VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

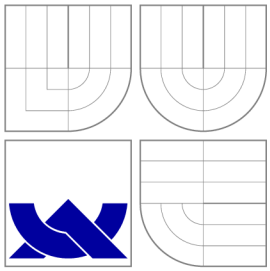
NÁSTROJ PRO PODPORU ČASOVÉHO PLÁNOVÁNÍ V PROJEKTECH: METODY CPM A PERT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

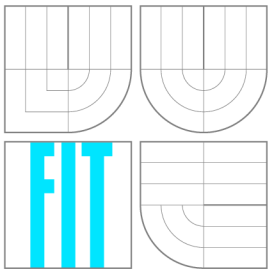
AUTOR PRÁCE
AUTHOR

JAROSLAV BENEŠ

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

NÁSTROJ PRO PODPORU ČASOVÉHO PLÁNOVÁNÍ V PROJEKTECH: METODY CPM A PERT

A TOOL FOR TIME PLANNING SUPPORT IN PROJECTS: CPM AND PERT METHODS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAROSLAV BENEŠ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ŠÁRKA KVĚTOŇOVÁ, Ph.D.

BRNO 2014

Abstrakt

Práce se zabývá tématem projektového řízení. Nejprve je vysvětlena teorie týkající se problematiky projektového řízení a principy metod CPM (Critical Path Method) a PERT (Program Evaluation and Review Technique). Dále je na základě analýzy dostupných nástrojů na trhu proveden návrh nové aplikace a jeho implementace. Aplikace byla implementována v jazyce Java. Závěrem jsou zhodnoceny dosažené výsledky a je uveden návrh možných rozšíření.

Abstract

This bachelor's thesis deals with project management. At first is explained theory of project management and principles of method CPM (Critical Path Method) and PERT (Program Evaluation and Review Technique). Based on the analysis of existing tools for planning was made design and implementation of our aplication. The system has been implemented in programming language Java. Conclusion is made summary of the achieved results and possible future extension.

Klíčová slova

projekt, projektový management, řízení projektu, trojimperativ, životní cyklus projektu, plán projektu, CPM, PERT, nástroje pro řízení projektu

Keywords

Project, Project Management, Project Triple Constraint, Project Life Cycle, Project Plan, Critical Path Method, Program Evalution and Review Technique, Project Management Tools

Citace

Jaroslav Beneš: Nástroj pro podporu časového plánování v projektech: Metody CPM a PERT, bakalářská práce, Brno, FIT VUT v Brně, 2014

Nástroj pro podporu časového plánování v projektech: Metody CPM a PERT

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně
pod vedením paní Ing. Šárky Květoňové, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jaroslav Beneš
20.května 2014

Poděkování

Rád bych poděkoval vedoucí mé práce Ing. Šárce Květoňové, Ph.D. za odborný přístup, poskytnuté rady a vstřícnost při spolupráci.

Obsah

Kapitola 1

Úvod

Projektové řízení, jak ho známe dnes, nesahá do daleké historie, a to i napříč skutečnosti, že spousta procesů konaných člověkem už ve středověku mělo charakter projektu. Ovšem i v té době měli lidé své metody, jak se vypořádat s rozsáhlejší akcí, či procesem náročnějším na organizaci, avšak tyto postupy se zásadně lišily od těch současných. V průběhu let se měnily požadavky na projekt, čemuž se musely přizpůsobit i používané metody. V dnešní době jsou největšími kritérii projektů omezení zdrojů a času. Doba je dynamická. Vše se děje příliš rychle, tudíž je čas pro všechny nesmírně důležitý. Doslova platí, že "čas jsou peníze". To je jeden z důvodů, proč je zapotřebí (správné) projektové řízení. Přestože je řízení projektů důležité ve všech odvětvích, v této práci se budu zabývat pouze oblastí informačních technologií.

Cílem této práce je navrhnout vlastní aplikaci pro časové plánování v projektech. V rámci toho je nezbytné prostudovat metody plánování projektů a na základě zvolených kritérií provést analýzu stávajících aplikací pro jejich podporu.

Problematika plánování projektů, včetně metod **CPM** (Critical Path Method) a **PERT** (Program Evaluation and Review Technique), je popsána v druhé kapitole práce. Definuje se v ní projekt a je uveden postup při časovém plánování. Třetí kapitola stanovuje kritéria, dle kterých budu hodnotit aplikace na trhu a ze kterých bude vycházet i návrh v následující kapitole. Návrh je popsán ve čtvrté kapitole a sloužil jako předloha při implementaci. V další kapitole je popsána konečná implementace společně s ukázkami grafického rozhraní aplikace. Postup a zjištěné chyby při testování jsou popsány v šesté kapitole, na kterou navazuje kapitola s návrhem možných rozšíření aplikace při dalším vývoji. V závěrečné osmé kapitole je shrnuto splnění jednotlivých bodů zadání a jsou uvedeny dosažené výsledky.

Kapitola 2

Časové plánování v projektech

Každý, kdo se dnes pohybuje na manažerských pozicích, si jistě dobře uvědomuje, jak důležité je znát moderní techniky projektového řízení. Vzhledem k tomu, že jeden z hlavních požadavků při sjednávání podmínek zainteresovanými stranami je co nejkratší čas, je časové plánování velmi důležitou součástí projektového řízení. Bohužel, jak je známo z praxe, ve velké spoustě případů dochází k překročení doby stanovené na projekt, což může být způsobeno právě nevhodným plánováním či špatným odhadem. Proto se neustále objevují programy, které se snaží tyto procesy ulehčit. Nehledě na to, že techniky projektového řízení mohou jednotlivci využít i v běžném životě.

V této kapitole proberu obecně plánování projektů. V první části uvádím základní pojmy související s problematikou, které budou následně použity. Jedná se o projektové řízení, co je to projekt, jaké má vlastnosti, jaký je jeho životní cyklus a vysvětlení **trojimperativu**. V další části je uveden, postup při časovém plánování. Zde jsou také podrobně popsány metody **CPM** a **PERT**.

2.1 Projektové řízení

Projektové řízení je používáno s cílem zkvalitnit řízení projektů. Používají jej především organizace, které jsou si vědomy ztrátovosti a chybnosti u projektů, ve kterých řízení není použito. Projektové řízení je tedy způsob, jakým se snažíme dosáhnout cílů projektu, co možná nejlepší cestou. Jedna z obecných definic říká: "Projektové řízení je uplatnění vědomostí, dovedností, nástrojů a technik na aktivity projektu za účelem dosažení projektových cílů"[7]. Objevilo se po té, co rozsah, obtížnost a neobvyklost projektu přesáhla určitou mez, po které již nebylo možné projekt zvládnout bez použití speciálních technik.

Projektové řízení může být řízení všeho, co má charakter projektu 2.2.2. Jeho základním znakem je, že vede vždy přímo ke konci. Každému členovi projektu přesně určí úkoly a pravomoc, a během jeho trvání kontroluje dosažené výsledky. Členové postupují ve svých úkolech podle známých metod jako jsou například CPM 2.3.3, PERT 2.3.4 a další. Většina metod je podporována počítačovými programy, na které budou uvedeny v kapitole analýzy 3.

Používání technik projektového řízení přináší velkou řadu aktiv. V první řadě se snižuje riziko neúspěchu při dosahování cílů. Dále zkracuje čas potřebný na dokončení akcí a snižuje celkové náklady. Organizace, která používá projektové řízení, má také větší příležitost podílet se na zahraničních zakázkách, jelikož většina západních firem považuje

tento způsob práce za standardní [13].

Pro podporu projektové řízení jsou v současnosti využívány možnosti současných počítačů, jejichž paměťová kapacita a výpočetní rychlost umožňuje zvýšit efektivitu a usnadnit celkovou práci projektovým manažerům.

2.2 Projekt

Nejdůležitějším pojmem v časovém plánování v projektech je projekt sám. Tento pojem označuje proces plánování a řízení projektu. Neexistuje však pouze jedna definice projektu a různé organizace si pod názvem projekt mohou představit odlišné věci.

2.2.1 Definice projektu

Projekt je jedinečný proces, který se skládá ze souboru koordinovaných a řízených činností s datem započetí a dokončení, které se vykonávají za účelem dosáhnutí cíle, včetně specifických požadavků na omezení zdrojů [10].

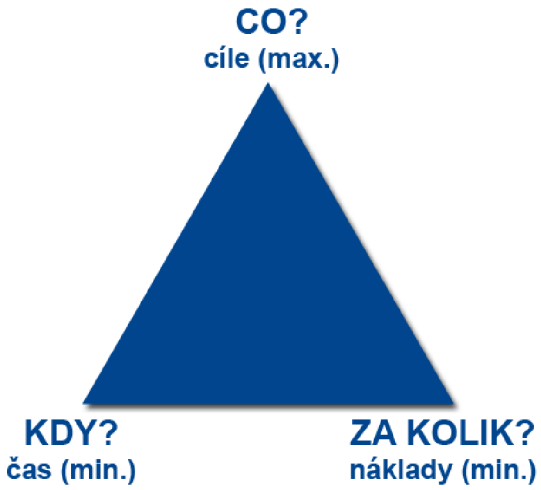
Projekt je v čase konečný soubor technologicky a organizačně navazujících činností a prostředků potřebných k vytvoření produktu či služby užitečné pro konkrétního zákazníka [10].

Jedinečný je každý projekt proto, jelikož se provádí pouze jednou, je dočasný a pracuje na něm vždy jiná skupina lidí. Existují čtyři typické znaky projektů, které pokud se vyskytují společně, odlišují projektové řízení od jiných manažerských činností. Projekty mají trojrozměrný cíl, zahrnují zdroje a realizují se v rámci organizace [10].

2.2.2 Vlastnosti projektu

Projekty se mohou týkat tisíce lidí nebo naopak i jednotlivce. Mohou trvat třeba jen týden, ale dokonce i celé roky. Projekty mohou existovat ve všech různých velikostech a formách. Abychom všechny mohly sjednotit do jednoho formátů, je potřeba zvolit atributy, dle kterých se bude projekt definovat [10]:

- **cíl projektu -** Každý projekt musí mít správně definovaný cíl. Většinou se jedná o nějaký nový produkt či službu.
- **termín projektu -** Každý projekt musí mít pevně určenou dobu začátku i dobu ukončení.
- **postupná konkretizace -** Na začátku tým sestaví plány, ve kterých je projekt definován obecně. Postupem času však dochází k upřesnění detailů, které se zanáší do původního plánu.
- **zdroje z různých oblastí -** Aby mohl projekt správně fungovat, je nutné mu přidělit zdroje jako například lidi, hardware nebo software. Jelikož jsou tyto prostředky vždy omezené, je nutné s nimi při plánování účelně nakládat.
- **hlavní zadavatel -** V každém projektu figuruje více "zainteresovaných stran"neboli osob či organizací, které jsou zapojeny do projektu. Je proto důležité, aby měl jeden roli hlavního zadavatele (vlastníka) projektu, který má zájem projekt realizovat, určuje směr řešení a uvolňuje na něj finanční prostředky.



Obrázek 2.1: Trojimperativ [14]

- **neurčitost -** Jelikož každý projekt je jedinečný, je obtížné přesně odhadnout a naplánovat všechny požadované faktory. Musíme také počítat s vnějšími vlivy jako je například dočasná indispozice člena týmu nebo zpoždění dodavatelských firem. V každém projektu je tudíž přítomna neurčitost.

2.2.3 Trojimperativ

V každém projektu figurují veličiny, které mírou jejich plnění určují vývoj projektu. Jedná se o rozsah projektu, minimální čas, za který lze projekt dokončit, a náklady. Tento model můžeme najít v publikacích pod pojmem **trojimperativ**. Jednotlivé veličiny jsou vzájemně závislé. Chceme-li například dokončit projekt v kratším čase, musí se to projevit minimálně na jedné další veličině. To znamená, že se nám zvýší náklady, nebo se nestihnou dokončit všechny cíle do plánovaného ukončení. Důležité je mít tyto tři veličiny vyvážené. Pro lepší představu je na obrázku 2.1 zobrazen trojimperativ jako trojúhelník.

Mimo tyto parametry trojimperativu mohou hrát důležitou roli i další elementy včetně kvality produktu. Někteří lidé tento element považují za stejně důležitý jako prvky trojimperativu, proto se tak vedle rozsahu, času a nákladů může objevovat také kvalita. V takovém případě hovoříme o takzvaném **čtveru omezením** [11]. Většina manažerů se však drží názoru, podle kterého je otázka kvality nedílnou součástí právě cílového rozsahu, času a nákladů na projekt.

Nesplnění všech bodů trojimperativu nevyhnutelně vede k neúspěšnému projektu. Jak by se ale člověk mohl mylně domnívat, projekt který splní všechny bodu trojimperativu, může také skončit neúspěchem. Jedná se o případ, kdy se sice splní všechny cíle ve stanoveném čase, avšak výsledek se ke konci může z nějakého důvodu stát nepoužitelným.

Jak vidíme, čas je jedním z nejdůležitějších parametrů projektů. Má také velký vliv na fyzický i psychický stav lidí, kteří se na projektu podílejí. V dnešní medicíně se často setkáváme se selháním lidského organismu z důvodu stresu a přepracování, což může být způsobeno právě špatným časovým rozvrhem, který nutí lidi pracovat přes čas. Kvalitní plánování času tedy vede nejen k úspěšným projektům, ale i ke zdraví.

2.2.4 Životní cyklus projektu

Všechny projekty jsou časově omezeny. Začátkem je specifikace problému, který je potřeba vyřešit a ukončen je rozpuštěním týmu. Vše, co se nachází mezi těmito okamžiky, se nazývá **životní cyklus projektu**. Obsahuje všechny fáze, kterými projekt v průběhu svého života prochází. Forma definice životního cyklu projektu se liší podle odvětví, ale i v rámci stejného odvětví bývá různá pro odlišné organizace a podniky [6]. **Životní cyklus produktu**, který vznikne, přesahuje životní cyklus projektu o jeho provoz, servis a likvidaci.

