VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

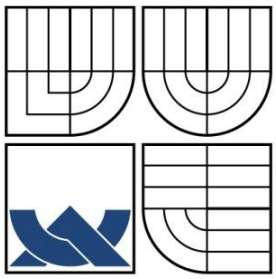
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

PLÁNOVÁNÍ A ŘÍZENÍ TÝMOVÝCH PROJEKTŮ

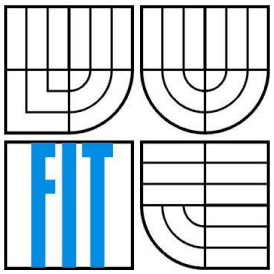
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. TOMÁŠ JELÍNEK
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

PLÁNOVÁNÍ A ŘÍZENÍ TÝMOVÝCH PROJEKTŮ

TEAM PROJECT PLANNING AND MANAGEMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. TOMÁŠ JELÍNEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. FILIP ORSÁG, Ph.D.

BRNO 2007

Abstrakt

Cílem této diplomové práce je implementovat systém pro plánování a řízení týmových projektů s webovým rozhraním. V teoretické části práce jsou vysvětleny základní pojmy, porovnány některé existující nástroje řízení a plánování projektů a také jsou zmíněny existující metodologie, podle kterých se projekty mohou řídit. Praktická část zahrnuje popis návrhu a implementace vytvořené aplikace pro řízení projektů a jeho porovnání s existujícími aplikacemi.

Klíčová slova

Řízení projektů, životní cyklus, Ganttův diagram, CPM, kritická cesta, PERT, UML, management, PHP, PEAR, SQL, Apache

Abstract

The aim of this thesis is implementation of information system designed for plan and project management with web user interface. From a theoretical part we first describe necessary terminology, compare existing tools used for plan and project management. Some existing methodology for project management is also mentioned. From pragmatical point of view we describe design and implementation of created application for project management. Other concurrent applications for project management are also mentioned.

Keywords

Project management, life cycle, Gantt chart, CPM, critical path, PERT, UML, management, PHP, PEAR, SQL, Apache

Citace

Jelínek Tomáš: Plánování a řízení týmových projektů. Brno, 2007, diplomová práce, FIT VUT v Brně.

Plánování a řízení týmových projektů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Filipa Orsága, Ph.D.
Další informace mi poskytla RNDr. Jitka Kreslíková, CSc.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Tomáš Jelínek
22.5.2007

Poděkování

Děkuji vedoucímu diplomové práce Ing. Filipu Orságovi, Ph.D. z Ústavu inteligentních systémů a RNDr. Jitce Kreslíkové, CSc. za metodické a cíleně orientované vedení při plnění úkolů realizovaných v rámci diplomové práce.

Obsah

Obsah 1
1 Úvod 3
2 Základní pojmy 4
2.1 Projekt 4
2.2 Proces 5
2.3 Kvalita 6
2.4 Management 8
2.5 Manažer (Manager) 9
2.6 Projektové řízení 9
2.7 Trojimperativ 10
2.8 Životní cyklus projektu 12
3 Nástroje a techniky využívané při plánování 14
3.1 Struktura prací - WBS (Work Breakdown Structure) 14
3.2 Metoda kritické cesty – CPM (Critical Path Method) 15
3.2.1 Příklad CPM 16
3.3 Metoda vyhodnocování a kontroly programu - PERT (Program Evaluation and Review Technique 19
3.4 Porovnání PERT a CPM 20
3.5 Kritický řetězec 20
3.6 Ganttův Diagram 20
3.7 MPM síť (Metra Potential Method) 21
4 Exitující aplikace pro plánování a řízení projektů 22
4.1 Iterity Project 22
4.1.1 Co umožňuje Iterity Project? 22
4.1.2 Klientské prostředí aplikace 23
4.2 Imendio Planner 23
4.3 Microsoft Office Project 24
4.4 Achievo 25
4.5 AutoCont Enterprise Project Management 27
4.5.1 Co AutoCont Enterprise Project Management umožňuje 27
4.6 Shrnutí 28
5 Návrh a Implementace 29
5.1 Návrh jednoduchého systému pro správu projektů 29
5.1.1 Use case diagram navrhovaného systému 29

5.1.2 Diagram tříd ... 30
5.2 Implementace ... 30
5.2.1 Implementační jazyk ... 30
5.2.2 Databázový server ... 32
5.2.3 PEAR ... 33
5.2.4 Webový server ... 36
5.3 Instalace ... 38
5.4 Struktura aplikace ... 39
5.5 Bezpečnost ... 40
5.5.1 Analýza možných útoků ... 40
5.5.2 Zabezpečení aplikace pomocí HTTP ... 42
5.5.3 Zabezpečení pomocí HTTPS ... 43
5.6 Dosažené výsledky ... 43
6 Závěr ... 45
6.1 Možnosti dalšího vývoje ... 45
6.2 Shrnutí ... 46
Literatura ... 47
Seznam příloh ... 49
Příloha 1 Obsah CD ... 50
Příloha 2 Instalace PEAR ... 51

1 Úvod

Počátky projektového řízení sahají do pozdních let 19. století. Do moderní podoby, ve které jej známe dnes, se vyvinulo na začátku 60. let minulého století. Podniky si tehdy začaly uvědomovat, že je nutné více se zaměřit na kvalitní řízení projektů. Také si začaly uvědomovat přínosy, jenž vyplývaly z kvalitní organizace práce projektů. S rostoucím počtem zaměstnanců a specializací jednotlivých profesí bylo potřeba koordinovat práci mezi odděleními i jednotlivými profesemi. Projektové řízení se tehdy uplatňovalo i v nově vznikajících oborech lidské činnosti, jako příklad lze uvést řízení vesmírného programu USA.

V současné době se stále více jednorázových a vývojových prací provádí formou projektu. Cíle projektů mohou být různé (vývoj nového software, reorganizace společnosti, reklamní kampaň, atd.). Nelze říct, že projekty vznikají jen při některých činnostech. Naopak, projekt může vzniknout prakticky v jakékoliv firmě nebo instituci, která se zabývá nějakou činností, to znamená, že se nemusí jednat jen o projekty z oblasti informačních technologií. Běžným jevem je, že firma nebo pracovní skupina pracuje i na několika projektech zároveň.

Při řízení projektů dříve převládala zkušenost, intuice a organizační schopnost vedoucího projektu. V 21. století však tvrdá konkurence nutí společnosti hledat stále nové oblasti, kde mohou získat konkurenční výhodu. Proto i řízení projektů je v současnosti významně podporováno informačními systémy, které dokážou podpořit flexibilitu, rychlost reakcí na změnu na trhu a organizaci projektu.

Oblast projektového řízení se v posledních letech dynamicky vyvíjí. Snahou je přijít na nové metody, které by ještě více urychlily a zefektivnily plánování projektu. Ani dnes, kdy můžeme díky novým technologiím a matematickým postupům přesně plánovat, není jednoduché vytvořit kvalitní plán projektu.

Projekty dnes najdeme opravdu všude. Většina marketingových kampaní, obchodních aktivit i strategických plánů nemá podobu jednorázových činností, ale naopak organizačně složitých a návazných procesů. Právě tyto složité procesy jsou velmi náročné na řízení, koordinaci a organizační zajištění. Soubor dílčích činností, které je třeba skloubit, aby celý úkol byl úspěšně zvládnut, tvoří společně komplexní projekt. Jádrem řízení projektu je rozhodování o tom, kdy má určitá činnost proběhnout, jak dlouhou dobu potřebujeme k jejímu zvládnutí, kdo bude za celou akci odpovědný a kteří pracovníci se budou na její realizaci podílet.

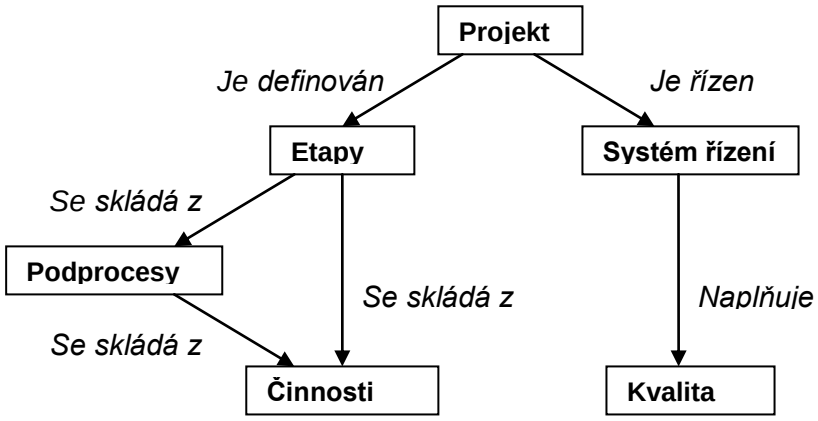
Tyto projekty jsou pro firmy většinou velmi důležité, proto mnohdy investují nemalé finanční prostředky do speciálního softwarového vybavení, určeného k řízení projektů. Projektový management je velice rozsáhlý obor a rozhodně se nejedná o triviální záležitost. Tento pojem zahrnuje mnohem více než jen používání speciálního software. Ten slouží pouze k usnadnění nelehké a velmi zodpovědné práce při řízení projektového týmu.

Na trhu existuje mnoho velkých, složitých a drahých softwarových nástrojů pro podporu projektového řízení. Některé jsou obecně zaměřené, některé se specializují na určitou speciální oblast. Výběr konkrétní aplikace je individuální záležitost. Před započetím projektu je vhodné se zamyslet nad tím, jaký software použít pro daný projekt. Změna software v průběhu projektu je totiž velmi problematická a nákladná.

Tato práce se věnuje především definici základních pojmů a porovnání nejznámějšího software. V poslední části textu se budu věnovat návrhu konkrétní aplikace, kterou se budu více zabývat ve své diplomové práci.

2 Základní pojmy

V této kapitole budou vysvětleny základní pojmy týkající se projektového řízení. Vztah mezi některými z nich je na **obr. 2.1**.



Obr. 2.1 Vztahy mezi některými základními pojmy

2.1 Projekt

O projektech dnes slyšíme velice často. Toto slovo však znamená pro různé profesní skupiny různé rozdílné věci. Nás zajímá slovo projekt a řízení projektů v souladu s jejich významem v anglicky a německy mluvících zemích. Nejprve vymezíme, co projekt skutečně je a co je podstatou jeho řízení.

V normě ISO 10006 [1] je projekt definován jako jedinečný proces sestávající z řady koordinovaných a řízených činností s daty zahájení a ukončení, prováděný pro dosažení jedinečného cíle, který vyhovuje specifickým požadavkům, včetně omezení daných časem, náklady a zdroji.

V literatuře se objevují i jiné definice projektu.

- Jedinečná soustava činností směřujících k předem stanovenému cíli, který má určitý začátek i konec. Vyžaduje spolupráci různých profesí, váže či spotřebovává jejich kapacity a využívá je pro vytvoření výstupu. Projekt představuje tři roviny, ve kterých se pohybujeme. Projekt je trojdimenzionální - tzv. trojimperativ. (**obr. 2.7**) [15]
- Projekt je řízený proces aplikace úkolů a zdrojů s definovaným cílem v určeném časovém rámci. [16]
- Dočasně vyvinuté úsilí, vynaložené na vytvoření jedinečného produktu nebo služby. [16]
- Síť činností mající formální začátek a konec, přidělené zdroje a směřující k vytvoření určitého produktu. Často má také stanoven rozpočet, v rámci kterého musí být stanovených cílů dosaženo. S vytvořením tohoto produktu je vždy spojeno určité riziko. [16]
- Projekt má 4 typické znaky, které, pokud se vyskytují společně, odlišují řízení projektu od jiných manažerských činností. Projekty mají trojrozměrný cíl, jsou jedinečné, zahrnují zdroje a realizují se v rámci organizace. [2]

Všechny tyto definice můžeme shrnout třeba takto: Projekt představuje množinou koordinovaných činností, která je jedinečná. Realizovat ho může buď jednotlivec nebo organizace. Cílem je dosáhnout daného cíle v určeném čase (mohou to být dny, týdny, měsíce, ale i roky) a pokud

možno nepřekročit stanovený rozpočet projektu. Předpokládá se také dosažení určité kvality výsledného produktu. Rozdílem mezi klasickou prací a projektem je v tom, že výsledkem by měl být jedinečný produkt nebo služba.

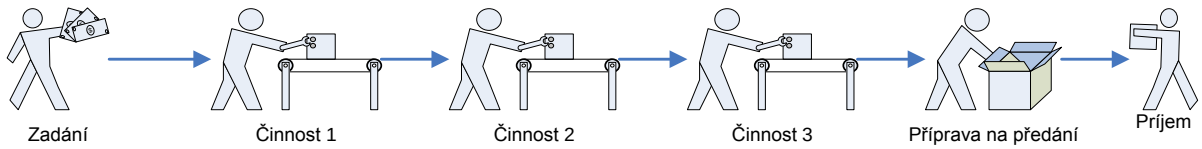
Projekt by měl podle [3] splňovat tato kritéria:

- Je jedinečným dílem, které se liší v nějakém ohledu od podobných projektů
- Činnosti se týkají více než dvou osob
- Doba trvání činnosti zabere více než dva týdny
- Vyžadují se nové postupy nebo nové technologie
- Rozpočet je těsný
- Některé úkoly jsou závislé na dokončení jiných úkolů
- Vývoj produktu se skládá z několika fází, které se musí koordinovat
- Je požadováno měření kvality

2.2 Proces

Proces je uspořádaná množina činností, která na základě jednoho, či více vstupů, tvoří opakovatelným způsobem požadované výstupy (na **obr. 2.3** znázorněno graficky).

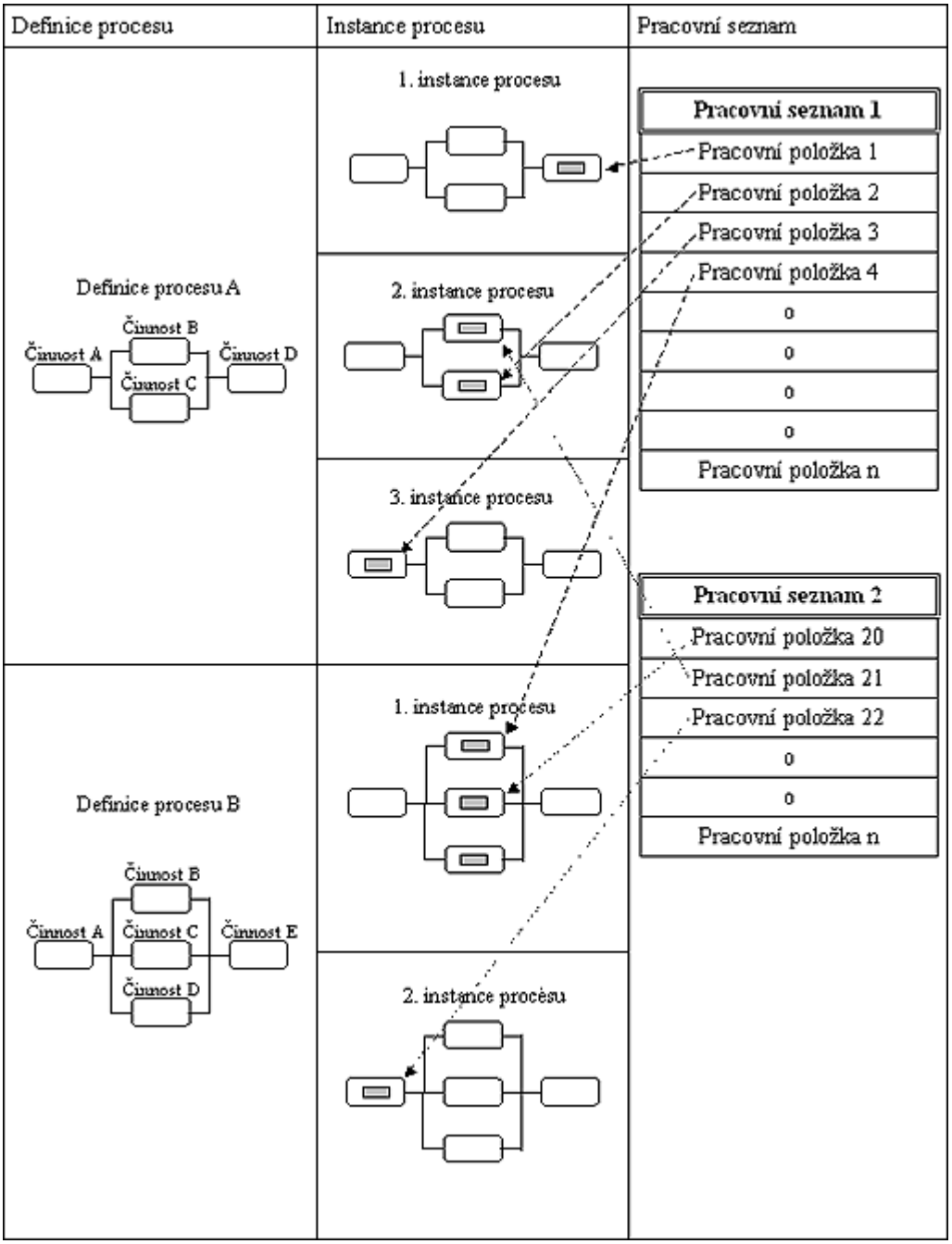
Za proces je vždy považována určitá posloupnost činností, které postupně vykonávají různí pracovníci napříč organizační strukturou organizace. Proces má jasně vymezený začátek a konec, definované vstupy a výstupy a hlavním cílem je uspokojit potřeby zákazníka tohoto procesu (**obr. 2.2**). Takový proces musí být opakovatelný a může být popsán na různých úrovních abstrakce.



Obr. 2.2 Postupné procesy vzniku výrobku procesy – výsledek předchozího procesu je vstupem následujícího

Základní vlastnosti procesu podle [30]:

1. skládá se z uspořádaných činností
2. má jednoznačně definovaný počátek a konec
3. transformuje vstupy na výstupy
4. využívá zdroje
5. je opakovatelný



Obr. 2.3 Vztah mezi procesy a pracovními seznamy (činnosti)(podle [10])

2.3 Kvalita

Jakost je možno definovat jako „stupeň splnění požadavků souborem inherentních znaků". (Norma ČSN ISO 9000:2000 [31]). Vývoj názorů na to, co je to kvalita se postupně vyvíjel. Nejprve se kontrolovalo jen to, zda je výrobek bezchybný. Později se začalo srovnávat s konkurenčními výrobky. Kvalitu výrobku je možné zvyšovat, ale s kvalitou roste i jeho cena a může se tedy snadno stát, že se stane nekonkurenceschopným.

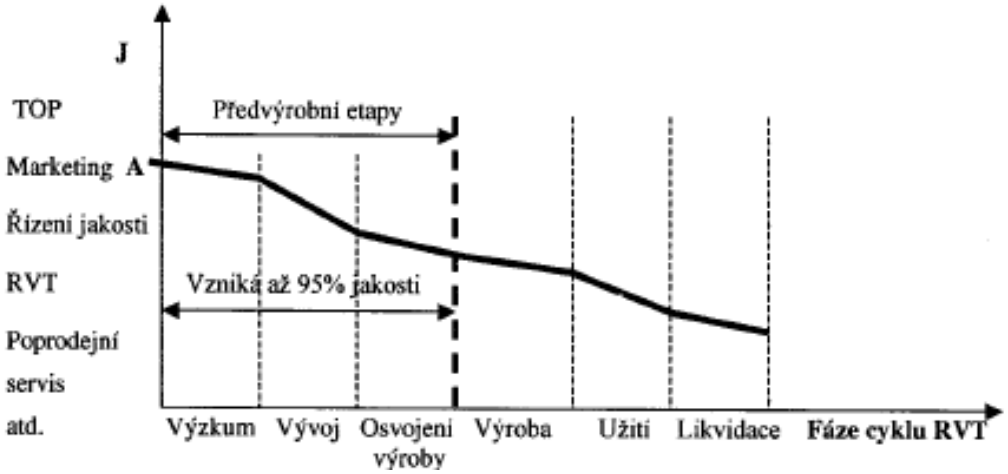
Kvalita nebo také jakost, je součástí dimenze provedení, v současné době je kvalita chápána obecně jako uspokojení potřeb zákazníka. Kvalita se dotýká nejen vlastních produktů, ale i procesů

produkt realizujících, zákazník tedy posuzuje jak kvalitu výsledných produktů, tak i souvisejících procesů nebo jejich částí.

