitř metody.

	SHF_P...	SHF_ProdShift.BeginTime	SHF_ProdShift.EndTime	SHF_ProdShift.DMT_DeviceStructId.Name	SHF_ProdShift.SHF_ShiftId.Code	SHF_ProdShift.SHF_ShiftId.Name	SHF_Prod
1	2275	2015-12-07 05:00:00.0000000	2015-12-07 13:00:00.0000000	Lis AR 500/200 - 106	Shf_R	Směna R	237
2	2276	2015-12-07 13:00:00.0000000	2015-12-07 21:00:00.0000000	Lis AR 500/200 - 106	Shf_O	Směna O	237
3	2277	2015-12-07 21:00:00.0000000	2015-12-08 05:00:00.0000000	Lis AR 500/200 - 106	Shf_N	Směna N	237
4	2278	2015-12-08 05:00:00.0000000	2015-12-08 13:00:00.0000000	Lis AR 500/200 - 106	1	Smena 1_Fajtová	237
5	2279	2015-12-08 13:00:00.0000000	2015-12-08 21:00:00.0000000	Lis AR 500/200 - 106	2	Smena 2_Chadima	237
6	2280	2015-12-08 21:00:00.0000000	2015-12-09 05:00:00.0000000	Lis AR 500/200 - 106	3	Smena 3_Mrkosová	237
7	2281	2015-12-09 05:00:00.0000000	2015-12-09 13:00:00.0000000	Lis AR 500/200 - 106	Shf_R	Směna R	237
8	2282	2015-12-09 13:00:00.0000000	2015-12-09 21:00:00.0000000	Lis AR 500/200 - 106	Shf_O	Směna O	237
9	2283	2015-12-09 21:00:00.0000000	2015-12-10 05:00:00.0000000	Lis AR 500/200 - 106	Shf_N	Směna N	237

Obr. 21 Výstup SQL dotazu pro výběr dat

Obrázek znázorňuje výstup z vytvořené Výstupem SQL procedury umožňující načtení dat. Kompletní procedura je k nahlédnutí v přiložených souborech.

Získaná data je nutné převést do podoby čitelné klientovi a ty mu předat formou odpovědi na požadavek. To zajišťuje následující ukázkový kód funkce *LoadGanttData*.

```
[WebMethod]
public string LoadGanttData(string startDate, string endDate)
{
    //konverze parametrů funkce do SQL-kompatibilních
    DateTime? startTs = null;
    DateTime? endTs = null;
    //vyvolání procedury pro získání dat
    DataTable dt = SqlProc.GanttLoadShiftData.ExecuteTable( startTs, endTs );
    //datový typ prezentující datovou strukturu
    Dictionary<string, object> response = new Dictionary<string, object>( );
    Dictionary<string, Row> rows = new Dictionary<string, Row>( );
```

```
    foreach (DataRow tr in dt.Rows) //pro každý řádek získaných dat
    {
    Stripe stripe = new Stripe(...);
    //pokud ještě neexistuje řádek vytvoříme jej
    Row row = new Row( rowId, rowTitle );
    //vložíme vytvořené pruhy
    row.stripes.Add( stripe.stripeId.ToString( ), stripe );
    }
    return js.Serialize( response );//vytvoření JSON odpovědi
}
```

Při volání funkce je nutné předat parametry – počáteční a koncový čas, které jsou použity v SQL proceduře pro filtrování dat při výběru z databáze. Kód prochází získaná data řádek po řádku a vytváří interní datovou strukturu v podobě kolekce řádků *GanttRow* viz 4.3.8 Diagram tříd. Odpovědí je pak řetězec typu JSON, který obsahuje načtená data předávaná klientu. Podoba odpovědi je popsána v kapitole 4.3.9.4 Návrh komunikace.

Vytvořená webová služba je dostupná na určité URL adrese, která umožňuje její vyvolání. V tomto případě to může být například URL adresa */GanttData.asmx/LoadGanttData*, kterou je pro správné fungování aplikace nutné vložit do HTML definice Ganttova diagramu.

4.5.3.2 Uložení dat diagramu

Pro možnost uložení změn v diagramu je nutné implementovat metodu *SaveGanttData*, která provede uložení pozměněných dat v databázi. Parametrem této funkce bude objekt JSON reprezentující datovou strukturu diagramu. Výsledek operace bude předán pomocí objektu JSON.