Jednotlivé fáze je potřeba regulovat a dohlížet na ně. Dále je nutné, aby za kvalitu a realizaci projektu odpovídala vždy právě jedna osoba. V závěru projektu je nutné provést zhodnocení projektu, v níž se zaznamenají faktory, které mohou v budoucnu přispět k snížení rizikových událostí při práci na projektech. Životní cyklus projektu lze rozdělit do těchto tří fází [4]:

1. Předprojektová fáze

Jedná se o nejdůležitější fázi z celého projektu. Úkolem předprojektové fáze je zhodnotit, zda je vhodné projekt uskutečnit. Posuzuje se zde proveditelnost daného záměru a provádí se předběžné plánování. Je nutné stanovit strategii, která povede k dosáhnutí stanovených cílů. Součástí této fáze je i vypracování různých analýz a studií, které poslouží jako podklady k dokumentům této fáze. V předprojektové fázi jsou obvyklé dva hlavní typy dokumentů:

- **studie příležitosti** (Opportunity Study) - Tento dokument je částečný základ předprojektové fáze. Je nutné zvážit současný i budoucí stav trhu, situaci ve firmě apod. Využívají se zde agregované údaje, expertní odhady, a součástí může být také **SWOT analýza**, která pomáhá identifikovat slabé a silné stránky projektu a příležitosti a hrozby projektu. Výsledkem je doporučení, do jaké míry je vhodné projekt realizovat. V případě kladného doporučení se zde objevuje první podrobnější charakteristika projektu.
- **studie proveditelnosti** (Feasibility Study) - Tato studie by měla určit nejvhodnější způsob jak realizovat projekt. Upřesňuje se zde celý obsah projektu, plánují se termíny začátku i konce projektu a dochází k odhadu zdrojů. Cílem studie je soubor všech cest, které vedou k úspěšnému dokončení projektu. Cesty jsou ohodnoceny podle různých elementů, jako je například počet potřebných zdrojů, doba trvání nebo výše celkových nákladů. Dále je zde míra, do jaké je vhodné projekt realizovat a případně nejvhodnější cesta k cíli.

V této fázi je stanoveno, k jakému cíli je třeba dojít a jakým způsobem. Nejdůležitějším faktorem je však rozhodnutí, zda-li projekt spustit. Pokud dojde k realizaci, analýza je prováděna i v průběhu projektu, se záměrem zjistit, zda je stále aktuální projekt dokončit. V průběhu projektu se může stát, že dokončení projektu již není potřeba. V takovém případě je práce na projektu ukončena, s cílem ušetřit zdroje.

2. Projektová fáze

Do této fáze spadá vše, co se odehrává od začátku realizace projektu po jeho předání zákazníkovi. Při zahájení projektu se upřesňují požadované cíle, výstupy, provádí se přiřazování zdrojů, kompetence atd. To vše je vhodné zpracovat do příslušného dokumentu, který dále slouží jako technicko-organizační parametr projektu. Jmenovaný

tým následně dle zadání sestaví plán projektu. Tomuto kroku se však budu věnovat posléze 2.3.

Před samotnou realizací projektu je vhodné uspořádat schůzku a pozvat všechny zainteresované strany. Provádí se zde rekapitulace plánu projektu a je představen harmonogram projektu. V průběhu je nutné sledovat stav projektu v porovnání s plány. Dojde-li k výraznějším odchylkám, je třeba na změny reagovat. V případě menších odchylek postačí drobná korekční opatření. Při výraznějším problému lze vytvořit zcela nový základní plán projektu [4].

Při ukončení realizace dochází k předání výsledného produktu zákazníkovi. K této příležitosti vypracovává projektový tým i závěrečnou zprávu projektu, ve které jsou shrnuty poznatky z realizace projektu spolu s případnými doporučeními. Realizace projektu je tímto ukončena, zdroje jsou uvolněny a je možné je použít do jiných projektů.

3. Poprojektová fáze

Tato fáze začíná po ukončení celého projektu. Poznatky, kterých bylo při práci dosaženo je vhodné zaznamenat, a zefektivnit tak práci na dalších projektech. Analyzuje se celý průběh projektu a zhodnotí se dobré i špatné zkušenosti. Aby mohla být poprojektová fáze provedena nezávisle, provádí ji jiný tým, než na projektu pracoval. Kromě vlastní práce na projektu se vyhodnocují i externí pracovníci, kteří se na projektu podíleli. Ve výsledku se může s některými z nich trvale přerušit spolupráce. U některých projektů se všechna aktiva mohou dostavit až po uplynutí několika měsíců. V takovém případě je nutné naplánovat zhodnocení až po tomto termínu.

2.3 Časové plánování

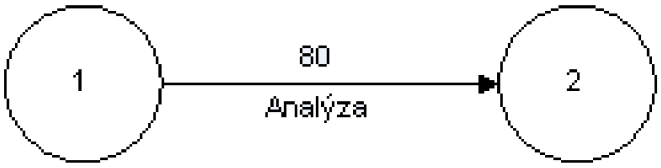
Při plánování projektu se v první řadě určí **cíle**. Zde se odpoví na všechny otázky typu: na co je projekt zaměřen a jaký má být konkrétní výsledek. Na základě toho je nyní možné vytvořit **strukturu projektu**. Většinou se jedná o popis toho, co je potřeba v rámci projektu realizovat. Nyní se dostáváme k časovému plánování, ale je brán v potaz trojimperativ 2.2.3 plánuje se zároveň jak čas tak zdroje. V plánování projektů se nejdříve definují všechny činnosti, které je v projektu potřeba vykonat, přiřadí se jim odhad doby trvání a vytvoří se časový harmonogram.

2.3.1 Definování činností

První krok časového plánování přímo navazuje na strukturu projektu. Je třeba získat výčet všech dílčích činností, které je potřeba vykonat, a to průchodem všech položek struktury. Při tomto procesu se návaznost ještě zcela zanedbává. V mnohých případech dochází k distribuci položky na více činností. Takto se pokračuje, dokud projekt není hierarchicky strukturován do úrovní s potřebnou složitostí. V tomto kroku se pouze definují jednotlivé kroky projektu.

Až nyní se definuje jejich logické návaznosti. Je potřeba vytvořit jakousi posloupnost činností, které na sebe logicky navazují. To však může být ovlivněno mnoha faktory, jako jsou například termíny dodání materiálu či závislost na jiné činnosti. Obecně existují čtyři vazby mezi činnostmi:

- **konec-začátek** - nejjednodušší příklad vazby (také nejčastější), předcházející činnost musí skončit, aby další mohla začít;



Obrázek 2.2: Hranově definovaný síťový graf [10]

- **konec-konec** - předcházející činnost musí skončit, aby následující mohly skončit, následující činnost tak může být započatá i před koncem předchozí;
- **začátek-konec** - následující činnost může skončit až po té, co začne předcházející;
- **začátek-začátek** - následující činnost může začít hned, když začne předcházející;

Nyní jsou činnosti v logické návaznosti a tvoří jakýsi postup. V této fázi projektu je také dobré definovat rozhraní jednotlivých činností, tj. jak spolu budou jednotlivé činnosti komunikovat a jaké materiály si budou vzájemně předávat. Každá činnost má jasně stanoveno jaká data a v jakém formátu dostává na vstup a přesně danou činnost k vykonání, z čehož vyplývá i formát výstupních dat. Pro dobrou orientaci v závislostech činností je dobré vzniklý postup reprezentovat vhodným způsobem. Ve většině případů je použit síťový graf.

Síťový graf

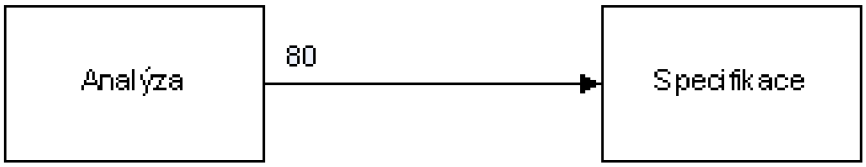
Síťový graf zobrazuje projekt formou diagramu, který zároveň vyjadřuje technologické vazby mezi dílčími činnostmi. Jedná se o nejvíce používané zobrazení činnosti projektů bez znalosti času, a proto je nejvíce podporován komerčními nástroji (např. MS Project). Projekt je možno interpretovat síťovým grafem dvěma různými způsoby:

- **hranově definovaný síťový graf** - V tomto případě jsou to ohodnocené hrany, které realizují činnosti projektu. Uzly zde znázorňují okamžik konce a začátku činnosti. Nevýhodou tohoto grafu může být ten fakt, že pro interpretaci některých vazeb jsme nuceni použít neexistující ohodnocené hrany.
- **uzlově definovaný síťový graf** - Činnost je zde reprezentovaná uzlem. Hrana potom znázorňuje závislost mezi činnostmi. Jedná se o nejrozšířenější způsob zobrazení. Používá se ve většině případů.

Dále musí každý síťový graf splňovat tři podmínky:

1. **pouze jeden začátek grafu**
2. **pouze jeden konec grafu**
3. **šipky orientované zleva doprava reprezentují čas** = nelze vytvářet cykly

Jednotlivé fáze projektu mohou být reprezentovány **milníky**. Jedná se o důležitou pomůcku při řazení činností. Většinou je to stav, kdy skončí jedna fáze a následující ještě



Obrázek 2.3: Uzlově definovaný síťový graf [10]

nezačala. V tomto bodu se hodnotí současný stav projektu. Je možné vybrat jinou variantu, zopakovat poslední etapu či ukončit celý projekt. Milníky jsou využity i ve smlouvě projektu například na vymezení vztahu mezi dodavatelem a odběratelem a mohou být zobrazeny i v síťovém grafu.

2.3.2 Odhad doby trvání

V této chvíli je nutné stanovit potřebný čas pro dokončení jednotlivých činností. Jelikož se zpravidla jedná o neopakující se činnosti, pracuje se pouze s odhadem, který sebou nese jistou chybnost. Tento pojem je označován jako **doba trvání**. Pro každou činnost by měla být vybrána minimálně jedna osoba, jenž má v dané problematice potřebné znalosti, je schopna zvážit všechny rizika a může tak provést reálný odhad. Těžko by mohl například člověk který neviděl auto odhadnout, jak dlouho trvá výměna oleje. Zároveň však tato osoba musí vzít v potaz množství zdrojů, které jsou pro daný projekt v daný čas k dispozici, a také ohodnotit jejich stav, kvalitu a produktivitu. Pravdou však zůstává, že i po zahrnutí nepřeberného množství faktorů a dosáhnutí velmi kvalitní hodnoty, výsledek zůstává stále jen odhadem.

K odhadu doby trvání lze přistupovat z více stran. Existuje mnoho postupů, kterými lze dojít k výsledné hodnotě. Lze provézt například jedno číselný odhad na základě osobní zkušenosti. To lze provést v případě, kdy má člověk s podobnou činností čerstvou osobní zkušenost. Dále je zde také možnost odhadnout dobu na základě dokumentace předchozích projektů. Máme-li dostupnou dokumentaci projektu, ve kterém se vyskytovala podobná činnost, lze na jejím základě provést odhad i naší činnosti. Takovému postupu se říká analogické odhadování. Dalším možným způsobem, je metoda PERT. Tato metoda je blíže popsána v následující části práce 2.3.4. Podobných postupů existuje ještě celá řada. V této práci je však není třeba uvádět.

Poté, co jsou vypočteny odhady ke všem činnostem projektu, se přistoupí k další fázi. K této fázi se však lze vrátit ještě v průběhu projektu, v případě větších nesrovnalostí v plánu. V takovém případě se podle aktuálního stavu provede nový odhad doby trvání a pokračuje se od tohoto kroku stejným postupem.

2.3.3 Metoda CPM

Critical Path metod, zkráceně CPM je metoda používající se k transformaci síťového grafu na časový harmonogram. Metoda CPM byla vymyšlena v 50. letech jako deterministicky matematický model, který počítá celkové trvání projektu se vzájemně provázanými a závislými činnostmi.

Pro aplikování metody potřebujeme znát síťový graf, který obsahuje všechny činnosti

a jejich vazby, odhad doby trvání všech činností, požadavky na zdroje, termíny klíčových událostí a kalendář projektu a zdrojů. Ten říká, kdy je možno které činnosti provést. Při tvorbě časového harmonogramu se mohou pro všechny činnosti vypočítávat tyto údaje [4]:

- **termíny -** Udávají nejrychlejší možný začátek a konec, i nejpozdější možný začátek i konec.
- **trvání projektu -** Doba trvání od začátku do konce projektu.
- **celková rezerva -** Časový interval, o který se může činnost zdržet, aniž by ohrozila kritickou cestu. Projekty s nulovou rezervou bývají kritické.
- **volná rezerva -** Časový interval, o kolik se může činnost opozdit, aniž by ohrozila činnost následující.
- **kritická cesta -** Kritická je totožná s nejkratší možnou délkou trvání projektu. Tvoří ji kritické činnosti, které na sebe bezprostředně navazují. Opozdí-li se některá činnost na kritické cestě, dojde k prodloužení doby realizace celého projektu. Jedná se o nejdelší cestu v síti, která vede od počátku až k cíli.

Dojde-li se k nevyhovující kritické cestě, například nevhodnou dobou trvání projektu, je nutné upravit síťový graf. To lze například provést změnou vazeb mezi činnostmi či modifikace zdrojů.

K výpočtu termínů nejrychleji možného začátku a konce a nejpozději možného začátku a konce, slouží dva speciální algoritmy, které se nazývají vpřed a vzad [5].

