Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra systémového inženýrství

Diplomová práce

Vývoj aplikace pro řešení metod VERT a PERT

Daniel Chamrada

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Daniel Chamrada

Projektové řízení

Název práce

Vývoj aplikace pro řešení metod VERT a PERT

Název anglicky

Development of application for solving methods VERT and PERT

Cíle práce

Cílem této práce je vytvoření jednoduché aplikace, které bude možné využít k řízení projektů menšího rozsahu. Unikátnost této aplikace bude spočívat především v kompatibilitě s jednotlivými projektovými metodami. Kromě všeobecně známé metody PERT (Program Evaluation and Review Technique) totiž dokáže aplikace vyřešit zadané úlohy i za použití metody VERT (Venture Evaluation and Review Technique).
Smyslem ani cílem této práce není vytvořit profesionální aplikaci, která by obstála na dnešním extrémně konkurenčním trhu, jež nabízí bezpočet produktů pro efektivní řízení projektů a na jejímž vývoji by se musela podílet řada odborníků a investorů. Aplikace by ve své finální podobě měla sloužit především v akademické sféře – pro snazší pochopení projektové metody VERT, případně pro porovnání jednotlivých matematických přístupů při řešení vybraného ekonomického problému z hlediska efektivity řešení.

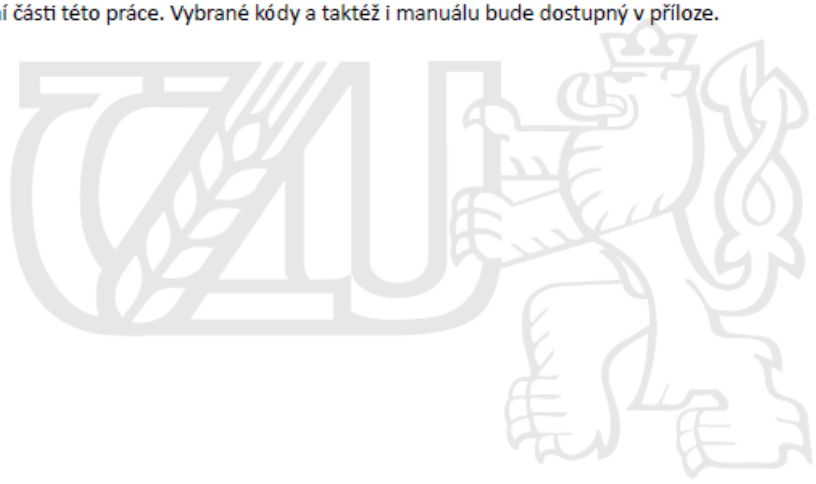
Metodika

Pro vypracování této diplomové práce bude nezbytné nastudovat z odborné literatury dostatek informací o zmíněných metodách PERT a VERT. Především u druhé zmíněné metody je literatura značně limitována co do množství, aktuálnosti a dostupnosti, o jazykové omezenosti nemluvě. Dále bude nezbytné prostudovat dostatek materiálů o jednoduchém programování. Pro tyto účely je diplomová práce hierarchicky rozdělena do celkem šesti navazujících kapitol, ve kterých se čtenář postupně seznamuje se surovou teorií řízením projektů, jeho metodami, simulacemi, programováním i na závěr finálním seznámení s hotovou aplikací.

Prvním krokem je získávání literatury a dalších relevantních zdrojů pro studium a zpracování literární rešerše. Následně proběhne studium další disciplíny, nezbytné pro zpracování této práce, a tedy programování. Finální aplikace bude naprogramována v jazyce JAVA.

Po důkladném prostudování všech materiálů budou nejprve postupně zpracovány první čtyři kapitol, tedy teoretická část závěrečné práce. Ta bude obsahovat ve své první části ucelený přehled teoretických znalostí o projektovém řízení, ve své druhé části bude následována informacemi o projektových metodách a jejich aplikaci. Vzhledem k hloubkové analýze metody VERT bude tomuto analytickému přístupu ponechána celá následující kapitoly, což by mělo stačit pro dostatečné vysvětlení tohoto přístupu, včetně retrospektivního pohledu do historie, na modifikace dané metody a také na možnosti jejího praktického použití v současném obchodně orientovaném světě.

S postupným vývojem aplikace (dílčími kroky) se začne již během studia tohoto programovacího jazyka. Veškeré významné postupy budou upřesněny a komentovány v páté kapitole, tedy praktické části této práce a jednotlivé možnosti ovládání této aplikace včetně ukázkové úlohy se stanou součástí šesté kapitoly a tedy poslední části této práce. Vybrané kódy a taktéž i manuálu bude dostupný v příloze.



Doporučený rozsah práce

60 – 80 str.

Klíčová slova

vert, pert, aplikace, teorie síťové analýzy, řízení projektů, stochastické metody

---

Doporučené zdroje informací

FÁBRY, Jan. Matematické modelování. Praha: Oeconomica, 2007. ISBN 978-80-245-1266-2.

FIALA, Petr. Řízení projektů. Vyd. 2., přeprac. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0.

HOUŠKA, Milan. Simulační modely I. V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X.

KERZNER, Harold. Applied project management: best practices on implementation. New York: Wiley, c2000. ISBN 0-471-36352-9.

KOEGH, James. Java bez předchozích znalostí: průvodce pro samouky. Brno: Computer Press, 2005. ISBN 80-251-0839-2.

LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7.

MOELLER, Gerald L. a Lester A. DIGMAN. Operations Planning with VERT. Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. – Aug., 1981), 676-697.

MOELLER, Gerald L. Venture Evaluation and Review Technique: Decision Models Directorate, US Army Armament Material Readiness Command. Rock Island, Illinois, 1979.

ŠUBRT, Tomáš a Jan BARTOŠKA. Projektové řízení: (měkké a pokročilé přístupy). V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2007. ISBN 978-80-213-1725-3.

ŠUBRT, Tomáš a Pavlína LANGROVÁ. Projektové řízení: (základy a matematické metody). Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0.

Předběžný termín obhajoby

2016/17 LS – PEF

Vedoucí práce

doc. Ing. Tomáš Šubrt, Ph.D.

Garantující pracoviště

Katedra systémového inženýrství

Elektronicky schváleno dne 28. 2. 2017

doc. Ing. Tomáš Šubrt, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 3. 3. 2017

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 03. 03. 2017

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Vývoj aplikace pro řešení metod VERT a PERT" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 31.3.2017 _____

Poděkování

Rád bych touto cestou poděkoval panu doc. Ing. Tomáši Šubrtovi, Ph.D. za cenné rady a připomínky k mé diplomové práci, které ji výrazně obohatily. Dále bych chtěl poděkovat svému konzultantovi, Bc. Pavlu Malému, za jeho pomoc při vývoji aplikace. Poděkování patří také partnerce a celé mé rodině za soustavnou podporu ve studiu.

Vývoj aplikace pro řešení metod VERT a PERT

Souhrn

Tato diplomová práce se věnuje vývoji jednoduché aplikace, která poslouží jako univerzální nástroj pro řešení sofistikovanějších úloh projektového řízení prostřednictvím analytických metod VERT a PERT, které jsou vhodné pro řešení úloh se stochastickou délkou trvání jednotlivých činností.

V teoretické části práce se autor obsáhle věnuje teoretické stránce projektového řízení. Pokračuje charakteristikou nejčastějších metod a detailnějším rozborem stochastické metody VERT, která byla vyvinuta v 70. letech 20. století a která je v takovémto rozsahu vysvětlena v českém jazyce vůbec poprvé, ve své více než 50 leté historii. Poslední kapitola teoretické části práce se věnuje simulacím a simulačním postupům.

V praktické části práce je detailně rozebrán postup programování jednoduché aplikace pod názvem PEVESO, která je určena pro řízení komplexnějších úloh matematického programování v oblasti projektového řízení. Závěrečná kapitola práce pak slouží jako manuál pro ovládání aplikace a zahrnuje jednoduché typy pro ovládání daného programu.

Klíčová slova: VERT, PERT, aplikace, teorie síťové analýzy, řízení projektů, stochastické metody

Development of application for solving methods VERT and PERT

Summary

This thesis deals with the development of a simple application that works as a universal tool for solving sophisticated tasks in project management using analytical methods VERT and PERT, which are suitable for solving projects, defined by the stochastic duration of each activity.

In the theoretical part, the author extensively deals with the theoretical side of project management. Continues with characteristic of the most common methods and more detailed analysis of stochastic method VERT, which was developed in the 70s of the 20th century. This method is explained in this magnitude and in Czech language for the first time in its more than 50-year history. The last chapter of theoretical part is concerned with simulations and simulation procedures.

In the practical part of this thesis is analyzed in detail the process of programming a simple application called PEVESO, which is designed to manage complex tasks of mathematical programming in project management. Final chapter serves as a guide for managing the application and includes simple tips for using the program.