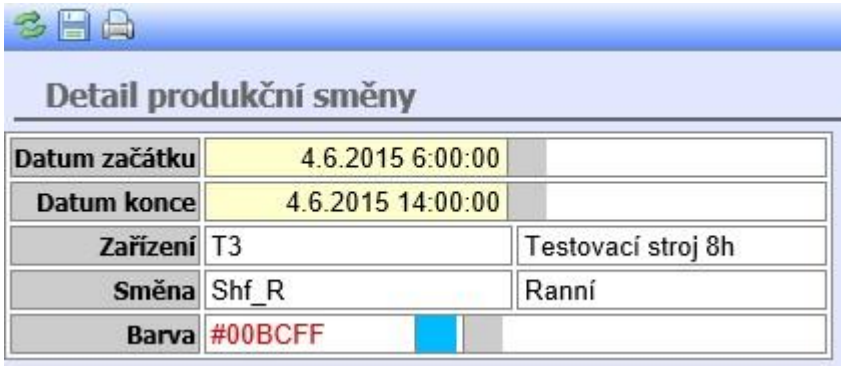
```
[WebMethod]
public string SaveGanttData( Stripe stripe )
{
    //vytvoření objektu GanttStripe z JSON parametrů
    //vyvolání procedury k aktualizaci
    int updated = (int)SqlProc.GanttSaveShiftData.ExecuteValue(stripeId,
startTs, endTs, rowId);
    //datová struktura odpověďi
    Dictionary<string, object> response = new Dictionary<string, object>( );
    //pokud byl aktualizován 1 záznam, jinak vygenerujeme chybu
    if (updated == 1)
    {
        response.Add( "response", "Succes" );
        response.Add( "message", "Ok" );
    }
    return js.Serialize( response );
}
```

Předávaný parametr reprezentuje aktualizovaný objekt, jehož vlastnosti jsou předány jako parametry do aktualizační procedury. SQL procedura pro aktualizaci záznamu je spuštěna, a na základě počtu ovlivněných řádků je vygenerována odpověď. Kompletní kód metody a SQL procedury je k nahlédnutí v příloze.

Popisovaná webová služba je přístupná na URL adrese, která je v tomto případě */GanttData.asmx/LoadGanttData*. V případě požadavku na ukládání dat diagramu je nutné tuto službu implementovat a její URL definovat v HTML elementu pro načtení Ganttova diagramu.

4.5.3.3 Zobrazení detailních informací

V případě požadavků na zobrazení detailu pruhu je nutné implementovat metodu *GetGanttDetail*, pro načtení dodatečných informací. Detailní informace jsou zobrazeny pomocí vnořené HTML stránky pomocí elementu *iframe*. Tím je zachována volnost způsobu zobrazení detailu.



Obr. 22 Formulář detailu produkční směny

V tomto případě dojde k zobrazení detailu pomocí standardního formuláře v systému COMES, kterým je upravená *aspx* stránka. Identifikátor pruhu, který má být zobrazen, je předán parametrem v URL adrese. Na základě přijatého identifikátoru dojde k načtení a zobrazení údajů příslušného záznamu. V případě poslání identifikátoru „*-1*“ dojde k vygenerování stránky s možností založení nového záznamu.

Vytvořený formulář, pro zobrazení detailu je přístupný na URL adrese */StripeDetail.aspx?id=2275*, kterou je nutné pro správnou funkčnost vložit do HTML elementu definujícího Ganttův diagram uvnitř stránky. V tomto případě dojde k zobrazení směny s identifikátorem *2275*.