Existuje mnoho definic kvality, zde je několik významných [31]:

1) Kvalita je schopnost plnit požadavky uživatele a veřejného zájmu prostřednictvím souhrnu vlastností, vyjadřujících způsobilost výrobku plnit funkce, pro něž je určen. (Crosby)
2) Kvalita výrobku nebo služby je stupeň způsobilosti, aby vyhovovala účelu jejího využití. (Juran)
3) Kvalita je souhrn vlastností výrobku nebo služby, rozhodujících pro plnění její funkce za předepsaných provozních podmínek a při nejnižších vynaložených nákladech, (americká norma)
4) Kvalita je minimem ztrát, které výrobek od okamžiku své expedice společnosti způsobí. (Taguchi)
5) Kvalita je optimum ve vztahu k požadavkům uživatele a vynaloženým nákladům na výrobek nebo službu. Jakost se rozkládá na dvě složky, tj. na soubor vlastností výrobku stanovených předvýrobou (projekcí, konstrukcí), které jsou rozhodné pro stupeň, v jakém výrobek splňuje přání zákazníka a na schopnost výroby realizovat záměry projekce (Feigenbaum).
6) Kvalita je míra výsledku, která může být kategorizována v různých třídách (Veber s.18.)
7) Kvalita nejsou náklady, které výrobce vkládá do svého výrobku, nýbrž užitek, který z něho získá kupující (Walter Masing).

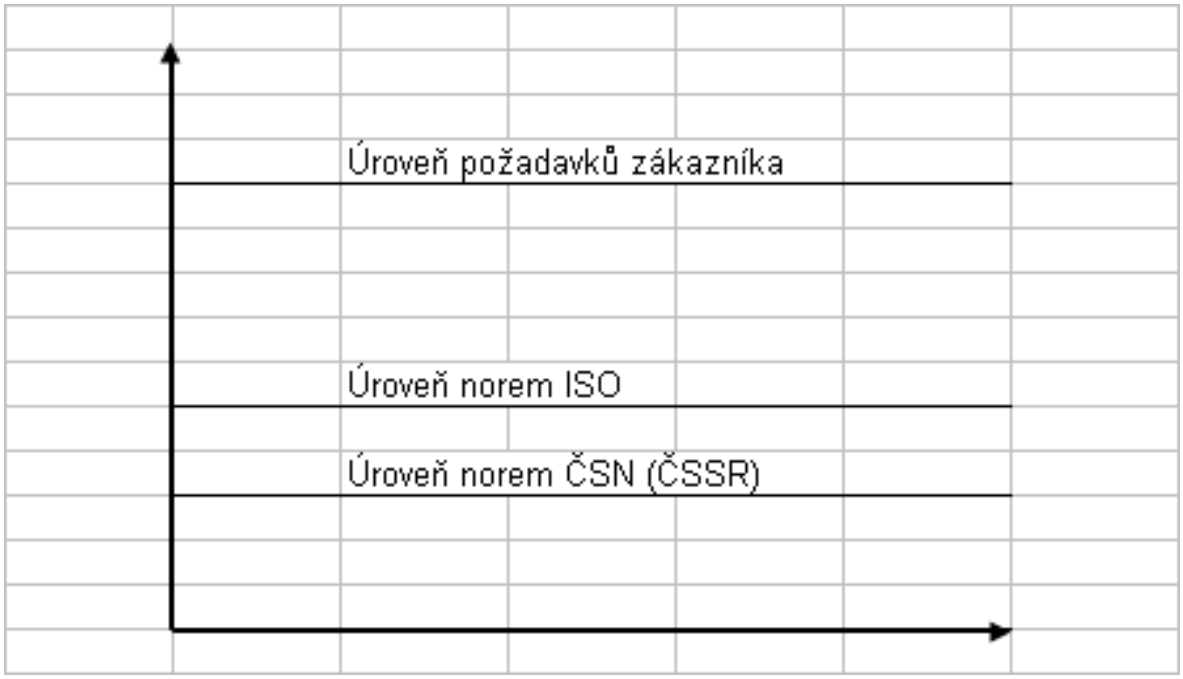
V současné době se rozhodující charakteristikou pojetí kvality stává její rozšíření o nové faktory. Je nutné přistupovat ke kvalitě komplexně, protože již nejde jen o jakost výrobku, ale i jakost výroby a vlastně o jakost celého cyklu RVT (výzkum - vývoj - osvojení výroby - výroba - užití - - likvidace). Z výzkumů plyne, že 95% kvality (jakosti) vzniká ve vývoji a výzkumu (tzn. v projektové části RVT – **obr. 2.4**).



Obr. 2.4: Průběh úrovně kvality v cyklu RVT (převzato z [14])

Dynamicky se měnící požadavky uživatelů musí být zdrojem inspirace pro vývojová oddělení firem. Úkolem norem ČSN a ISO je zajistit požadovanou úroveň řízení jakosti dané firmy. Můžeme říct, že splnění těchto norem představuje podmínku nutnou, nikoliv však postačující pro úspěch

výrobku na trhu (**obr. 2.5**). Úspěšnější je ten výrobek, který je přesnější, spolehlivější, má lépe zajištěn servis, vyšší technické parametry, atd. Skutečný konkurenční boj se pohybuje vysoko nad úrovní norem, protože požadavky zákazníků jsou mnohem vyšší, než je stanovují normy. Podle [14] je kvalita výrobku v tržním prostředí vlastnost výrobku, determinovaná stupněm splnění potřeb zákazníka.



Obr. 2.5 Splnění norem je podmínkou nutnou, nikoliv postačující pro kvalitní výrobek (ČSSR je uvedeno pro srovnání se současnými ISO normami) [14]

2.4 Management

Slovo management je převzato z angličtiny a je odvozeno od slůvka manager, čili řídit, zvládnout, dokázat.

Management můžeme definovat jako proces tvorby a udržování prostředí, ve kterém jednotlivci pracují společně ve skupinách a účinně dosahují vybraných cílů. Stejně tak však můžeme konstatovat, že management je vlastně procesem plánování, organizování, vedení a kontroly organizačních činností, zaměřených na dosažení organizačních cílů. Definic je však mnoho a značně se liší, mnohé jsou dokonce protichůdné.

Ve většině definic managementu je kladen důraz na:

1. **Vedení lidí** - podle Americké společnosti pro management, management znamená vykonávání úkolů prostřednictvím práce jiných. To bývá dnes interpretováno jako: „Management je umění dosahovat cíle organizace prací druhých."
2. **Specifické funkce vykonávané vedoucími pracovníky** - zdůrazněny jsou specifické funkce prováděné vedoucími pracovníky. Podle K. Millera je „…management soubor typických činností, které manažer vykonává, jako je rozhodování, organizování, plánování, kontrolování, vedení lidí, koordinace, motivování, atd."

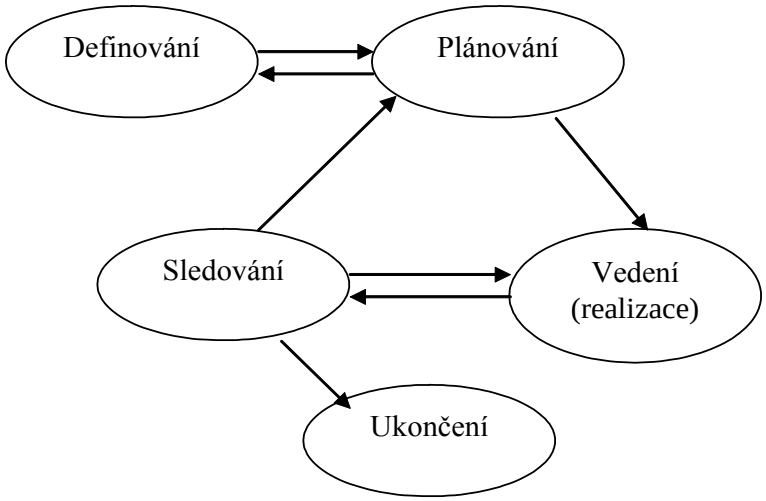
3. **Předmět učení a jeho účel (Knowledge management)** – příkladem může být pojetí např. J. A. Pearcea: „Management je proces optimalizace využití lidských, materiálních a finančních zdrojů k dosažení organizačních cílů."

2.5 Manažer (Manager)

Manažer je profese člověka, který je zodpovědný za dosahování cílů, které byly stanoveny jeho organizační jednotce. Využívá při tom kolektiv spolupracovníků. Každý manažer musí mít dostatečné odborné i osobnostní předpoklady. V rámci organizace mohou existovat různé manažerské úrovně.

2.6 Projektové řízení

Projektové řízení je uplatnění znalostí, dovedností, nástrojů a technik v projektových činnostech s cílem splnit nebo překročit potřeby zájmových skupin a jejich očekávání od projektu. Snaží se dosáhnout cílů a optimalizuje využívání zdrojů (čas, peníze, lidé, materiály, energie, prostor, …).



Obr. 2.6 Základní životní cyklus projektu

Řízení projektů se skládá z pěti odlišných manažerských činností (základní životní cyklus projektu):

1. **Definování projektových cílů** – nadefinované cíle by měly být měřitelné a dosažitelné.
2. **Plánování** – jak tým dosáhne splnění podmínek trojimperativu (viz. dále), tzn. Specifikuje provedení a kvalitu, časový plán a finanční rozpočet.
3. **Vedení** – uplatnění managementu řízení lidských zdrojů. Snaží se členy týmu vést k tomu, aby svoji práci vykonávali efektivně a včas, tj. bez zbytečných prodlev.
4. **Sledování (monitoring)** – kontrola stavu a postupu prací na projektu. Je nutné odhalit včas odchylky od plánu a rychle přistoupit k jejich korekci. To je často provedeno korekcí plánu a může to vést až ke změně potřeby jednotlivých zdrojů (tzv. změnové řízení).
5. **Ukončení** – ověření, že hotový úkol odpovídá aktuální definici toho, co se mělo udělat a uzavření všech činností na projektu. Po ukončení projektu se zpravidla provádí kontrola výsledků. A dosažený cíl se porovnává se zadanými požadavky. Často se provádějí různé

druhy testů, případně i zkušební provoz. Celkové ukončení všech činností, které s projektem přímo nebo nepřímo souvisejí, je většinou doprovázeno předávacím protokolem pro zákazníka. Jeho součástí mohou být další náležitosti, které jsou součástí zadání (např.: způsob odstranění závad a nedodělků, dokumenty stanovující dobu a podmínky záruky, technická dokumentace, manuály, zaškolení pracovníků, atd.).

První dva kroky je možné sloučit, případně je provádět v opačném pořadí. Nejčastější však je pořadí, které je zde uvedeno.

Při řízení projektu a činností, které s ním souvisejí, se snažíme neztratit kontrolu nad těmito proměnnými:

- **Čas** – pro dokončení projektu je potřebné určité množství času, které bývá rozloženo pro analytické účely na čas požadovaný pro splnění částí projektu a čas potřebný pro dokončení každé činnosti.
- **Cena** – cena je přímo úměrná k času, který je požadován. Náklady se vypočtou jako čas násobený sazbou na členy týmu, kteří jsou do projektu zapojeni. Co možná nejpřesnější odhad času je velice důležitý pro výpočet ceny, která je pro zákazníka velice důležitá.
- **Rozsah** – blíže specifikuje podobu konečného výsledku. Tzn. celková definice toho, co se od projektu očekává, aby zvládal a určitý popis toho, jak by měl konečný výsledek vypadat.
- **Kvalita** - množství času vloženého do jednotlivých úkolů určuje celkovou kvalitu projektu. Každá činnost vyžaduje určité množství času k dokončení. Při nedostatku času může být ohrožena kvalita výsledku. V průběhu rozsáhlého projektu může mít kvalita velký dopad na čas a náklady a naopak.
- **Riziko** - možná pravděpodobnost, že nastane chyba. Většině rizik nebo potenciálním chybám se dá předcházet, když se přidělí dostatek času a zdrojů.

Tyto proměnné mohou být zadávány zákazníkem. Proměnné, které zákazník nezadá, se obvykle dopočítávají z těch, které jsou zadány. Obvykle se těchto 5 proměnných redukuje na 3 – čas, náklady (cena), provedení (rozsah a kvalita). Riziko se vždy dopočítává. Redukovaná trojice se nazývá **trojimperativ.**

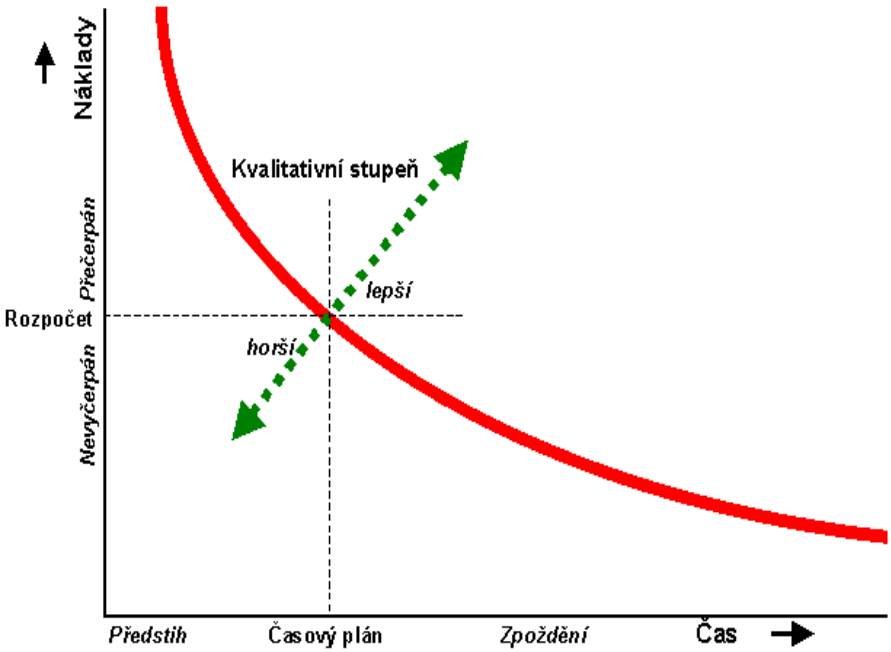
K projektovému řízení je dobré si říct i něco o projektovém týmu. Projektový tým je skupina lidí pracujících na určitém projektu. Mnohdy je složen z lidí různých profesí, kteří se snaží dosáhnout nějakého společného cíle.

2.7 Trojimperativ

Tento pojem se vyskytuje v odborných textech o projektovém řízení velice často. Trojimperativ definuje projekt třemi proměnnými, popsanými v předchozím textu (čas, cena, rozsah a kvalita). Kvalitu a rozsah můžeme shrnout do pojmu specifikace provedení nebo jen provedení, někde je také uváděn jen pojem kvalita. A právě na těchto třech dimenzích závisí nejvíce úspěch projektu. Úspěšné řízení projektů znamená dosáhnout požadované parametry provedení v daném termínu, nebo ideálně ještě před ním, a v rámci rozpočtových nákladů. Náklady se obvykle počítají v korunách, eurech, dolarech, či kterékoli jiné měně, ale mohou být také někdy uvedeny jako počet celkem

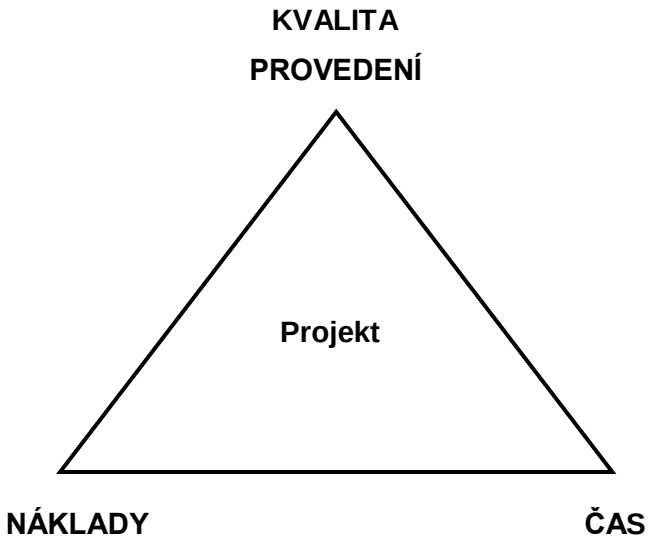
odpracovaných hodin (to je běžné hlavně v projektech z oblasti informačních technologií). Je třeba dosáhnout současně tří nezávislých cílů, tzn. dimenzí trojimperativu:

- věcná, kvalitativní (**Co** se má udělat)
- časová (**Kdy** se to má udělat)
- nákladová (**Za kolik** se to má udělat)



Obr. 2.7 Důsledky trojimperativu

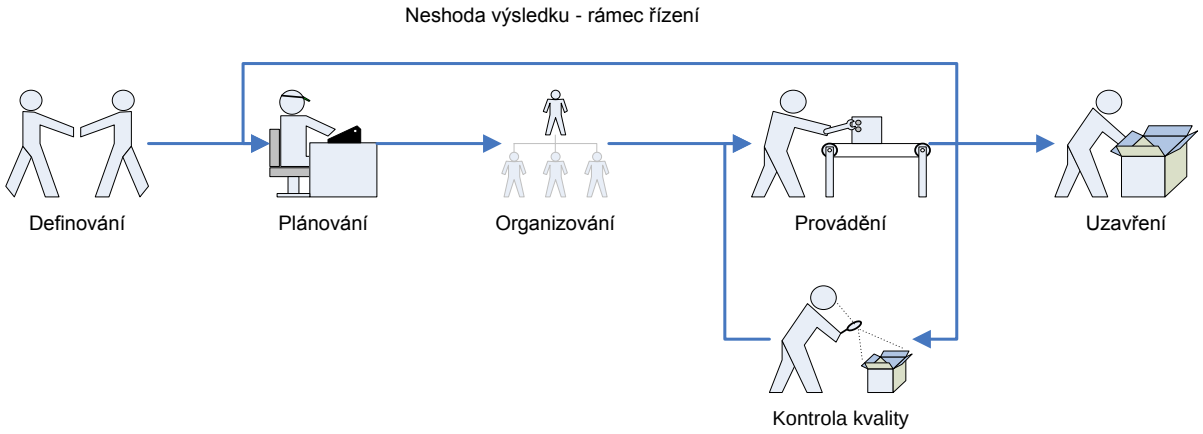
Dimenze lze také znázornit graficky – **obr. 2.8.**



Obr. 2.8 Trojimperativ znázorněn pomocí trojúhelníku

2.8 Životní cyklus projektu

Životní cyklus projektu dobře ilustruje **obr. 2.9**, který zachycuje nejvýznamnější etapy a kroky projektu. Podrobně je životní cyklus projektu znázorněn na **obr. 2.10**. V horním řádku jsou nejdůležitější etapy, v ostatních řádcích jsou jednotlivé procesy, které do daných etap spadají. Šipky naznačují jejich návaznosti. Uzavření etapy projektu se obvykle vyznačuje vyhodnocením a případným přezkoumáním klíčových činností a plnění projektu s cílem určit, jestli může projekt pokračovat další fází. Při uzavření etapy se také snažíme najít chyby a s minimálními náklady je tak zjistit a opravit.



Obr. 2.9 Životní cyklus projektu

Životní cyklus projektu se obvykle dělí:

- **Zahájení projektu:** Je to fáze, kde se vybere, vyhodnotí a definuje koncept projektu.
- **Plánování projektu:** Zde je koncept ověřen a zapracován do pracovního plánu pro impelementaci.
- **Organizování:** Fáze ve které se provádí řídící činnosti, kontroluje se, zda je projekt jasně definován.
- **Sledování a kontrola:** V této fázi se provádí implementace a kontroluje se splnění požadavků a funkčnost.
- **Uzavření projektu:** Je fáze projektu, kde se celý projekt dokončí a zdokumentuje, součástí je i předání zákazníkovi.