Keywords: VERT, PERT, application, theory of network analysis, project management, stochastic methods

Obsah

Úvod

V dnešní extrémně konkurenční době se i renomované společnosti s bohatou historií a nepředstavitelným zázemím mohou proměnit během několika okamžiků z pozice lídra na trhu na firmu s regionálním působením. Doba si žádá inovace, zlepšování, dynamiku a neustálý přínos pro zainteresované strany. Pro udržení dobrého postavení na trhu se většina hráčů v soukromé sféře snaží každý den ujistit stávající i nové klienty o své dokonalosti. K těmto krokům často slouží projektová oddělení firem, která za podpory vysokého rozpočtu představují novinky a vymoženosti, jež jsou pro tyto klienty přitažlivé.

Je dobré vědět, že za každou z těchto novinek se skrývají měsíce i roky práce, příprav, analýz, vyhodnocení, testů a zkušebních provozů. Projektové oddělení, které je součástí takřka každé dnešní firmy, využívá silné zázemí kvalifikovaných expertů a odborníků, stejně tak jako kvalitního a často drahého software, který sumarizuje časový plán jednotlivých činností, náklady, alokované zdroje, rizika nebo příležitosti. Licence na používání těchto nástrojů často představují značnou finanční zátěž v rozpočtu společnosti, avšak i za této podmínky jsou extrémně efektivním nástrojem.

Pod výše uvedenými přístupy rozumíme zpravidla část projektového řízení, kterou můžeme nazvat jako *Hard project management* – neboli matematické metody, které slouží k sestavení základního projektu z hlediska jeho logiky, času nebo nákladů. Neméně významnou částí je *Soft project management*, pod kterým si můžeme představit manažerské přístupy a rozhodování na základě znalostí a zkušeností, které vychází z předchozích projektů, dynamiky firmy a schopnosti zaměstnanců nebo jednotlivých oddělení, které jsou do daného projektu zainteresované.

Vzhledem ke své různorodosti a neustálému vývoji této vědní disciplíny tak řada (často sofistikovanějších) metod nikdy nedostala příležitost ukázat svůj reálný potenciál, a proto se tyto metody nikdy neobjevily jako součást programového vybavení. Jednou z těchto metod je metoda VERT (*Venture Evaluation and Review Technique*), která vznikla počátkem 70. let 20. století a podobně jako řada jiných metod nebyla nikdy součástí žádného významnějšího projektového software, používaného v soukromé sféře. Ačkoliv je metoda schopná podávat extrémně přesné výsledky, nestala se nikdy všeobecně užívaným matematickým přístupem, podobně jako například PERT (*Program Evaluation and Review Technique*) nebo GERT (*Graphical Evaluation and Review Technique*), a to pravděpodobně z důvodu vysoké složitosti a časové nákladnosti výpočtu.

Cíl práce a metodika

Významná většina závěrečných prací, které jsou každoročně prezentovány na vysokoškolských institucích, obsahují často široký vlastní výzkum, jenž daný systém nejen prověří, ale většinou i obohatí. Nicméně, autor této závěrečné práce se rozhodl pro práci s čistě teoretickým tématem. Na rozdíl od spolupráce s externím subjektem je tedy hlavní přínos shledán v propojení dvou studijních oborů – programování a projektové řízení, které v této své jedinečné kombinaci daly vzniknout této závěrečné práci.

1.1 Cíl práce

Cílem této práce je vytvoření jednoduché aplikace, kterou bude možné využít k řízení projektů menšího rozsahu. Unikátnost této aplikace bude spočívat především v kompatibilitě s jednotlivými projektovými metodami. Kromě všeobecně známé metody PERT (*Program Evaluation and Review Technique*) totiž dokáže aplikace vyřešit zadané úlohy i za použití metody VERT (*Venture Evaluation and Review Technique*).
Smyslem ani cílem této práce není vytvořit profesionální aplikaci, která by obstála na dnešním extrémně konkurenčním trhu, jež nabízí bezpočet produktů pro efektivní řízení projektů a na jejímž vývoji by se musela podílet řada odborníků a investorů. Aplikace by ve své finální podobě měla sloužit především v akademické sféře - pro snazší pochopení projektové metody VERT, případně pro porovnání jednotlivých matematických přístupů při řešení vybraného ekonomického problému z hlediska efektivity řešení.

1.2 Metodika

Pro vypracování této diplomové práce bude nezbytné nastudovat z odborné literatury dostatek informací o zmíněných metodách PERT a VERT. Především u druhé zmíněné metody je literatura značně limitována co do množství, aktuálnosti a dostupnosti, o jazykové omezenosti nemluvě. Dále bude nezbytné prostudovat dostatek materiálů o jednoduchém programování. Pro tyto účely je diplomová práce hierarchicky rozdělena do celkem šesti navazujících kapitol, ve kterých se čtenář postupně seznamuje se surovou teorií řízením projektů, jeho metodami, simulacemi, programováním i na závěr finálním seznámením s hotovou aplikací.

Prvním krokem je získávání literatury a dalších relevantních zdrojů pro studium a zpracování literární rešerše. Následně proběhne studium další disciplíny nezbytné pro zpracování této práce, a tedy programování. Finální aplikace bude naprogramována v jazyce JAVA.

Po důkladném prostudování všech materiálů budou nejprve postupně zpracovány první čtyři kapitoly, tedy teoretická část závěrečné práce. Ta bude obsahovat ve své první části ucelený přehled teoretických znalostí o projektovém řízení, ve své druhé části bude následována informacemi o projektových metodách a jejich aplikaci. Vzhledem k hloubkové analýze metody VERT bude tomuto analytickému přístupu ponechána celá následující kapitola, což by mělo stačit pro dostatečné vysvětlení tohoto přístupu včetně retrospektivního pohledu do historie, na modifikaci dané metody a také na možnosti jejího praktického použití v současném obchodně orientovaném světě.

S postupným vývojem aplikace (dílčími kroky) se začne již během studia tohoto programovacího jazyka. Veškeré významné postupy budou upřesněny a komentovány v páté kapitole, tedy praktické části této práce a jednotlivé možnosti ovládání této aplikace včetně ukázkové úlohy se stanou součástí šesté kapitoly, a tedy poslední části této práce. Vybrané kódy a taktéž i manuál budou dostupné v příloze.

2 Teoretická východiska

První kapitola je věnována představení teorie řízení projektů. Hluboká literární rešerše představí nejprve projektové řízení prostřednictvím základních definic českých a zahraničních autorů. Stručně představená charakteristika a terminologie projektového řízení usnadní orientaci v dalších kapitolách této práce. Ze široka je však vysvětlen životní cyklus projektového řízení a projektu samotného. Na projekt je nahlíženo z pohledu několika českých a zahraničních autorů.

2.1 Základní definice

Naprosto klíčové pojmy, které mají souvislost s touto závěrečnou prací, jsou pojmy „*projekt"* a „*projektové řízení*". Vzhledem ke své významnosti budou oba pojmy na následujících řádcích definovány řadou autorů z ČR i ze zahraničí. Autory těchto definicí jsou doc. Ing. Tomáš Šubrt, Ph.D., prof. RNDr. Ing. Petr Fiala, Csc, MBA., Ing. Alena Svozilová, MBA a ze zahraničních autorů pak Harold Kerzner, Ph.D. a Margot Note.

V prostorách České zemědělské univerzity v Praze je projektové řízení definováno Tomášem Šubrtem v základních skriptech následovně: „*Projektové řízení je plánování, organizování a řízení činností a jejich zdrojů v rámci uceleného projektu za respektování časových, zdrojových a nákladových omezení (obvykle s cílem dosažení maximálního ekonomického efektu*.)" Samotný „projekt" pak Tomáš Šubrt definuje následovně: „*Projektem nazýváme posloupnost činností, které je třeba provést k dosažení stanoveného cíle*".[1]

Jiná definice z akademické sféry od Petra Fialy říká, že „*Projekt je výsledek materiální nebo nemateriální povahy založený na strategickém plánu, navržený, organizovaný a realizovaný pod řízením někoho v zájmu vlastníka nebo zadavatele.*" Nicméně, tato definice je podmíněna dalšími skutečnostmi, mezi které patří všeobecná použitelnost projektu po dobu určenou zadavatelem projektu, časové ohraničení trvání projektu, omezenost zdrojů nebo absence informací o výsledku projektu na jeho začátku. Řízení

[1] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0. str. 3; 6

projektů pak Fiala definuje jako „*Soubor modelů, metod, postupů, nástrojů a technik pro plánování a řízení realizace složitých projektů.*“[2]

Alena Svozilová si stojí za názorem, že při aplikaci vybraných metod a pravidel projektového managementu může být jako projekt vnímán téměř jakýkoli sled úkolů. Sama pak stojí za definicí: „*Projekt je řízeným procesem, který má svůj začátek a konec a přesná pravidla řízení a regulace.*“ V opačném případě se jedná o sled úkolů, který (na rozdíl od projektu) může dosáhnout neočekávaného výsledku. Stejně tak nemusí celkový objem vstupů odpovídat hodnotě výstupu.[3]