4.5.4 Aplikace řešení

V současné době je vše připravené k aplikaci Ganttova diagramu na vybranou oblast informačního systému. Pro zahájení vykreslování Ganttova diagramu, je potřeba inicializovat vykreslovací objekt. K tomu dojde vložením HTML elementu *div* (viz následující ukázka) s příslušností ve třídě *GanttArea* do obsahu webové stránky určené k vykreslování Ganttova diagramu.

```
<div class="GanttArea" id="Gantt1" width="900" height="400"
    GanttTitle="Směnový model" showGrid="false"
    showMouseCursor="true" enableDialog="true"
    enableOption="true" enableDragDrop="true"
    enableEdit="true" showStripeTitle="true" enableGridSnap="false"
    ganttLoadDataUrl="/GanttData.asmx/LoadGanttData"
    ganttSaveDataUrl="/GanttData.asmx/SaveGanttData"
    stripeDetailUrl="/StripeDetail.aspx"
    viewStartTs="1451606400000" viewEndTs="1454630400000"
    startTs="1451606400000" endTs="1459900800000" />
```

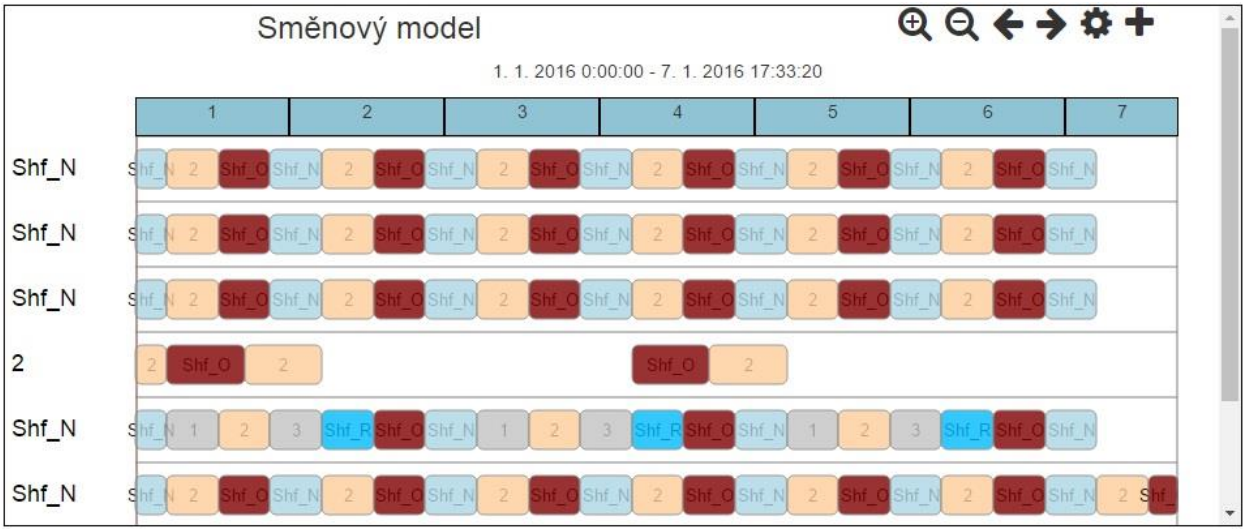
K inicializaci objektu vykreslujícího obsah diagramu dojde pomocí následujícího JavaScriptového zápisu. Každá vlastnost elementu prezentuje atribut třídy, v případě vyplnění špatné hodnoty, nebo zapomenutí definovat danou vlastnost bude načtena výchozí hodnota. Pro zobrazení diagramu v nejjednodušší podobě nutné minimálně přiřadit vlastnosti *ganttLoadDataUrl* URL odkaz na webovou službu obsluhující požadavek na získání dat dle návrhu.

```
var gnt = $("#Gantt1").gantt();
```

Na základě hodnot atributů HTML elementu s identifikátorem *Gantt1*, který definuje widget, dojde k vytvoření objektu s příslušným nastavením. Uvnitř tohoto elementu dojde k vykreslení dodatečného obsahu, pro zobrazení Ganttova diagramu, který je vygenerován v JavaScriptové knihovně. Objekt pro vykreslování je prezentován proměnnou *gnt,* pomocí které může dojít k programovému volání jednotlivých metod widgetu z venčí. To umožní vývojářům další možnosti využití.

Při realizaci nasazení komponenty v systému došlo ke zjištění skutečnosti, že webové služby nelze využívat navrženým způsobem. Problém byl v konfiguraci webového serveru, který neumožňoval volání webových služeb z jiného, než lokálního klienta pomocí http požadavku POST. K odstranění problematiky došlo změnou konfigurace v konfiguračním souboru.