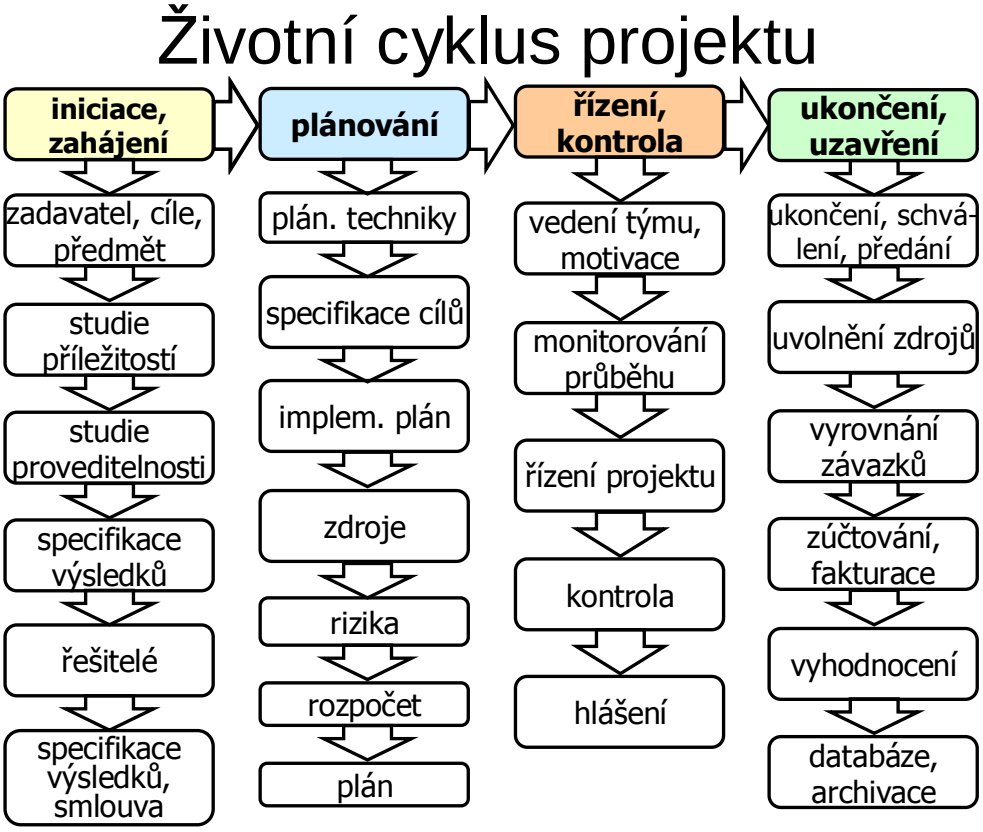
Ve fázi **zahájení projektu** by mělo vzniknout poměrně mnoho dokumentů, ve kterých se definuje budoucí podoba produktu. Prvním dokumentem, který vzniká, je definice problému (ukázka na přiloženém CD). V této fázi vznikají i další dokumenty. např. studie proveditelnosti, podrobná specifikace, studie příležitosti, úvodní studie atd.

Plánování projektu se považuje za nejkritičtější. Tato činnost trvá v průběhu celého projektu (na **obr. 2.9** naznačeno jako smyčka neshody výsledku). V této fázi se plánuje rozsah a vytváří se struktura prací (WSB), definuje se plán, vytváří se rozpočet, definují se požadavky na kvalitu a vzniká plán zdrojů. Dokumenty musí být přehledné s jasným výkladem.

Organizování je třetí fází, tuto fázi někteří autoři slučují s řízením a kontrolou. V této fázi se vytváří projektový tým, je proškolen a kontroluje se, zda má tým všechny potřebné zdroje pro úspěšné zvládnutí projektu.

Řízení a kontrola je předposlední fází. Řeší se zde uplatňování změn, ověřuje se rozsah a kontroluje se dodržování časového plánu. Součástí je i práce s týmem a komunikace. Průběžně se také dohlíží na plnění požadavků na kvalitu a vede se potřebná dokumentace (výkazy práce, zprávy o testování, monitorování atd.).

Poslední fází je **uzavření projektu.** Tato fáze je často přehlížena, ale přesto je velmi důležitá. Zjišťují se v ní výsledky projektu, které mohou být použity pro zdokonalení dalších projektů. Fáze sestává z ukončení projektu, archivace zkušeností, vyhodnocení výsledku projektu atd.



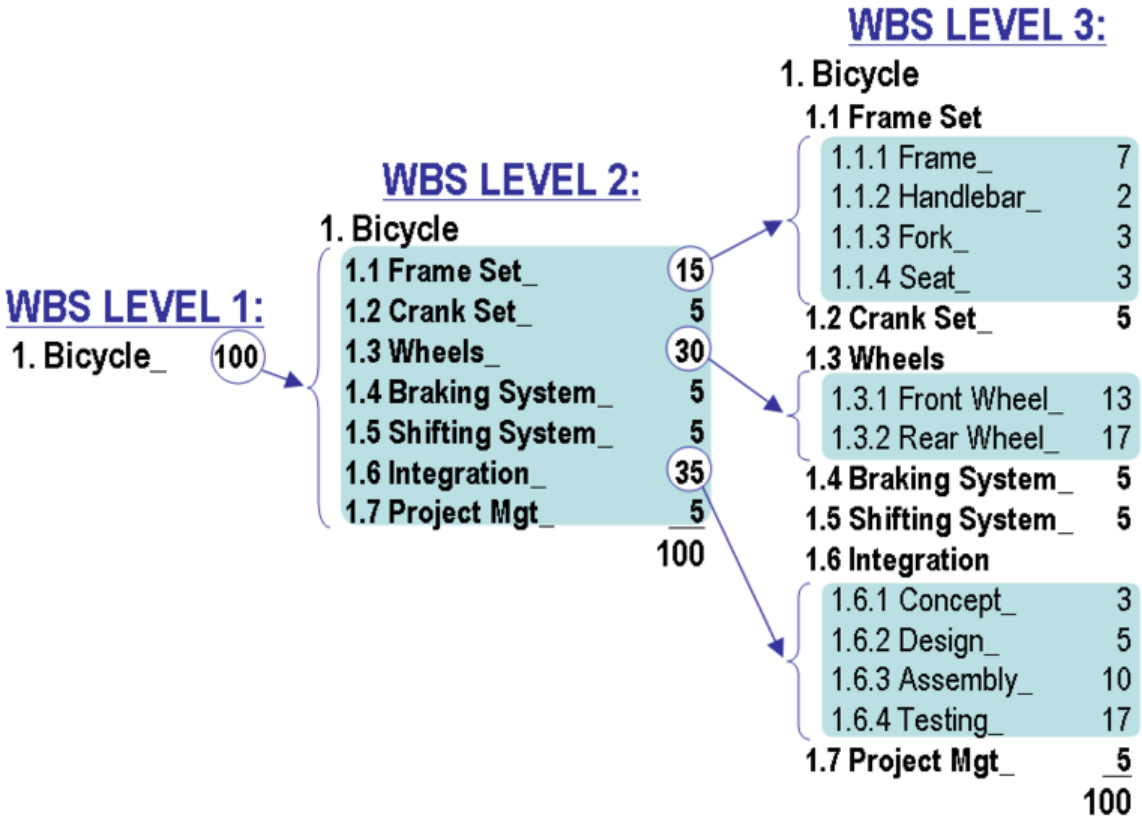
Obr. 2.10 Životní cyklus projektu podle [2]

3 Nástroje a techniky využívané při plánování

Při plánování projektů se používají různé nástroje a techniky. Většinou se jedná o různé způsoby grafické prezentace postupu práce na projektu.

3.1 Struktura prací - WBS (Work Breakdown Structure)

Složitý projekt se dá rozložit na jednotlivé části v hierarchické struktuře, známé jako struktura členění prací (Work breakdown Structure) nebo zkráceně WBS [5]. Taková struktura definuje úlohy, které mohou být dokončeny nezávisle na jiných úlohách, usnadňující rozmístění zdrojů, přidělení zodpovědností, měření a řízení projektu. Činnost, která není ve WBS, nepatří do rozsahu prací projektu. WSB bývá zobrazována jako stromová struktura (**obr. 3.1**) nebo ve formě tabulky (**tab. 3.1**).



Obr. 3.1 WSB jako strom (převzato z [5])

Úroveň 1	Úroveň 2	Úroveň 3	Úroveň 4
Úloha 1	Podúloha 1.1	Balík prací 1.1.1	Práce 1.1.1.1
			Práce 1.1.1.2
		Balík prací 1.1.2	Práce 1.1.2.1
			Práce 1.1.2.2
	Podúloha 1.2	Balík prací 1.2.1	Práce 1.2.1.1
		Balík prací 1.2.2	Práce 1.2.2.1
			Práce 1.2.2.2
Úloha 2	Podúloha 2.1	Balík prací 2.1.1	Práce 2.1.1.1
		Balík prací 2.1.2	Práce 2.1.2.1
			Práce 2.1.2.2

Tab. 3.1 WSB

Každá organizace používá svou vlastní upravenou variantu WBS podle úrovně její hierarchie. Některé organizace se odkazují na různé podúrovně projektu jako úlohy, podúlohy a balíky prací na nejnižší úrovni, jak je znázorněno v tabulce 4.1 nahoře. Jiné používají např. termíny, fáze, záznamy a činnosti atp.

WBS může mít různou strukturu, například podle fází v životním cyklu projektu nebo podle dodávek. Čím vyšší úroveň ve struktuře, tím víc osob se na ní obvykle podílí, naopak na nejnižších úrovních hierarchii často pracují pouze jednotlivci.

Rozklad projektu na díly usnadňuje rozvržení a přidělení jednotlivých zodpovědností, proto by se měla velká pozornost při vytváření WSB věnovat použití řádné úrovně detailů.

WBS je základem při plánování projektu. Je vytvořena předtím, než se určí závislosti a odhadne se trvání činností. WBS se může použít také pro identifikaci úloh v CPM a PERT modelech při plánování projektů.

3.2 Metoda kritické cesty – CPM (Critical Path Method)

Postup výpočtu kritické cesty je poměrně jednoduchý. Jak již bylo uvedeno v předchozím textu, každý projekt je třeba nejprve rozdělit na jednotlivé činnosti, které na sebe navazují a mají určitý nárok na čas a prostředky.

Nejnázornějším zobrazením jednotlivých činností a vazeb mezi nimi je hranově orientovaný síťový graf. Na **obr. 3.2** je příklad grafu projektu skládajícího se z činností A až L. Časové okamžiky, ve kterých činnosti začínají (resp. končí) se nazývají kontrolní body.

Při tvorbě grafu musí být dodrženo:

- Mezi libovolnou dvojicí kontrolních bodů vede maximálně jedna činnost. Pokud tomu tak v praxi není, lze pomocí tzv. fiktivních činností (činností s nulovou dobou trvání a s nulovými náklady) přejít na nový síťový graf projektu, pro který je již tento předpoklad splněn.

- Graf má právě jeden počáteční uzel, ve kterém činnosti pouze začínají (činnost A)

- Graf má právě jeden koncový uzel, ve kterém činnosti pouze končí (činnost L).

- Každá hrana grafu je ohodnocena – ke každé hraně (činnosti) je přiřazena její časová náročnost (jedná se o hranově ohodnocený síťový graf). Například činnost mezi uzly A a B je ohodnocena dobou trvání 4.

Často studovaným problémem bývá právě „nalezení kritické cesty“. Kritická cesta je definována jako (časově) nejdelší možná cesta z počátečního uzlu grafu do koncového uzlu grafu. V jednodušších úlohách je každá hrana grafu (činnost) (i,j) ohodnocena právě jedním časovým údajem (očekávanou dobou trvání činnosti) T_{ij}.

Když jsou známé veškeré očekávané doby trvání T_{ij}, lze vypočítat nejdříve možný začátek (ZM) doby činností začínajících v i-tém kontrolním uzlu (T_i). V uzlu A je triviálně $T_1 = 0$. Činnosti v i-tém bodě mohou začít až poté, co jsou dokončeny všechny činnosti, které v i-tém bodě končí. Neboli:

$$T_i = \max_k (T_k + T_{ki}) \quad \text{pro } k< i,\ i = 2,3,\ldots,n$$

Nejdříve možný konec (KM) celého projektu je určen hodnotou T_n.

V dalším kroku je třeba vypočítat nejpozději přípustný konec (KP) činností končících v i-tém bodě (T_i^*). V n-tém bodě se stanoví $T_n^* = T_i$. Činnosti, které v i-tém bodě končí, musí skončit nejpozději do té doby, než jsou zahájeny činnosti v i-tém uzlu začínající. Neboli:

$$T_i^* = \max_j (T_j + T_{ij}) \quad \text{pro } k< j,\ i = 2,3,\ldots,n\text{-}1$$

Kritická cesta je tvořena nepřetržitým sledem činností z bodu 1 do bodu n po takových uzlech, pro které platí $T_i^* = T_i$.

Každý projekt má minimálně jednu kritickou cestu. Každá kritická cesta se skládá ze seznamu činností, na které by se měl manager projektu nejvíce zaměřit, pokud chce zabezpečit včasné dokončení projektu. Každé zpoždění těchto činností má za následek zpoždění celého projektu.

Metoda kritické cesty je velice často používána při plánování různých projektů a prací. V současné době se k plánování projektů používá hlavně specializovaný software (např. MS Project). Pomocí takové aplikace lze snadno vytvořit plány kritické cesty. Plány mohou být vytvořeny podle různých kritérií. Na základě zadaných dat program vykreslí přehledný Ganttův diagram.

3.2.1 Příklad CPM

Stavba domu je náročnou záležitostí. V následujícím textu je uvedena smyšlená stavba rodinného domu. Čas je uveden v závorkách za jednotlivými činnostmi a je v týdnech. Celý příklad je řešen metodou kritické cesty (CPM).

i	j	T	ZM	KM	ZP	KP	R
A	B	4	0	4	0	4	**0**
B	E	5	4	9	4	9	**0**
B	F	3	4	7	9	12	5
B	C	4	4	8	6	10	2
C	D	4	8	12	10	14	2
D	J	15	14	29	14	29	**0**
E	F	3	9	12	9	12	**0**
E	G	2	9	11	9	11	**0**
F	D	2	12	14	12	14	**0**
G	F	1	11	12	11	12	**0**
G	H	10	11	21	33	43	22
H	K	8	43	51	43	51	**0**
H	L	8	43	51	57	65	14
I	H	5	38	43	38	43	**0**
I	K	6	38	44	45	51	7
J	I	9	29	38	29	38	**0**
J	K	3	29	32	48	51	19
K	L	14	51	65	51	65	**0**

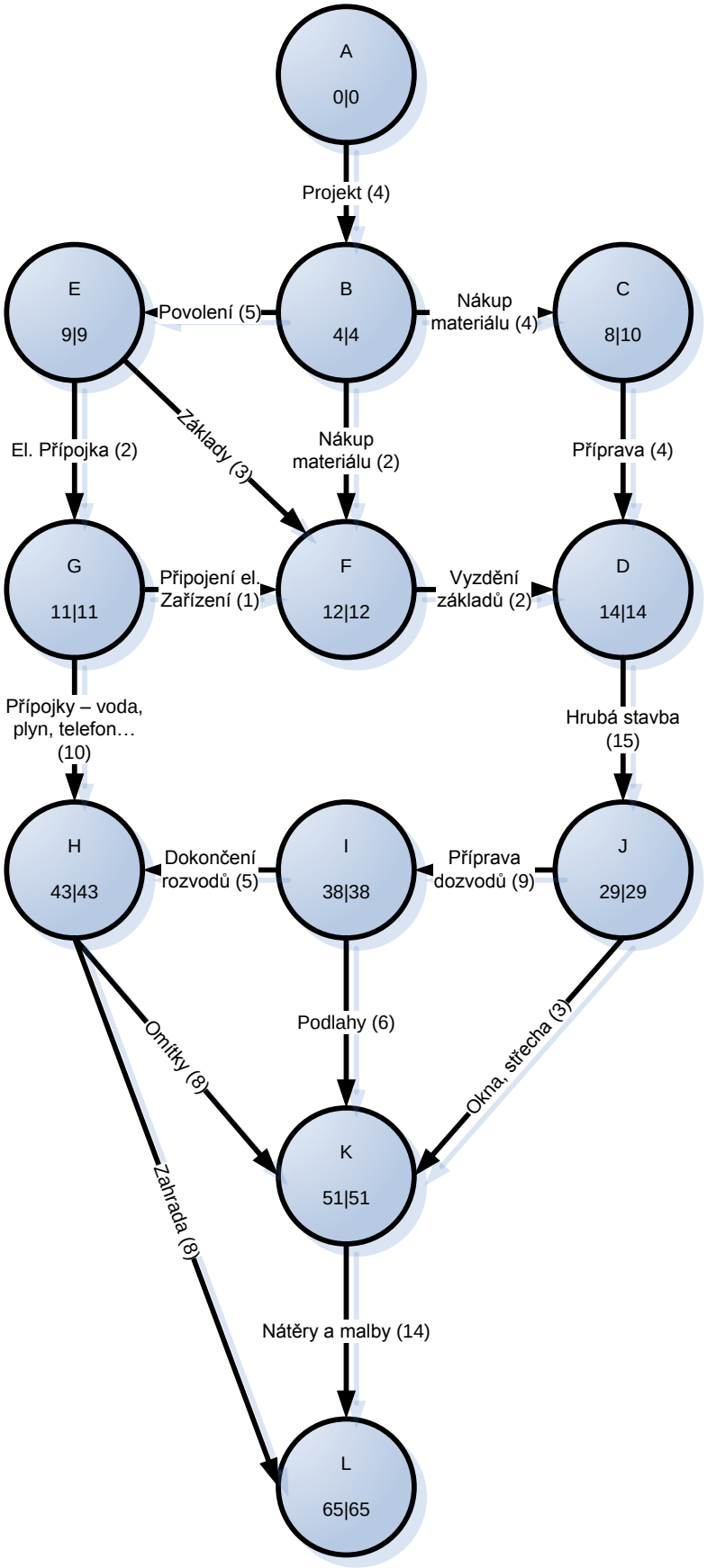
Tab. 4.2 Příklad výpočtu CMP

Tabulka obsahuje všechny důležité údaje:
i ……..Počátek činnosti – jeden z uzlů A-K (A je začátek celé stavby)
j ……..Konec činnosti – jeden z uzlů B-L (L je konce celé stavby)
Tij ……..Čas potřebný pro vykonání dané činnosti
ZM – Začátek Možný
ZP – Začátek Přípustný
KM – Konec Možný
KP – Konec Přípustný
R …….je časová rezerva (pokud je rezerva 0 je daná činnost součást kritické cesty), vypočítáme je tak že od KP odečteme hodnotu KM.

Graf kritické cesty s názvy činností a potřebným časem. Každý uzel obsahuje název (písmeno A-L), a ve spodním řádku časový údaj o možném začátku a přípustném konci. Z tabulky můžeme sestavit kritickou cestu tak, že ji složíme činností, které mají v tabulce u hodnoty R nulu. V našem případě jsou 2 možnosti kritické cesty tzn.:

A-B-E-F-D-J-I-H-K-L
A-B-E-G-F-D-J-I-H-K-L

Stavba domu trvá celkem 65 týdnů, tedy o něco déle než rok.

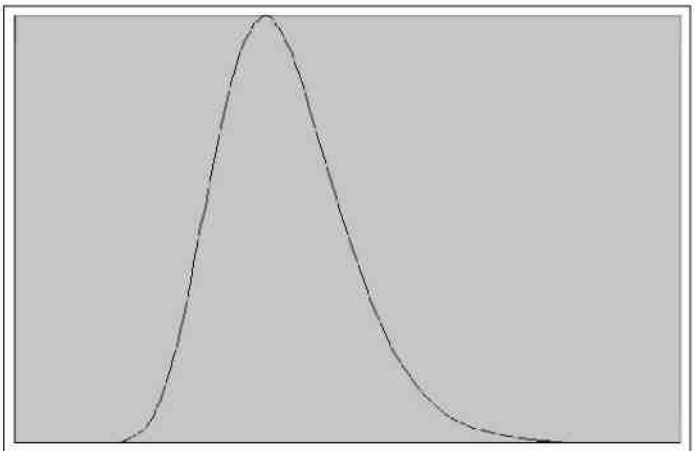


Obr. 3.2 Graf k výpočtu CPM (stavba domu)

3.3 Metoda vyhodnocování a kontroly programu - PERT (Program Evaluation and Review Technique

U metody kritické cesty byla každá hrana grafu (i,j) ohodnocena právě jedním časovým údajem (očekávanou dobou trvání činnosti) T_{ij}. Při odhadování jediného parametru je riziko, že plán bude příliš ambiciózní nebo naopak příliš pesimistický. Pro řadu úkolů navíc může být obtížné získat přesné odhady. V takovém případě je vhodné použít metodu PERT.

U této metody je každé hraně (i,j) přiřazena náhodná veličina reprezentující dobu trvání této činnosti. Předpokládá se, že doby trvání jednotlivých činností jsou na sobě nezávislé. Na **obr. 3.3** je znázorněn obvyklý tvar hustoty pravděpodobnosti rozdělení doby trvání.