Harold Kerzner ve své zahraniční publikaci „*Applied Project Management: Best Practices on Implementation“* definuje pojem „Projektové řízení“ následovně: „*Projektové řízení lze pojmout jako plánování, přiřazování a sledování dané řady integrovaných úkolů, které jsou v souladu s úspěšným dosažením cíle celého projektu během realizace a dále jsou v zájmu všech zainteresovaných stran*.“[4] Dle jeho názoru, svět ještě neobjevil a v nejbližší době ani neobjeví, jak vysoký význam může mít projektové řízení v dnešním vyspělém obchodně orientovaném světě.[5]

Poslední z uvedených definic pochází od americké autorky Margot Note, která projekt definuje následovně: „*Projekt je řada jedinečných, mnohostranných a vzájemně souvisejících činností, které vedou k danému cíli, jenž musí být dosažen ve stanoveném čase, s omezenými finančními možnostmi a dle dalších zadaných specifikací*.“[6] Ačkoli může mít několik projektů více společných rysů, každý z nich je charakterizován svými unikátními atributy. Projekt je unikátní a od data svého začátku vytváří něco, co je dokončeno ve stanoveném termínu.[7]

---

[2] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X., str. 12-13

[3] SVOZILOVÁ, Alena. *Projektový management.* Praha: Grada, 2006. Expert (Grada). ISBN 80-247-1501-5., str. 21

[4] V originále: Project management can be defined as the planning, scheduling, and controlling of a series of integrated tasks such that the objectives of the project are achieved successfully and in the best interest of the project´s stakeholders.

[5] KERZNER, Harold. *Applied project management: best practices on implementation.* New York: Wiley, c2000. ISBN 0471363529., str. 2

[6] V originále: A project is a series of unique, multifaceted, and related activities with a purpose that must be accomplished at a paricular time, within cost constraints, and according to specifications.

[7] MARGOT, Note. *Project management for information professionals.* Vyd. 1. Chandos Publishing, 2015. ISBN 978-0-08-100127-1., str. 1

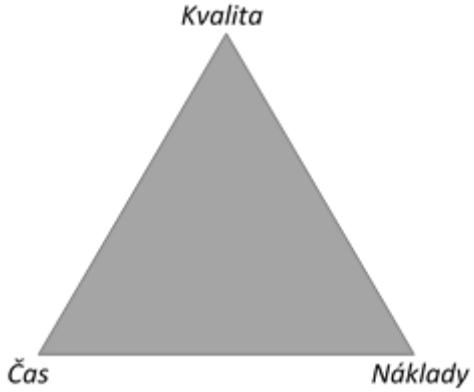
2.2 Charakteristické rysy projektu

Při důkladném pročítání výše uvedených definic odborných autorů si nelze nevšimnout vzájemné podobnosti při pohledu ať už na projekt či na celý obor projektového řízení. Z tohoto důvodu je vhodné uvést veškeré charakteristické rysy projektu. Každý projekt má jasně definovaný svůj začátek a konec, přičemž v termínu svého konce je dosaženo jasně definovaného cíle, kvůli kterému daný projekt vznikl.

Projekt se dále vyznačuje použitím pružné organizační struktury, která je schopná okamžitě reagovat na nečekaně nastalé události, které s projektem vždy svazují významnou míru nejistoty a vytváří rizika. Složení řešitelského týmu projektu je proměnlivé a pro efektivní řízení pracovního postupu je nezbytné, aby byl tento tým řízen vertikálně (od shora dolů) a nikoliv horizontálně, jako je tomu zvykem v tradičních odvětví managementu, neboť zahrnuje rozsáhlé plánování a koordinaci.

Z hlediska sestavování jednotlivých analýz jsou však nejvýznamnějšími rysy projektu hodnoty jeho číselných charakteristik, které umožňují ohodnotit délku jeho trvání, analyzovat náklady a dále disponibilní zdroje, celkový rozsah projektu a kvalita. Při řízení projektů je nezbytné sledovat čas vzhledem ke stanovenému plánu, náklady ve srovnání se stanoveným rozpočtem a také kvalitu projektu, která určuje stupeň dosažení nastavených cílů. Čas, náklady a kvalita tedy mohou být chápány jako základní ukazatele projektu a vzhledem k vzájemné závislosti je nezbytné při plánování projektu tyto ukazatele vyvažovat. Tuto situaci dokonale zachycuje projektový trojúhelník na obrázku č. 1.[8]

Obrázek 1: Projektový trojúhelník (trojimperativ)



Zdroj: FIALA, Petr. *Řízení projektů.* 2008, str. 11

[8] FIALA, Petr. *Řízení projektů.* Vyd. 2., přep. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0. str 10-11

2.3 Terminologie řízení projektů

Celá vědní disciplína řízení projektů je provázána specifickou terminologií. Základními termíny jsou bezpochyby „projekt“ a „řízení projektů“. Vzhledem k tomu, že oběma termínům byly ze široka věnovány předchozí podkapitoly, nebude se jimi autor v této části již zabývat. Podobně nebudou rozebírány ani jednotlivé charakteristiky.[9]

- **Činnost** - Činnost můžeme určit jako specifický proces, během kterého se vstupy transformují na výstupy. Každá činnost má své vlastní charakteristiky, které postupně čerpá a logicky navazuje na předchozí a následné činnosti v rámci celého projektu. Vstupy činností jsou lidské a materiální zdroje, výstupy činností pak jsou služby, výrobky nebo zkvalitněné dílčí procesy.
- **Cíl** - Cíl projektu vyjadřuje změny, které projekt zahrnuje a důvod, proč se projekt realizuje. Podle předních autorů by měl být cíl tzv. SMART - *Specific* (konkrétní), *Measurable* (měřitelný), *Achievable* (dosažitelný), *Realistic* (realistický), *Time-bound* (časově ohraničený). K dosažení stanoveného cíle je nezbytné využití lidských a materiálních zdrojů, které jsou k projektu přiřazeny v omezeném množství. Pro stanovení cíle projektu je nezbytné zodpovědět základní otázky:
    - Co má být projektem dosaženo (veřejně prospěšná hodnota díla)
    - Jak bude průběh projektu plánován a sledován (projektové ukazatele)
    - Omezení ukazatelů (finanční, časové, ekologické a energetické omezení)
    - Priority cílů a příslušných úkolů pro přiřazení dostupných zdrojů
    - Koordinační požadavky
- **Subprojekt** - Subprojekt si můžeme představit jako větší množství dílčích úkolů se stejným charakteristickým rysem, ale zobrazené v hlavním projektu jako jediný úkol. Použití subprojektů je vhodné především při dekompozici projektu na menší, snadno řiditelné jednotky. Své využití má dále ve standardizaci často opakovaných úkolů a pro redukci matematických výpočtů při řešení rozsáhlých projektů.
- **Milník** - Milník je specifická podoba činnosti, která má nulovou délku trvání a která v rámci projektu označuje významný termín. Milníky slouží jako kontrolní body a umožňují sledovat postup a dokončenost projektu.

[9] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0. str. 6-9

2.4 Životní cyklus projektového managementu

Harold Kerzner vnímá řadu asociací mezi Projektovým řízením a TQM (*Total Quality Management*), neboť oboje jsou řídicí systémy, jež vyžadují rozsáhlé přípravy a odborné vzdělávání. Jako důsledek životního cyklu projektového řízení si představuje vývoj společnosti v dané oblasti s každým dalším dokončeným projektem, přičemž ideálem je dosáhnout stádia zralosti.[10]

Dosažení vynikajících výsledků (a tedy stádia zralosti) v oblasti řízení projektů může být dosaženo během několika let nebo několika desetiletí. Tohoto stavu nelze dosáhnout beze změny, přičemž rychlost těchto změn je klíčová. Stejně jako systémy řízení kvality, vyžaduje i projektové řízení vzdělávání pracovníků a odborné přípravy. Vzdělávací proces musí začínat u TOP managementu a postupně zapojovat zaměstnance na nižších úrovních k plné integraci v celé společnosti.

Je dobré vědět, že postupný, výše uvedený přechod, je součástí každé moderní společnosti. Křivka procesu učení se měří v letech, přičemž u efektivní a schopné společnosti může být celý proces splněn se štěstím za dva roky, zatímco průměrná firma se dostane do finálního stádia za pět let.