Po absolvování popsaných kroku již může dojít k samotnému vykreslení dat ve formě Ganttova diagramu. Po otevření webové stránky obsahující HTML definici Ganttova diagramu, dojde k zahájení činnosti komponenty a vykreslení diagramu.



Obr. 23 Vykreslení produkčních směn jako Ganttův diagram

Obrázek znázorňuje finální podobu Ganttova diagramu, který je nasazen pro prezentaci a plánování směnového modelu v systému COMES. Zvolené data jsou vykreslena do znázorněné podoby. Pomocí komponenty je možné provádět změny, a využívat veškerou popsanou funkcionalitu.

5 Závěr

Diplomová práce se zabývala problematikou rozšíření firemního informačního systému o komponentu pro vykreslování Ganttova diagramu. Cílem bylo vytvořit univerzální, přenositelné řešení pro vykreslení dat v neznámém formátu do grafické podoby Ganttova diagramu. Motivací pro řešení této problematiky byl externí firmou zadaný požadavek na rozšíření funkcionality informačního systému COMES.

Pro úspěšné řešení problematiky došlo v teoretické části k seznámení s několika základními pojmy. Znalost problematiky informačních systémů, Ganttova diagramu a rozšiřovaného systému COMES, je důležitá pro zahájení prací na tvorbě návrhu řešení.

Stěžejní část práce se zabývala tvorbou a realizací vlastního návrhu rozšíření systému o komponentu Ganttova diagramu. V této části práce došlo k bližší specifikaci problému analyzováním stávajícího systému a definicí klientských požadavků na novou funkcionalitu systému. Z důvodů existence více možných způsobů řešení došlo k rozboru jednotlivých technických prostředků za účelem výběru toho nejvhodnějšího.

Výstupem návrhové části je specifikace vlastního návrhu řešení problematiky pomocí textového popisu a UML diagramů. V této fázi projektu je kladen důraz na splnění všech stanovených požadavků na rozšíření systému. Byl zde také popsán technologický způsob realizace návrhu, tak aby mohlo dojít k nasazení ve specifických provozních podmínkách systému COMES.

Během realizace návrhu došlo k implementaci pomocí konkrétních technologických prostředků a nasazení řešení jako rozšíření systému COMES. Realizační část práce popisuje způsob implementace jednotlivých návrhových struktur, postup při zavádění rozšíření a veškeré důležité skutečnosti.

Po zprovoznění řešení již mohlo dojít k testování komponenty Ganttova diagramu na praktickém příkladu za účelem ověření funkčnosti.

5.1 Zhodnocení řešení

Realizované řešení správně vykreslovalo zvolená data v podobě Ganttova diagramu a umožňovalo využívat veškerou funkcionalitu navrženým způsobem. Po nasazení nedošlo ke zjištění žádných závažných nedostatků a na základě těchto výsledků práce může být konstatováno, že došlo ke splnění všech vytyčených cílů. Zejména požadavky definované zadavatelem byly splněny dohodnutým způsobem. O této skutečnosti je možné se přesvědčit na demonstračním příkladu.

Existují však oblasti, které při návrhu nebyly brány v úvahu a na základě jejich neřešení se v konečné fázi mohou projevit drobné nedostatky. Některé je však možné nazvat specifickými vlastnostmi aplikace.

Slabost tohoto řešení spočívá v nutnosti vytvářet webovou službu při potřebě vykreslení nové struktury dat. Může být vytvořeno i více služeb v případě potřeby při propojení komponenty s více různými datovými strukturami.

Nevýhodu implementace služeb kompenzuje široká míra přenositelnosti aplikace, kterou je možné univerzálně nasadit v systémech různých architektur a odlišných technologických řešení pro zobrazení informací nejrůznějšího formátu.

Další nevýhodou je problematičtější zobrazení v různých webových prohlížečích, kdy může dojít k pokusu o zobrazení Ganttova diagramu uživatelem používající nepodporovaný prohlížeč. Proto došlo ke stanovení požadavků na zobrazování pouze pomocí konkrétních podporovaných prohlížečů, nelze však zajistit že uživatel skutečně použije podporovaný prohlížeč. Nicméně ve většině testovaných prohlížečů bylo možné využít funkcionalitu komponenty Ganttova diagramu, byly shledány pouze drobné rozdíly v způsobu zobrazení.