Obr. 3.3 Hustota pravděpodobnosti rozdělení

Náhodná veličina $X(T_{ij})$ je charakterizována svojí střední hodnotou EX (ET_{ij}) a rozptylem VarX($VarT_{ij}$). V původní metodě PERT dobu trvání reprezentuje rozdělení β. Hustota tohoto rozdělení je nenulová jen na intervalu <a, b>. Pro každou činnost je třeba zadat tři parametry:
a - odpovídá optimistickému odhadu
b - odpovídá pesimistickému odhadu
m - odpovídá nejpravděpodobnější hodnotě

Střední hodnota β - rozdělení je $\mathrm{EX}(a,b,m)=\frac{a+4m+b}{6}$ a rozptyl je $\mathrm{VarX}(a,b,m)=\frac{(b-a)^2}{36}$.

V modifikované metodě PERT lze použít logaritmicko-normální rozdělení. Hustota má podobný tvar jako u β - rozdělení. Pro každou činnost je třeba zadat jen dva parametry - m, b.
m - odpovídá nejpravděpodobnější hodnotě
b - odpovídá pesimistickému odhadu.

Střední hodnota logaritmicko-normálního rozdělení $EX=\exp(\mu+\sigma^2/2)$ a rozptyl $\mathrm{VarY}=\exp(2\mu+\sigma^2)+(\exp(\sigma^2)-1)$, kde $\sigma\cong\sqrt{1-\ln\frac{m}{b}}-1$ a $\mu\cong\ln b-2\sigma$.

Jsou-li známy (či odhadnuty) parametry pro všechny činnosti projektu, lze zjistit pomocí v předchozím textu popsaných algoritmů kritickou cestu projektu. Stačí místo hodnot očekávaných dob trvání (T_{ij}) dosadit střední hodnoty ET_{ij} vypočítané z výše uvedených vztahů.

Z předpokladu nezávislosti plyne, že střední hodnota projektu je rovna součtu středních hodnot jednotlivých činností ležících na kritické cestě grafu. To platí i pro rozptyl projektu. Pokud existuje více kritických cest, je třeba vybrat tu s největším rozptylem.

3.4 Porovnání PERT a CPM

PERT jedna z metod síťové analýzy projektu. Je obdobou metody CPM. Liší se tím, že délky trvání jednotlivých dílčích činností projektu nejsou určeny jednoznačně, nýbrž jako stochastické veličiny (obvykle se udává optimistický odhad, pesimistický odhad a nejpravděpodobnější odhad doby trvání).

3.5 Kritický řetězec

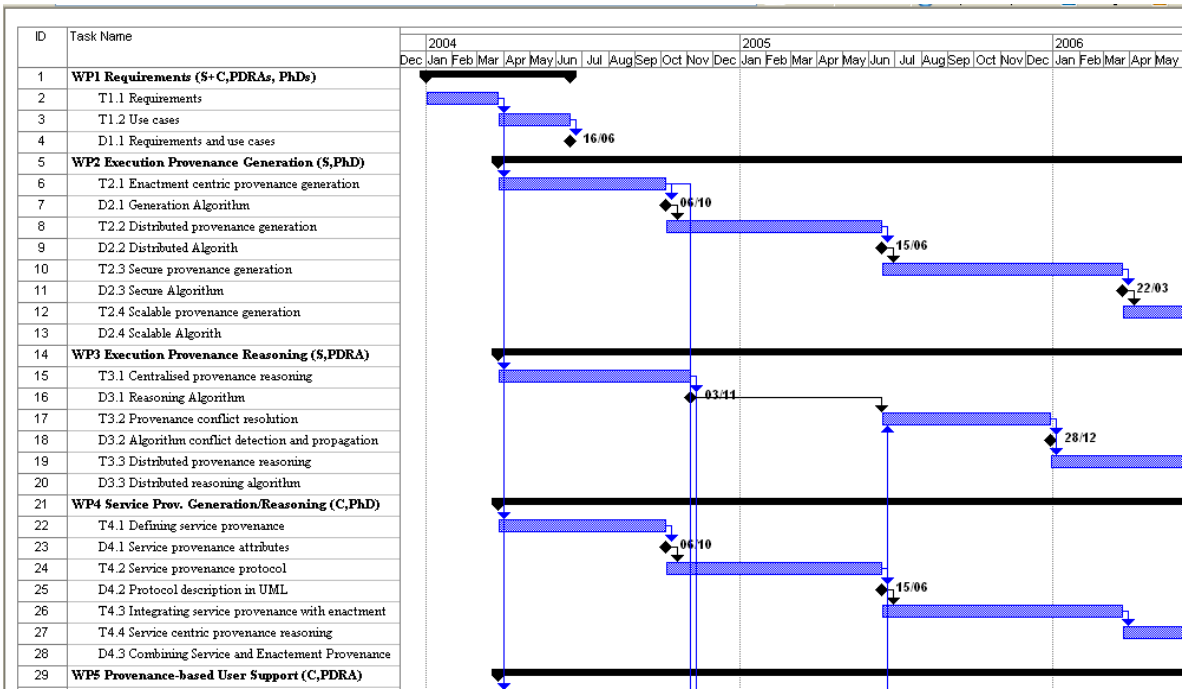
Řízení projektu pomocí kritického řetězce (CCPM) využívá metody a algoritmy, které byly vyvinuty v roce 1997 Eliyahuem M. Goldrattem. V řízení projektů je kritický řetězec posloupností činností projektu, které jsou závislé jak na prioritách, tak na zdrojích. To znemožňuje, aby byl projekt dokončen v kratším čase, když má stanoven konečný počet zdrojů. Kdyby byly zdroje k dispozici v nekonečném množství, pak by byl kritický řetězec stejný jako kritická cesta. Kritický řetězec se používá jako alternativa ke kritické cestě.

Odlišnosti kritického řetězce od kritické cesty jsou:

- Použití implicitních závislostí zdrojů (nejsou zahrnuty do projektové sítě, ale musí se rozpoznat)
- Nedostatečně hledá optimální řešení. To znamená, že najde dobré řešení, které považuje za dostatečné
- Neexistuje žádná známá analytická metoda nalezení absolutního optima (nejkratší kritický řetězec)
- Nejistota v odhadech je větší než rozdíl mezi optimálním a téměř optimálním řešením

3.6 Ganttův Diagram

V roce 1910 Henry Gantt (1861-1919) vyvinul nástroj pro zobrazení procesu projektu. Jednalo se o speciální graf, jeho první použití bylo sledování procesů projektu pro stavbu lodí. Dnes má Ganttův nástroj tvar horizontálního sloupcového grafu a je známý jako Ganttův graf, jehož základní snímek je zobrazen na **obr. 3.4** [3].



Obr. 3.4 Ganttův diagram [3]

Horizontální osa diagramu je časová stupnice. Jednotky na ose závisí na projektu, jsou obvykle udávány v týdnech nebo měsících. Řádky sloupců v diagramu značí časové úseky, kdy probíhá daná činnost. Činnosti mohou být vzájemně propojeny šipkami, které značí jejich vzájemnou návaznost.

3.7 MPM síť (Metra Potential Method)

MPM je další síťová technika, která byla poprvé použita v padesátých letech pro stavbu křížové lodi Le France a poté se používala například při stavbách atomových elektráren. MPM patří do kategorie metod s událostními uzly.

Při vytváření sítí typu MPM se události zobrazují jako uzly a závislosti mezi událostmi se značí šipkami (hranami). Milníky přejaté z WBS, se zobrazují jako události s dobou 0. Nejdůležitější pro sestavení MPM sítě je si všimnout, že uzly jsou propojeny ve vztahu začátek-začátek. Každá šipka, která vychází z uzlu, musí vyústˇovat do jiných uzlů. Minimální časové odstupy mezi událostmi se zobrazují popsáním šipek. Dvě události mohou být propojeny maximálně dvěmi šipkami (pozitivní a negativní). MPM sítě mohou obsahovat cykly. Víc informací je v [3].

4 Exitující aplikace pro plánování a řízení projektů

4.1 Iterity Project

Iterity Project je software vytvořený pro společnosti, které realizují své podnikání převážně formou různých projektů či zakázek. Je tedy určen především pro developerské kanceláře, architektonické a realitní kanceláře, projektové kanceláře, vývojářské firmy a organizace zaměřené na facility managment.

Aplikace pokrývá všechny fáze projektu od investičního záměru či plánování, přes realizaci a až po předání klientovi. Iterity Project umožňuje řídit celý průběh projektu, organizaci, ale také týmovou spolupráci ve společnosti. V systému lze všechny procesy zautomatizovat a podpořit tak jejich hladký průběh a návaznost dílčích úkolů. Tím dochází ke snížení rizika chyb a k eliminaci problémů při rozdělování úkolů.

Cílem vývojářů společnosti Iterity bylo vytvořit produkt, který by umožňoval příjemnější a snadnější kontrolu nad projektem vedoucímu zakázky. Ten může sledovat výkonnost individuálních pracovníků, přerozdělovat jejich úkoly nebo vyhodnocovat parciální etapy projektu z hlediska nákladů, ale i již získaných výnosů. Může se tak plně soustředit na strategické cíle projektu, aniž by se musel zatěžovat běžnými operativními záležitostmi.

Pro jednotlivé členy projektového týmu znamená tento produkt naopak zjednodušení jejich každodenní činnosti. Jedná se o operativní aktivity jako vnější i vnitrofiremní komunikace, sledování vlastních obchodních příležitostí, zadaných úkolů a schůzek nebo automatické vytváření různých smluv či dokumentů.

4.1.1 Co umožňuje Iterity Project?

- řízení týmové spolupráce
- plánování času a úkolů
- workflow
- řízení a evidence obchodních příležitostí
- efektivní zasílání a dohledávání zpráv
- získání přehledu o spotřebě a plánu času zaměstnanců společnosti
- přehlednou správu historicky ukládaných dokumentů a jejich sdílení mezi uživateli
- flexibilní tvorbu komplexních sestav potřebných pro reporting
- automatickou tvorbu libovolných dokumentů na základě šablon sestavených uživatelem
- sdílení a správu historicky ukládaných kontaktů
- sledování historie vztahů s klienty společnosti, obchodními partnery či zaměstnanci

Iterity Project umožňují řídit současně různé projekty, jejich jednotlivé části a kroky, určovat odpovědné osoby a jejich pracovní týmy. V systému jsou projekty zakládány pomocí šablon, jenž definují typ projektu a formulář, který náleží projektu. Na základě toho může organizace vytvářet libovolné vlastní typy projektů jako např. provozní projekty vývojového či produkčního charakteru, obchodní příležitosti, smlouvy, zakázky, právní kauzy, poskytované služby, apod. Projekt může být evidován ve vazbě na partnera, zákazníka nebo jiné projekty.

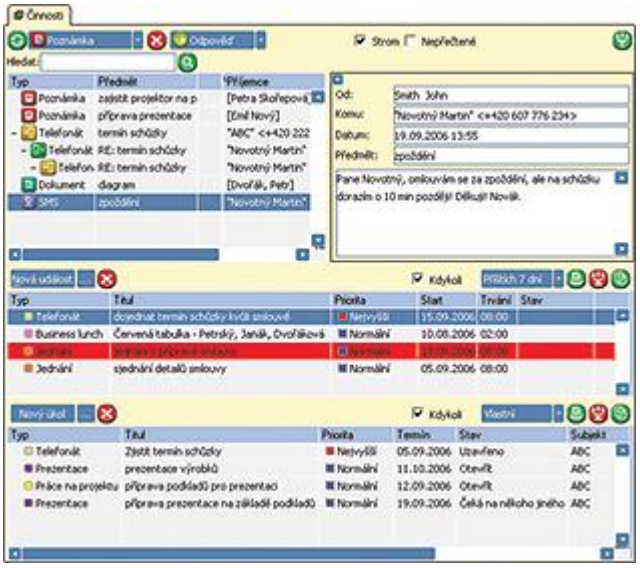
4.1.2 Klientské prostředí aplikace

Po přihlášení uživatele do aplikace se mu zobrazí v takové podobě, v jaké ji naposledy opustil. Rozměry oken, nastavení filtrů, ale i další vizuální prvky zůstávají uchovány v podobě, ve které byly opuštěny při posledním ukončení aplikace.

Na titulní straně se zobrazují nejčastěji používané funkce. Tuto stranu lze nastavit podle požadavků uživatelů, kteří s aplikací pracují. Stejně je možné nastavit i ostatní části uživatelského prostředí. Titulní strana slouží k rychlému přehledu aktuálních událostí. Zobrazují se zde informace o nových zprávách**,** naplánovaných schůzkách v kalendáři pro aktuální den, nesplněných či zadaných úkolech**,** apod.

Na titulní straně je zároveň možné zobrazit výsledek libovolné sestavy, např. „Nové zakázky tento týden" nebo „Klíčoví zákazníci", apod. Záleží pouze na uživateli, jak si titulní stranu přizpůsobí svým potřebám.

Jednotlivým projektům i dílčím činnostem lze přidělit odpovědnou osobu a realizační týmy. Vyhodnocování projektů je velmi jednoduché především díky vytváření libovolného množství různých sestav, které zobrazují veškeré informace vztahující se k projektu.



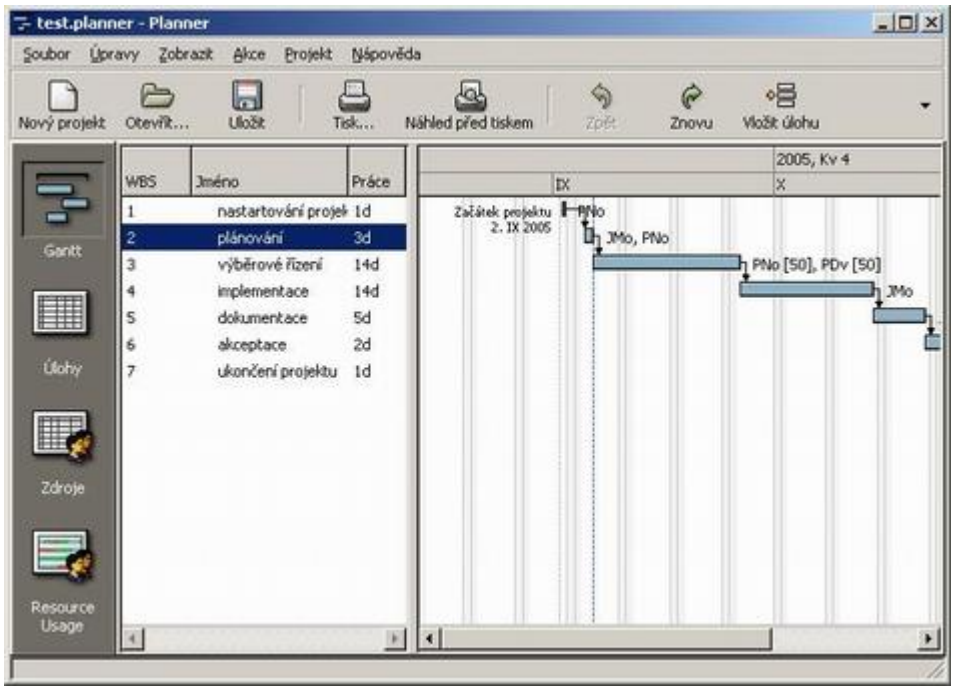
Obr. 4.1 Ukázka uživatelského rozhraní Iterity Projectu

4.2 Imendio Planner

Imendio Planner je malý program, který je dodáván jako součást grafického prostředí Gnome v distribucích Linuxu. Později vznikla i jeho varianta určená pro platformu Windows - WinPlanner. Program není určen pro profesionální projektové manažery. Je velice jednoduchý neumí toho mnoho, ale pracuje se zdroji, úkoly a kreslí Ganttovy diagramy a využití zdrojů. Pro spoustu malých, třeba studentských, projektů ale stačí. Protože je součástí Gnome, je tento program zdarma a je obsažen ve většině linuxových distribucí, případně je možné ho zdarma stáhnout z webu autorů. Při instalaci verze pro MS Windows (WinPlanner) je třeba nejprve nainstalovat GTK+ runtime, minimálně verze 2.6.4.

Velkou výhodou tohoto jednoduchého software je možnost importu dat z MS Project XML. Program je opravdu velice jednoduchý a intuitativní, a uživatelské prostředí je přehledné. I bez manuálu lze velice rychle zadat zdroje (ikona „Zdroje" na levém panelu) a úkoly (ikona „Úkoly" na levém panelu) ze kterých se automaticky vykresluje Gantův diagram. V diagramu je pak možné pomocí myši editovat závislosti jednotlivých úloh, přiřazovat zdroje apod. To se okamžitě promítá do tabulek s úkoly a zdroji. Ve všech podoknech je k dispozici užitečná funkce „zpět" pro případ nechtěné úpravy.

Tento program se nejvíce hodí pro malé projekty, pro které by se běžně plán projektu vůbec nedělal. Při jeho použití je však možné i u těchto projektů dosáhnout zkvalitnění. Je zdarma, nedochází tedy ke zbytečnému navyšování rozpočtu jednoduchých projektů a vzhledem k jednoduchosti obsluhy nepotřebuje uživatel žádné drahé školení.



Obr. 4.2 Ukázka uživatelského rozhraní WinPlanneru

4.3 Microsoft Office Project

MS Project je určen pro řízení projektů a je navržen tak, aby pomáhal manažerům projektů při tvorbě plánů, rozdělování úkolů členům týmu, kontrolování postupu práce na projektu, a analyzování nákladů. V roce 1995 společnost Microsoft uvolnila první verzi Microsoft Project Windows 95. V dalších letech postupně následovali verze 98, 2000 a 2003.

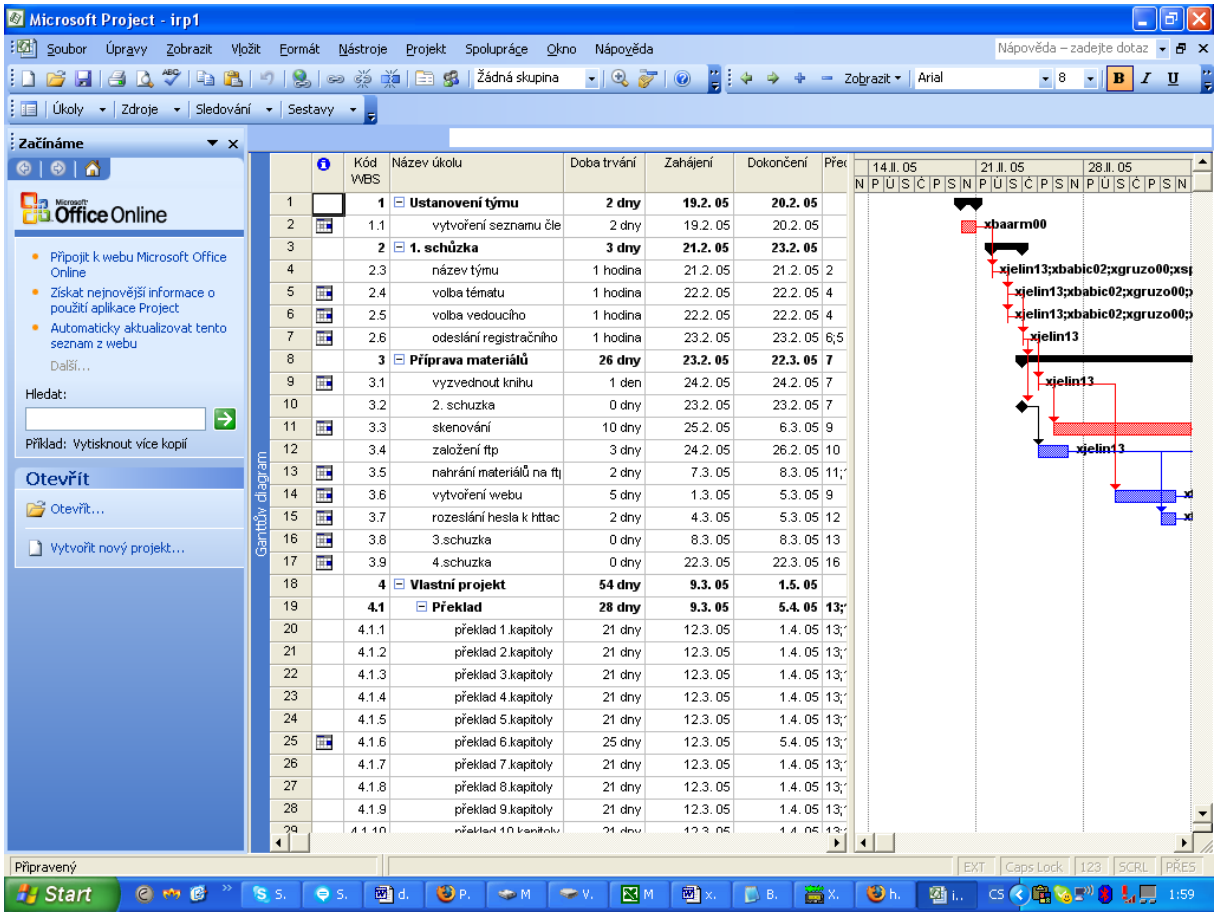
V posledních verzích Microsoft Office, byly rozšířeny schopnosti MS Project uvedením MS Office Project Server a MS Project Web Access. MS Office Project Server ukládá data projektu do centrální databáze a umožňuje uživateli zobrazit a aktualizovat tato data přes internet. MS Project Web Access dovoluje autorizovaným uživatelům přistoupit k projektům pomocí internetu a zahrnuje časové listy, grafické analýzy a pracovní náklady zdrojů a administrativní nástroje.