Jak je ale možné ověřit dosažení stádia zralosti? Na rozdíl od kultur, kde je možné zralost určit na základě šedivých vlasů nebo věku, není možné v projektovém řízení takto banálně rozhodnout. V projektovém řízení toto stádium nemůžeme určit přesně, ale můžeme určit pravděpodobnost, že se daná firma blízko tohoto stádia nachází. To se projevuje vyvíjením systémů a procesů s opakujícími se charaktery, které zvyšují pravděpodobnost dokončení projektu. Sami o sobě nezaručují úspěch, ale pouze zvyšují možnost úspěchu. V tabulce č. 1 pod tímto textem se nachází seznam a charakteristika fází životního cyklu projektového řízení v organizaci.[11]

[10] KERZNER, Harold. *Applied project management: best practices on implementation.* New York: Wiley, c2000. ISBN 0471363529. str. 13-14

[11] KERZNER, Harold. *Applied project management: best practices on implementation.* New York: Wiley, c2000. ISBN 0471363529.str. 32-33

Tabulka 1: Životní cyklus projektového řízení dle Harolda Kerznera

ZÁRODEK	PŘIJETÍ ŘÍDÍCÍM MGMT.	PŘIJETÍ LINIOVÝM MGMT.	RŮST	DOSPĚLOST
Rozpoznání potřeb	Získat značnou podporu řídícího mgmt.	Získat značnou podporu liniov. mgmt.	Rozpoznatelné užívání fází životního cyklu	Vývoj systému pro sledování času a nákladů
Rozpoznání přínosů	Dosáhnout přijetí řízení projektu	Dosáhnout závazku liniov. mgmt.	Vývoj metodologie pro řízení projektů	Nástroje pro ovládání času a nákladů
Rozpoznání žádostí	Zavést sponzorství projektu	Poskytnout vzdělání liniov. mgmt.	Dát závazek k pravidelnému plánování	Vývoj vzděláv. programu pro posílení PŘ
Rozpoznání toho, co je třeba udělat	Ochota změnit způsob podnikání	Ochota poskytnout školení v proj. řízení	Minimalizovat rozsah finálních úprav	Systém sledování projektu

Zdroj: KERZNER, Harold. *Applied project management: best practices on implementation.* 2000, str. 33

Ve vybraných případech je možné se dostat ke stádiu zralosti bez toho, že by daná organizace prošla všemi fázemi. Jedná se však o velmi neobvyklý jev. Je dobré zmínit, že Harold Kerzner doplňuje, aby součástí veškerých výše uvedených skutečností byly záznamy o provedených změnách: „*Bez dokumentů zachycujících „Lessons Learned“ se může společnost rychle vrátit ze stádia zralosti do nižších fází projektového řízení. Získané znalosti se mohou ztratit a dřívější chyby se opakují.“*[12]

2.5 Životní cyklus projektu

Životní cyklus projektu charakterizuje vývoj projektu na časovém rozsahu, jenž trvá od formulace projektu až po zavedení změny do běžného provozu, případně jeho ukončení a verifikace. Životní cyklus zahrnuje vznik, existenci i likvidaci daného díla ve všech možných souvislostech. Je tedy samozřejmé, že projekt řídíme v cyklu, který se skládá z několika fází. Je dobré vědět, že tyto fáze nekopírují ani nepředstavují činnosti projektu.[13]

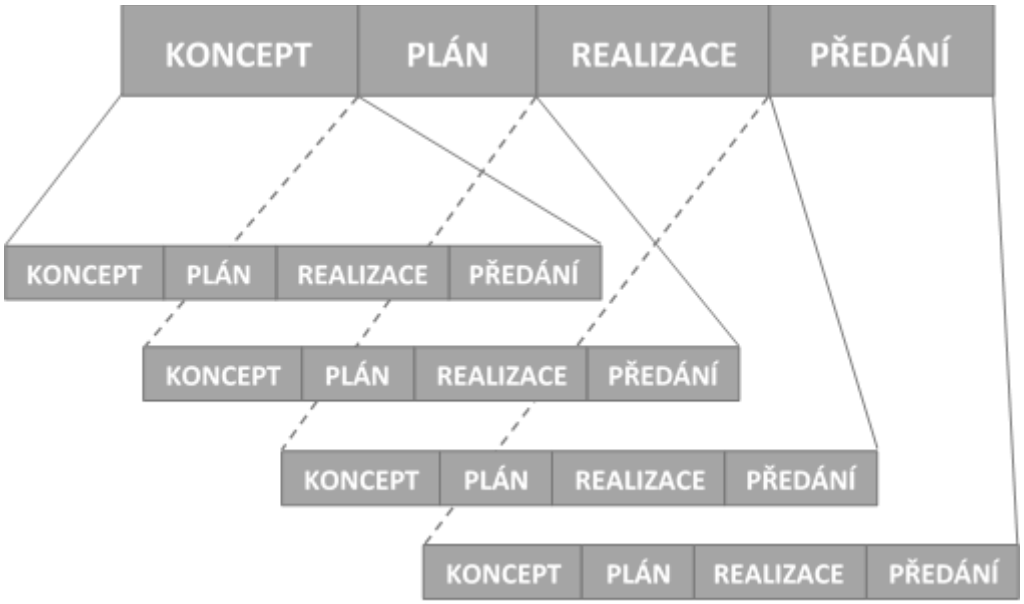
[12] KERZNER, Harold. *Applied project management: best practices on implementation.* New York: Wiley, c2000. ISBN 0471363529. str. 39
[13] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0. str. 7

Životní cyklus projektu uvedla ve svých publikacích řada autorů. Autor práce se rozhodl představit podobu životního cyklu projektu podle *Petra Fialy, Aleny Svozilové* a trojice autorů *Sang M. Lee, Gerald L. Moellera* a *Lester A. Digmanna*. Je vhodné doplnit, že zajímavý pohled na projekt nabízí ve své publikaci „Projektový management" autoři *Václav Dolanský, Vladimír Měkota* a *Vladimír Němec*, kteří dělí jednotlivé fáze z hlediska financování projektu nebo *D. I. Cleland* a *W. R. King*, kteří vnímají životní cyklus projektu jako silně podpůrný nástroj projektového manažera.

2.5.1 Životní cyklus projektu dle Petra Fialy

V rámci svého životního cyklu prochází projekt, podle Petra Fialy, většinou čtyřmi až osmi fázemi. Čtyři základní fáze, které na sebe navazují a které je možné dále strukturovat na obdobné fáze, se nazývají koncept, plán, realizace a předání. Obrázek č. 2 zobrazuje zmíněnou strukturalizaci fází na nižší úroveň.

Obrázek 2: Životní cyklus projektu dle Petra Fialy



Zdroj: FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* 2004, str. 25

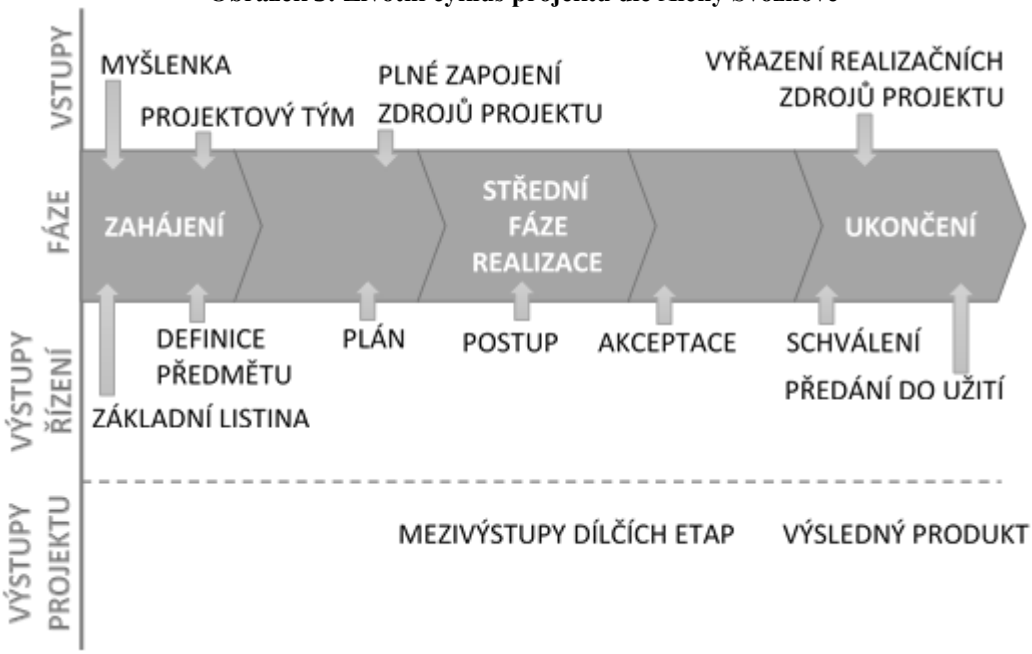
Ve chvíli stanovení jednotlivých fází je vhodné provést jejich analýzu a stanovit vstupy, procesy, klíčové činnosti, zlomové okamžiky a výstupy.[14]

[14] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X., str. 24-25

2.5.2 Životní cyklus projektu dle Aleny Svozilové

Podle Aleny Svozilové je cílem rozdělení daných aktivit do logického časového sledu zlepšení podmínek pro kontrolu daných procesů. Rozdělení projektu na jednotlivé fáze zpřehledňuje orientaci všech účastníků, kteří jsou do daného projektu zainteresováni a zvyšuje tak pravděpodobnost celkového úspěchu projektu. Fáze životního cyklu projektu tak jednoznačně vymezují, kdo se podílí na projektu v jeho jednotlivých časových úsecích, jaký typ práce má být v daném čase vykonán a které konkrétní výstupy jsou v daných fázích generovány, jak jsou ověřovány a hodnoceny. Grafické znázornění typického rozložení fází životního cyklu projektu je zobrazeno pod textem na obrázku č. 3.