5.2 Náměty na další rozšíření

Komponenta pro vykreslování Ganttova diagramu splňuje veškerou požadovanou funkcionalitu. Na základě poznatků získaných během tvorby a implementace návrhu řešení, bylo zjištěno několik skutečností, na jejichž základě může dojít k vylepšení komponenty. Jedná se o přidání nové funkcionality, zlepšení ergonomie práce s diagramem či odstranění některých objevených nedostatků. Implementace zlepšení by umožnila širší a efektivnější využití komponenty jako pracovního nástroje.

Prvním námětem pro rozšíření komponenty by mohla být možnost zobrazení plnění průběhu operací. Každý pruh diagramu by mohl být doplněn o znázornění závislosti uběhlé doby operace proti plánované. Tím by mohl uživatel sledovat průběh operací a včas rozpoznat případné zpoždění činnosti.

Některé propracovanější nástroje pro tvorbu Ganttových diagramů umožňují zobrazovat vazby mezi jednotlivými pruhy. Na základě inspirací touto funkcionalitou by bylo vhodné rozšířit komponentu o znázornění návazností mezi jednotlivými činnostmi.

Další vhodnou funkcionalitou, o kterou by mohlo být řešení rozšířeno je export dat do souboru. Tím by mohlo být například umožněno uložení stávajícího stavu nebo načtení nových údajů z, respektive do souboru.

Zajímavé by mohlo být implementovat některé analytické či algoritmické metody projektového řízení, za účelem získání podrobnějších informací o stavu projektu.

Výstupem komponenty měl být Ganttův diagram v elektronické formě, nicméně během zavádění řešení bylo zjištěno, že může dojít k tisku diagramu. Pro správné vykreslení diagramu v takovém případě by mělo dojít k zajištění optimalizaci pro tisk.

6 Literatura

KOCH, Miloš. Management informačních systémů. Vyd. 3., přeprac. Brno: Akademické nakladatelství CERM, 2010, 171 s. ISBN 978-80-214-4157-6.

Vladimír Šmíd. Pojem informačního systému. Management informačního systému. [online]. [cit. 2016-02-07]. Dostupné z:http://www.fi.muni.cz/~smid/mis-infsys.htm

SODOMKA, Petr. Informační systémy v podnikové praxi. Vyd. 1. Brno: Computer Press, 2006, 351 s. ISBN 80-251-1200-4.

BASL, Josef a Roman BLAŽÍČEK. Podnikové informační systémy: podnik v informační společnosti. 2., výrazně přeprac. a rozš. vyd. Praha: Grada, 2008, 283 s. Management v informační společnosti. ISBN 978-80-247-2279-5.

Rábová Ivana. Informační systémy. Elektronické studijní materiály. [online]. 2006 [cit. 2016-02-10]. Dostupné z: https://is.mendelu.cz/eknihovna/opory/index.pl?opora=177.

RÁBOVÁ, Ivana. Podnikové informační systémy a technologie jejich vývoje. V Tribun EU vyd. 1. Brno: Tribun EU, 2008, 139 s. ISBN 978-80-7399-599-7.

VOŘÍŠEK, Jiří. Informační systémy a jejich řízení. 3. vyd. Praha: Bankovní institut vysoká škola, 2007, iv, 278 s. ISBN 978-80-7265-100-9.

Compas Automatizace. Profil firmy. Compas Automatizace. [online]. © 2016 [cit. 2016-02-07]. Dostupné z: http://www.compas.cz/o-nas/profil-firmy .

MES solution COMES. O systému Comes. Comes. [online]. © 2016 [cit. 2016-02-07]. Dostupné z: http://www.compas.cz/o-nas/profil-firmy.

Compas Automatizace. Výrobní IT. Compas Automatizace. [online]. © 2016 [cit. 2016-02-07]. Dostupné z: http://www.compas.cz/produkty-a-sluzby/vyrobni-it.

Třívrstvá architektura. ManagementMania. [online]. 2013 [cit. 2016-02-15]. Dostupné z: https://managementmania.com/cs/trivrstva-architektura-three-tier-architecture.

Řízení projektů (Project Management). ManagementMania. [online]. 2013 [cit. 2016-02-15]. Dostupné z: https://managementmania.com/cs/metody-rizeni-projektu.