MS Project umožňuje rozdělit uživatele do tříd, které mohou mít odlišné přístupové úrovně k projektům, pohledům a jiným datům.

Pomocí aplikace lze snadno vytvořit plány kritické cesty. Plány mohou být vytvořeny podle různých kritérií. Na základě zadaných dat program vykreslí přehledný Ganttův diagram.

MS Project umožňuje vytvářet rozpočty založené na přiřazení práce a sazeb zdrojů. Jakmile jsou zdroje přiřazeny úlohám a je stanovena cena práce, program vypočítá náklady. Definice zdrojů (lidé, vybavení a materiály) se dají sdílet mezi projekty. Každý zdroj může mít svůj vlastní kalendář, který definuje, ve kterých dnech je zdroj k dispozici.

Protože software funguje jako část balíku MS Office, je možné vyměňovat data mezi jednotlivými produkty tohoto kancelářského balíku (např. s PowerPointem nebo Visiem).



Obr. 4.3 Ukázka uživatelského rozhraní MS Projectu 2003

4.4 Achievo

Achievo je flexibilní síťový nástroj pro řízení projektů a zdrojů vhodné i pro větší firmy. Je k dispozici jako open source. Achievo je naprogramován v PHP a používá session pro přihlašování uživatelů jako u mého programu. Přesto, že je napsaný v PHP, je objektově orientovaný. Achievo vyžaduje grafickou knihovnu GD2 pro vytváření různých grafů. Produkt Achievo pro řízení projektů je možné stáhnout z http://www.achievo.org/

Achilevo umožňuje:

1. Vytvářet zprávy a dokumenty o projektu

Achievo nabízí několik způsobů plánování projektu (plánování projektů, fází, zaměstnanců). Umožňuje sledování času, aby se dalo lehce určit, kolik času se strávilo na projektu. Je k dispozici několik statistik pro porovnání se skutečným stavem realizace projektu.

2. Odhad a plánování

Čas, který se stráví na poradách, se obvykle podceňuje. Achievo nabízí statistiky, které uživatele informují, kolik lidé strávili průměrně na poradách, návrhu a implementaci atd.

3. Řízení nákladů a přidělování činností

Achievo nabízí několik způsobů práce, které lze ovládat a nastavit vedoucím projektu. Lidé mohou zaznamenávat čas a jakmile skončí činnost, mohou uzavřít týden, po němž může správní oddělení vyúčtovat odpracované hodiny pro výpočet mzdy. Toto lze však nakonfigurovat i tak, že se o uzavření stará vedoucí projektu. Zaměstnanci jsou upozorňováni emailem, pokud jsou pozadu.

4. Přiřazování zdrojů

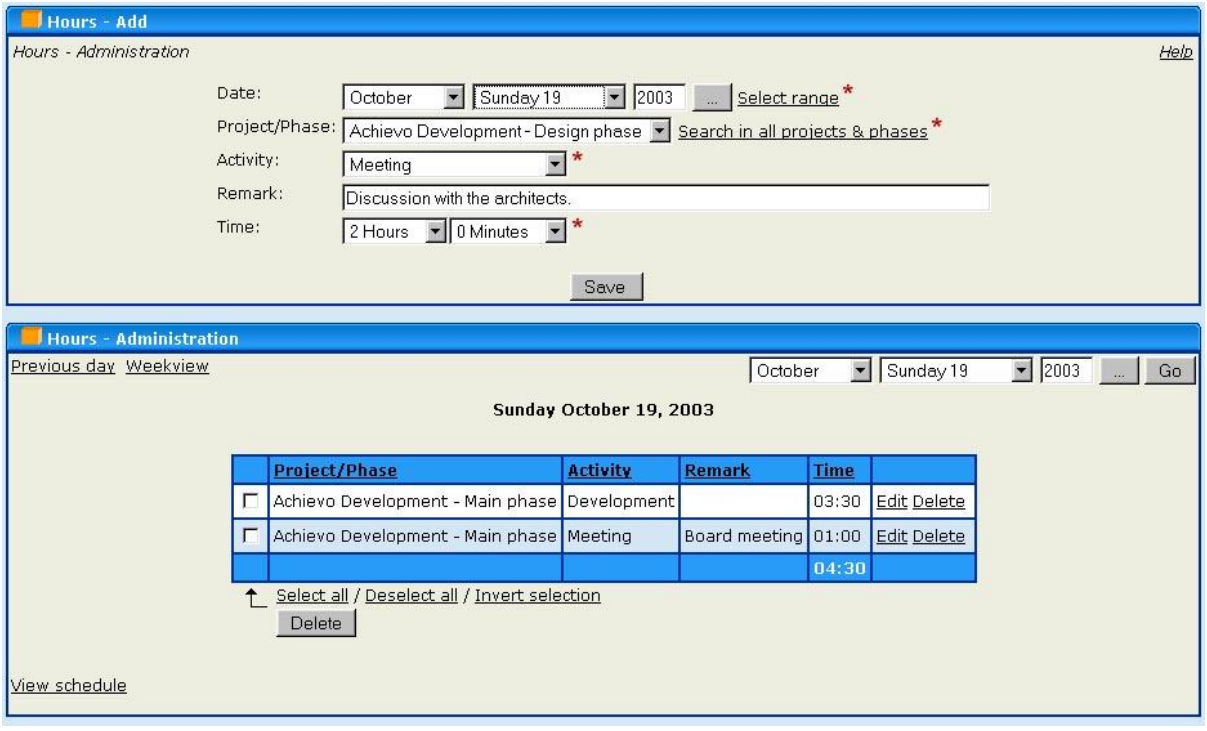
Nástroj plánování zdrojů dovoluje manažerům projektů plánovat zaměstnancům činnosti až do týdenní úrovně. Porovnáním hodinových sazeb lidí s odhady projektu manažer okamžitě vidí, jak by se náklady změnily, kdyby udělal nějakou změnu v projektu. Porovnáním odhadovaných časů se skutečnými časy manažer může také vidět, jaký důsledek to bude mít na termín dokončení projektu.

5. Komunikace v rámci projektu

Achievo nabízí možnost umístit poznámky do projektů, aby se udržela informace pohromadě.

6. Spolupráce projektového týmu

Software nabízí soustředěný program, s možností nahlédnout do plánů kolegů (pokud to povolili ostatním). Je zde vytvořen modul pro sledování seznamu úkolů s možností přidělení úkolů zdrojům. Změny je možné provádět během celého projektu.



Obr. 4.4 Ukázka uživatelského rozhraní WinPlanneru

4.5 AutoCont Enterprise Project Management

Společnost AutoCont CZ nabízí komplexní řešení pro podporu projektového řízení v podnicích s názvem Enterprise Project Management. Při jeho tvorbě AutoCont CZ čerpal ze svých dlouhodobých zkušenosti s vedením široké palety projektů v oblasti IT (HW, SW i infrastruktury, dodávek a implementace různých typů informačních systémů ERP, IDM, projektů vývoje nových SW aplikací i zajištění IT podpory pro celosvětové konference apod.).

Společnost AutoCont CZ kombinuje své zkušenosti z řízení projektů s nejnovějšími technologiemi Microsoft v oblasti projektového řízení a s jejich využitím vytváří další systémy pro podporu projektového řízení.

Základem úspěšného řešení je kvalitní metodika projektového řízení v podniku. Proto má AutoCont CZ propracovanou technickou podporu pro nasazení svého systému projektového řízení u zákazníků. To umožňuje vytvořit konkrétní řešení systému projektového řízení v podniku.

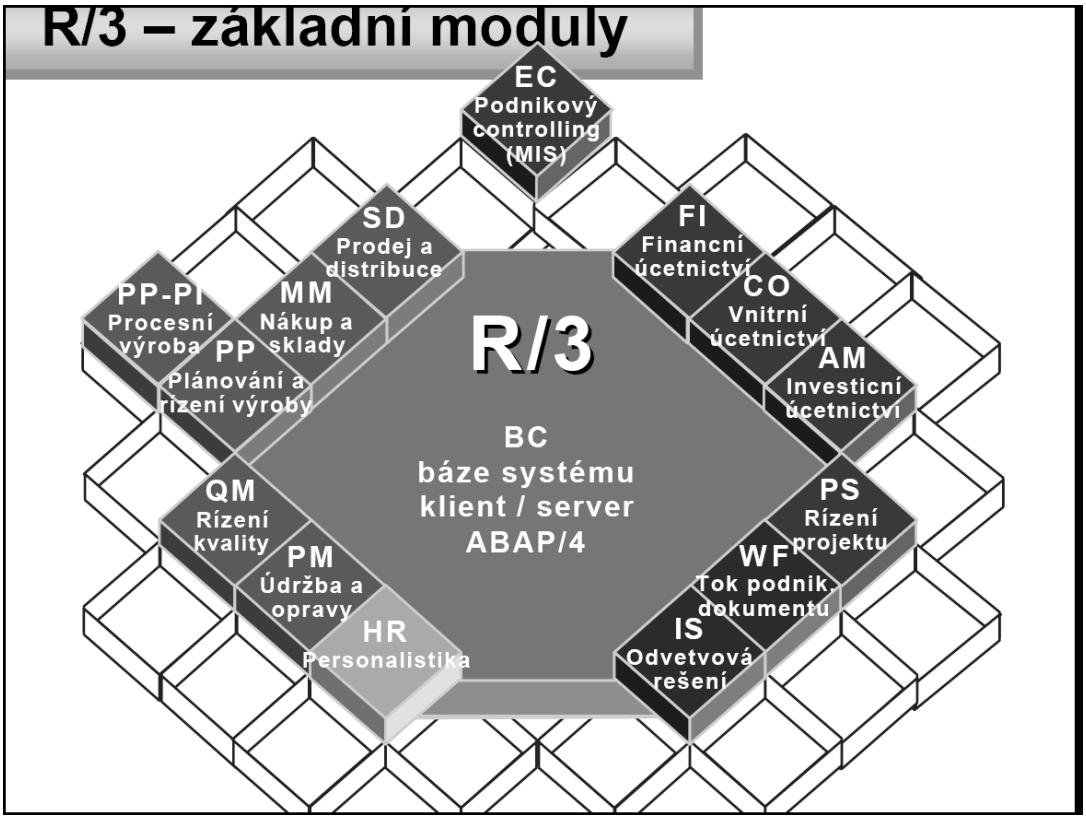
4.5.1 Co AutoCont Enterprise Project Management umožňuje

- Definici projektových zdrojů - Všechny podnikové zdroje jsou evidovány a popsány v centrálním podnikovém fondu zdrojů. Tak je zajištěna okamžitá informovanost o parametrech zdrojů a o jejich vytíženosti napříč všemi projekty v podniku.
- Tvorba a sledování projektových harmonogramů - Projektový manažer vytvoří projektový plán a k jednotlivým činnostem přidělí zdroje z centrálního fondu zdrojů. Zpracovaný projektový plán a informace o projektu pak publikuje na projektovém serveru. Při vlastní realizaci sleduje průběh projektu na základě informací od členů celého projektového týmu.
- Členové projektového týmu - Systém automaticky odesílá e-mailem jednotlivým členům projektového týmu upozornění na přiřazení nového úkolu na projektu. Uživatelé se potom prostřednictvím intranetu přes Project Web Access přihlásí k projektovému serveru. Zde vykážou uskutečněnou práci na projektových úkolech a stav jejich plnění. Současně mají přístup k projektovým dokumentům, k seznamu problémů a rizik a k dalším projektovým informacím. Členům projektových týmů nabízí systém integraci s kalendářem v MS Outlook a tím i efektivní sledování pracovního času a vykazování prací na projektech.
- Projektová dokumentace - Projektový server umožňuje publikování projektové dokumentace a práci s ní. V systému lze dokumenty verzovat a pro každý projektový dokument vytvořit pracovní prostor pro týmovou spolupráci. Lze rovněž sdílet poznámky a případně upozorňovat na vzniklé problémy a rizika na projektech.
- Plánování zdrojů - Nasazení pracovníků na projekty většinou neodpovídá organizačnímu rozdělení pracovníků v rámci organizace – do divizí, oddělení, úseků atd. Projektový server poskytuje manažerům zdrojů, pod které pracovníci organizačně spadají, přehled o využití svých pracovníků na projektech organizace, i když tyto projekty sami neřídí.
- Analýza projektů - Vedoucí pracovníci a management společnosti mají přístup jak k dílčím tak i k souhrnným informacím o stavu projektů a o vytížení podnikových zdrojů. Systém umožňuje vytvoření nejrůznějších analytických pohledů a modelů napříč celým portfoliem projektů.
- Technologie řešení - Hlavní informační základna všech projektových informací je centrální projektový server MS Office Project Server 2003, ke kterému se přihlašují všichni účastníci projektového řízení. Zde jsou ukládána všechna data projektového systému. Jednotliví uživatelé přistupují k informacím podle typu své role v projektu buď pomocí klientské

aplikace MS Office Project Professional 2003 nebo pomocí síťového klienta MS Office Project Web Access. Vedení projektových dokumentů, problémů a rizik zajišťuje služba Windows SharePoint Services. Pro vlastní databázový stroj je využíván MS SQL 2000 Server.

4.6 Shrnutí

V této kapitole je uvedeno několik zástupců více či méně konkurenčních nástrojů pro projektové řízení. Jsou zde jak zástupci jednoduchých jednoúčelových aplikací (WinPlanner), tak složité a propracované systémy (Iterity Project). Jsou zde produkty komerční i nekomerční, síťové i lokální (spustitelné jen na lokálním PC), méně či více specializované na určitou oblast. Snažil jsem se, aby každý z popisovaných produktů zastupoval jinou skupinu aplikací.



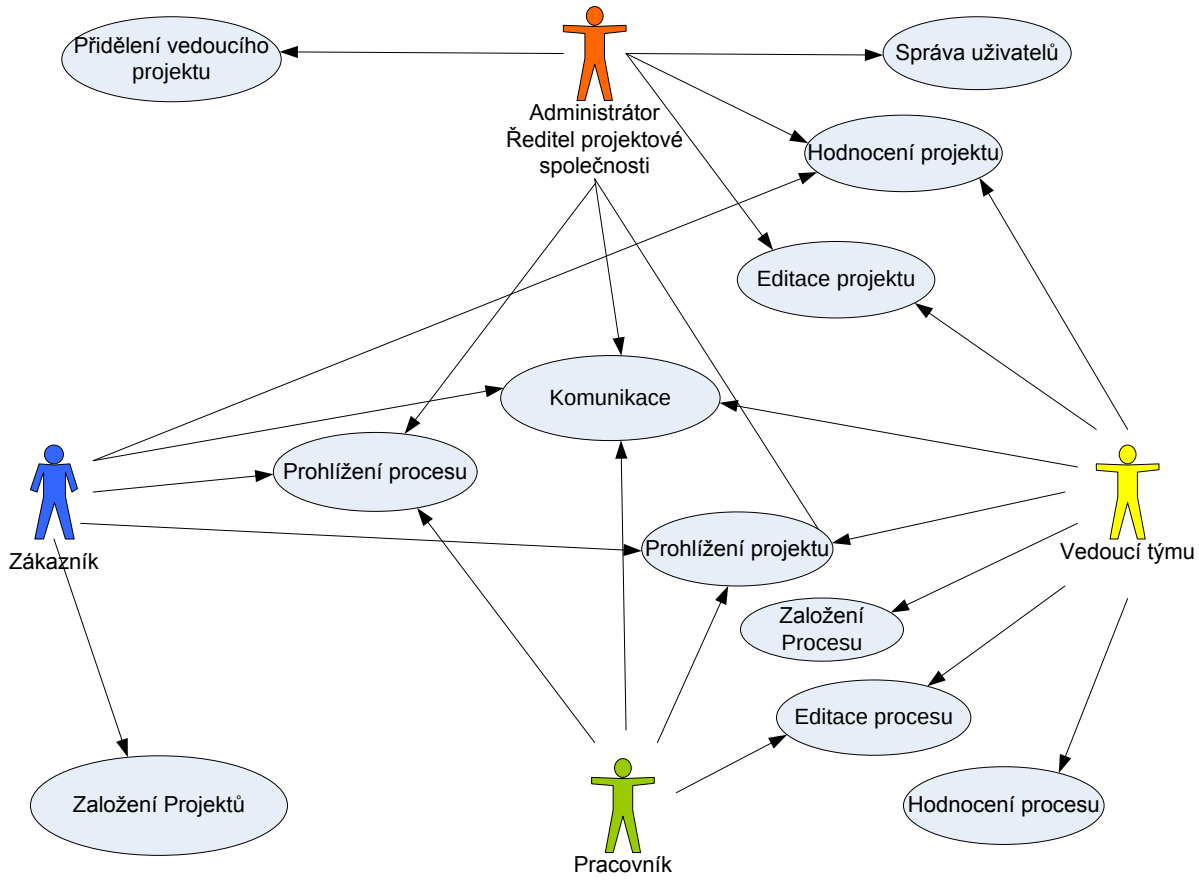
Obr. 4.5 Struktura podnikového informačního systému SAP R/3 (převzato z [16])

V současné době je stále běžnější, že jsou nástroje pro podporu řízení projektů součástí podnikových informačních systémů (případně jsou dodávány jako volitelné moduly). Příkladem takové integrace nástrojů pro podporu projektového řízení může být i jeden z nejznámějších podnikových informačních systémů SAP R/3 (**obr. 4.5**). Společnosti vyvíjející podobné informační systémy mají velký zájem na tom, poskytnout svým zákazníkům kompletní softwarovou podporu pro jejich činnost. Aplikace pro projektové řízení, která spolupracuje s dalším software ve společnosti, je velice výhodná. Umožňuje například automatické zpracování informací o odpracovaných hodinách, jednotnou komunikaci se zákazníkem, kontrolu nadřízených pracovníků, objednávky materiálu přes software skladu atd. Navíc umožňuje zahrnout informace z projektů do celkových výkazů.

5 Návrh a Implementace

5.1 Návrh jednoduchého systému pro správu projektů

Navrhovaný systém pro řízení projektu by měl umožnit rozdělení projektu na fáze a činnosti. Měl by umět přiřadit jednotlivým činnostem zdroje a kontrolovat průběh činností. Ve stádiu přípravy by měl umožnit nastavení zahájení a ukončení činností. Sytém by měl také umožňovat správu dokumentů, které se týkají projektu. Ve fázi provádění by měl systém umožnit aktualizaci zadaných údajů a hlášení o zpoždění činností. Při ukončování projektu by měl systém nabídnout celkové shrnutí projektu ve zprávě a různé statistiky. Měl by též poskytnout grafické naplánování činností pomocí jednoduchého Ganttova diagramu, případně síťového grafu, pro lepší představu o průběhu projektu.



Obr. 5.1 Diagram užití

.

5.1.1 Use case diagram navrhovaného systému

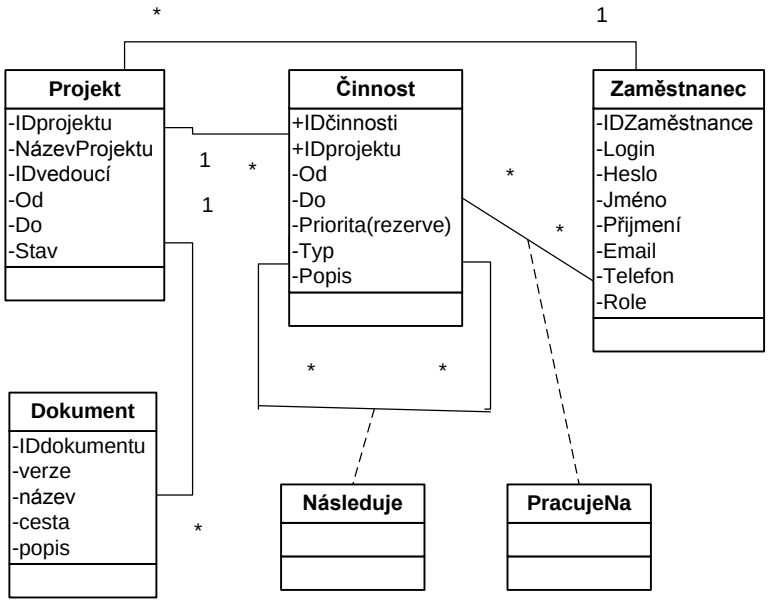
Návrh aplikace počítá se čtyřmi druhy uživatelů. Administrátor na **obr. 5.1** může pracovat s uživateli a projekty. Smí vkládat, editovat, mazat a aktualizovat projekty a informace o osobách. Pracovník má omezená práva pouze na prohlížení dat uložených v databázi. Nemůže mazat, ani aktualizovat údaje

vložené vedoucím, může však vkládat informace o stavu své práce. Zákazník může vkládat projekty a kontrolovat jejich průběh. Na konci projektu jej může ohodnotit. Všichni uživatelé mohou spolu komunikovat.