Obrázek 3: Životní cyklus projektu dle Aleny Svozilové



Zdroj: SVOZILOVÁ, Alena. *Projektový management.* 2006, str. 38

Jak je jasně vidět na výše uvedeném obrázku (č. 3), tak v jednotlivých fázích projektu dochází k postupnému čerpání zdrojů, přičemž souvisle s jejich čerpáním dochází ke generování výstupů projektu, které jsou v souladu s cílem projektu. Přechod mezi fázemi bývá uskutečněn na základě schvalovacího procesu, kde je rozhodnuto o dalším osudu vývoje projektu. Dalším vývojem může být pokračování dle původně zpracovaného plánu, přijetí opatření a provedení korekcí, ale také pozastavení či zrušení projektu.[15]

[15] SVOZILOVÁ, Alena. *Projektový management.* Praha: Grada, 2006. Expert (Grada). ISBN 80-247-1501-5., str. 38

2.5.3 Životní cyklus projektu dle S. M. Lee, G. L. Moellera a L. A. Digmana

Trojice autorů nahlíží na projekt podobně jako na produkt a směle tvrdí, že se z pohledu životního cyklu jeden od druhého příliš neliší. Tento přístup by ale měl být brát s rezervou, neboť citace pochází z 80. let minulého století. Přesto je, dle názoru autora, tento přístup vhodné uvést, neboť vychází z publikace, ve které je následně detailně charakterizována matematická metoda VERT, která je klíčovým tématem této práce.

Podobně jako tvrdí Fiala, tak i výše uvedení autoři dělí životní cyklus do čtyř odlišných, vzájemně se však překrývajících částí. (Tím se liší od výše uvedeného názoru Aleny Svozilové, podle které musí být jednotlivé fáze přesně časově ohraničeny a musí jim být přiřazeny jednotlivé výstupy.) Těmito fázemi jsou fáze koncepční, vývojová fáze, produkční fáze a provozní fáze. Trvání těchto fázích se může vzájemně lišit v závislosti na charakteru projektu, případně produktu. Detailněji jsou jednotlivé fáze projektu rozebrány na následujících řádcích.[16]

Tabulka 2: Životní cyklus projektu dle Sang M. Lee, Gerald L. Moellera a Lester A. Digmana

Parametry projektu	KONCEPČNÍ FÁZE	VÝVOJOVÁ FÁZE	PRODUKČNÍ FÁZE	PROVOZNÍ FÁZE
ČAS	Potřebný čas pro daný koncept	Milník; Odhad času k dokončení	Čas dodání	Funkční dodavatelské & logistické řešení
NÁKLADY	Rozpočet pro daný koncept	Náklady na vývoj; Odhad nákladů k dokončení	Jednotkové náklady	Přijatelné provozní náklady
VÝKON	Požadované provozní vlastnosti daného konceptu	Úspěch designu; Výkon prototypů	Shoda se zadáním	Spolehlivost a udržitelnost
RIZIKO	Pravděp. dostat. času, nákladů a výkonu pro každý koncept	Pravděpodobnost úspěšného splnění výše uvedeného	Pravděpodobnost úspěšného splnění výše uvedeného	Hodnocení úspěšných konceptů

Zdroj: LEE, Sang M. a kol. *Network Analysis for Management Decision.* 1982, str. 17

[16] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach.* Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7., str. 14

2.5.3.1 Koncepční fáze

Součástí koncepční fáze projektu je vytvoření studie ziskovosti a studie proveditelnosti. Ta zahrnuje 4 jednotlivá spektra, jejichž pravděpodobnost musí být zkoumána. Mezi nimi je:

- Pravděpodobnost, že konkurenční výrobky budou na základě svého designu nebo vlastností pro koncového zákazníka atraktivnější.
- Pravděpodobnost, že výrobky konkurenční společnosti budou splňovat potřeby zákazníka stejně jako naše.
- Rozsah pravděpodobné poptávky.
- Pravděpodobnost, že bude projekt/produkt navržen, vyvinut, vyroben a provozován dle své potřeby, v rámci časových a nákladových omezení a také k přijatelné úrovni rizika.

Po dokončení této fáze by měl být připraven kompletní plán (časová a nákladová analýza, očekávaný výkon, identifikovaná rizika) a kromě toho by měl být připraven konkrétní plán pro dokončení zbývajících etap.

2.5.3.2 Vývojová fáze

Po výběru vhodného konceptu začíná vývojová fáze. Tato činnost zahrnuje převážně práci inženýrů, jejichž cílem je sestavení vhodného prototypu výrobku. Výstupem by měla být sada podrobných nákresů a specifikací, které jsou dostačující pro spuštění výroby daného produktu.

2.5.3.3 Produkční fáze

Přestože produkční fáze běžně navazuje na fázi vývoje produktu, v některých případech se mohou obě fáze překrývat, především pokud je čas kritickým faktorem. Tato fáze zahrnuje výrobu a dodání daného množství produktu ve stanovené kvalitě, za daného provozního výkonu a nákladu na jednotku.

2.5.3.4 Provozní fáze

Provozní fáze se obvykle překrývá s výrobní fází a na jejím počátku dochází k dodání položek ke koncovému spotřebiteli. Úspěšná provozní fáze je charakteristická neustálou touhou spotřebitele po výrobku, přičemž poptávka převyšuje nabídku. K uzavření provozní

fáze dochází ve chvíli, kdy se produkt stává zastaralým a nevyhovuje potřebám uživatele. Likvidace produktu dokončuje jeho životní cyklus.

Jak již bylo zmíněno, ačkoliv jsou výše uvedené klíčové fáze životního cyklu charakteristické spíše pro produkt, můžeme zaznamenat výskyt obdobných fází i u projektů. Nicméně, v těchto případech existuje daleko vyšší tlak na nejvýznamnější koncepční fázi.[17]

[17] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach.* Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7., str. 15-16

3 Analýza řízení projektů

V této kapitole je představen projektový management v analytickém přístupu. Prostor je nejprve věnován rozsáhlejší historii vývoje řízení projektů. V další části se autor stručně věnuje představení teorie grafů. Na závěr jsou prezentovány základní metody pro řízení projektů – CPM, MPM, PERT, GERT a VERT.

3.1 Historie řízení projektů

Přes veškerou snahu není možné přesně specifikovat, odkdy je možné datovat reálný začátek projektového řízení. Již v období starověku vznikaly po celém světě pravděpodobně nejstarší známé stavební projekty, které si dodnes uchovaly svou původní podobu. Přesto se oficiální začátek této disciplíny datuje k počátku 20. století. V následujících 5 podkapitolách je chronologicky zachycen vývoj projektového řízení od starověku až do začátku 21. století.

3.1.1 Starověké a středověké projekty

Mezi nejstarší díla, která můžeme v dnešní době klasifikovat jako projekty, patří dochované historické stavby ze Starověkého Egypta v podobě pyramid, Velkou čínskou zeď vystavěnou na obranu proti Mongolům nebo městečko Machu-Picchu a Aztécké pyramidy ve Střední resp. Jižní Americe. Ačkoli nemáme v dnešní době dochované znalosti technik řízení, je jisté, že tyto stavby vznikaly koordinací pracovního úsilí.

Vzhledem k výše uvedenému tak není příliš odvážné tvrdit, že prvním doloženým „projektovým manažerem“ byl Vitruvius[18], římský architekt, stavební a vojenský inženýr, který žil v 1. stol. př. Kr. Dalším, kdo ve svých stavbách vycházel z dokonale připravených plánů, byl anglický architekt a návrhář Christopher Wren. Ten navrhl v 17. století přes 50 londýnských kostelů, včetně katedrály sv. Pavla. V oblasti projektového managementu se proslavil především tím, že ke své práci vytvářel nadměrné množství plánů, které ukládal a dle potřeby konzultoval se zadavateli stavby[19]. Pomyslnou trojici uzavírá nejvýznamnější

[18] Plným jménem Marcus Vitruvius Pollio
[19] Tento soubor dokumentů by mohl být vnímán jako prapočátek projektové dokumentace

britský konstruktér a podnikatel Isambard Kingdom Brunel. Ten je známý svými monstrózními stavbami z 19. století, mezi které patří řada mostů a tunelů, a také vytvoření Great Western Railway. Jeho největším dílem je však stavba nejdelší lodi své doby na světě – zaoceánského parníku Great Eastern.[20]

3.1.2 Počátky projektového řízení

Skutečné zavedení projektového řízení je spojeno s používáním Ganttových diagramů. Ty jsou jednoduchým vizuálním prostředkem, který Henry Gantt zavedl v roce 1910 pro řízení stavby lodí. Tento způsob plánování a řízení je dodnes velmi často používaný, neboť vyniká svou jednoduchostí a přehledností.[21] Málokdo však ví, že obdobnou metodu nezávisle na Ganttovi publikoval (v angličtině) v roce 1931 polský ekonom a inženýr Karol Adamiecki, který stejnou metodu pod názvem *harmonograf* již řadu let využíval. Ten se teorii řízení projektů věnoval od roku 1896, avšak tou dobou publikoval pouze v polštině a ruštině, přičemž oba jazyky byly pro západní svět v té době neznámé.[22]

Neméně významnou osobu této éry je Henri Fayol, který rozpoznal a popsal pět původních funkcí managementu – plánovat, organizovat, přikazovat, koordinovat a kontrolovat. Gantt i Fayol byli studenti učení F. W. Taylora, který ve svých pracích položil základy alokace zdrojů nebo WBS[23] (Work Breakdown Structure).