Ganttův diagram. In: Wikipedia [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-04-19]. Dostupné z: https://cs.wikipedia.org/wiki/Ganttův_diagram

James Cadle a Donald Yeates, Project Management for Information Systems, pátá edice, Pears Education Limited, 2008, ISBN 978-0-13-206858-1.

Ganttův diagram. ManagementMania. [online]. 2013 [cit. 2016-02-15]. Dostupné z: https://managementmania.com/cs/ganttuv-diagram.

ROSENAU, M. D. Řízení projektů. Brno: Computer Press, 2007. ISBN 978-80-251-1506-0

Systémová integrace. 2011, 18(3). ISSN 1210-9479.

MES (Manufacturing Execution System) [online]. 2011 [cit. 2016-05-02]. Dostupné z: http://www.gerhard-schubert.com/en/mes_manufacturing_execution_system/29.

CEJPEK, Jiří. Informace, komunikace a myšlení: úvod do informační vědy. 2., přeprac. vyd. Praha: Karolinum, 2005. ISBN 80-246-1037-X.

RYBIČKA, Jiří a Petra ČAČKOVÁ. Informatika pro ekonomy. 1. vyd. Praha: Alfa Nakladatelství, 2009, 251 s. Informatika (Alfa Nakladatelství). ISBN 978-8087197-24-0.

FOWLER, Martin. Patterns of enterprise application architecture. Boston: Addison-Wesley, c2003. ISBN 0321127420.

ROSENAU, M D. Řízení projektů. 3. vyd. Brno: Computer Press, 2010. 344 s. ISBN 978-80-251-1506-0.

HRAZDILOVÁ BOČKOVÁ, Kateřina. Projektové řízení: Učebnice. 1. Martin Koláček – E-knihy jedou, 2016. ISBN 978-80-7512-431-9.

SVOZILOVÁ, A. Projektový management /2. vyd. Praha: Grada Publishing, a.s., 2011 ISBN 978-80-247-3611-2 [3] ROSENAU, M. D. Řízení projektů. Brno: Computer Press, 2007. ISBN 978-80-251- 1506-0.

Online Gantt Chart Software. TeamGantt [online]. 2016 [cit. 2016-04-19]. Dostupné z: www.tomsplanner.com/

Online Gantt Chart - Project Planning software. Tom's planner [online]. 2016 [cit. 2016-04-19]. Dostupné z: www.tomsplanner.com/

Software pro řízení projektů| Microsoft Project. Office [online]. 2016 [cit. 2016-04-19]. Dostupné z: https://products.office.com/cs-cz/Project/project-and-portfolio-management-software

GanttProject: free desktop project management app. GanttProject [online]. 2016 [cit. 2016-04-19]. Dostupné z: www.ganttproject.biz

HTML: The Markup Language (an HTML language reference). W3C [online]. 2013 [cit. 2016-05-17]. Dostupné z: www.w3.org/TR/html-markup/

Cascading Style Sheets. W3C [online]. 2016 [cit. 2016-04-23]. Dostupné z: www.w3.org/Style/CSS/

JavaScript Web APIs - W3C. W3C [online]. 2016 [cit. 2016-04-23]. Dostupné z: www.w3.org/standards/webdesign/script

JQuery. JQuery [online]. 2016 [cit. 2016-04-23]. Dostupné z: https://jquery.com/

JQuery UI. JQuery UI [online]. 2016 [cit. 2016-04-23]. Dostupné z: www.jqueryui.com/

Úvod do JSON. JSON [online]. 2016 [cit. 2016-04-23]. Dostupné z: www.json.org/json-cz.html

Getting Started - Ajax. Mozilla developer network [online]. 2016 [cit. 2016-04-23]. Dostupné z: www.developer.mozilla.org/en-US/docs/AJAX/Getting_Started

ASP.NET Overview. Microsoft developer network [online]. [cit. 2016-04-26]. Dostupné z: www.msdn.microsoft.com/en-us/library/4w3ex9c2.aspx