5.1.2 Diagram tříd

Návrh diagramu tříd je na **obr. 5.2**. Projekt je nejdůležitější třídou, jeho primárním klíčem je IDProjekt, což znamená identifikační číslo projektu. Třída Zaměstnanec má jako primární klíč atribut IDZaměstnance. Přihlašování do systému je řešeno pomocí HTTP autorizace. Projekt má pouze jednoho vedoucího (IDVedoucí) ze třídy Zaměstnanec, který je za něj plně zodpovědný.

V tabulce Dokument se ukládají data o názvu, verzi a cestě k souborům, které uživatel přiložil. Třída Následuje je potřebná pro síťový graf, kde se uchovávají vazby mezi činnostmi, aby se mohl potom zobrazit jejich časový sled.



Obr. 5.2 Diagram tříd

5.2 Implementace

Tato kapitola je věnována popisu zvolených řešení pro systém pro plánování a řízení týmových projektů.

5.2.1 Implementační jazyk

Výběr implementačního jazyka ovlivní i výběr dalších nástrojů a aplikací, které budeme používat (např. databázový a webový server). V úvahu přicházejí ASP .NET, JSP a PHP. Protože by systém pro plánování a řízení týmových projektů měl být přenositelný, a to hlavně v unix-like systémech, implementace pomocí .NET technologie není příliš vhodná.

JSP a PHP jsou realizovatelné na platformě Unix i na Windows. Pomocí JSP a frameworků, kterých se v dnešní době nabízí celá řada (Struts, Spring), by se dal celý systém realizovat, ale jednoznačně nejrozšířenějším jazykem je PHP. Důvody pro použití PHP shrnuje následující část.

PHP

PHP (rekurzivní akronym pro PHP: Hypertext Preprocessor) je široce používaný mnohoúčelový skriptovací jazyk, šířený pod Open Source licencí. Je zvláště vhodný pro vývoj webových aplikací. PHP bylo původně zaměřeno jen na skripty na straně webového serveru, méně známá je skutečnost, že PHP lze dnes úspěšně použít pro psaní spustitelných aplikací a skriptů.

Existují tři hlavní oblasti, v nichž je možné použít PHP skripty:

- Skriptování na straně serveru je nejpoužívanější nasazení PHP. Pro správnou funkci potřebujeme tři věci. Parser PHP (CGI nebo modul serveru), webový server na straně serveru a webový prohlížeč na straně klienta.
- Skriptování v příkazovém řádku. PHP skript lze provozovat bez webového serveru a prohlížeče. Pro toto použití potřebujeme pouze PHP parser. Tento typ použití je ideální jako náhrada pro psaní BASH skriptů u unix-like systémů. Použít jej lze i v Naplavovaných úlohách (ve Windows). Tyto skripty mohou být rovněž použity v úlohách pro práci s textem a XML soubory.
- Psaní desktopových aplikací. PHP asi není ten nejlepší jazyk pro tvorbu desktopových aplikací s grafickým rozhraním, ale PHP je velmi rozšířený skriptovací jazyk, který umí i mnoho programátorů amatérů a jeho znalost tedy mohou využít i při vytváření aplikací na straně klienta. Takový program můžeme napsat pomocí PHP-GTK. PHP-GTK je rozšíření jazyka PHP. Jeho cílem je dokázat, že PHP jako programovací jazyk je použitelné i pro vytváření klientských GUI aplikací. (GUI = grafické uživatelské rozhraní; okna, tlačítka, menu, křížek v pravém horním rohu apod.). PHP-GTK umožňuje rovněž napsat aplikaci pro více platforem.

PHP lze použít na různých operačních systémech, jako jsou GNU Linux, mnoho variant Unixu (včetně HP-UXu, Solarisu a OpenBSD), Microsoft Windows, Mac OS X, RISC OS a ve spoustě dalších. PHP má v současné době podporu pro většinu existujících webových serveru (Apache, Microsoft Internet Information Server, Personal Web Server, servery Netscape a iPlanet, Oreilly Website Pro, Caudium, Xitami, OmniHTTPd a další). Pro většinu z nich existují PHP moduly.

Jak je vidět, s PHP si můžeme svobodně zvolit téměř jakýkoliv operační systém i webový server. Dále si můžeme vybrat, zda použijeme procedurální nebo objektově orientované programovaní, případně směs obojího. Ačkoli v PHP 4 nejsou implementovány všechny standardní vlastnosti OOP, mnoho knihoven a rozsáhlých aplikací je napsaných výhradně za použití technik OOP. PHP 5 opravuje nedostatky tykající se OOP a zavadí kompletní objektový model.

Další výhodou PHP je to, že nejsme omezeni pouze na vystup HTML. PHP umožňuje vytvářet obrázky, souboru PDF a dokonce Flash animace (je nutné použít knihovny libswf a Ming). Výstupem může být jakýkoli XML soubor. PHP umí tyto soubory nejen vypisovat, ale také automaticky generovat a ukládat je do souborového systému.

Jedna z nejsilnějších a nejvýznamnějších vlastností PHP je jeho podpora pro širokou škálu databází. Více o PHP je uvedeno v [20], [21], [22].

5.2.2 Databázový server

Databáze je základ tohoto systému, začneme tedy u výběru databázového serveru. Možností pro volbu databázových serverů existuje mnoho. Při jeho volbě je dobré přihlédnout k jazyku, ve kterém bude aplikace naprogramována.

V předchozím textu jsme zvolili skriptovací jazyk PHP. V současné době podporuje PHP přímo tyto SQL servery: Adabas D, Informix, Interbase, mSQL, MySQL, Sybase, Oracle, PostgreSQL, Solid, MSSQL a prostřednictvím ODBC teoreticky jakýkoliv jiný SQL server. Více o podporovaných databázích lze nalézt v [27].

Při výběru jsem porovnával nejvhodnější databázové systémy Oracle[17], PostgreSQL[18] a MySQL[19].

Oracle

Oracle by samozřejmě, jako jeden z nejlepších databázových serverů, neměl žádné problémy s touto aplikací. Ale vzhledem k tomu, že systém má být přenositelný a snadno aplikovatelný bez zbytečných požadavků na software a hardware, Oracle není vhodnou volbou. Nasazení takovéto aplikace lze považovat za „chození s kanónem na vrabce".

MySQL

MySQL je jeden z nejrozšířenějších databázových serverů pro projekty tohoto rozsahu. Je dostatečně rychlý pro vyhledávání a je podporován různými jazyky pro web (JSP, ASP .NET, PHP). Nevýhoda MySQL je v tom, že to není klasický Open Source produkt, ale stojí za ním komerční společnost a proto jeho použití není vždy zdarma.

PostgreSQL

PostgreSQL je pravděpodobně nejpokrokovějším open source RDBMS4. Jeho přednostmi je podpora všech moderních operačních systémů včetně os/2 a Novell. Podporu pro PostgreSQL má většina programovacích jazyků - například PHP, Perl, Pascal, Python, libpq nebo Embedded SQL pro C/C++, Java (JDBC) a další. Výkonnostně je na tom PostgreSQL podobně jako další komerční i open source databáze, v něčem je rychlejší, jindy pomalejší. V porovnaní s MySQL a podobnými databázovými systémy, je PostgreSQL rychlejší při víceuživatelském přístupu, složitějších dotazech a zatížení read/write dotazy. MySQL je rychlejší v jednodušších dotazech s malým počtem uživatelů. MySQL navíc nepodporuje mnohé vlastnosti zmíněné v sekci vlastnosti. Dále v MySQL verze 4 nepodporuje vnořené selecty a některé množinové operace [7].

Výběr databázového serveru

Přestože z předchozího textu jasně vyplývá, že nejvhodnějším databázovým serverem je PostgreSQL, protože za vývojem ostatních není Open Source komunita, ale komerční společnosti, které mohou měnit svoje licence apod., po průzkumu nabídky hostingu na našem trhu (především freehostingu) je nejlepší volbou MySQL. MySQL podporují téměř všichni poskytovatelé. Navíc existuje mnoho balíčků, které automaticky nainstalují trojici MySQL + PHP + Apache (někdy je tato trojice označována jako svatá trojice). Dokonce existují i distribuce linuxu, pomocí kterých lze velice snadno vytvořit internetový server s těmito třemi produkty. Takový softwarový bundle bývá označován jako LAMP (Linux + Apache + MySQL + PHP). Příkladem je například distribuce Ubuntu, kde je při instalaci možné zvolit volbu „Instalovat LAMP server".

Přestože je při implementaci použit databázový server MySQL, je velice snadné přizpůsobit aplikaci pro PostgreSQL. Tato změna nám však neumožňuje plně využít pokročilé nástroje, které nám PostgreSQL nabízí.

5.2.3 PEAR

Řešením problému s výběrem databázového serveru může být použití PEAR (PHP Extension and Application Reposiotory). PEAR je framework a distribuční systém knihoven a rozšíření pro aplikace psané v jazyce PHP. PEAR řeší některé užitečné úlohy, jako jsou právě komunikace s databázovým serverem odesílání e-mailů nebo přihlašování uživatelů. Kompletní dokumentaci i balík tříd lze stáhnout z [28]. PEAR obsahuje také jednoduchý manažer tohoto balíku a konfigurační webové rozhraní. S jeho pomocí lze velice snadno PEAR nainstalovat, aktualizovat i konfigurovat. PEAR i s manažerem je oficiální částí PHP od verze 4.3.0.

PEAR vznikl v roce 2000 na základě diskuse na PHP Developers' Meeting. Od té doby urazil poměrně dlouhou cestu. Současný archiv balíčků obsahuje víc než 250 open source knihoven a rozšíření roztříděných tématicky do několika kategorií (např. databáze, souborový systém, HTTP, matematika, text, xml, web services apod.). Archiv doplňuje sofistikovaný distribuční systém, kvalitní dokumentace a návody k samotnému frameworku, mailing-listy a v neposlední řadě IRC kanál sloužící jak uživatelům, tak vývojářům, kteří se snaží o začlenění svých částí kódu do repositáře PEARu.

PEAR definuje čtyři zvláštní skupiny balíčků. Jednou z nich je repositář PECL (PHP Extension Community Library), jehož obsahem je rozšíření pro samotné PHP. PECL definuje vlastní styl psaní kódu, ovšem distribuční systém včetně struktury balíčků sdílí s PEARem. Mimo PECL jsou velmi důležité tzv. PHP Foundation Classes (PFC), což jsou balíčky, které standardní distribuce PHP instalují spolu s nástroji pro práci s archivem. PFC obsahuje pouze stabilní balíčky, které nejsou vázány na konkrétní systém a jsou díky standardizovanému API do budoucna snadno rozšiřitelné. Právě tyto balíčky jsou standardně dodávány s PHP.

Každý balíček, který se stane součástí PEARu musí projít poměrně složitou procedurou, než je do něj začleněn. Takový balíček musel splnit poměrně přísné konvence. Konvence například stanovují, jak bude vypadat kód, ale i co můžeme očekávat od jeho autora. Pokud se někdo rozhodnete rozšířit PEAR o vlastní knihovnu, musí si důkladně prostudovat kompletní dokumentaci na webových stránkách projektu [28].

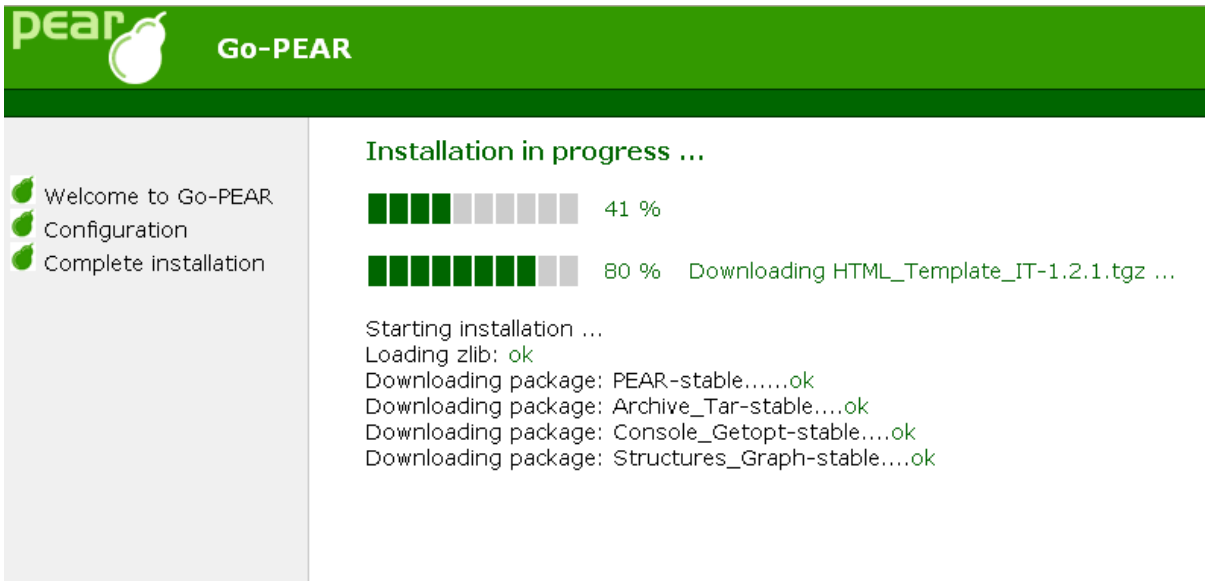
Na začátku vývoje každého balíčku je nutné přesně stanovit, k čemu bude knihovna nebo rozšíření sloužit. Podle toho jej autor musí správně pojmenovat a zařadit do kategorie. Svůj návrh pošle do e-mailové konference a počká na vyjádření, zda se pro PEAR nepřipravuje podobný či stejný balíček. Členové se také vyjádří k tomu, zda vůbec bude nový balíček přínosem. Cílem je, aby se nestalo, že by archiv PEAR obsahoval stejné nebo hodně podobné balíčky. Přesto se však stává, že se některé balíčky ve svých funkcích minimálně částečně překrývají (např. DB a ADODB). Pokud se vývojáři webové aplikace některý balíček hodí, neměl by zapomenout zkontrolovat jeho závislosti na ostatních a zjistit, s jakou licencí je balíček nabízen k použití. Autor se vytvořením zavazuje poskytnout podporu, opravovat nahlášené chyby a také by měl rozšiřovat kód o nové funkce, které uživatelé navrhnou. Pokud se stane, že někdo objeví chybu, měl by ji určitě nahlásit správci balíčku. K tomu účelu slouží formulář umístěný v distribučním systému.

Jedním z požadavků kladených na všechny balíčky je dodržení konvence formátování řídících struktur, názvů proměnných, tříd atd. To znamená, že zdrojový kód musí odpovídat stylu, jehož popis je k dispozici v dokumentaci PEARu. Styl psaní kódu se docela podobá doporučením, která používá

Sunu pro programovací jazyk Java. Balíčky není vhodné přímo upravovat, protože pak přicházíme o možnost snadné aktualizace PEARu.

Autor balíčku je mimo jiné také povinen zveřejnit kompletní dokumentaci zdrojových kódů a to ve formátu Docbook XML nebo ve formátu prostého textu. Dokumentaci je možné najít jak na webu [28], tak také v adresáři, kde je PEAR nainstalován. Instalační dokumentace obvykle představuje jen návod jak balíček nainstalovat a začít používat. V distribučním systému je pro každý balíček podstránka s dokumentací celého API.

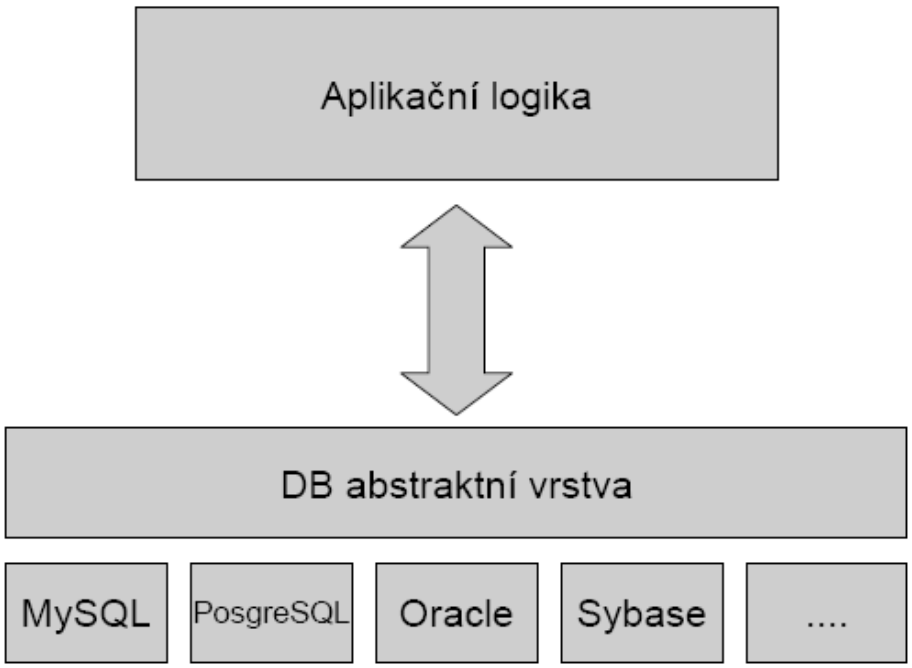
Vývoj balíčku probíhá pod dohledem ostatních vývojářů, kteří také nakonec rozhodují o jeho začlenění do archivu. Jakmile je autor spokojen se současným stavem balíčku, může jej zařadit do hlasovacího procesu. Během hlasování se projekt až na výjimečné případy nesmí měnit a první verze je tak před vydáním dána na milost ostatním. Počet hlasů, který je potřebný pro přijetí balíčku do PEARu je relativně malý (stačí 5 kladných hlasů za 7 dní), nicméně projekt je od svých prvopočátků sledován, takže výsledná kvalita by měla být zřejmá už před hlasováním.



Obr. 5.3 Instalace PEARu pomocí webového rozhraní

Distribuční systém balíčků nabízí mimo funkcí pro získávání a aktualizaci zdrojových kódů také jednotné webové rozhraní pro každý balíček, ve kterém lze nalézt informace o jeho stavu. Je možné si zde také prohlédnout dokumentaci, zjistit informace o týmu vývojářů, kteří jej mají na starosti, nebo nahlásit nalezené chyby. S distribučním systémem se dá pracovat buď přes webové rozhraní (**obr. 5.3**), nebo přes správce balíčků, který je obvykle nainstalován spolu s balíkem PHP. Obě prostředí ovšem obsahují funkce, které nenabízí to druhé, ale základní funkce mají společné. Podrobnosti k instalaci jsou v příloze 2.

Přístup k databázím s PEAR DB



Obr. 5.4 Grafické znázornění funkce DB (převzato z [27])

Typ_databáze	Databáze
fbsql	FrontBase
ibase	InterBase
informix	Informix
msql	Microsoft SQL Server
mssql	SQL
mysql	MySQL
oci8	Oracle 7, 8, 8i
odbc	ODBC
pgsql	PostgreSQL
sqlite	PostgreSQL
sybase	Sysbase

Tab. 5.1 Hodnoty pro typ databáze DSN

Balík PEAR mimo jiné obsahuje také balíček DB. DB je sada metod pro práci s relačními databázemi, bez ohledu na to o jaký druh se jedná. To je velmi zajímavá vlastnost, která PHP chybí. V PHP se totiž při přístupu k databázi používají db-specific funkce. To způsobuje obtížnou migraci na jiný typ databáze. Řešením by bylo udělat jednotné databázové rozhraní, tedy jakousi abstraktní vrstvu pro všechny databáze (**obr. 5.4**). Právě takovým rozhraním je DB.