Pravděpodobně nejznámějším projektem této doby, u kterého byly aplikovány výše uvedené „nové“ poznatky z řízení projektů, je stavba Hooverovy přehrady na řece Colorado. Stavba tohoto kolosu na hranicích Nevady a Arizony započala 20. dubna 1931 a podílelo se na ní 5000 dělníků, kteří pokládali 5 stop široké betonové pláty v 72 hodinových intervalech. Vedoucí stavby, Francis T. Crowe, použil Ganttovy diagramy při plánování a řízení stavby této elektrárny. Stavba byla dokončena v rekordním čase za 5 let, o 2 roky dříve, než bylo plánováno.[24]

---

[20] LOCK, Dennis. *Project management*. Vyd. 9. Burlington: Gower Publishing, 2007. ISBN 0-566-08772-3. str. 82

[21] FIALA, Petr. *Řízení projektů*. Vyd. 2., přeprac. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0. str. 7

[22] MARSH, Edward R. *The Harmonogram of Karol Adamiecki*. The Academy of Management Journal. Academy of Management, 1975, Vol. 18, No. 2, 358-364. str. 358-361

[23] Hierarchický rozklad cíle projektu

[24] The Hoover Dam: Bechtel, Crowe, Kaiser, Khan and Shea 1936 (completed). *Project Management History*; Dostupné online z: http://projectmanagementhistory.com/The_Hoover_Dam.html [cit. 31.1.2017]

3.1.3 Řízení projektů jako matematická disciplína

První boom projektového řízení nastal v 50. a 60. letech 20. století. Tato doba je spjata s vývojem řady nových analytických přístupů v souvislosti s rozvojem kosmických a vojenských projektů. Postupně tak vznikly metody CPM (1956), která řeší časovou analýzu projektu hledáním kritické cesty, následně PERT (1958), která se vyznačuje stochastickou délkou trvání činností a později PDM (1961). Zároveň vznikají 2 možnosti zobrazení síťových grafů,[25] a to hranově orientované grafy, kde hrany představují činnosti a uzlově orientované grafy, kde uzly představují činnosti. V roce 1962 vznikla první myšlenka na vytvoření metody s variabilními rozhodovacími uzly, a tak o 4 roky později byla publikována metoda GERT (1966), která je stochastická v síťovém grafu.[26]

Toto období je charakteristické aplikací „tvrdých" přístupů projektového managementu, které je založené na matematických operacích a znalost matematiky je tak předpokladem pro práci s projekty. Vzniká bezpočet (především stochastických) metod „na míru", které jsou implementovány do organizací. Na závěr je nezbytné upozornit, že tato doba je neodmyslitelně spjata s prvními počítači, a tak vznikalo i první softwarové vybavení pro řízení projektů. To však často umožňovalo pouze základní výpočty délky trvání.[27]

3.1.4 Rozvoj počítačové techniky

Na konci 60. let přišel silný rozvoj výpočetní techniky a s ním i aplikace zcela nové možnosti při řízení projektů – využití simulací. O tuto funkci měla zájem především armáda Spojených Států, která si na Princetonské univerzitě objednala program MATHNET (1970). Jednalo se o první software, který byl schopný provést analýzu rizika, avšak (z důvodu nedostatku času na vývoj) obsahoval řadu drobných chyb. Z tohoto důvodu vznikly tři alternativní produkty, kterými jsou RISCA (1970)[28], základy původního programu byly využity pro nový software STATNET (1971) a o rok později vznikl SOLVNET (1972).[29] Ve všech třech případech se jednalo o pokročilé nástroje, které využívaly řadu statistických rozdělení a logických uzlů, avšak nebyli vzájemně

[25] Grafického znázornění projektů, více v podkapitole 3.2 – Teorie grafů
[26] FIALA, Petr. *Řízení projektů*. Vyd. 2., přeprac. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0., str 7
[27] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach*. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 30-31
[28] Risk Information System Cost Analysis
[29] Více v podkapitole 4.1 – Historie a vývoj metody VERT

kompatibilní. Tím vznikl požadavek na vývoj zcela nové metody VERT (1972), která by byla schopná veškeré požadované funkce zahrnout. Přípravou této metody byl pověřen Gerald L. Moeller, který navíc dokázal v metodě VERT jasně definovat matematický vztah mezi kterýmikoli dvěma činnostmi.[30]

Řízení projektů se v 70. letech ale rozšířilo i na další odvětví a se vznikem pozice „Projektového manažera" začalo být projektové řízení chápáno jako profese. S tím souvisel i vznik prvních profesních společností a postupem času i sofistikovanějších počítačových programů pro veřejnost (např. *Harvard Project Manager*).[31]

3.1.5 Řízení projektů jako manažerská disciplína

Od začátku 90. let se s inovací organizačních struktur začalo projektové oddělení zahrnovat do každé významnější organizace. S tím souviselo utváření těchto projektových týmů, které se podílely na práci ve formě projektů, jež pružně reagovaly na požadavky trhu. Na rozdíl od období 50. – 70. let již není kladen důraz na znalost projektových metod, podložených matematikou, ale spíše na měkké přístupy – analytické myšlení, komunikační dovednosti, umění multitaskingu nebo řízení rizik.[32]

Na hranici matematických a manažerských přístupů byla na konci 20. století vytvořena izraelským fyzikem E. Goldrattem metoda kritického řetězu (CCPM). Ta vychází z běžného pohledu do praxe, při které je upravena práce a zapojení jednotlivých zdrojů podle několika pravidel (Multitasking, Studentův syndrom, Princip štafetového běžce, Čerpání časových nárazníků,…). Při aplikace těchto pravidel zkracuje metoda kritického řetězu délku trvání projektu zhruba o 30 % bez potřeby navyšování zdrojů.[33]

Od roku 2010 zažívá projektový management v zemích střední a východní Evropy menší boom a pozice projektového manažera je opakovaně vyhodnocována jako jedna z nejžádanějších. S tím, bohužel, také klesá úroveň těchto odborníků, kteří často nemají dostatek znalostí a zkušeností a postrádají jakýkoliv matematický základ, obvykle nezbytný pro práci v tomto oboru.

---

[30] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach*. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 31-33

[31] FIALA, Petr. *Řízení projektů*. Vyd. 2., přeprac. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0., str 8

[32] FIALA, Petr. *Řízení projektů*. Vyd. 2., přeprac. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0., str 8

[33] Goldratt.cz. *Kritický řetěz* (CC); Dostupné online z: http://goldratt.cz/teorie-omezeni/kriticky-retez, [cit. 31.1.2017]

3.2 Teorie grafů

Teorie grafů je klíčovou složkou řízení projektů při řešení vybraných úloh „na papíře.“ Na následujících řádcích jsou stručně sepsány základní informace týkající se této problematiky. Kromě nezbytných základních pojmů je čtenář seznámen se síťovým grafem, který je používán pro grafické znázornění projektu a také s orientací grafu.
Po získání seznamu činností je možné takřka každý projekt zakreslit graficky. Z hlediska matematiky můžeme graf vnímat jako množinu G, která se skládá ze dvou podmnožin – množiny uzlů U a množiny hran H, tedy:

$$G = \{U, H\}$$

Uzly jsou obvykle užívány ke znázornění prvků či objektů v reálném systému, zatímco hrany reprezentují vztahy mezi nimi. V řízení projektů jsou činnosti většinou znázorněny pomocí hran (např. CPM), ale můžou být znázorněny i pomocí uzlů (např. MPM).[34]

3.2.1 Základní pojmy

Základními termíny teorie grafů jsou bezpochyby pojmy *uzel* a *hrana*. *Uzel* se zobrazuje prostřednictvím kolečka s vepsaným číslem, obecně i, j,… *Hrana* je spojnicí dvou uzlů a značí se čísly spojených uzlů, obecně (i, j). Hrana se dále dělí na *orientované* a *neorientované* (které jsou zachyceny na obrázku č. 4 pod textem), přičemž v řízení projektů se vyskytují pouze orientované hrany.