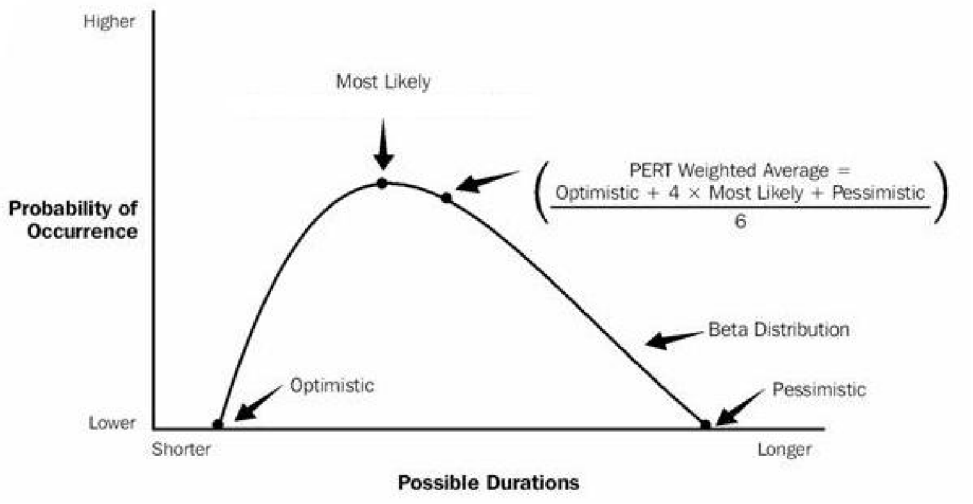
Algoritmus vpřed

Pomocí tohoto algoritmu lze spočítat termíny nejdříve možného začátku a nejdříve možného konce činnosti. Vpřed se nazývá, jelikož se postupuje od začátku plánu ke konci. Nejdříve je nutné znát termín zahájení projektu. Tento čas se nastaví ke všem činnostem kterými projekt začíná. Následně se určí konec těchto činností, tj. termín začátku plus doba trvání činnosti. Po-té můžeme přistoupit k následující činnosti. V případě, že má činnost pouze jeden předcházející uzel, termín nejdříve možného začátku je shodný s datem konce předcházející události. Pokud však některé činnosti předchází více uzlů, k výpočtu termínů této činnosti je možné pokračovat až po stanovení termínů u všech těchto uzlů. Jako termín nejdříve možného začátku se pak použije datum z předchůdce který končí jako poslední, jelikož se musí počkat na dokončení všech předcházejících částí.

Tímto způsobem se projde celý projekt a jsou určeny nejdříve možné termíny pro všechny činnosti projektu.

Algoritmus vzad

Algoritmus vzad slouží k výpočtu nejpozději přípustných termínů v projektu. Postupuje se zde od poslední činnosti k první. V předchozím algoritmu jsme došli až k poslední činnosti, pro kterou jsme určili nejdříve možné termíny začátku a konce a které se budou shodovat i s nejpozději přípustnými termíny v tomto bodě. Odtud se bude postupovat zpětně skrze jednotlivé činnosti analyticky dle algoritmu vpřed 2.3.3. Na základě získaných termínů lze určit celkové časové rezervy pro všechny činnosti projektu.



Obrázek 2.4: způsob výpočtu pomocí metody Pert [1]

2.3.4 Metoda PERT

Metoda **Pert** neboli *Program Evaluation and Review Technique* je možno použít k odhadu složitých činností, které mají stochastický charakter. Na dobu trvání každé stochastické činnosti se pohlíží jako na náhodnou proměnnou s konkrétním rozložením pravděpodobnosti. Vzhledem charakteru činností řešených v softwarovém inženýrství bylo zvoleno pravděpodobnostní rozdělení *beta*, jelikož nejlépe vystihuje proměnlivost provozních podmínek. Jak lze vidět na obrázku 2.4, rozložení je mírně podobné normálnímu rozložení. Obě jsou spojité, mají pouze jeden vrchol a jsou mírně asymetrické. Hlavním rozdílem je však oboustranná ohraničenost beta rozdělení.

Cílem metody je vypočítat nejpravděpodobnější dobu trvání činnosti. Jako první se každé činnosti přiřadí tři odhady[8].

1. **optimistický odhad** t_o**-** nejkratší doba trvání činnosti, za žádných okolností nemůže dojít k dokončení činnosti za kratší dobu, nepřipouští možnost vzniku poruch
2. **normální odhad** t_n**-** nejpravděpodobnější hodnota odhadu trvání činnosti, jedná se o odhad, že činnost probíhá za normálních podmínek, tj. neobjeví se žádné větší překážky
3. **pesimistický odhad** t_p**-** nejdelší možná doba za kterou lze činnost dokončit, předpokládá se, že dojde k chybě na všech místech kde je to možné

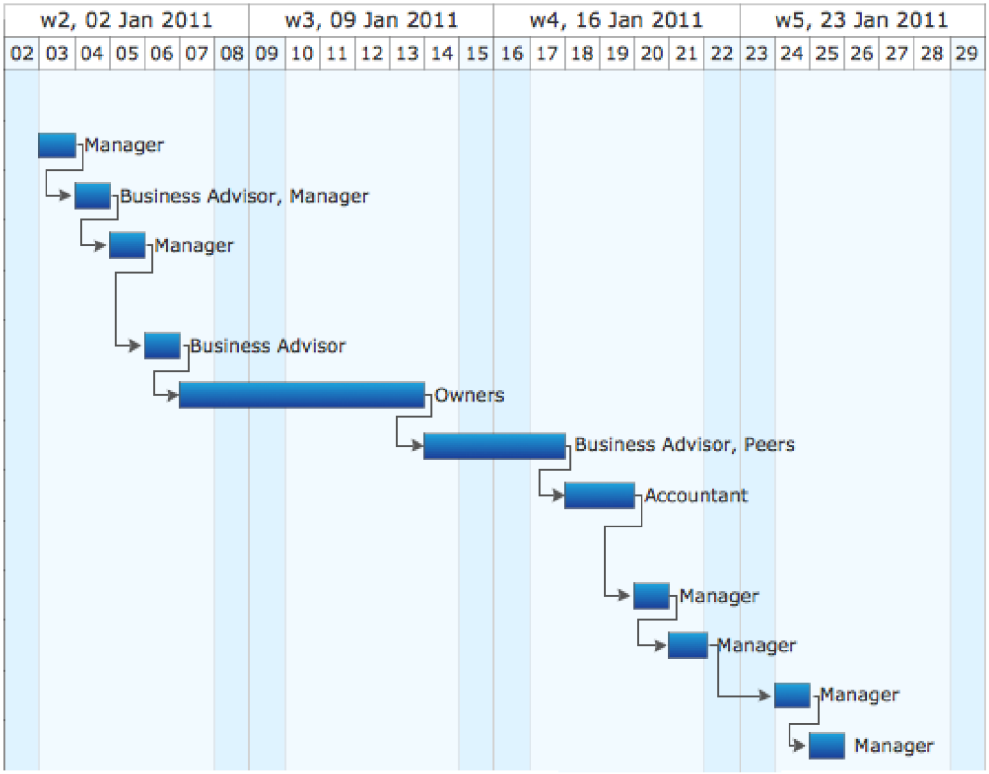
Pomocí těchto tří odhadů, je možno vypočítat střední hodnotu T, neboli dobu trvání činnosti dle PERT, a rozptyl. Výpočty lze provézt dle vzorců:

Očekávaná doba trvání projektu:

$$T = \frac{t_o + 4t_n + t_p}{6} \tag{2.1}$$

Rozptyl doby trvání činností:

$$\sigma^2(T) = \left(\frac{t_p - t_a}{6}\right)^2 \tag{2.2}$$



Obrázek 2.5: Ganttův diagram [2]

Čím vyšší je hodnota rozptylu, tím větší je pravděpodobnost že skutečná hodnota doby trvání činnosti se bude více lišit od její střední hodnoty [15]. Doba trvání celého projektu lze vyjádřit jako součet středních dob trvání činností na kritické cestě 2.3.3.

2.3.5 Ganttův diagram

Nejpoužívanějším grafickým zobrazením plánu projektu je Ganttův diagram. Ten byl rozšířen během první světové války průkopníkem H. L. Ganttem, po kterém je pojmenován, a sloužil pro zobrazení naplánovaných činností a jejich posloupnosti v čase. V té době se jednalo o velmi revoluční přístup ke grafickému znázornění. Dnes je ale Ganttovo zobrazení běžnou formou.

V diagramu je na horizontální ose uveden čas ve kterém se plánuje. Dle délky celkového trvání projektu se zvolí měřítko a zobrazují se odpovídající podrobnosti. V řádcích jsou zobrazeny jednotlivé činnosti, které svojí velikostí protínají časovou osu v délce určující jejich dobu trvání. V každé činnosti lze potom zobrazit patřičný průběh. Dlouhou dobu sloužil Ganttův diagram pouze ke znázornění činností. V moderním plánování projektů je však možno do něj zakomponovat i vzájemné vztahy mezi nimi.

Největší nevýhodou je zobrazení rozsáhlého projektu. Zatímco pro menší projekty je diagram přehledný a jednoduchý, při zobrazení většího množství činností se diagram stává téměř nečitelným. Orientace v takovémto diagramu je velmi pomalá a není ani vhodná pro reprezentaci na obrazovkách dnešních počítačů, jelikož poměr mezi potřebnou plochou, a počtu zobrazených informací není ideální.

Kapitola 3

Analýza současných nástrojů

V této kapitole si zvolím kritéria, kterých bych při tvorbě nového nástroje chtěl dosáhnout. Kritéria potom porovnám s programy, které jsou již na trhu. Dále najdu jejich další přednosti i nedostatky. Výsledky z analýzy po tom uplatním při návrhu naší výsledné aplikace. Pro lepší orientaci jsem se rozhodl rozdělit programy do dvou skupin podle toho, zda-li je aplikace bezplatně přístupná či nikoliv.

3.1 Kritéria

Dnešního manažera lze charakterizovat jedním slovem: vytíženost. Manažer bývá ve své práci velmi zaneprázdněn, tudíž se nabízí požadavek na velmi jednoduché ovládání, které uživatel bez větších potíží zvládne sám. Čím bývá řešení intuitivnější tím, je to lepší. V ideálním případě by uživatel neměl mít potřebu vyhledat manuál. S tím souvisí i vkládání dat. Uživatel musí mít jednoduchý přístup k vložení nových činností, správě jejich atributů a tvorbě vzájemné vazby. Program by měl rozumným způsobem reprezentovat uložená data uživateli a provádět sám i operace související s plánováním času.

Dalším kritériem je přenositelnost uchovávaných dat. Uživatel potřebuje pracovat s daty na různých místech, a proto je nutné data vhodně exportovat a následně i zpět importovat. To by mělo být umožněno nejen skrze lokální disk, ale i pomocí připojení k internetu. Projektový manažer se tak připojí ke svému projektu z jakéhokoliv počítače s připojením na internet.

Z praxe je také známo, že většinu plánů je potřeba přizpůsobit aktuální situaci. Je proto nutné aby následná editace byla jednoduchá, přehledná a co možná nejefektivnější. Uživatel by při změně jedné hodnoty neměl ručně přepočítávat data v následujících činnostech.

Po dokončení projektu by spolupracovníci měli mít možnost připojit se k plánu a zadat informace týkající se jim přidělené činnosti. Je tedy nutné aby aplikace obsahovala i jistou správu rolí, kde by měli různí uživatelé odlišná práva.

K projektovému řízení je mnohdy používán právě program Microsoft Project. Často navíc dochází k situaci, kdy na projektovém plánu pracuje víc lidí. Je proto důležité, aby bylo možné data mezi Microsoft Project a danou aplikací libovolně vyměňovat.

3.2 Open Source aplikace

Do této kategorie patří programy, které je možno volně stáhnout na internetu. Většinou se jedná o jednoduché aplikace s výrazně menší funkčností, které vyvíjí skupiny vývojářů bez nároků na honorář. Tyto aplikace jsou určeny začínajícím firmám a menším projektům, které nemají dostatek financí pro zakoupení plnohodnotného softwaru.

3.2.1 ProjectLibre

Tento software je poměrně mladý. Vznikl v roce 2012, kdy se skupina vývojářů rozhodla pokračovat ve stopách jeho předchůdce OpenProj. Ten byl svého času nejlepší bezplatnou offline náhradou aplikace Microsoft Project. Vývoj však skončil v roce 2008 u verze OpenProj 1.4. ProjectLibre je klientský nástroj založený na jazce Java, a firma InfoWord jej v roce 2013 umístila mezi deset nejlepších open source programů v soutěži "Bossie Awards". Aplikace je multiplatformní a je přeložena do mnoha světových jazyků mezi kterými nechybí ani čeština. Pro analýzu jsem použil aktuální verzi ProjectLibre 1.5.8 vydanou v prosinci roku 2013.

Tento program je z pohledu své funkčnosti velmi jednoduchý. Program využívá základní plánovací funkcionalitu, kterou známe z jiných produktů. Obsahuje základní prvky, které projektový manažer k tvorbě časovému plánování potřebuje. Podporuje velké množství reprezentace vytvořeného harmonogramu a umožňuje editaci v každém z nich. Umožňuje organizaci zdrojů a jejich přidělování činnostem. Velkou výhodou je kompatibilita s programem Microsoft Project. Časový plán je možné obousměrně mezi těmito programy převádět. Další užitečnou funkcí může být možnost exportovat časový plán do formátu PDF. V tomto roce by se již měla objevit nová verze programu, které již nebude vyžadovat instalaci a bude tak zcela přenosná. Umožní tak uživatelům nosit ji stále při sobě například na USB a spustit ji kdekoli na počítači.