S pomocí DB můžeme velice snadno migrovat mezi různými databázovými servery. V **tab. 5.1** je uveden seznam všech podporovaných databází a hodnoty, které se doplňují místo typ_db v řádku kódu $dbh = DB::connect('typ_db://user:password@host/database'); v kódu:

```
<?php
// Load the DB code
require 'DB.php';

// Connect to the database
$dbh = DB::connect('typ_db://user:password@host/database');

// Send a SELECT query to the database
$sth = $dbh->query('SELECT flavor, price, calories FROM ice_cream');

// Check if any rows were returned
if ($sth->numRows()) {
   print "<table>";
   print "<tr><th>Ice Cream Flavor</th><th>Price per Serving</th><th>Calories~CCC
 per Serving</th></tr>";
   // Retrieve each row
   while ($row = $sth->fetchRow()) {

      // And print out the elements in the row
      print "<tr><td>$row[0]</td><td>$row[1]</td><td>$row[2]</td></tr>\n";
   }
   print "</table>";
} else {
    print "No results";
}
```

5.2.4 Webový server

Web server je počítač nebo počítačový program, který je odpovědný za vyřizování požadavků HTTP od klientů (webové prohlížeče). Vyřízením požadavku se rozumí odeslání webové stránky klientovi, který ji pak zobrazí.

Jednotlivé webové servery se mohou značně lišit. Presto mají několik společných vlastností. Každý webový server je připojen k počítačové síti a přijímá požadavky od klientů. Tyto požadavky vyřizuje a počítači, který požadavek vznesl, vrací odpovědi. Odpověď obvykle představuje nějaký HTML dokument. Může to být i dokument v jiném formátu - text, obrázek apod.

Webový server má dvě možnosti, jak získávat informace, které vrací klientům:

- Jsou to buď předem připravené datové soubory (HTML stránky), které webový server bez změny odešle klientovi (tzv. statický obsah)
- Na základě požadavku klienta jsou data shromážděna (přečtena ze souboru, databáze, nebo nějakého koncového zařízení), zformátována a připravena k prezentaci u klienta (tzv. dynamický obsah)

K dynamickému vytváření obsahu se používá celá řada technologií (PERL, PHP, ASP, ASP .NET apod.). Statický obsah je schopen server poskytnout významně rychleji než dynamický.

Na druhé straně pomocí dynamického obsahu lze poskytovat mnohem větší obsah informací a lze reagovat i na různé „ad hoc" dotazy klientu. Proto se v praxi v mnoha případech oba způsoby poskytovaní obsahu kombinují

Nejrozšířenějšími webovými servery, které zabezpečují službu webového serveru, jsou:

- Apache HTTP Server
- Internet Information Services
- Sun Java System Web Server

Server Apache

Apache HTTP Server je softwarový webový server s otevřeným kódem pro Linux, BSD, Microsoft Windows a další platformy. V současné době se jedná asi o celosvětově nejrozšířenější aplikaci svého druhu.

Apache je volně šířitelný a je dostupný pro téměř všechny významné platformy. HTTP server Apache je modulární a má 2 základní moduly: httpd_core – což je vlastní jádro webserveru a mod_so který slouží pro dynamické načítání ostatních modulů. Díky tomu má široký rozsah programovacích jazyků na straně serveru, podpory autentifikace a dalších funkcí. Podporuje většinu pro web určených skriptovacích jazyků jako je Perl, Python, Tcl a PHP. Oblíbené moduly autentifikace jsou mod_access, mod_auth a mod_digest (více o nich v části věnované bezpečnosti nebo v [24] a [26]). Příkladem dalších modulu je podpora SSL a TLS (mod_ssl), proxy modul, URL přepisovací engine (modrewrite), podpora filtrovaní (modinclude a mod_extfilter) atd.

Nová verze Apache 2 přináší řadu vylepšení a zvýšení výkonu oproti předchozí verzi. Má schopnost běžet v hybridním thread/process režimu na libovolné platformě, která oba režimy podporuje, podporuje I/O filtrovaní a IPv6 (více [25]).

Internet Information Services

Microsoft Internet Information Services je balík Internetových služeb, který je součástí některých verzí Microsoft Windows. Je to druhý nejpoužívanější webový server na světě. Původně byl dodáván jako součást server edice Windows NT 4.0 a byl součástí i navazujících upgradů. Postupně byl také součástí Windows 2000 server a Windows Server 2003. V případě Visty je IIS dostupný jen v některých verzích - nemůžeme s ním počítat v obou verzích Home (Basic, Premium). Naopak verze Business, Enterprise a Ultimate IIS obsahují.

Verze IIS 6.0 obsahuje servery FTP, SMTP, NNTP a HTTP(S) a je dostupný jen pro Windows Server 2003. Omezená verze IIS 5.1, která podporuje jednu webovou prezentaci a omezené množství připojení, je dodávána s Windows XP Professional. Na uživatele Windows Vista čeká IIS 7.0, který nemá nastaven limit v počtu připojení, ale omezení zatížení aktivních souběžných požadavků. Podpora ASP (Active Server Pages) začala s IIS 3.0. Podpora PHP je odzkoušena od IIS 5.1 (PHP 4.1.3) (více [22]).

Sun Java System Web Server

Sun Java System Web Server (Sun ONE Web Server, který byl dříve znám jako iPlanet Web Server) je bezpečný, spolehlivý a snadno použitelný webový server, který je určen pro střední a velké obchodní aplikace. Je k dispozici pro většinu operačních systémů, Java System Web Server poskytuje vývojové platformy pro webové služby, Java Server Pages (JSP) a technologii Java Servlet, Microsoft Active Server Pages, PHP a CGI.

Výběr webového serveru

Jelikož všechny zmiňované servery mají podporu jazyka PHP, můžeme zvolit kterýkoliv z nich. Z důvodu největšího rozšíření serveru Apache, jsem aplikaci testoval právě na tomto serveru.

5.3 Instalace

Instalace aplikace je velmi jednoduchá. Je třeba pouze nahrát všechny potřebné soubory do příslušné složky webového serveru. Obvykle je to složka „WWW". Poté je třeba upravit soubor `config.php`. Definované konstanty je nutné změnit tímto způsobem:

1) Nejprve je nutné definovat nastavení serveru pro odesílání pošty.

```
define ("SMTP_SERVER", "smtp.iol.cz");  // smtp server
```

2) Pokud server nevyžaduje ověření, nastavíme:

```
define ("SMTP_AUTH", "0");
```

3) Pokud ověření vyžaduje, použijeme následující nastavení:

```
define ("SMTP_AUTH", "1");        // zapnuti autorizace
define ("SMTP_USERNAME", "kix"); // login
define ("SMTP_PASSWORD", "kix"); // heslo
```

4) Dále nastavíme e-mail a jméno, které má být u odesílaných e-mailů jako odesilatel:

```
define ("SMTP_SENDER", "admin@project.cz");
define ("SMTP_SENDER_NAME", "GR8 solutinot");
```

5) Je nutné nadefinovat cestu ke složce, kde je aplikace uložena:

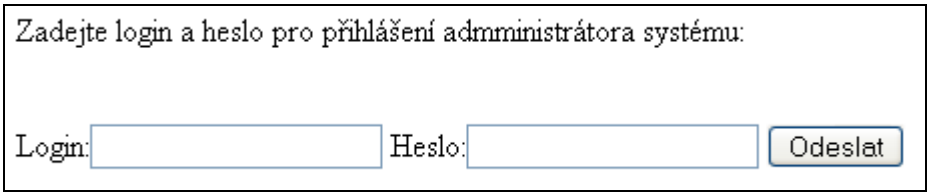
```
define ("SLOZKA","/var/www/moje");
```

6) Posledním krokem je nastavení přihlašovacích údajů k databázi ve tvaru typ_db://login:heslo@host/databáze:

```
define ("SQL", "mysql://root:123456@localhost/moje_db");
```

V další fázi je nutné zajistit, aby mohl webový server zapisovat (vytvářet nové složky a soubory) do složky, kde je uložena naše aplikace.

Pokud máme soubor `config.php` upravený a server může zapisovat do složky s daty, můžeme přistoupit k serveru pomocí webového prohlížeče a to na adresu `adresa_serveru/cesta_k_aplikaci/install.html`. Zobrazí se nám formulář, do kterého je nutné vyplnit přihlašovací údaje pro administrátora (**obr. 5.5**). Administrátor je v systému jen jeden a jeho úlohou je správa uživatelů a projektů. Administrátorem je obvykle ředitel společnosti. Po odeslání formuláře se dokončí instalace a s aplikací je možné začít pracovat.



Obr. 5.5 Instalace aplikace – vytvoření administrátorského účtu

5.4 Struktura aplikace

U většiny aplikací musí administrátor vytvořit všechny uživatelské účty, vyžaduje to od něj poměrně náročné zadávání údajů při vkládání každého nového uživatele. Naše aplikace tento problém řeší tím, že se uživatel zaregistruje sám, ale dokud není jeho účet schválen administrátorem, není mu umožněno přihlásit se. Administrátor je o každém novém uživateli informován emailem.

Administrátor si ověří údaje a pravdivost registrace, poté nový uživatelský účet zařadí do některé ze dvou kategorií (Zaměstnanec nebo Klient) (**obr. 5.6**). Dokud není účet zařazen, uživatel se nemůže přihlásit.

ID	Login	Jm?no	P?ijmeni	E-mail	Telefon	Firma	Role	
64	mix	Miroslav	Jelínek	mije2@seznam.cz	606492971		klient	Odstranit
66	dix	David	Jelínek	daje2@seznam.cz	606492966		pracovnik	Odstranit
65	tix	Tomáš	Jelínek	toje2@seznam.cz	606492965			Odstranit

Obr. 5.6 Práce s aplikací– administrace uživatelů

Pokud je uživateli přiděleno jedno ze dvou práv, může se přihlásit do aplikace a začít pracovat. Na **obr. 5.7** je znázorněn postup práce s aplikací. Zákazník (klient) potřebuje něco vyrobit. Zadá proto do systému informace o své zakázce a tím vlastně založí nový projekt. O založení projektu je informován administrátor aplikace. Ten projektu přiřadí vedoucího (někdo ze zaměstnanců). Vedoucí projektu je opět informován e-mailem o přidělení projektu.

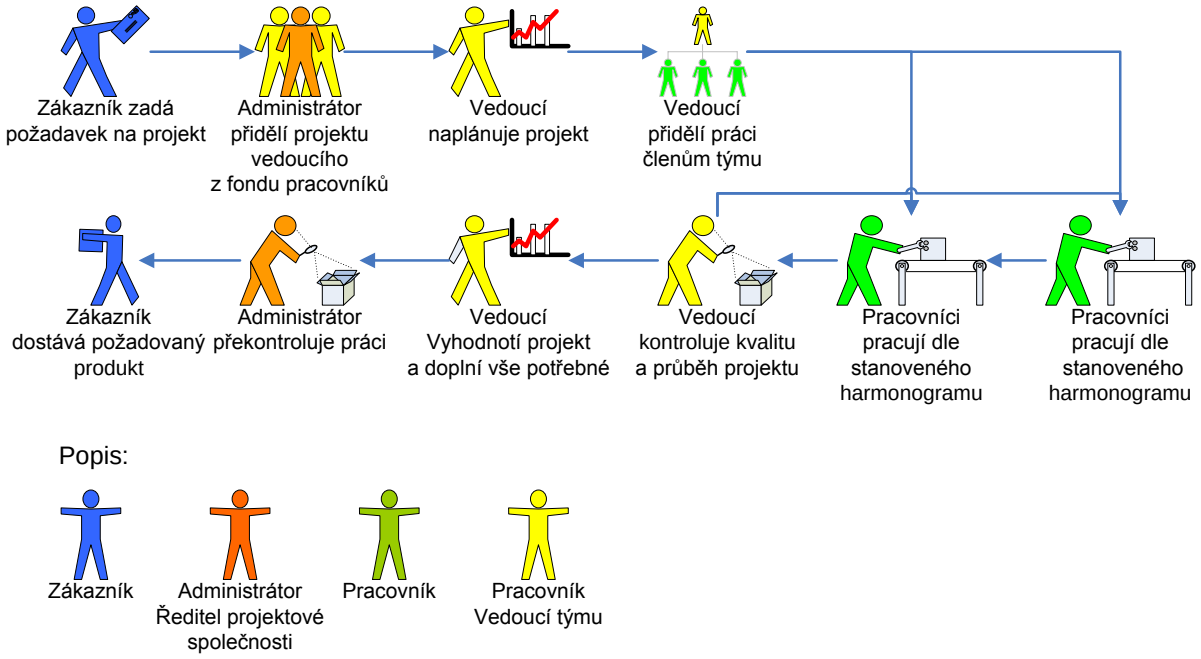
Po obdržení e-mailu o tom, že se stává vedoucím nějakého projektu, začne projekt plánovat:

- Se zákazníkem domluví finální specifikaci výsledného produktu.
- Vytvoří plán prací
- Přidělí zdroje
- Komunikuje se členy svého týmu, se zákazníkem i administrátorem

Následuje samotná práce na produktu. Vedoucí projektu kontroluje ostatní pracovníky a řeší případnou nekvalitní práci. V případě, že nastane nějaký problém, tak jej řeší. Celý tým mezi sebou

může komunikovat pomocí projektového chatu a sdílet soubory s daty (soubory jsou ukládány na server).

Po ukončení vývoje produktu je vše zhodnoceno a předáno zákazníkovi. Ten vyplní formulář, ve kterém uvede informace o své spokojenosti s výsledným produktem.



Obr. 5.7 Postup práce na projektu

Podrobné informace o ovládání aplikace jsou v uživatelském manuálu na přiloženém CD.

5.5 Bezpečnost

Problematika zabezpečení provozu a dat uložených v informačních systémech je širokou oblastí, která pokrývá nejen použité programové vybavení, ale také hardware včetně síťových prostředků a organizace práce. Bezpečnost bývá při hodnocení a srovnávání informačních systémů považována za jednu z nejdůležitějších oblastí, protože málokdo je ochoten přijít o mnohdy velice důležitá data. Snahy o zabezpečení informačního systému při jeho vývoji zpravidla začínají již při návrhu systému. Od samého počátku je nutné navrhovat systém tak, aby byla data chráněna nejen před ztrátou, ale je také nutné zajistit, aby se nedostala třeba do rukou konkurence.

5.5.1 Analýza možných útoků

V následujícím textu je podrobně vysvětleno, jaká bezpečnostní rizika se mohou týkat aplikace pro řízení a plánování projektů.

Nejprve je nutné si uvědomit, proti čemu se vlastně bráníme a kdo jsou naši nepřátelé (rizika a útočníci). Základní typy útoků podle [29] jsou tyto:

1) **Vetřelec se pokouší získat naše data** – Vetřelec se může dostat k důvěrným datům třeba o vývoji nového produktu, který připravujeme a předat tyto informace naší konkurenci. Může se také dostat ke jménům a adresám našich zákazníků i k informacím o bankovních účtech

a pokud nebudeme dost opatrní, možná i k jejich obsahu. Právě zveřejněním a zneužitím získaných dat dochází k největším škodám. To, že se o našich aktivitách a připravovaných službách a výrobcích dozví jeden vetřelec, ještě nemusí být tak závažné. Pokud ale tyto informace někde zveřejní a dostane se k nim konkurence, stane se pro nás tato ztráta informací velkým problémem.

2) **Vetřelec provádí změny v datech** – Tento typ útoku je asi nejhorší, znamená pro nás totiž okamžitě velké škody. Vetřelec může třeba změnit plány a informace pro vývoj budoucích služeb a výrobků. Navíc se může velice snadno stát, že si této úmyslné zlomyslné změny dlouhou dobu nikdo nevšimne. Může se také stát, že vetřelec vůbec netuší, jak velkou škodu může způsobit
3) **Vetřelec maže data** – Zde jsou důsledky útoku jasné každému. Můžeme je omezit tak, že data budeme důsledně a správně zálohovat. I zde může nastat problém pozdního objevení takového útoku.
4) **Vetřelec se snaží o odepření služby (DoS útoky)** – U tohoto útoku se vetřelec snaží způsobit nedostupnost nebo alespoň o snížení dostupnosti dat přetížením sítě nebo serveru.
5) **Vetřelec se snaží získané skutečnosti použít k dalším útokům** – Pokud vetřelec získá přístup na cizí server, může toho využít např. při DoS útoku na nějaký další server.

Každý z těchto typů útoku nám může způsobit velké problémy a může vést třeba i ke krachu firmy apod.

Abychom mohli dobře analyzovat možná rizika, je dobré si uvědomit, kdo by mohl na náš systém zaútočit. Podle [29] to mohou být:

1) **Crackeři a hackeři** – Do této skupiny patří jak lidé, kteří nemají v úmyslu udělat něco špatného, tak ti, kteří tuto činnost dělají vědomě a s úmyslem někoho poškodit. Jejich útoky se objevují v podstatě nahodile a tato skupina útočníků se většinou snaží zviditelnit. Právě proto se snaží napadat systémy významných firem, kde budou následky jejich útoku „na očích" velké skupiny lidí (zákazníků těchto firem).
2) **Nespokojení současní zaměstnanci** – Také tyto útoky se dají velice obtížně předvídat. Pachatelé se ale dají snadno nalézt při správném auditu ve společnosti. Dopadení jednoho takového pracovníka způsobí strach u ostatních. Tím lze na nějakou dobu minimalizovat pravděpodobnost dalšího podobného útoku. Obranou proti takovýmto útokům je časté zálohování.
3) **Nespokojení bývalí zaměstnanci** – Většina takových útoků se dá předvídat a důsledným dodržováním určitých pravidel jim lze předejít. Při odchodu zaměstnance z firmy je nutné včas a důsledně zrušit všechna jeho přístupová oprávnění k systému. O zákazu přístupu by také měli být informováni všichni zaměstnanci, aby nevědomky neposkytli bývalému zaměstnanci přístup.
4) **Konkurence** - Ostatní firmy pracující ve stejném oboru se samozřejmě snaží monitorovat své konkurenty a získat o nich co nejvíce informací. Zajímají je nové nápady, připravované výrobky a služby, ale i informace, na jejichž základě by mohly přetáhnout zákazníky. Podobným způsobem se ale nemusí chovat jen firmy, které se snaží zjistit informace o konkurenci. Příkladem mohou být například tzv. lovci mozků (headhunters) z personálních agentur, kteří se naopak snaží přetáhnout nejschopnější zaměstnance.
5) **Špioni** – Do této skupiny útočníků patří ti, kteří se zabývají např. tzv. průmyslovou špionáží.

6) **Kriminálnicí, extrémisté a teroristé** – Na rozdíl od první skupiny útočníků, kteří nemají žádné vyšší cíle, se tato skupina snaží buď o vlastní obohacení nebo tuto činnost provádějí např. z morálního přesvědčení (např. boj proti globalizaci).

Přestože se zdá, že otázkami bezpečnosti se zabývám příliš podrobně, opak je pravdou. Předchozí text měl být pouze zamyšlením nad tím, jak by mohlo dopadnout podcenění bezpečnostních opatření. Musíme si uvědomit, že právě vývoj a výzkum jsou například právě pro konkurenci velice lákavé činnosti. Mnohdy konkurence neváhá investovat nemalé finanční prostředky k získaní informací o těchto činnostech a nezdráhá se použít i protizákonné praktiky.

5.5.2 Zabezpečení aplikace pomocí HTTP

Při zabečení aplikace máme mnoho možností. Můžeme využít jednoduchého zabezpečení pomocí PHP (např. s využitím session), můžeme využít možností, které nám nabízí webový server Apache nebo můžeme využít dalších možností (např. VPN apod.).

Jak je uvedeno výše, aplikace pro řízení projektů má velké požadavky na zabezpečení, proto není vhodné spoléhat na poměrně slabé řešení pomocí PHP. Je mnohem vhodnější využít možnosti, které nám nabízí webový server Apache. Ten již nějakou dobu podporuje autentizaci pomocí protokolu HTTP.

Základní autentizace pomocí protokolu http je poměrně snadná, k autentizaci uživatele se používá mechanizmus výzvy a odpovědi. Autentizace probíhá takto:

1) Autentizace začne v okamžiku, kdy webový prohlížeč požádá server o chráněný lokátor URL.
2) Webový server vrací v odpovědi hlavičku 401 společně s WWW-Autenticate. To znamená, že pro přístup na danou URL požaduje autentizaci.
3) Prohlížeč zobrazí uživateli okno pro zadání přihlašovacích údajů.
4) Uživatel zadá jméno a heslo a stiskne OK. Prohlížeč odešle na server jméno a heslo. Server zkontroluje, jestli jsou zadané údaje platné.
5) Pokud se jméno s heslem shoduje s údaji osoby, která má na danou URL přístup, server pošle prohlížeči požadovanou stránku. Pokud údaje platné nejsou, server postupuje podle znovu od bodu 2).