Obrázek 4: Neorientovaná a orientovaná hrana

Zdroj: FÁBRY, Jan. *Matematické modelování*. 2007, str. 53

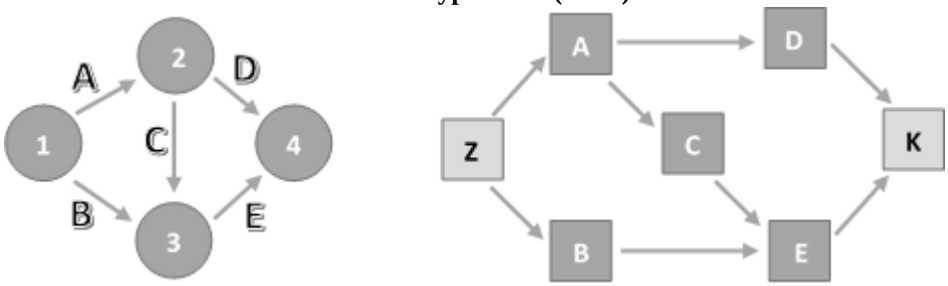
Síťový graf je specifickým grafem v teorii grafů. Jedná se o graf, který je orientovaný, souvislý a uzlově či hranově nezáporně ohodnocený. Síťový graf obsahuje dva speciální uzly, které se nazývají vstup a výstup, přičemž vstup je uzlem, ze kterého uzly pouze vystupují a výstup je uzel, do kterého hrany pouze vstupují. [35]

[34] FÁBRY, Jan. *Matematické modelování*. Praha: Oeconomica, 2007. ISBN 978-80-245-1266-2. str. 51
[35] FÁBRY, Jan. *Matematické modelování*. Praha: Oeconomica, 2007. ISBN 978-80-245-1266-2. str. 53-55

3.2.2 Projektová síť

Jako projektová síť se nazývá sestavení síťového grafu, který graficky zachycuje modelovaný problém a který slouží k řešení dané úlohy. Projektová síť pak může být dvojího typu – grafy AOA a AON, přičemž jednotlivé možnosti se liší zobrazením činností na hranách a na uzlech. Graficky jsou jednotlivé podoby grafů zachyceny na obrázku č. 5 pod tímto textem.

Obrázek 5: Graf typu AOA (vlevo) a AON



Zdroj: Vlastní zpracování

Grafy typu AOA (Activity On Arc) jsou grafy, které představují své činnosti na hranách. Toto znázornění je obvyklejší především v Evropě a je využíváno u řady základních projektových metod (např. CPM, PERT).

Proti tomu je graf typu AON (Activity on Node) specifický tím, že své činnosti představuje uzly. Tento typ znázornění je používán především v Severní Americe, kde je používán pro zobrazení metody PDM (obdoba evropské MPM). Předností tohoto typu zobrazení je snazší představení daného problému pro uživatele, neznalého projektového řízení.[36]

3.3 Základní metody pro řízení projektů

V tomto oddíle jsou představeny základní metody pro řízení projektů. Mezi nimi je naprosto základní a nejstarší metoda CPM, následně zástupce uzlově orientované metody MPM. Další v pořadí je metoda PERT, metoda stochastická v délce trvání a GERT, metoda stochastická v síťovém grafu. Poslední představenou metodou je VERT, která sice nepatří mezi známější metody, avšak je těžištěm této práce.

[36] TAVARES, L. Valadares. *Advanced Models for Project Management.* New York: Science & Business Media, 1999. ISBN 978-1-4613-4649-4. str 23-25

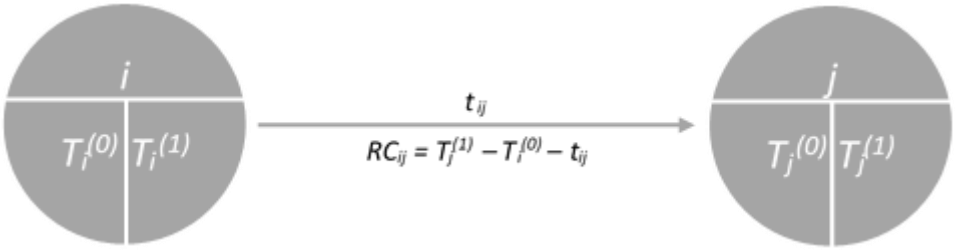
3.3.1 CPM

Metoda CPM (*Critical Path Method*) je jednou z nejjednodušších a nejstarších metod, které lze použít při plánování projektu. CPM vznikla ve Spojených státech a řeší časovou analýzu projektu při deterministické struktuře sítě a deterministického časového ohodnocení činností. Graficky je CPM zobrazena s činnostmi na hranách (AOA) a její činnosti jsou na základě logické návaznosti a délky trvání děleny na kritické a nekritické. Spojením kritických činnosti vznikne kritická cesta, stanovení délky trvání projektu.

Pro provedení výpočtu pomocí CPM je zavedeno následující označení:

- ij činnost (mezi uzly i, j)
- t_{ij} délka trvání činnosti (i, j)
- $t_i^{(0)}$ termín nejdříve možného začátku činnosti (i, j)
- $t_j^{(0)}$ termín nejdříve možného konce činnosti (i, j)
- $t_i^{(1)}$ termín nejpozději přípustného začátku činnosti (i, j)
- $t_j^{(1)}$ termín nejpozději přípustného konce činnosti (i, j)
- $T_i^{(0)}$ nejdříve možný termín uzlu i
- $T_i^{(1)}$ nejpozději přípustný termín uzlu i
- T_P plánovaná délka trvání projektu

Obrázek 6: Metoda CPM - výpočet v grafu



Zdroj: FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* 2004, str. 91

Vstupními údaji při výpočtu jsou činnosti (i, j) včetně časové návaznosti a délky jejich trvání. Eventuální možností je zadat přímo plánovanou délku trvání projektu či v kalendáři určit plánovaný termín zahájení a ukončení projektu. Při výpočtu se pro jednotlivé činnosti grafu zjišťují termíny $t_i^{(0)}$, $t_j^{(0)}$, $t_i^{(1)}$ a $t_j^{(1)}$ a pro uzly termíny $T_i^{(0)}$ a $T_j^{(0)}$. Získáním těchto časových údajů je možné určit časové rezervy projektu a kritickou cestu. Samotný výpočet se, podobně jako u jiných metod, skládá ze dvou nezávislých fází - výpočtu vpřed

a výpočtu vzad a výše uvedené zjišťované časové termíny získáme vždy pouze jedním z těchto výpočtů.[37]

3.3.1.1 Výpočet vpřed

Výpočet vpřed probíhá od začátku projektu k jeho konci a jeho cílem je zjistit nejdříve možné termíny (tedy $t_i^{(0)}$, $t_j^{(0)}$ a $T_i^{(0)}$).
Prvním krokem je určení nejdříve možného termínu zahájení projektu, což zahrnuje všechny činnosti vycházející z prvního (počátečního) uzlu.

$$t_1^{(0)} = T_1^{(0)} = 0$$

Následujícím krokem je určení nejdříve možných konců dalších činností. Vypočítáme je postupným kumulováním po sobě jdoucích hran dle následujícího vzorce.

$$t_j^{(0)} = t_i^{(0)} + t_{ij}$$

K realizaci každého uzlu dochází až ve chvíli, kdy jsou dokončeny veškeré činnosti do něj vstupující. Nejdříve možný termín realizace uzlu tak můžeme stanovit jako.

$$t_j^{(0)} = t_i^{(0)} + t_{ij}$$

Pro další činnosti určíme nejdříve možné začátky.

$$t_i^{(0)} = T_i^{(0)}$$

Pomocí výše uvedených vzorců se provede výpočet vpřed a určí se nejdříve možné termíny všech činností a uzlů. Termín $T_n^{(0)}$ udává nejdříve možný termín dokončení daného projektu.[38]

3.3.1.2 Výpočet vzad

Výpočet vzad probíhá od konce projektu k jeho začátku a jeho cílem je zjistit nejpozději přípustné termíny (tedy $t_i^{(1)}$, $t_j^{(1)}$ a $T_i^{(1)}$).

---

[37] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 85-86
[38] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 86

Pomocí předchozího výpočtu je možné stanovit nejpozději přípustný konec projektu, kde hodnota $T_n^{(0)}$ byla zjištěna při výpočtu vpřed.

Analogicky k výpočtu vpřed se dopočítávají nejpozději přípustné termíny všech činností a uzlů podle následujících vzorců.[39]

$$t_i^{(1)} = t_j^{(1)} - t_{ij}$$

$$T_i^{(1)} = max\ t_i^{(1)}$$

$$t_j^{(1)} = T_j^{(1)}$$

3.3.1.3 Výpočet rezerv

Časová rezerva RC_{ij} je jednoznačně stanovena pro každou činnost v projektu. Je možné ji vypočítat na základě získaných údajů z výpočtu vpřed a výpočtu vzad.

$$RC_{ij} = T_j^{(1)} - T_i^{(0)} - t_{ij} = t_j^{(1)} - t_i^{(0)} - t_{ij}$$

Vypočtená hodnota určuje časovou rezervu, kterou je možné čerpat u činností, aniž by se ohrozil termín dokončení projektu. U kritických činností je tato rezerva vždy rovna nule.[40]

3.3.2 MPM

Metoda MPM (*Metra Potencial Method*) je nejčastějším zástupcem uzlově orientovaných grafů (AON), přičemž takřka ve všech ostatních ohledech je podobná metodě CPM. MPM vznikla ve Francii, kde byla prvně použita při stavbě jaderné elektrárny na řece Loiře. Obdobnou, avšak zjednodušenou verzí MPM, je metoda PDM, která vznikla ve Spojených Státech. Cílem metody je stejně jako v předchozím případě určit časové rezervy a nalézt kritickou cestu. Stejně jako CPM, i MPM se zabývá časovou analýzou projektu, kde předpokládá deterministickou strukturu sítě i ohodnocení hran, nicméně, s využitím potenciálů hran umožňuje modelovat a počítat projekty s intervalovým zadáním některých časových parametrů.