Faktem však zůstává, že program lze použít pouze offline. Pokud se chce uživatel s vytvořeným plánem podělit, musí projekt uložit a v této formě jej dopravit druhému spolupracovníkovi. Jedná se ale o bezplatný produkt, který nabízí optimální vyvážení mezi složitostí programu a množství nabízených funkcí a je vhodný především do menších, nízko nákladových projektů.

3.2.2 GanttProject

Jako druhý program v této kategorii jsem zvolil GanttProject, jelikož byl uváděn v nejednom žebříčku jako jeden z nejlepších programů pro projektový management, které jsou dostupné zcela zdarma. Je vyvíjen od ledna roku 2003, kdy jej Alexandr Tomas uvedl na univerzitě Paris-Est Marne-la-Vallée ve Francii. Během let sice projekt převzal Dmitry Barashev, ale je i nadále vyvíjen. Pro analýzu byla použita poslední verze GanttProject 2.6.5, která vyšla v únoru tohoto roku. Je napsán v Javě, tudíž je možné spustit jej na jakékoliv platformě. Ke spuštění je však zapotřebí verzi Javy 1.4.2 nebo novější.

GanttProject umožňuje všechny základní operace, které projektový manažer potřebuje včetně správy zdrojů. Uživatelské rozhraní je zde velmi intuitivní. Po prvních vteřinách je člověk schopen program ovládat do posledního detailu. Je to nejspíš i na úkor zobrazených dat v základním zobrazení projektu. Z informací o projektu zde uživatel vidí pouze název činnosti a datum začátku i konce. Pro jakékoliv další informace je nucen otevřít okno vlastností činnosti.

Za zmínku také stojí, že program umí exportovat data do různých formátů jako je třeba HTML či PNG. S importováním je to však horší. Program by měl zajistit kompatibilitu dat s programem Microsoft Project, a to dokonce skrze tři různé formáty. Vytvořil jsem tedy v Microsoft Project plán, který jsem následně vyexportoval do všech tří formátů a dále jsem se je pokusil importovat do programu GanttProject. Po importu prvních dvou formátů se sice načetla částečně některá data, avšak se špatnými časy či vazbami. Poslední formát xml, paradoxně dopadl nejhůře. V tomto případě skončil celý program s chybovým hlášením. Ne jinak tomu bylo i při opačném postupu.

Kooperaci s Microsoft Project zde tedy můžeme považovat za nefunkční. Vlastní ukládání a zpětné načítání dat ale pracuje správně. Množství funkcí je sice omezeno pouze na základní operace, ale tento fakt spolu s výbornou organizací prvků na obrazovce dělá z programu skvělou aplikaci pro začínající projektové manažery.

3.3 Komerční aplikace

Programy patřící do této kategorie už obsahují nástroje pro kompletní časovou analýzu. Díky tomu provázejí projekt po jeho celou dobu trvání. Programy dokážou vyhodnocovat zadané úkoly v souvislosti s časem a reflektují současný stav projektu. Porovnání obsahu programu a reality by nám tak dá představu, jak si projekt vede v rámci časových termínů.

3.3.1 Microsoft Project

Jak je patrné z názvu, jedná se o produkt firmy Microsoft založené v roce 1975. Firma chtěla přijít s nástrojem na projektové řízení, a proto uvedla na trh v roce 1984 Microsoft Project, v té době pro operační systém DOS. Od té doby se na trhu objevilo nespočet verzí. Zatím poslední verze, kterou jsem použil i k analýze programu, byla vydaná v roce 2013. Napříč tomu, že Microsoft prezentoval software vždy jako součást sady Microsoft office, nelze jej v žádné verzi Office nalézt. Prodává se zvlášť a cena licence pouze na jeden počítač se pohybuje kolem 20 000 Kč. K analýze jsem však mohl použít plnou verzi, která je dostupná v rámci licence Dream Spark, pro všechny studenty Fakulty Informačních technologií VUT v Brně.

Microsoft Project je aplikace pro plánování a projektové řízení, který je na profesionální úrovni. Jednou z hlavních výhod je týmové plánování a synchronizace plánů. Na projektu tak může pracovat více lidí současně či projekt může uživatel spravovat téměř odkudkoliv. Aplikace obsahuje velmi rozsáhlou paletu nástrojů pro projektové řízení. Pro mě jsou ale důležité nástroje týkající se především času. Podpora metod CPM a PERT je integrována. Program umožňuje nejen více typů zobrazení časových plánů, ale i možnost aktivní správy v jakémkoliv z nich. Je tedy pouze na člověku zda upřednostňuje práci v síťovém grafu, Ganttově grafu či seznamů úkolů. Dále je zde také možnost spravovat jednotlivé týmy pracovníků a přidělovat jim činnosti. V závislosti na zdrojích, termínech či předchůdcích je možno nechat plánování čistě jen na produktu.

Problémem tohoto programu je však jeho složitost. Uživatel ač velmi dobře znalý informačních technologií může mít problém se v uživatelském rozhraní orientovat. Některé prvky nelze na první pohled snadno nalézt a celková organizace může působit zmateně. V programu, jenž ale obsahuje tak rozsáhlé množství funkcí se tomuto jevu lze jen těžko vyhnout. Program obsahuje kvalitní nápovědu a i na internetu lze nalézt spoustu tutoriálů k naučení práce s produktem.

3.3.2 PERT Chart EXPERT

Posledním programem naší analýzy je PERT Charts Expert vydaný společností Critical Tools Inc. Tato organizace spatřila světlo světa v roce 1991 a stala se certifikovaným partnerem společnosti Microsoft. Zaměřuje se tedy na programy pro Windows určené k projektovému managementu. V současné době využívá software firmy přes 25 000 organizací po celém světě včetně IBM, Microsoftu, Siemens, General Motors a dalších.

První verze programu PERT Chart Expert 1.0 vyšla začátkem roku 1996. Menší úpravy následovali řadu let. K hlavnímu přelomu však došlo v roce 2001, kdy vyšla verze 2.0 s mnohými změnami. Současná a také analyzovaná verze je z roku 2012 s označením 2.8a. Jedná se však o americký produkt, který u českých prodejců není dostupný. Objednání přímo u firmy Critical Tools stojí $199 za jednu licenci. Pro tyto účely jsem použil třicetidenní trial verzi s neomezenou funkčností.

Zřejmě největší výhodou je plná kompatibilita s MS Project. Tvůrci tak nejsou omezováni ani platformou a mohou tvořit a spravovat plány bez ohledu na software. Pokud máme například v MS Project pouze seznam činností s příslušnými daty, PERT Chart EXPERT automaticky vytvoří PERT graf. Program lze však plně využívat i samostatně. Také nabízí práci v mnoha zobrazovacích režimech podle množství zobrazovaných dat. Každá činnost je reprezentována jako objekt, k jehož vlastnostem lze přistupovat a spravovat tak stav i v průběhu projektu. Objekty jsou vzájemně provázány, takže každá změna je reflektována do celého projektu. Například při editaci doby trvání některé činnosti je vždy aktualizována kritická cesta. Na výbornou je v programu provedeno i rozhraní. Program i práce s ním je velmi jednoduchá. Je zde implementováno velmi netradiční rozhraní, kdy uživatel nepotřebuje žádné přepínání mezi kontextem. Rozdílnou prací s myší vykonává nejrůznější úkony. Uživatel tak může tvořit činnosti, vazby, přesouvat objekty a editovat data, aniž by musel přepínat kontext myši.

Nevýhodou u této aplikace může být pro některé vysoká cena za poměrně jednoduchý program. Dále zde také chybí možnost interní synchronizace. Pokud chce člověk s někým spolupracovat, musí program uložit a po té mu tento soubor poslat. Je ale pravdou, že na rozdíl od MS Project tento program postrádá spoustu dodatečných nástrojů, na druhou stranu absence právě takovýchto prvků dělají PERT Chart EXPERT jednoduchým programem pro manažery, kteří nemají možnost strávit nějaký čas prostudováním manuálů náročnějších programů.

3.4 Zhodnocení současného stavu

Všechny programy které se v současnosti objevují na trhu nesplňují zcela přesně zvolená kritéria. Jsou zde některé, co nabízí uživateli dle konkrétního kritéria naprosto excelentní implementaci. Co se však týká jiných, výrazně pokulhávají. Jde o to stanovit mez, do jaké je proveditelné konkrétní kritérium a současně mohla být splněna i ta ostatní.

Dle nabízených funkcí se jeví jako nejvíce vhodná aplikace MS Project. Je však zbytečně složitá a cena pro některé firmy příliš vysoká. Co se týká nekomerčních aplikací, vítězem je ProjectLibre. I přes pár nedostatků nejvíce vyhovovala zvoleným kritériím a měla by mít k naší vyvíjené aplikaci nejblíže. Kdybychom ale brali v potaz pouze uživatelské rozhraní, na prvním místě by byla aplikace GanttProject. Její umístění položek, jednoduchost i grafická podoba prvků ve formuláři byla nejvíce přívětivá uživateli. Velkou nevýhodou nekomerčních aplikací je také ten fakt, že neposkytují žádné nástroje pro správu plánu po dokončení plánování.

Kapitola 4

Návrh

Cílem této kapitoly je vytvořit návrh aplikace na základě požadavků a kritérií zvolených v předchozí kapitole, který definuje jak bude výsledná aplikace vypadat. Vytvořený návrh poslouží jako předloha pro samotnou implementaci.

4.1 Návrh architektury aplikace

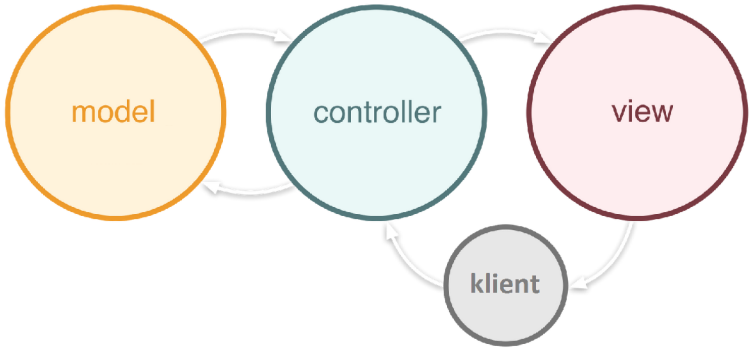
Architektura určuje celkovou strukturu vyvíjené aplikace. Popisuje, jakým způsobem lze aplikaci rozdělit a do jakých logických celků je uspořádána. Jedná se o základní kostru celé aplikace. Architekturu aplikace je potřeba definovat ještě před začátkem návrhu, aby možné provést rozdělení dle vybraného modelu architektury. Změna architektury v průběhu implementace bývá velmi těžko proveditelná a může znamenat i zahození celého projektu. Rozdělení architektury je důležitou součástí především větších projektů, kde na jednotlivých komponentách pracují rozdílní lidé.

Pro naši aplikaci jsem zvolil architekturu **Model View Controller** neboli **MVC**, která má základní myšlenku oddělit logiku od výstupu. Jedná se o velmi populární model, určen k návrhu aplikací všech druhů. Aplikace, ve kterých je implementována logika, vizualizační funkce i interakce s databázemi v jednom souboru, jsou náročné na údržbu. Soubor je nepřehledný a práce s ním je výrazně pomalejší. Při rozdělení aplikace na jednotlivé komponenty je ale editace některé z částí snáze proveditelná a má minimální vliv na ostatní.

MVC rozděluji aplikaci na tři komponenty, které se nazývají Model, View a Controller [9].

- **Model -** Do Modelu patří všechny logické operace aplikace, ať už se jedná o aritmetické výpočty, databázové dotazy či validace dat. Není zde obsažena nejmenší zmínka o způsobu předávání dat s uživatelem aplikace, je to jádro aplikace, které dostane již konkrétní data v definované podobě a převede je za pomoci posloupnosti činností na výstupní data. Model neví, odkud se data berou, ani jak bude dále s daty naloženo. Model má plnou odpovědnost za validitu dat celé aplikace.

- **View -** Tato komponenta vyobrazuje obsah modelu, většinou na grafický výstup aplikace. Můžeme si jej představit také jako grafické rozhraní aplikace. Zobrazení musí být neustále aktuální. Při jakékoliv změně modelu se musí View aktualizovat



Obrázek 4.1: Rozdělení vrstev architektury MVC [12]

(překreslit). Současně slouží pro příjem událostí od uživatele, které následně předává Controlleru

- **Controller -** Jedná se o řídící část modelu. Zde jsou definovány reakce na události, které vyvolá uživatel. Dle akce uživatele je zavolána příslušná metoda, která mění stav modelu[3]. Většinou dochází i k aktualizaci View.

Tato ustálená architektura dělá z aplikace dobře udržovatelný modulární program s eliminací duplicitního kódu. Je vhodná k implementaci víceuživatelských aplikací, jelikož při zavedení nového klienta do programu stačí vytvořit pouze nový view. Model a Controller tak zůstává zcela bez zásahu.

4.2 Funkce aplikace

Aby byla aplikace bezpečná, každý uživatel se smí dostat pouze k těm datům se kterými má dovoleno pracovat. Funkce programu by měla být rozdílná pro uživatele s různým oprávněním. V tomto případě postačí dvě úrovně přístupu, které budou přiděleny aktérům podle jejich role v projektu. Aplikaci bude možno využívat jako projektový manažer nebo jako zaměstnanec.

4.2.1 Projektový manažer

Projektovým manažerem se lze stát, pokud vytváří nový projekt nebo otvírá již existující a zná přístupové údaje k této roli. V případě zakládání nového projektu si zvolí své přístupové údaje a současně nastaví základní atributy projektu. Následně bude mít k dispozici celou funkčnost aplikace. Je jediným aktérem který má na starost kompletní správu všech činností, včetně jejich vytváření a editaci.