Použitý modul `mod_auth` je standardním autentizačním modulem Apache. Lze pomocí něj autentizovat uživatele, jejichž přihlašovací údaje jsou v textovém souboru ve tvaru:

```
Uživatel:zašifrované_heslo
```

Autentizace pomocí textového souboru se doporučuje maximálně pro několik tisíc uživatelů, což plně vyhovuje požadavkům aplikace pro projektové řízení. Velkou výhodou tohoto způsobu autentizace je také to, že jej podporuje většina webhostingu a není těžké jej nastavit ani na „čistě" nainstalovaném serveru (mnohdy je vše nastaveno hned po nainstalování Apache).

Zabezpečení přístupu k HTML stránkám, které jsou na určité URL se provede tak, ze se do složky (adresáře), kde se stránky nacházejí vloží soubor s názvem `.htaccess` (pokud není v konfiguračním souboru apache nastaven jiný název). Ten bude označovat tuto oblast jako chráněnou. Soubor bude obsahovat například tento text:

```
AuthName "Chranena oblast"
AuthType Basic
```

```
AuthUserFile /home/hesla/.htpasswd
require valid-user
```

Tento soubor říká serveru, že přístup k souborům v adresáři kde je uložen je autorizován dle údajů v souboru `.htpasswd` (soubor s hesly). Při pokusu o zobrazení nějaké stránky z podstromu kde je soubor `.htaccess` vypíše prohlížeč dotaz na jméno a heslo opravňující návštěvníky k přístupu do Chráněné oblasti.

Příklad obsahu souboru `.htpasswd`:

```
bond:Xy9KgHmOmCESc
trubka:qWKFFmkF7LPjQ
```

Přihlásí-li se tedy někdo do chráněné oblast jménem bond s heslem James nebo pod jménem trubka a s heslem dcs.48, budou mu požadované stránky zobrazeny. Soubor s hesly musí být přístupný pro WWW server. To znamená, že musí být čitelný (v našem případě musí mít i práva pro zápis, aby se dala měnit hesla a přidávat uživatelé) pro uživatele www.

O vytváření souborů `.htaccess` se nemusíme starat, pokud jsme webovému serveru umožnili zápis do složky s aplikací, vytvoří si aplikace sama příslušné soubory při instalaci.

5.5.3 Zabezpečení pomocí HTTPS

Zabezpečení pomocí protokolu HTTP je nedostatečné, umožňuje totiž snadné odposlechnutí hesla, které je zasílané v nezabezpečené formě. HTTPS je vylepšení protokolu HTTP, která poskytuje zvýšenou bezpečnost před odposloucháváním či podvržením dat. HTTPS není zvláštní protokol, data jsou přenášena pomocí standardního http. Nejsou však přenášena běžným způsobem, ale jsou šifrována pomocí SSL nebo TLS. To zaručuje ochranu proti packet-sniffingu i man-in-the-middle útokům. HTTPS komunikuje implicitně (pokud není nastaveno jinak) prostřednictvím TCP portu 443 (u nechráněného HTTP je to port 80).

Pro komunikaci pomocí HTTPS musí server vlastnit certifikát. Certifikáty můžou být vytvořeny například pomocí nástroje ssl-ca z balíku OpenSSL. Certifikát by měl být podepsán tzv. certifikační autoritou, která zaručí, že vlastník certifikátu se nevydává za nikoho jiného. Webové prohlížeče jsou většinou vybaveny podpisovými certifikáty největších podpisových autorit (např. VeriSign), ale není to nutnost. Je možné si také udělat vlastní certifikační autoritu například pro pobočku firmy, případně pro celou firmu. Podrobnosti o nastavení HTTPS u Apache jsou v [26].

5.6 Dosažené výsledky

Protože moje aplikace byla vyvíjena z velké části až v letním semestru, bylo by nutné ji před „ostrým nasazením" ještě dopracovat a některé části částečně přepracovat. Také by bylo vhodné delší testování, aby se objevilo co nejvíce chyb před „ostrým nasazením".

Aplikace si neklade za cíl splnit požadavky velkých společností. S časem, který jsem této aplikaci mohl věnovat, to ani není možné. Od samého začátku byla aplikace navrhována tak, aby umožňovala jednoduché plánování tam, kde se dosud žádný podobný software nepoužívá. Není tedy určena pro zaběhlé společnosti, které se zabývají projektovým řízením již mnoho let.

Primární určení aplikace je tam, kde nejsou finanční prostředky na nákup drahého software, jako jsou například různé nadšenecké projekty z oblasti vývoje software, ekologie, modelářství, cestovního ruchu, dobrovolného záchranářství atd.

Příkladem projektu pro tuto aplikaci může být například generální oprava hasičského automobilu u sboru dobrovolných hasičů. Velitel nebo starosta založí projekt na opravu automobilu, jako vedoucího projektu přidělí některého z velitelů družstev. Ten společně se svým družstvem vytvoří plán projektu a podle něj pak celé družstvo postupuje. Opravu provádí částečně svépomocí, částečně objednají u specializované firmy. Vedoucí projektu postup prací koordinují a dohlíží na kvalitu a termíny. Tuto aplikaci ale nemusejí využívat jen pro plánování prací na technice. Mohou ji využívat také třeba při plánování přípravy na soutěže nebo při přípravě společenských akcí. Aplikace se hodí pro jakoukoli strukturovanou činnost, kterou je dobré předem naplánovat.

Cílem bylo vytvořit aplikaci jednoduchou, snadno a intuitativně ovladatelnou, před jejímž použitím není nutné dlouze studovat rozsáhlý manuál. S tím souvisí i snaha minimalizovat funkce. V ostatních podobných aplikacích je mnoho funkcí, které většina uživatelů nikdy nepoužije. Tyto funkce pouze komplikují ovládání těchto aplikací.

V následující kapitole je podrobně rozepsáno, co by bylo ještě vhodné dodělat, případně opravit nebo upravit.

6 Závěr

6.1 Možnosti dalšího vývoje

Programová podpora pro plánování projektů by měla zjednodušit práci projektového managera i všech členů projektového týmu. Aplikace vytvořená v rámci tohoto projektu je sice plně funkční, pro praktické využití by však měla být rozšířena a měly by být opraveny některé nedostatky.

Bylo by vhodné zkvalitnit uživatelské rozhraní, vylepšit vzhled aplikace, přidat možnost přizpůsobení konkrétnímu uživateli atd. Velkou nevýhodou u současné webové aplikace je dlouhá reakce při komunikaci klienta se serverem, který vyhodnocuje a zpracovává informace od uživatele. Jedním z řešení by bylo použití technologie AJAX [4].

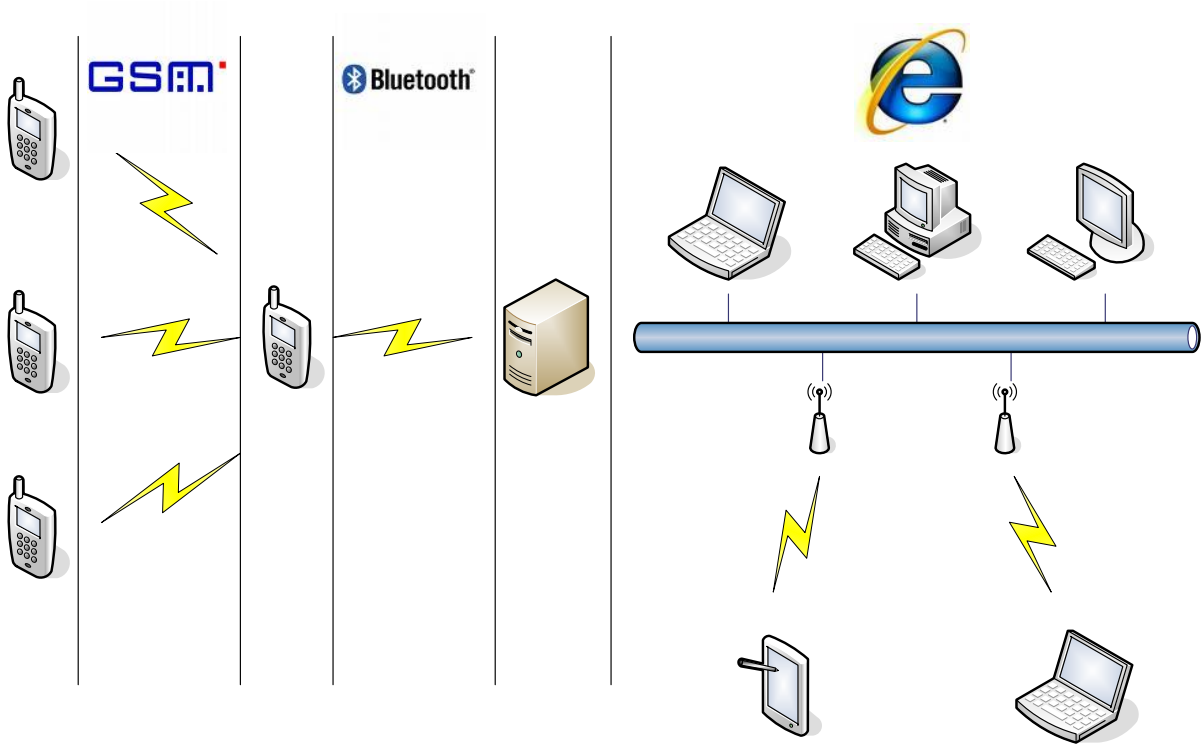
Největší výhodou technologie AJAX je právě odstranění nutnosti znovunačtení a překreslení celé stránky při každé operaci, kterému se u klasického modelu WWW stránek nevyhneme. Pokud například uživatel klikne na tlačítko pro vyhodnocení formuláře, celá stránka se musí znovu načíst ze serveru, přestože se na ní aktualizuje jen jediný údaj a zbytek obsahu zůstává stejný. Prostřednictvím AJAXu proběhne odeslání dat od uživatele na pozadí a server zašle jen ty části stránky, které se změnily, a jen tyto části se uživateli na stránce aktualizují a překreslí. Uživatel tak může pracovat mnohem plynuleji a rychleji, téměř jako by pracoval s běžnou desktopovou aplikací. Tento přístup také snižuje zátěž webového serveru, protože nemusí při každém požadavku sestavit celý HTML dokument, ale pracuje pouze s daty, které se na stránce mění. To také výrazně snižuje zatížení sítě a databáze, ze které se nemusí stále dokola načítat stejná data.

Nevýhodou AJAXu je nutnost používat relativně moderní prohlížeče WWW stránek, které podporují potřebné technologie. Všechny dnes běžné prohlížeče však tyto technologie alespoň v základu podporují a pokud chce uživatel rozumně pracovat s internetem, tak již dávno takový prohlížeč používá.

Další možností, jak vylepšit práci na projektu, je místo e-mailů vytvořit k aplikaci přístup pomocí mobilního telefonu. I v dnešní době existují lidé, kteří nemají přístup k počítači s internetem, většina z nich ale má k dispozici mobilní telefon. Proto by bylo vhodné umožnit jim přístup k aplikaci pomocí mobilního telefonu, např. pomocí WAPu. Také by bylo dobré umožnit těmto uživatelům zasílání informací pomocí SMS na jejich mobilní telefon. Příklad takového řešení je na **obr. 6.1.** K serveru (uprostřed) je připojen mobilní telefon nebo GSM modul (v našem případě se jedná o mobilní telefon, který se serverem komunikuje pomocí bluetooth).

Použití GSM modulu je obvykle nejdražší řešení, je to však řešení kompletní a spolehlivé. Druhou možností je připojit k serveru standardní mobilní telefon s hardwarovým modemem pomocí datového kabelu. U tohoto řešení může nastat problém s napájením mobilního telefonu. Při komunikaci serveru s telefonem můžeme také využít bezdrátové propojení pomocí bluetooth. Toto řešení je asi nejlevnější a nejjednodušší na realizaci. Jeho velkou nevýhodou je však bezpečnost bezdrátového spoje.

Poslední možností je využít SMS bran mobilních operátorů. Toto řešení je však problematické v tom, že operátor může kdykoliv změnit URL umístění svojí brány.



Obr. 6.1 Schéma komunikace

6.2 Shrnutí

V teoretické části této práce jsem vysvětlil několik základních pojmů z projektového řízení, životní cyklus projektu a zmínil jsem se také o některých nástrojích a technikách využívaných při plánování projektů.

V další části jsem představil některé komerční i nekomerční nástroje pro plánování a řízení projektů.

V poslední části jsem se věnoval návrhu a implementaci aplikace pro podporu projektového řízení. Nejprve vznikl návrh pomocí jazyka UML, podle tohoto návrhu byl potom vytvořen samotný program. Mým cílem bylo vyzkoušet si i některé nové a pro mě dosud neznámé věci. Příkladem může být například kolekce PHP tříd PEAR, která umožňuje snadné řešení různých jinak poměrně složitě řešitelných problémů.

V závěrečné kapitole jsou uvedeny některé možnosti a návrhy na další vylepšení. Pokud se je povede v budoucnu dokončit, mohl by vzniknout ojedinělý produkt, který nemá v kategorii nekomerčního software konkurenci.

Literatura

[1] ISO 10006, Management jakosti – Směrnice jakosti v managementu projektu, [online]. 1999 [cit. 2007-1-1]. Dostupné z: <http://www.pmpartners.com/resources/iso10006.html> .

[2] Rosenau, M. D.: Řízení projektů, Computer Press, Brno, 2003, ISBN 80-7226-218-1.

[3] Galbavý, L.: Diplomová práce: Řízení projektů, FIT VUT, Brno, 2006.

[4] Darie, C., Brinzarea, B., Cherecheş-Toşa, F., Bucica, M.: AJAX a PHP tvoříme interaktivní webové aplikace PROFESIONÁLNĚ, Brno, Zoner Press, 2006, ISBN 80-86815-47-1.

[5] Wikipedia, otevřená encyklopedie [online]. [cit. 2007-1-1]. Dostupné z: <www.wikipedia.org>.

[6] McCarthy, J.: Softwarové projekty, Computer Press, Praha, 1999, ISBN 80-7226-164-0.

[7] Maslakowski, M.: Naučte se MySQL za 21 dní, Computer Press, Praha, 2001, ISBN 80-7226-448-6.

[8] Arlow, J., Neustadt, I.: UML 2 a unifikovaný proces vývoje aplikací, Computer Press, Praha, 2007, ISBN: 978-80-251-1503-9.

[9] Cederqvist, P.: Version Management with CVS: For CVS 1.12.13 [online]. 1993-2005 [cit. 2006-03-28]. Dostupné z URL: <http://ftp.gnu.org/non-gnu/cvs/source/feature/1.12.13/cederqvist-1.12.13.pdf>.

[10] Němec, V.: Projektový management, Grada, 2002. ISBN 80-247-0392-0.

[11] Veber, J. a kol.: Management. Management Press, Praha, 2001. ISBN 80-7261-029-5.

[12] Coombs, P.: IT Project Estimation A Practical Guide to the Costing of Software, Cambridge University Press, London, 2003, ISBN 0-521-53285-X.

[13] Nokes, S., Major, I., Greenwood, A., Goodman, M.: The Definitive Guide to Project Management, Financial Times Prentice Hall, Glasgow, 2003, ISBN 0-273-66397-6.

[14] Bartes,F. Řízení jakosti. VUT v Brně, Fakulta podnikatelská. Brno, 2004, ISBN 80-86510-92-1

[15] Lasák, P.: Projektový management, [online]. [cit. 2007-1-1]. Dostupné z URL: http://www.lasakovi.com/pavel/dovednosti/projektovy-management.

[16] Kreslíková, J., Martínek, Z.: Management projektů MPR (Studijní opora), FIT VUT, Brno, 2007.

[17] Oracle [online]. [cit. 2007-1-1]. Dostupné z: <http://www.oracle.com>.

[18] Dokumentace a instalace PostgreSQL [online]. [cit. 2007-1-1]. Dostupné z: <http://www.postgresql.org/>.

[19] MySQL [online]. [cit. 2007-1-1]. Dostupné z: <http://www.mysql.com>.

[20] Gilmore J., W.: Velká kniha PHP a MySQL 5 - kompendium znalostí pro začátečníky i profesionály, Zoner press, Brno, 2005, ISBN 80-86815-20-X.

[21] Schlossnagle, G.: Pokročilé programování v PHP 5, Zoner press, Brno, 2004, ISBN 80-86815-14-5.

[22] Jesus, C., Harish, R., Sascha, S, Chris, S., Deepak, V.: Programujeme PHP profesionálně, Computer Press, Praha, 2001, ISBN 80-7226-310-2.

[23] DeLisle, M,.: phpMyAdmin - Efektivní správa MySQL, Zoner Press, Brno, 2004, ISBN 80-86815-09-9.

[24] Kolektiv autorů: APACHE & PHP/FI, Neokokorex spol. s.r.o., Praha, 1998, ISBN 80-902230-2-8.

[25] Satrapa, P.: IPv6, Neokokorex spol. s.r.o., Praha, 2002, ISBN 80-86330-10-9.

[26] Kabir,M., J.: Apache server 2 Kompletní příručka administrátora, Computer Press, Brno, 2004, ISBN 80-251-0319-6.

[27] Sklar, D.: PHP - moduly, rozšíření a akcelerátory, Zoner press, Brno, 2005, ISBN 80-86815-19-6.

[28] PEAR [online]. [cit. 2007-2-1]. Dostupné z: <http://pear.php.net/>.

[29] Toxen, B.: Bezpečnost v Linuxu - Prevence a odvracení napadení systému, Computer press, Brno, 2003, ISBN 80-7226-716-7.

[30] ISO 9000, [online]. 2000 [cit. 2007-4-1]. Dostupné z: <http://www.pmpartners.com/resources/iso9000.html>

Seznam příloh

Příloha 1. CD
Příloha 2. Instalace PEAR

Příloha 1

Obsah CD

- Doc\ - elektronická verze tohoto textu, uživatelský manuál
- Prog\ - kompletní verze aplikace
- SQL\ - data pro vytvoření databáze

Příloha 2

Instalace PEAR

Jak je uvedeno výše, instalaci lze provést několika způsoby.

Instalace provedená ve správci balíčků

Jak je uvedeno výše PHP ve verzi 4.3.0 a vyšší standardně obsahuje PEAR i správce balíčků, (pokud nebyla při překladu použita volba without-pear). Pokud PHP správce balíčků přesto neobsahuje, je možné jej nainstalovat ručně.
V Unixovém shellu stačí zadat:

```
lynx -source http://go-pear.org/ | php
```

Tímto příkazem se provede stažení instalačního skriptu z URL http://go-pear.org/, skript stáhne PEAR, nakonfiguruje jej a nainstaluje. Pokud nemáme linx, můžeme PEAR nainstalovat tak, že v libovolném webovém prohlížeči stáhneme kód stránky http://go-pear.org/, uložíme jej do souboru /tmp/go-pear.bat a spustíme jej:

```
php /tmp/go-pear.php
```

V MS Windows lze pear nainstalovat pomocí dávkového souboru go-pear.bat:

```
C:\> c:\php\go-pear.bat
```

Jakmile je správce nainstalován, můžete s ním začít pracovat. Seznam argumentů spouštěcího příkazu je výstupem skriptu bez parametrů, tzn. stačí do příkazového řádku napsat:

pear

Seznam dostupných balíčků, které jsou součástí nainstalovaného PEARu, lze zobrazit příkazem:

pear remote-list

V okně terminálů se zobrazí názvy všech balíčků a čísla jejich verzí. Balíček, který chceme nainstalovat, ovšem není zobrazen. Je to dáno tím. Že chování správce ovlivňují konfigurační proměnné a jednou z nich je i preferred_state, která určuje, jakou verzi (alpha, beta, devel, stable, snapshot) chceme přednostně instalovat. Určitě šikovná volba, nicméně ovlivní i seznam balíčků, který se zobrazí (vyhledávání pomocí pear search neovlivňuje).

Instalace v prostředí webhostingu

K webhostingu zpravidla nedostanete přístup k shellu a instalaci tedy musíte provést ručně. Není to však nic obtížného. Stačí stáhnout zdrojové kódy vybraného balíčku a přes FTP je uploadujte např. do adresáře /include ve webovém prostoru. Ve skriptech se musí pomocí funkce *ini_set* nastavit proměnná "include_path" na /include.

Tento způsob instalace s sebou nese malý nedostatek. Je třeba ověřovat závislosti a "nainstalovat" také závislé balíčky. Samotný balíček PEAR definuje pro PHP4 také často používané speciální funkce, které fungují jako destruktory. Další informace k instalaci lze nalézt na webu [27].