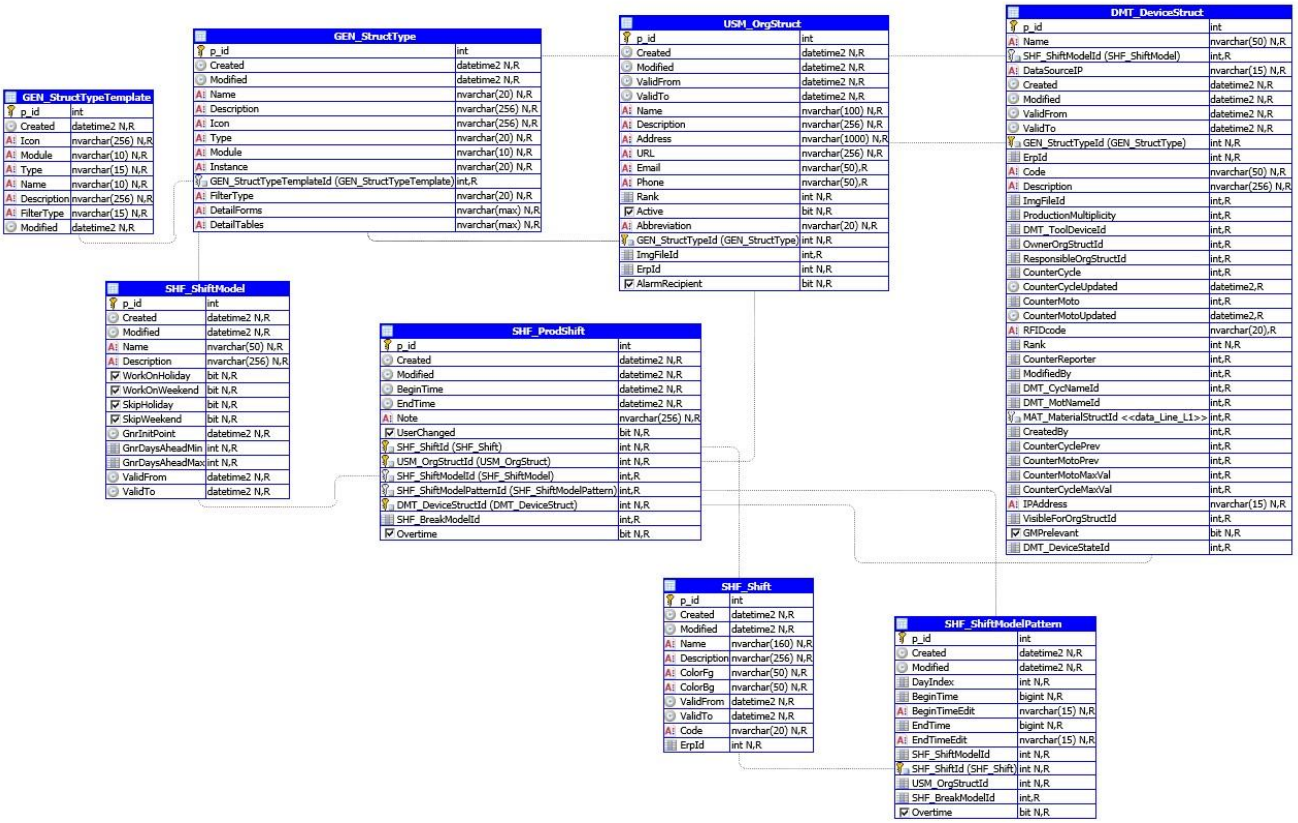
C#. Microsoft developer network [online]. 2016 [cit. 2016-04-23]. Dostupné z: www.msdn.microsoft.com/en-us/library/kx37x362.aspx

ARLOW, J. -- NEUSTADT, I. UML 2 a unifikovaný proces vývoje aplikací : objektově orientovaná analýza a návrh prakticky. 2. vyd. Brno: Computer Press, 2007. 567 s. ISBN 978-80-251-1503-9

Přílohy

A Struktura databáze pro směnový model

Následující obrázek popisuje strukturu části databáze z oblasti směn a zařízení informačního systému COMES. Znalost struktury této části databáze je důležitá pro nasazení komponenty Ganttova diagramu v konkrétním případu nasazení.



Obr. 24 Struktura části databáze potřebné pro vykreslování výrobních směn do Ganttova diagramu