Činnosti budou reprezentovány tabulkou kde budou uloženy společně se svými atributy. Dále také vytváří zdroje a přiděluje je k jednotlivým činnostem. Celý projekt může také vytvořit importováním ze souboru xml, nebo lze soubor exportovat do stejného formátu. Celý projekt může být uložit i do online databáze, nebo jej z databáze odebrat. Projektový manažer samozřejmě může vykonávat všechny funkce, které smí provádět zaměstnanec. Zaměstnance v projektu přidává taktéž projektový manažer a má právo je kdykoliv během projektu editovat.

4.2.2 Zaměstnanec

Zaměstnanec má v projektu nejméně privilegií. Má právo k zobrazení celého projektu včetně detailů všech činností. Editovat ale může pouze činnosti ke kterým je přiřazen projektovým manažerem. Zaměstnanec nemá právo zasahovat do celého plánu, tj. přidávání i odebírání činností je zcela v režii projektového manažera.

Hlavním důvodem proč by měl mít zaměstnanec přístup k editaci atributů své činnosti plánu je, aby mohl informovat ostatní o stavu práce svého týmu. V mnohých případech týmy pracující na odlišných částech během projektu nemusí potkat. Je však nutná vzájemná znalost aktuálního stavu. Dojde-li tak v jedné části projektu k výraznějšímu odchýlení od plánu, skutečnost se zaznamená do plánu a vedoucí ostatních týmu jsou na ně okamžitě schopni reagovat.

4.3 Návrh uživatelského rozhraní

Návrh uživatelského rozhraní aplikace je velmi důležitou součástí, jelikož se zde rozhoduje o celém prostředí programu. Skrze toto rozhraní komunikuje s aplikací a to jak při zadávání či pozorování dat. Proto je velmi vhodné návrh koncipovat tak, aby byl koncovému uživateli šitý na míru. Co když ale program není určen konkrétnímu uživateli? V takovém případě je potřeba zvážit na jakou cílovou skupinu se zaměříme. Za naši cílovou skupinu jsem zvolil projektové manažery menších firem.

Při spuštění aplikace se otevře hlavní okno kde si uživatel vybere, jestli chce započít nový projekt nebo si chce otevřít již stávající. Zadá-li uživatel možnost vytvořit nový projekt, otevře se nové okno, ve kterém se nastaví základní atributy projektu. Poté nebo při otevření již stávajícího projektu se zobrazí hlavní okno aplikace.

Hlavní okno aplikace reprezentuje celý projekt a bude umožňovat jeho tvorbu i editaci. Bude zde menu, které umožní přístup ke všem funkcím programu. Položky v menu budou seskupeny podle jejich účelu. Dále zde umístím panel nástrojů ve kterém bude mimo jiné možnost vybrat si, jestli chceme pracovat s plánem či spravovat zdroje. Ke spravování zdrojů bude mít ovšem přístup pouze projektový manažer. Na panelu nástrojů implementuji ikony znázorňující činnost, která se spustí při kliknutí na ni a níže umístím okno ve kterém zobrazím procesy. Toto okno se bude lišit dle režimu, ve kterém se bude uživatel nacházet.

V režimu spravování zdrojů se bude v okně nacházet tabulka, do které uloží uživatel prvky, včetně jejich atributů. Aby bylo možné zdroj použít, bude nutné vyplnit všechny povinné atributy, jako jsou přihlašovací údaje osob. Další atributy mohou zůstat prázdné.

V druhém režimu plánuji rozdělit okno vertikálně do dvou oblastí. V levé části se bude nacházet správa činností. K tomu je vhodný strukturovaný výpis, který můžeme opět reprezentovat pomocí tabulky. Do jednoho sloupce umístím název činností a v dalších jednotlivé informace o nich, jako např. datum zahájení, datum ukončení nebo jejich stav. Zde bude uživatel tvořit veškeré činnosti a editovat jejich vlastnosti. Do pravé části umístím zobrazení harmonogramu v grafické podobě. Uživatel si bude moci vybrat z více zobrazení, která by měla být také multifunkční. Například při kliknutí na objekt, který reprezentuje činnost, se objeví vyskakovací okno reprezentující jeho atributy. Uživatel by měl také mít možnost zobrazit si plán ve více grafických formátech. Nové zobrazení by se mělo otvírat v dalším režimu.

V posledním režimu bude celkový souhrn informací o projektu. Implementuji zde zobrazení důležitých termínů, přidělených zdrojů a celkové statistiky projektu. Další podpůrné funkce budou realizovány jako vyskakovací okna s interaktivními formuláři.

Kapitola 5

Implementace

V této kapitole se budu zabývat implementací aplikace pro podporu časového plánování. Tato fáze vychází z návrhu a zabývá se samotným vývojem. Pod pojmem implementace se však neskrývá pouze programování. Popíši zde svoje způsoby řešení zajímavých částí aplikace či jakým způsobem jsem se vypořádal s problémy které při tvorbě aplikace vznikly.

Tato kapitola je rozčleněná do tří částí. V první jsou popsány použité implementační nástroje. V další části je uveden způsob interní reprezentace dat a v poslední se nachází popis funkčních prvků, společně s grafickými částmi aplikace.

5.1 Implementační nástroje

Aplikaci jsem realizoval pomocí jazyka Java. Podle něj jsou zvoleny i všechny ostatní technologie a postupy, které jsou v práci zahrnuty. Jazyk jsem vybíral dle svých zkušeností a také podle toho, aby bylo možné výsledný program spustit na různých platformách. Hlavním důvodem byla ale přenositelnost programu.

5.1.1 Java

Jedná se o programovací jazyk od firmy Sun Microsystem, který byl světu představen v roce 1995. Přestože se tehdy jednalo o zcela nový jazyk, velké množství programátorů v něm začaly programovat své aplikace a ještě před rokem 2000 jej využívalo více než tři milióny programátorů. V současné době se jedná o nejpoužívanější programovací jazyk. Před pár lety převzala kontrolu nad vývojem firma Oracle, která jej stále řídí a v letošním roce vydala novou verzi 8.

Díky své přenositelnosti je používán k vývoji programů, které musí běžet na více platformách, jelikož se jedná o interpretovaný jazyk. V praxi to znamená, že je program nejprve přeložen do mezikódu, který je následně analyzován a interpretován programem **Java Virtual Machine** (JVM). Program lze tedy spustit všude tam, kde se příslušné JVM nachází. Při vyvíjení aplikací v Javě je dále potřeba mít nainstalován **Java Developement Kit** (JDK). Ten poskytuje nástroje k přeložení zdrojového kódu, generování dokumentace, ladění programu a dalším věcem.

Java je objektově orientovaný programovací jazyk, který je mnohými instituty užíván k výuce tohoto typu programování. Jeho syntaxe má základ v jazycích C a C++. Nenalezneme zde ale žádné složité konstrukce, nýbrž spoustu užitečných rozšíření. Jazyk Java je tak velmi jednoduchý a přehledný. Podporuje zpracování vícevláknových aplikací a napříč tomu, že se jedná o interpretovaný jazyk, ztráta výkonu není nijak veliká.

V naší aplikaci jsou z velké části použity knihovny, které jsou součástí JDK. V některých částech bylo ale nutné použít knihovny externí, jako například pro komunikaci s mysql serverem, k vykreslování grafů či k tvorbě a parsování xml formátu. Tyto knihovny a jejich použití budou popsány v této kapitole.

5.1.2 NetBeans

Netbeans začal jako studentský projekt v roce 1996 v České Republice. Jejich produkt, tehdy ještě od názvem Xelfi, vzbudil tolik zájmu, že se jej rozhodli vpustit na trh jako komerční. V roce 1999 přešel vývoj pod firmu Sun Microsystems, která je hlavním sponzorem. Od roku 2000 je produkt uvolněn pod licencí open-source. Vývoj ale stále probíhá převážně v Praze českými vývojáři.

Toto vývojové prostředí je napsáno v jazyce Java a je primárně určeno k vývoji aplikací v tomto jazyce. Je zde zabudována podpora pro tyto tři hlavní platformy Javy.

1. **J2SE (Java Standart Edition) -** původní verze, je postupně rozšiřována
2. **J2EE (Java Enterprise Edition) -** platforma určená na vývoj a provoz podnikových aplikací a informačních systémů
3. **J2ME (Java Micro Edition) -** podmnožina verze J2SE, určena k vývoji pro malá zařízení s omezenými prostředky

Během let však došlo k rozšíření a je možné jej použít i k práci s jazyky C++, PHP a další.

Pomocí NetBeans se dají efektivně vytvářet uživatelská rozhraní, přehledně zpracovávat rozsáhlé projekty, ladit je a dále šířit. Na trhu je také velké množství modulů, které je možné k NetBeans připojit a rozšířit tak jeho možnosti. NetBeans také nabízí výbornou korekturu projektu, která se hodí například při přejmenování prvků či jiných změn.

Náš software bude klasická okenní aplikace. Proto jsem k návrhu použil GUI Builder v aplikaci Netbeans. Díky němu je možné pohodlně vytvářet a organizovat všechny ovládací prvky aplikace, které jsou zde přístupné na paletě dostupných komponentů.

Tyto vlastnosti rozhodli o tom, že jsem NetBeans vybral jako prostředí k samotné implementaci. Umožnilo mi pohodlné vytvoření rozhraní jednotlivých oken aplikace, jednoduchou editaci kódu a přehlednou správu souborů projektu.

5.2 Implementace datové vrstvy

Při implementaci většiny dnešních aplikací je důležité začít implementací datového základu. Od něj se všechno odvíjí a právě s ním chce uživatel pracovat. Aplikace bude kombinovat několik typů uložení dat. Při práci s aplikací budou data uloženy v dočasných souborech, o což se stará jazyk Java. Při uložení dat na disk, či importování do programu MS Project, budou data uloženy do xml souboru, a pokud bude manažer chtít nabídnout hotový plán k nahlédnutí či editaci svým kolegům online, nahrajeme data do mysql databáze.

5.2.1 Reprezentace dat v jazyce Java

Výhodou objektově orientovaného programování je ten fakt, že jednotlivé třídy (objekty) jsou v abstrahované objekty z reálného světa. Proto jsem i já volil takovou strukturu, aby

reflektovala reálné prvky, které se vyskytují při plánování projektu. Všechny třídy jsou uloženy ve společném balíku *Model*.

Hlavní třída je uložena v souboru *Project* a jak již napovídá její název, jsou zde uloženy informace, které se týkají samotného projektu jako například datum zahájení, název či poznámky. Všechny proměnné zde i ve všech dalších třídách jsou privátní. Lze k nim přistupovat pouze pomocí implementovaných funkcí, které ošetří nežádoucí operace s daty.

Činnosti časového plánu jsou reprezentovány třídou *Activity*. Přehledně uchovává údaje jednotlivých činností včetně všech časových údajů zadaných jak uživatelem, tak spočtených. Zároveň si uchovává informace o činnostech předcházejících, hmotných a lidských zdrojích a o současném postupu, tj. v jaké fázi se činnost nachází.

Ve třídách *Human* a *Material* se nachází zdroje projektu. Rozhodl jsem se zvolit dvě rozdílné třídy, a to z důvodu odlišnosti ukládaných dat. Ve třídě *Human* jsou uložený lidské zdroje projektu, u kterých je potřeba ukládat informace jako jsou přihlašovací jméno, heslo, přidělené činnosti a identifikační číslo. Co se týká materiálních zdrojů, je zaznamenán název zdroje, jeho popis, identifikační číslo a odkaz na činnost, které je daný zdroj přidělen.

Poslední datovou třídou je *Predecesor*. Tato třída představuje vazbu mezi činností a jeho předchůdcem. Kolekce těchto tříd je uložena v každé činnosti. Obsahuje informace o tom, která činnost je předchůdce a v jaké logické návaznosti.

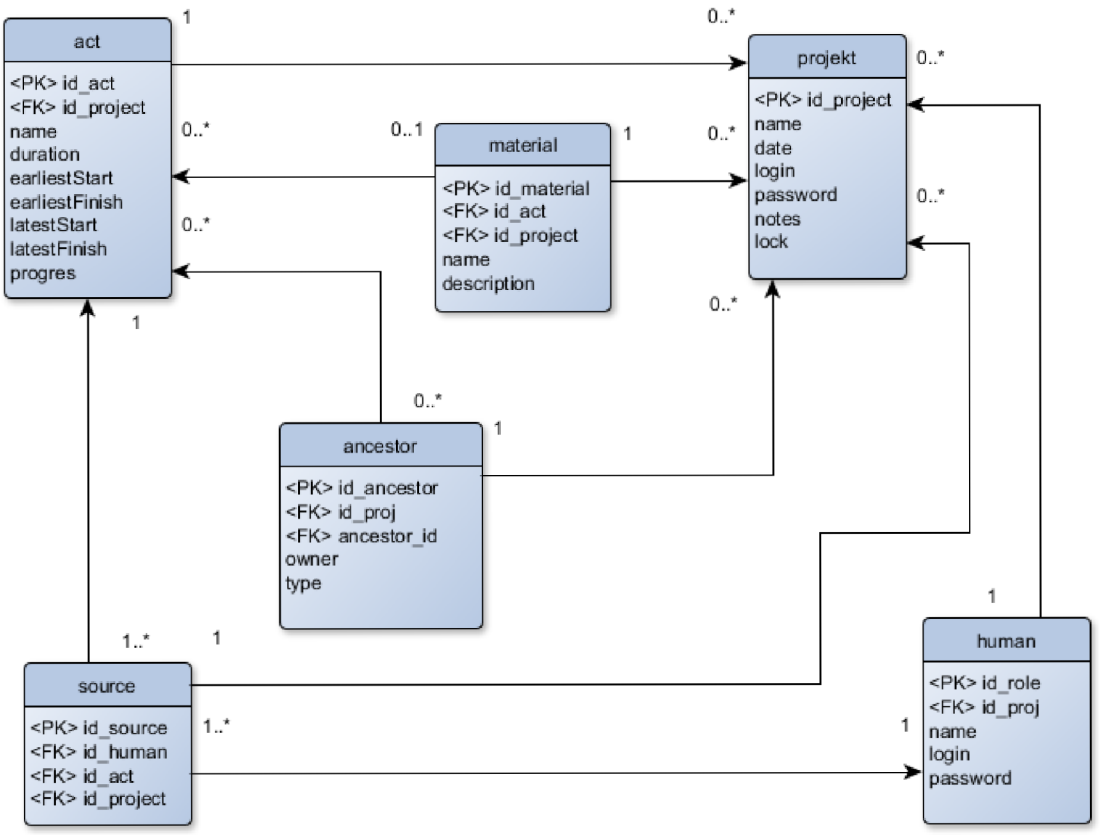
Ve třídě projektu se ještě nachází kolekce tříd *Activity, Human* i *Material*. Jsou zde uloženy všechny existující prvky projektu.

5.2.2 Implementace databáze

Struktura databáze je složena z 6 databázových tabulek, dle ER diagramu na obrázku 5.1, přičemž každá entita návrhu reprezentuje samostatnou tabulku databázové struktury. Také jsou zde uvedeny všechny atributy tabulek a vzájemné vazby mezi nimi. Vztahy mezi tabulkami jsou implementovány za pomoci cizích klíčů. Při manipulování s databázovými tabulkami tedy pracujeme s dvěma typy klíčů: primární (*PRIMARY KEY*) a cizí klíč (*FOREIGN KEY*). Každá tabulka má svůj primární klíč, který je v případě vztahu s další tabulkou spojen jako cizí klíč. Pro primární klíč musí být splněna omezení, které vyžadují jedinečnost každého klíče v rámci celé tabulky. Pokud by došlo k situaci, kdyby dvě položky obsahovaly stejný primární klíč, došlo by ke kolizi dat. Primární klíč nesmí být nikdy změněn, a to z důvodu ztráty reference. K zajištění unikátnosti primárního klíče slouží parametr *AUTO INCREMENT*. Ten nám vygeneruje unikátní klíč podle stávajícího obsahu tabulky. Primární klíč v tabulce projektu je mimo jiné i identifikátor, který je vyžadován při přihlášení k projektu.

Ukládání dalších atribut entity je implementováno jako instance daných typů proměnných, podobně jako v programovacích jazycích. Pro ukládání názvů, poznámek a všech textových hodnot je použit typ *VARCHAR*, který oproti typu *CHAR* zabírá méně místa na disku. Pro čísla a booleovské výrazy bylo použito typů *DOUBLE*, *INT* a *BOOLEAN* a pro uchování data proměnná typu *DATE*.

Funkce zajišťující operace s databázovým serverem jsou obsaženy ve třídě *DB*. Při její inicializaci je proveden pokus o přihlášení na databázový server pomocí funkce *getConnection()*. Ten je realizován na adrese *sql2.freemysqlhosting.net* s názvem *sql238251*. Při přihlášení je také potřeba zadat uživatelské jméno *sql238251* a heslo *tV6!aH4!*. Všechna tato data jsou uložena v aplikaci a jsou na server odesílána automaticky. Po úspěšném přihlášení lze možno vyvolat některou ze tří implementovaných funkcí.



Obrázek 5.1: ER diagram databáze

Pro připojení k již existujícímu projektu uloženému online se vyvolá funkce *loadData*. Ta vyžaduje dva argumenty, první je odkaz na projekt, do kterého se mají data nahrát a druhý je celé číslo označující projekt. Funkce jako první začne ukládat obecné informace o projektu, poté jednotlivé činnosti a jejich atributy, nastaví se předchůdci a jako poslední se uloží a přiřadí zdroje. Jakmile jsou všechna data staženy, dopočítají se potřebné prvky a je možné s projektem pracovat.

Ukládání dat na databázový server zajišťuje funkce *saveData*. Ta dostává v argumentu odkaz na projekt který je potřeba uložit. Postupuje se zde ve stejném pořadí jako při stahování dat. Jediným rozdílem je smazání již neaktuálních dat z databáze.

Poslední operace s databází se nachází ve funkci *access*. Ta je vyvolána při přihlašování k projektu. Tato funkce dostává trojrozměrné pole řetězců, ve kterém se nachází identifikátor projektu, přihlašovací jméno a heslo. Podle toho se na serveru nejprve vyhledá projekt s příslušným id a poté se hledá, zda-li a s jakými právy má uživatel přístup k projektu. To je reprezentováno v návratové funkci projektu.

Vzhledem k tomu že k databázi budou přistupovat především vedoucí jednotlivých činností v průběhu celého projektu a nastavovat pouze jeho aktuální stav, je malá šance, že by plán chtěli editovat současně. I tak je ale nutné implementovat alespoň jednoduché ošetření konfliktu. Proto jsem implementoval jednoduchý zámek, který druhého uživatele upozorní, že na projektu momentálně pracuje někdo jiný. Tím se vyhneme práci s neaktuální databází.

5.2.3 Reprezentace dat v XML

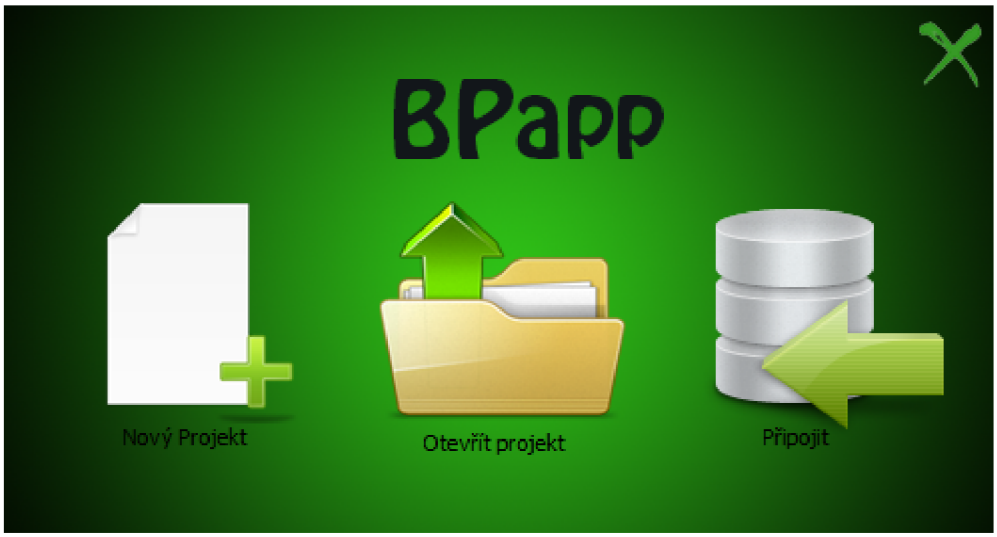
Přestože lze plán uložit na databázový server, manažer projektu se může mnohdy ocitnout v situaci, kdy nemá s počítačem přístup k internetu. V takovém případě by se aplikace omezila pouze na jednorázovou činnost. Abychom tomu zamezili, je třeba vytvořit strukturu souboru, do kterého by šel plán projektu uložit. Z požadavků na aplikaci také víme, že její nedílnou součástí musí být export do aplikace Microsof Project. Proto jsem se rozhodl tyto dva úkoly spojit a vždy ukládat data z naší aplikace do kompatibilního formátu s požadovanou aplikací.

Jelikož struktura xml je určená k importu a exportu dat z MS Project, je vhodné použít knihovnu, která skrze své funkce umožňuje přehledně pracovat s těmito soubory. V naší aplikaci byla použita knihovna *MPXJ*, konkrétně modul *MSPDI*, který je určen pro tvorbu souborových formátů vyhovující přesně našim požadavkům. Veškeré funkce, které souvisejí s ukládáním či načítáním dat z formátu xml, jsou implementovány ve třídě *xml*. V té se nachází *saveData* a *loadData*, které pracují zcela analogicky dle ukládání a načítání dat z databázového serveru.

5.3 Implementace hlavních funkcí

V této části si popisuji implementaci jednotlivých prvků aplikace, které byly realizovány podle kapitoly návrh 4. Jsou to funkce, které zajistí, aby vyvíjená aplikace splňovala všechny požadované vlastnosti, které jsme si definovali při analýze.

Funkce, které jsou následně popsány, jsou umístěny na vrstvě Controller, avšak pro lepší představu je budu popisovat současně s grafickými prvky na vrstvě View, jelikož právě pomocí těchto prvků přichází všechny události od uživatele. Na základě těchto událostí jsou následně vyvolány jednotlivé metody tříd aplikace. Ty jsou určeny k práci s datovými prvky na vrstvě Model.



Obrázek 5.2: Úvodní menu aplikace

5.3.1 Úvodní menu

Jak jsme zjistili již z analýzy, pro projektového manažera je důležitá přenositelnost uložených dat. Z toho důvodu je po spuštění aplikace uživatel vyzván, aby si vybral jakým způsobem chce nahrát data do aplikace 5.2. Obsluha má na výběr ze třech možností jak postupovat. Při výběru kterékoliv z nich toto okno automaticky mizí.

V prvním případě lze vytvořit nový projekt. Při takovém výběru se bude pokračovat na obrazovku se základním nastavením nového projektu. Jako další možností je otevřít projekt ze souboru. Tuto volbu lze aplikovat nejen v případě dřívějšího uložení z naší aplikace ale i k importu plánu z aplikace Microsoft Project. Při zvolení této položky se otevře okno pro výběr souboru k otevření a po jeho potvrzení je zobrazeno hlavní okno aplikace s načtenými daty. V opačném případě se vracíme zpět do úvodního menu. V obou těchto případech je uživatel automaticky v roli projektového manažera.

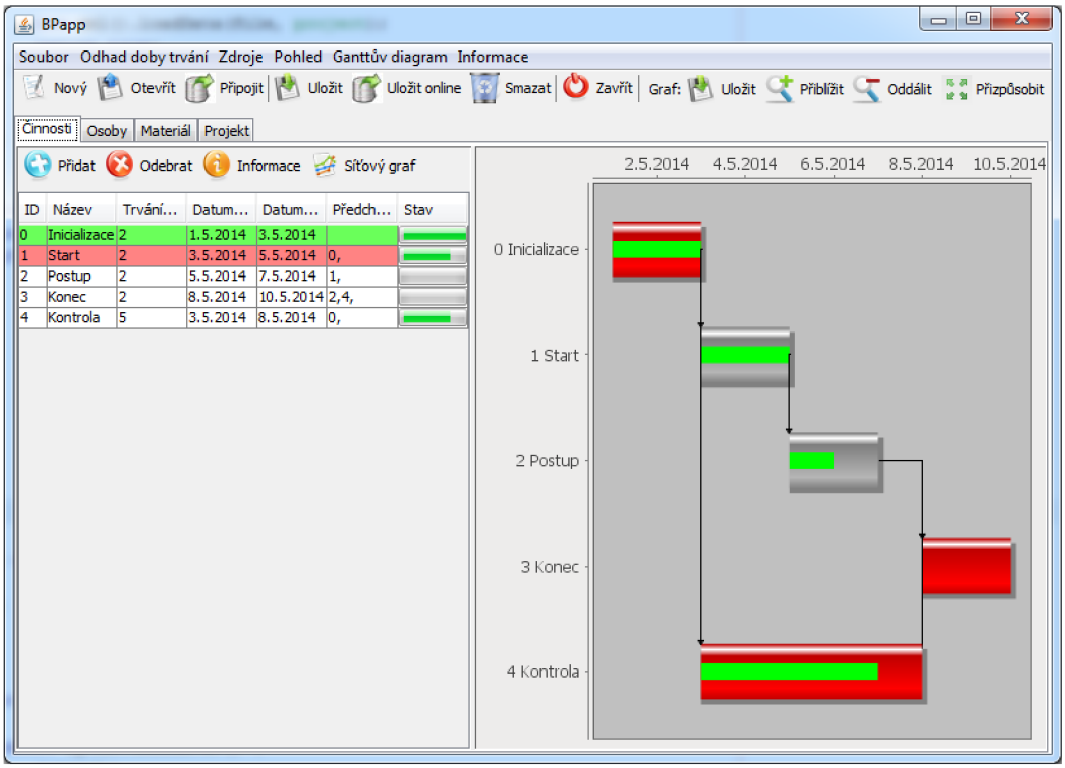
Posledním způsobem je připojení k projektu uloženém na databázovém serveru. Tato volba vyvolá okno, kde je uživatel vyzván k zadání přihlašovacích údajů. Jedná se o identifikační číslo projektu a přidělený login společně s heslem.

5.3.2 Vytvoření nového projektu

Vytvořením nového projektu se uživatel staví do role projektového manažera a může přidělovat přístup i ostatním členům. Nový projekt je možné vytvořit ihned po spuštění aplikace ale i v průběhu práce na jiném projektu. Při práci na jiném projektu je však nejprve nutné rozpracovaná data uložit, jelikož lze současně pracovat pouze na jednom projektu.

Pro vytvoření nového projektu uživatel musí vyplnit příslušný formulář, který definuje následující vlastnosti projektu:

- název
- datum zahájení
- popis



Obrázek 5.3: Hlavní okno aplikace

- výběr způsobu zadávání doby trvání

Vyjímaje položky *popis* jsou všechny prvky povinné. Při uživatelově příkazu k vytvoření projektu jsou zadaná data kontrolována a v případě nevyhovujícímu obsahu vyzvou uživatele k opakovanému zadání. K takovéto situaci ale může dojít pouze tehdy, zapomene-li uživatel některý z povinných údajů. Díky komponentě *jDateChooser* uživatel může vybrat datum pomocí kalendáře, přičemž je formát data zvolen automaticky.

Způsob zadávání doby trvání vybere uživatel pomocí dvou prvků *jRadioButton*, které patří do společné skupiny. Díky tomu lze vybrat zároveň pouze jednu z těchto variant. Na výběr má uživatel mezi přímým zadáváním a odhadem pomocí metody PERT. Je-li vše v pořádku, vytvoří se nová třída *Project*, do které se všechny informace uloží.

5.3.3 Hlavní okno aplikace

Po úspěšném vytvoření, otevření či připojení se k projektu je zobrazeno hlavní rozhraní projektu. Právě s tímto oknem bude uživatel po čas celého plánování v kontaktu a prostřednictvím něj bude zadávat své požadavky. Okno, které můžeme vidět na obrázku 5.3 je inicializováno třídou *MainFrame*.

Rozhraní je implementováno podle návrhu a je rozděleno do více částí. V horní části je umístěna lišta, ve které se nachází nástroje pro operace s celým projektem jako je například uložení, vytvoření nového projektu, ukončení projektu a tak dále. Pod touto lištou leží hlavní část naší aplikace, která je rozdělená do 4 záložek, dle jednotlivých prvků projektu.

První záložka je určena pro činnosti projektu společně s Ganttovým diagramem, druhá pro správu lidských zdrojů, třetí pro správu materiálních zdrojů a poslední pro souhrnné

informace o projektu. Máme-li hotový plán, je možné přidat ještě jednu záložku, pod kterou se nachází reprezentace činností v síťovém grafu.

5.3.4 Záložka činností

Přejdeme-li na záložku správa činností, rozprostře se okno rozdělené do dvou hlavních celků. Vlevo se nachází část pro editaci činností a na druhé straně leží Ganttův diagram. Tyto dvě části jsou úzce spjaty a každá změna provedená na levé straně je ihned interpretována i v části druhé. Obě komponenty leží v položce *Split Pane*, což umožňuje měnit šířku jedné z nich v závislosti na druhé, beze změny rozměrů celkového okna.

Správa činností

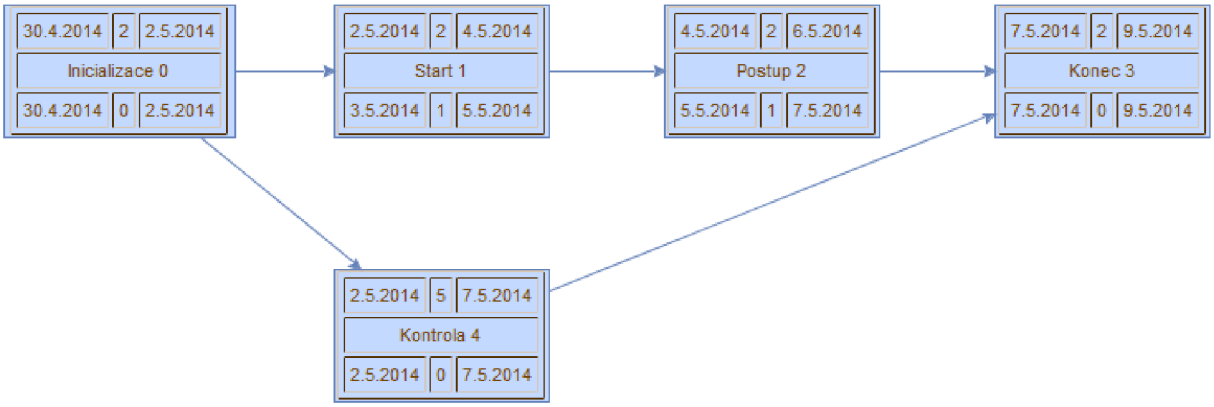
Činnosti jsou v aplikaci reprezentovány ve formě tabulky. Hlavním důvodem je přehlednost a možnost zobrazení atributů daného úkolu. Jednotlivé řádky tabulky znázorňují činnosti a sloupce údaje o ní. V první kolonce se nachází jedinečné identifikační číslo činnosti. Podle tohoto čísla je možné s činností pracovat. Vedle identifikačního čísla jsou zobrazeny další vybrané atributy činností jako například název, doba trvání, datum zahájení a ukončení či seznam předchůdců. Pro zobrazení podrobnějšího výpisu informací či pro editaci si uživatel může otevřít okno s podrobnějšími informacemi o projektu.

Řádky tabulky lze jednoduše přidávat či odebírat pomocí nástrojového panelu umístěného nad tabulkou nebo díky rozbalovacímu menu, které se otevře po kliknutí na tabulku levým tlačítkem myši. V obou nabídkách se nachází položka k zobrazení podrobnějších informací o činnosti. Při vyvolání této akce a akce smazání činnosti je nutné mít vybranou některou činnost z tabulky. Není-li tomu tak, uživatel je upozorněn vyskakovacím oknem.

Ganttův diagram

Při časovém plánování projektů je nedílnou součástí také vhodná reprezentace činností v závislosti na čase. Jak jsme se obeznámili již v kapitole 2.3.5, nejvhodnějším a nejčastěji používaným znázorněním je Ganttův diagram. Jsou v něm zobrazeny všechny činnosti projektu podle data zahájení a ukončení. Vykreslení Ganttova diagramu můžeme rozdělit do dvou fází. První je příprava dat, které slouží jako datový podklad pro diagram. Tuto část vykonává třída *CalculateDate* prostřednictvím svojí funkce *calculate*. Ta počítá potřebné termíny pro tvorbu diagramu za pomocí algoritmů popsaných podrobněji v kapitole 2.3.3. Spočtené údaje jsou uloženy do třídy *Activity* k příslušné činnosti. Zde se nastavuje i příznak, který označuje činnosti, které se nacházejí na kritické cestě plánu. Aplikace je navržena tak, aby umožnila správně vypočítat plány pro všechny vzájemně nezávislé shluky činností.

Druhou částí je samotné vykreslení grafu. K mému překvapení knihoven pro tvorbu Ganttova diagramu není velké množství. Musíme-li zanedbat ty, které nejsou licencovány jako open-source, zůstanou nám pouze dvě trošku rozšířenější knihovny. Bohužel, ale dokonce ani jedna z nich plně nevyhovovala našim požadavkům na funkčnost. Nakonec jsme museli vybrat tu, u které bude její chybějící část nejsnáze nahraditelná. Vítězem se tedy nakonec stala knihovna *JFreeChart*, která provádí vykreslování na standardním Java 2D API. Právě díky tomuto faktu je knihovna tolik rozšířená, umožňuje totiž její použití jak na desktopové aplikace, tak na webové aplikace ať už na straně klienta či



Obrázek 5.4: Ukázka zobrazení PERT diagramu

serveru. Tato knihovna podporuje velké množství grafů jako jsou bodové, spojnicové, kruhové, polární, sloupcové a tak dále. Nejdůležitějším je Ganttův diagram.

Každá činnost vyskytující se v plánu je obdélníkově vykreslena i v Ganttově diagramu. Šířka každého obdélníku určuje, dobu trvání události, což lze vyčíst z časové osy umístěné v horní části grafu. Její měřítko se automaticky změní podle zobrazených dat. Pro lepší orientaci v zobrazovaných činnostech je u všech zobrazených činností uveden i její název. Přestože zobrazení vazby činností je poměrně novinkou, pro přehlednost je dobré ji do diagramu uvést. Bohužel, je to právě tato schopnost, kterou JFreeChart nepodporuje. Díky rozšířenosti této knihovny naštěstí existuje spoustu nezávislých programátorů, kteří k ní vytváří nejrůznější patche. A právě jeden z nich, konkrétně od Francouze vystupující pod přezdívkou jrebillat, řeší i tuto nepříjemnost. Tato knihovna nám také umožňuje zobrazení průběhu činnosti uvnitř grafu, což je v aplikaci využito, jelikož je vhodné aby měl uživatel přehled jak o skutečném průběhu činnosti, tak i o plánovaném. V naší aplikaci je v Ganttově diagramu vyobrazen plánovaný průběh činnosti v zobrazenou dobu.

Tvorbu samotného grafu provádí funkce *createGanttChart*, která jako jeden z argumentů vyžaduje speciální struktury s naplněnými daty. Její vytvoření a naplnění je realizováno ve třídě *ChartData*. Zde se importují jednotlivé činnosti, nastaví se jejich plánovaný stav v aktuálním čase a poté nastaví případně vazby. Činnosti zde ukládají do takzvaných sérií. Při vykreslování grafu jsou barvy pro jednotlivé činnosti zvoleny podle jejich CPM příznaku, který udává, zda-li se daná činnost nachází na kritické cestě či nikoliv. Touto formou je umožněno zobrazení více kritických cest.

Při stisknutí pravého tlačítka myši nad oblastí grafu je zobrazeno rozbalovací menu. Zde lze použít některou z nabízených funkcí včetně tisku, uložení do formátu *png* či změny měřítka grafu. Nachází se zde i položka vlastnosti, kde je možné editovat základní parametry grafu včetně vzhledu i popisků. Některé prvky z rozbalovacího menu grafu, jsou vyvedeny i na panel nástrojů, jenž se nachází v pravé horní části obrazovky a je zobrazen pouze při vybrání záložky s grafem.

5.3.5 Záložka PERT diagram

Volitelná záložka obsahuje PERT diagram. Ten vychází z klasického síťového diagramu s tím rozdílem, že je každý uzel nahrazen tabulkou. Každá tabulka obsahuje název činnosti, data začátku i konce, dobu trvání a rezervu. Tabulky jsou dále seřazeny v čase a spojeny tak, jak jsme již zvyklí z klasických síťových diagramů.

Implementace tohoto grafu je umístěna do třídy *PERTChart*. Ta umožňuje tvorbu síťových grafů, přičemž obsahem každého uzlu může být HTML kód, který je při vykreslení interpretován. Tímto způsobem jsem také implementoval tabulku v uzlu. Celý diagram je možné exportovat do obrázku za pomocí tlačítka umístěného v horní nástrojové liště. V takovém případě je diagram uložen do formátu PNG. Jelikož se před nedávnem stala tato knihovna komerční, byl jsem nucen použít její starší verzi. I ta ale splňovala všechny požadavky. Její výsledek můžete vidět na obrázku 5.4.

5.3.6 Záložka osoby a materiál

Struktura těchto záložek je velmi podobná, a proto je budu popisovat dohromady. I přes podobnost zobrazení jsou ale interpretovaná data odlišná, a tudíž i v aplikaci umístěna odděleně. V obou případech se na záložce nachází přehledná tabulka s ovládacím panelem umístěným nad ní. V panelu se nachází dvě editační tlačítka pro přidání či odebrání položky v seznamu. Přidává-li se položka, uloží se na konec tabulky s předvyplněnými daty a je automaticky zařazena do projektu. Přiřazení k činnosti probíhá u činnosti samotné. Pro odebrání položky je nutné nejprve vybrat položku k odstranění. V opačném případě položka zůstává a uživatel je vyzván k jejímu vybrání.

Řádky tabulky představují jednotlivé zdroje a sloupce potom jejich atributy. V záložce osoby jsou v tabulce zobrazeny tyto atributy: identifikační číslo osoby, její název, přihlašovací údaje (login a heslo) a čísla přiřazených činností. Všechny údaje jsou uloženy ve třídě *Human* a jejich editace je umožněna přímým přepsáním hodnot v tabulce. Při přidělování lidských zdrojů uvažujeme mnohonásobné použití jednoho zdroje.

V záložce materiál se v tabulce jako atributy nachází: Identifikační číslo, název položky, její popis a identifikátor přidělené činnosti. Přidělení probíhá rovněž u činnosti samotné a na rozdíl od lidských zdrojů se počítá pouze s jedním možným použitím. Editace dat je zvolena formou úpravy buňky přímo v tabulce a každá změna je ihned reflektována ve třídě *Material*.

Právo editace těchto dvou tabulek má pouze projektový manažer. Po přihlášení v jiné roli záložka osob není zobrazena a to z důvodu ochrany osobních údajů osob pracujících na projektu. Záložka materiálu je sice zobrazena, avšak její tabulka i panely nástrojů jsou deaktivované.

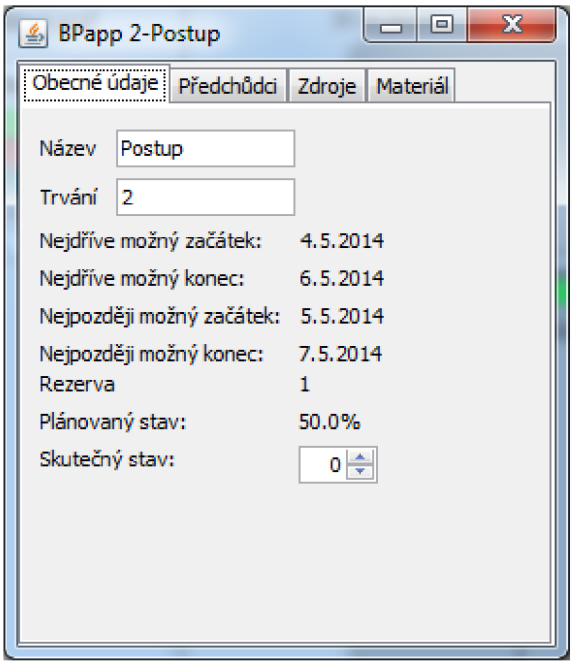
5.3.7 Záložka projekt

Na záložce projekt může uživatel naleznout všechny informace o projektu. Nachází se zde číselné i textové údaje. Zde může uživatel také editovat základní parametry projektu, jako jsou začátek nebo jeho název.

K záložce mají přístup uživatelé v obou rolích, avšak pouze projektový manažer má možnost cokoliv editovat. Ostatním uživatelům slouží pouze jako informativní část aplikace.

5.3.8 Informace o činnostech

Tabulka činností v hlavním okně aplikace je ideálním zobrazením pro přehled o všech činnostech plánu spolu se základními vlastnostmi v jednom čase. Informací a možností činností k zobrazení je ale daleko více. Proto je zde okno s informacemi o činnostech implementované třídou *ActEditForm*, které je jedinečné pro každou z nich. Toto okno je následně rozděleno do 4 - 5 záložek. Tento počet se liší podle zvolené metody zadávání doby odhadu.



Obrázek 5.5: Informace o činnosti - obecné informace

Na první záložce, která je vyobrazena na obrázku 5.5, jsou zobrazeny všechny číselné údaje. Nachází se zde jak zadaná data projektovým manažerem, tak i spočtená data aplikací jako například data nejpozději možného a nejdříve možného konce. Data zadaná ručně je možné ještě zde dodatečně upravit. Důležitým prvkem, který bude zajímat především osoby přidělené k činnosti, je nastavení aktuálního stavu činnosti. Tu uživatel zadává v procentech od hodnoty 0 do maximální hodnoty 100 včetně.

Na dalších třech záložkách lze potom přiřadit projektu předchůdce a lidské či materiální zdroje. Ve všech třech případech jsou data reprezentována tabulkou, do které je možno přidat nebo odebrat položku, pomocí tlačítek umístěných nad ní. Při zvolení možnosti přidání položky, se do tabulky vloží nový řádek s prázdnými hodnotami. Klikne-li uživatel na příslušný řádek, objeví se uživateli rozbalovací seznam, ze kterého si vybere, který prvek chce přidat. Například v záložce předchůdců obsahuje seznam identifikační číslo činnosti společně s jejím názvem. Zůstaneme-li u předchůdců, v druhé kolonce tabulky se nachází taktéž rozbalovací menu, ve kterém uživatel vybírá typ vazby mezi činnostmi.

Poslední, volitelnou záložkou je PERT. Ta je zobrazena pouze tehdy, je-li vybrána tato metoda zadávání. Jsou zde tři okénka pro zadání pesimistického, normálního a optimistického odhadu, pomocí kterého je následně vypočítána doba trvání činnosti.

V závislosti na roli uživatele, lze okno zobrazit třemi způsoby. V roli projektového manažera jsou všechny komponenty funkční a je možno libovolně editovat projekt. Má-li činnost přidělenou osobu, která otevře její informační okno, může sice nahlížet na všechny záložky, ale editovat pouze její aktuální stav a název. Posledním možným způsobem zobrazení je uživatel, který k ní nemá žádná práva. Ten smí pouze nahlížet na všechny stávající informace.

Kapitola 6

Testování

Testovací fáze je nezbytnou součástí při vývoji jakéhokoliv produktu. Jedná se o složitý dlouhodobý proces, při kterém je snaha o co největší přiblížení požadované kvalitě a to především z hlediska spolehlivosti a funkčnosti. V praxi to znamená, že je nutné odhalit co nejvíce chyb oproti očekávanému chování. Jelikož jsou dnes vyráběny stále složitější aplikace, také proces testování nabírá na svém rozsahu. Samozřejmě ale záleží pouze na vedoucím vývoje, do jaké míry se rozhodne testování provádět. U menších projektů nebo dokonce jednotlivců je většinou proveden pouze test základních funkčností. Naopak rozsáhlé projekty nebo komerční aplikace musí splňovat určité standardy. V takových případech fáze testování představuje až polovinu celkového času vývoje.

Testování mé aplikace jsem rozdělil do dvou kategorií:

- testování základní funkcionality aplikace
- testování akceptovatelnosti aplikace

Testy spadající do první kategorie byly prováděny s cílem odhalit chyby, které by způsobily nefunkčnost implementovaných funkcí v případě, kdy uživatel postupuje zcela správně. Tyto testy jsem prováděl v průběhu celého vývoje při dokončení jakéhokoliv celku. Test probíhal na základě znalosti vnitřní logiky programu a odhalené chyby byly ihned opraveny.

Při tomto testování byly zjištěny především chyby, týkající se aplikační funkcionality. Objevilo se pár chyb při práci s databází, při ukládání dat do souboru xml, ale i chyby způsobené mou nepozorností při implementování složitějších algoritmů.

Velkou součástí bylo také testování grafických prvků aplikace. Například po zásahu do knihovních funkcí bylo odhaleno zobrazení průběhu činnosti v grafu zcela mimo danou činnost. Co se týká grafu PERT, byly zde jednotlivé uzly vykreslovány jeden přes druhý, což bylo způsobeno špatně implementovaným algoritmem pro jejich rozložení.

Druhá kategorie testů byla určená k odhalení chyb vzniklých kvůli krajním situacím, které mohou během používání nastat. Jedná se tedy o jakousi akceptovatelnost systému na kritické stavy a nevyhovující vstupy. Vzhledem k tomu, že při tomto typu testování není nutné znát vnitřní implementaci, rozhodl jsem se použít i potencionální uživatele. Díky tomu byl odhalen větší počet chyb.

Testy byly zaměřené na ošetření uživatelských vstupů ale i na nedostatky vzniklé při vývoji. I přes velké množství zjištěných a opravených chyb nelze tvrdit, že je aplikace zcela bez chyb. Testy lze dokázat pouze to, že aplikace chyby obsahuje, avšak nejde potvrdit, že je nemá.

Kapitola 7

Další rozvoj systému

Poté co je každý produkt uveden do provozu, přicházejí nápady a impulsy na další rozvoj nebo vylepšení systému. Tyto požadavky je potřeba zanalyzovat a případně implementovat v produktu. Již v průběhu testování vznikly určité myšlenky jak tuto aplikaci rozšířit. Vzniklé myšlenky a pár dalších návrhů vám vylíčím v této kapitole.

Prvním rozšířením by mohla být editace plánu prostřednictvím grafů. Například v Ganttově diagramu by měl mít uživatel možnost editovat dobu trvání činnosti, změnit její datum přetažením ji na jiné místo či tvorbu nových vazeb mezi činnostmi podobně, jako nám to umožňují některé komerční aplikace. Také v PERT diagramu by uživatel mohl mít možnost přidávat nové uzly/činnosti a ty potom pomocí grafického rozhraní zakomponovat do celého časového plánu.

Dalším možným rozšířením je funkce zasílání zpráv. Každý uživatel by mohl zanechat zprávu týkající se projektu jinému uživateli. Zprávy lze také generovat automaticky, jako upozorňující hlášení při důležitých událostech plánu. Tato funkce by ovšem vyžadovala instalaci serveru pro posílané zprávy.

V souvislosti s předcházejícím návrhem by bylo nezbytné také rozšířit aplikaci o složitější správu konfliktů. V současnosti se při práci jednoho člena týmu celý projekt zablokuje. V ideálním případě, mohou v jedné chvíli na jednom projektu pracovat více lidí s tím, že by případně konflikty byly automaticky zpracovány.

V neposlední řadě se jedná o ukládání stavu plánu při jakékoliv editaci. Pokud by se tedy uživatel chtěl vrátit ke stavu před pěti provedenými kroky, měl by to provést pomocí kliknutí na příslušné tlačítko zpět. To by šlo implementovat buď pomocí ukládání celkového stavu plánu v každém kroku nebo ukládáním provedených činností a následně jejich zpětném vykonání.

Projektovým manažerům by také jistě přišlo vhod rozšířit množství ukládaných prvků. V reálných projektech se dnes vyskytuje spousta dalších faktorů, které projekt zcela zásadně ovlivní. Právě tyto a spoustu dalších by bylo vhodné do aplikace zakomponovat. Uživatelé aplikace by tak měli ještě komplexnější přehled o reálném stavu projektu.

Podobných rozšíření by se jistě našla celá řada. Ale v případě, pokračování ve vývoji aplikace, bych se zaměřil nejdříve na ty výše uvedené. Mohlo by se také stát, že v průběhu jejich implementace dojde k závěru, že rozšíření nejsou proveditelná a část aplikace by se musela zcela zahodit. Při dalším vývoji lze také předpokládat, že aplikace bude používána značným množství lidí, kteří by měli mít přehled o aktualitách spojených s aplikací. Bylo by tedy vhodné vytvořit internetovou stránku, kde by šla stáhnout aktuální verze aplikace a kde by byl prostor pro přání či vzkazy uživatelů. Případně by se tam mohlo vyskytovat diskuzní fórum, kde by si uživatelé mohli vzájemně radit.

Kapitola 8

Závěr

V práci jsem se věnoval aplikacím určeným k podpoře časového plánování. Byla představena problematika projektového řízení a techniky, které se používají při provádění časové analýzy projektu. Pomocí dat vycházejících z tohoto kroku jsem vybral kritéria, která by měl software v této oblasti splňovat. Dle daných kritérií jsem analyzoval současné aplikace na trhu a na tomto základě jsem vytvořil návrh implementace vlastní aplikace. Výsledkem je více uživatelská aplikace, která umožňuje tvorbu časového plánu a následnou správu projektu skrze jednoduché rozhraní. S plánem je možné pracovat jak bez připojení k internetu, tak i vzdáleně prostřednictvím databázového serveru. V rámci rozšíření je možné projekt kdykoli exportovat do aplikace MS Project.

K zajímavému zjištění jsem dospěl především v kapitole analýzy dostupných nástrojů na trhu. Z té vyplynulo, že aplikace, která by vyhovovala zvoleným požadavkům, na trhu chybí. Rozdělil jsem podle nich produkty do dvou kategorií, z nichž se v jedné nacházely velmi komplexní, až složité nástroje s nemalou cenou a v druhé jednoduché nástroje s volnou licencí, které postrádaly některou z požadovaných vlastností. Vzniklá aplikace obsahuje nejlepší vlastnosti z obou kategorií.

Díky této bakalářské práci jsem získal spoustu znalostí z projektového řízení které, i přes svou důležitost, nemá odpovídající místo u všech projektových manažerů. Dále jsem zjistil, jaké by mělo mít časové plánování místo v životě každého člověka, jelikož život sám je jeden velký projekt. Naopak při samotné implementaci jsem získal lepší znalosti práce s jazykem Java, a jaké úskalí obnáší používání neoficiálních knihoven od svobodných tvůrců, které jsem nikdy před tím při programování nepotřeboval.

Musím konstatovat, že samotný vývoj aplikace neprobíhal zcela ideálně, a to z důvodu mé vlastní nepozornosti. Při psaní kódu se v programu objevilo značné množství chyb, které se dřív nebo později projevily při testování a byly odladěny. Ve výsledku ale aplikace uspěla ve všech testech a neobjevily se žádné kritické chyby.

Aplikace by mohla být rozšířena o spoustu dalších prvků, které můžeme najít u jiných komerčních produktů. Vzhledem k požadavkům je ale aplikace zcela kompletní. Některé návrhy na rozšíření aplikace, které vznikly při fázi testování, jsou uvedeny v kapitole 7. Lze předpokládat i vznik dalších požadavků při rozšíření aplikace mezi uživatele.

Aplikace, která vznikla v rámci této práce, vyhovuje všem kritériím stanoveným při analýze a také všem požadavkům vyplývajících ze zadání a je možné ji použít při tvorbě a projektovém řízení v nejrůznějších odvětvích.

Literatura

[1] Ciano, B.: Using PERT for estimating tasks [online]. `http://brunociano.blogspot.cz/2007/08/using-pert-for-estimating-tasks.html`, [cit. 2014-3-8].

[2] Concept Draw: Project Chart [online]. `http://www.conceptdraw.com/samples/project-chart`, 2009-11-24 [cit. 2014-4-6].

[3] Dibyajyoti, K.: OAF MVC Architecture [online]. `http://myoraclecafe.com/2012/09/12/oaf-mvc-architecture/`, 2012-11-12 [cit. 2014-3-5].

[4] Doležal, J.; Machál, P.; Lacko, B.; aj.: *Projektový management podle IPMA, 2., aktualizované a doplněné vydání.* Praha: Grada Publishing, 2012, ISBN 978-80-247-4275-5.

[5] Jirchář, V.: Metoda CPM/PERT: Podkladový materiál pro předmět Základy a teorie grafů [online]. `http://kmlinux.fjfi.cvut.cz/ jahodfra/ztgb/prezentace/cpm_pert.pdf`, 2009-11-24 [cit. 2014-3-16].

[6] Korecký, M.; Trkovský, V.: *Management rizik projektů se zaměřením na projekty v průmyslových podnicích.* Praha: Grada Publishing, 2011, ISBN 978-80-247-3221-3.

[7] Project Management Institute: *A Guide to the Project Management Body of Knowledge, Third Edition.* Pennsylvania: Project Management Institute, 2004, ISBN 978-19-306-9945-8.

[8] Ralph L. Kliem, I. S.: *Project Management Methodology: A Practical Guide for the next Millennium.* New York: Marcel Dekker, 1997, ISBN 0-8247-0088-0.

[9] Robert, E.: Java SE Application Design With MVC [online]. `http://www.oracle.com/technetwork/articles/javase/index-142890.html`, 2008-03-01 [cit. 2014-3-5].

[10] Rosenau, J., Milton D.: *Řízení projektů.* Praha: Computer Press, 2000, ISBN 80-7226-218-1.

[11] Schwalbe, K.: *Řízení projektů v IT.* Praha: Computer Press, 2007, ISBN 978-80-251-1526-8.

[12] Tahelyani, S.: MVC Architecture in Android [online]. `http://gkonandroid.blogspot.cz/2013/11/mvc-architecture-in-android.html`, 2013-11-08 [cit. 2014-3-9].

[13] Taylor, J.: *Začínáme řídit projekty*. Praha: Computer Press, 2007, ISBN 978-80-251-1759-0.

[14] Šafář, P.: Hra o kvalitu [online]. `https://www.ibacz.eu/blog/-/blogs/hra-o-kvalitu`, 2012-07-12 [cit. 2014-2-21].

[15] Šubrt, T.: ZIP Projektové řízení: Podkladový materiál pro kurz ZIP [online]. `http://etext.czu.cz/php/skripta/skriptum.php?titul_key=77`.