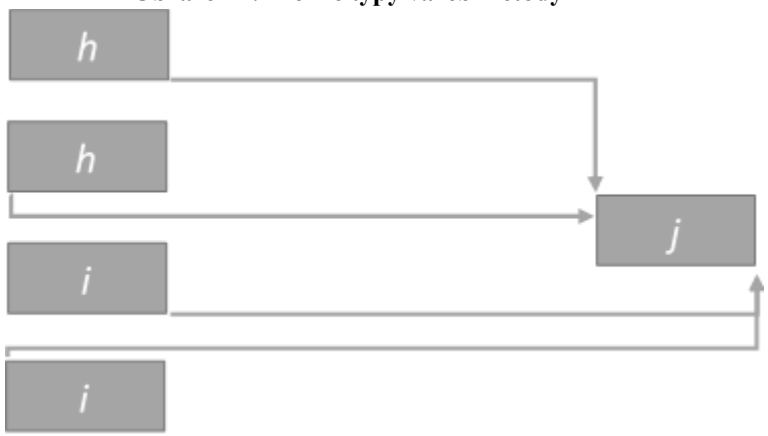
---

[39] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 87

[40] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 87

Nespornou výhodou metody MPM je její grafická podoba, což ocení především uživatelé méně znalí řízení projektů. Svou podobou totiž představuje projekt, tak jak jej vnímá běžný člověk (sled činností) a zároveň částečně připomíná Ganttův diagram. Zásadně se však liší od předchozí metody CPM v ohledu návaznosti činností. Zatímco v případě předchozí metody existuje pouze jedna možná vazba mezi dvěma činnostmi (F-S; po skončení předchozí začíná následující), MPM umožňuje použití dalších tří typů vazeb (S-S, S-F, F-F). Veškeré možnosti návaznosti činností jsou zachyceny na obrázku č. 7.[41]

Obrázek 7: Možné typy vazeb metody MPM



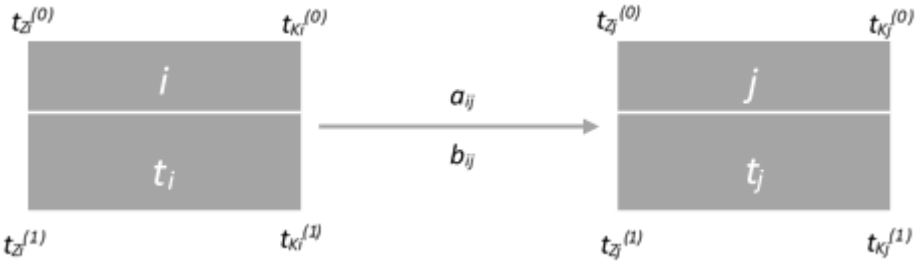
Zdroj: FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* 2004, str. 120

Pro provedení výpočtu pomocí MPM je zavedeno následující označení:

i	činnost
t_i	délka trvání činnosti
Z_i	začátek činnosti (i)
K_i	konec činnosti (i)
$t_{Zi}^{(0)}$	termín nejdříve možného začátku činnosti (Z_i)
$t_{Ki}^{(0)}$	termín nejdříve možného konce činnosti (K_i)
$t_{Zi}^{(1)}$	termín nejpozději přípustného začátku činnosti (Z_i)
$t_{Ki}^{(1)}$	termín nejpozději přípustného konce činnosti (K_i)
a_{ij}	minimální časový odstup událostí U_i a U_j
b_{ij}	maximální časový odstup událostí U_i a U_j

[41] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 119

Obrázek 8: Metoda MPM - Výpočet v grafu



Zdroj: FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* 2004, str. 122

Pro provedení výpočtu časové analýzy podle metody MPM je třeba znát veškeré činnosti *i*, délku jejich trvání t_i, vzájemné vazby mezi činnostmi *i-j* a časové odstupy činností a_{ij} a b_{ij}. Vhodné je doplnit zadání časem začátku všech počátečních činností a časem konců všech koncových činností. Vzhledem k tomu, že v grafickém znázornění se jedná o síťový graf typu AON, je vhodné jej začít a ukončit fiktivními činnostmi.

Následným provedením časové analýzy se získají termíny $t_{Zi}^{(0)}$, $t_{Ki}^{(0)}$, $t_{Zi}^{(1)}$ a $t_{Ki}^{(1)}$, pro všechny činnosti, které jsou zanesené v grafu. Výpočet opět probíhá ve dvou fázích - výpočtem vpřed a výpočtem vzad. V první fázi výpočtu se postupuje od počátečních činností ke koncovým a počítají se nejdříve možné termíny $t_{Zi}^{(0)}$ a $t_{Ki}^{(0)}$. Ve druhé fázi, výpočtu vzad, se postupuje od konce projektu k jeho začátku a zjišťují se nejpozději přípustné termíny $t_{Zi}^{(1)}$ a $t_{Ki}^{(1)}$.[42]

3.3.3 PERT

Metoda PERT (*Program Evaluation and Review Technique*) je dalším zástupcem hranově orientovaných metod. Vznikla v roce 1958 ve Spojených Státech, a to pro účely vojenského projektu *UMG-27 Polaris*, který se týkal odpalování balistických střel z ponorek. Vychází z metody CPM, od které se liší stochastickým časovým ohodnocením činností.

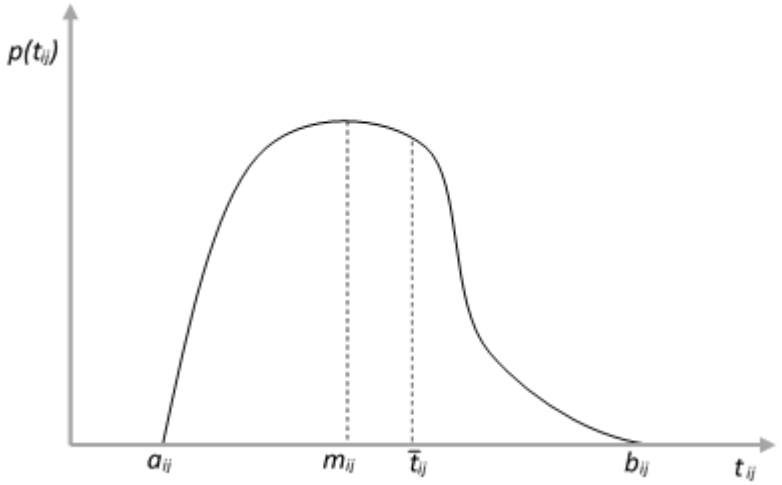
Obě dvě výše uvedené metody vycházejí z předpokladu, že činnosti projektu jsou charakterizovány jednoznačnou dobou jejich trvání. Přístup metody PERT se v tomto ohledu více snaží kopírovat reálný svět, kde se často pracuje s odhadovanými údaji, jež

[42] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 119

jsou zatíženy určitou chybou. V rámci metody PERT tak není možné zjistit přesnou dobu trvání daného projektu (deterministická struktura), nýbrž jeho očekávané dokončení s dostatečně vysokou pravděpodobností (stochastická struktura).[43]

Doba trvání libovolné činnosti není v případě modelů PERT přesně známá a je definována pouze s určitou pravděpodobností. PERT pro zjištění doby trvání činnosti využívá Beta rozdělení (β-rozdělení), které je podobné normálnímu rozdělení. β-rozdělení je spojité, jednovrcholové a většinou mírně asymetrické (příp. symetrické), na rozdíl od normálního rozdělení má konečné variační rozpětí.[44]

Obrázek 9: Beta rozdělení



Zdroj: FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* 2004, str. 96

Základním předpokladem pro výpočet je tak kvalifikovaný odhad délky trvání činností, který je k dispozici ve formě tří různých ukazatelů:

a_{ij} - optimistický odhad (nejkratší možný odhad trvání činnosti *i, j*)

b_{ij} - pesimistický odhad (nejdelší možný odhad trvání činnosti *i, j*)

m_{ij} - modální odhad (nejpravděpodobnější odhad trvání činnosti *i, j*)

Na základě znalostí výše uvedených ukazatelů je uživatel schopen vypočítat základní statistické charakteristiky činností. Těmi jsou střední hodnota doby trvání a rozptyl. Ty

[43] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0. str. 27

[44] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X. str. 95

odpovídají příslušným charakteristikám β-rozdělení a vypočtou se podle následujících vzorců:

$$\mu(t_{ij}) = t_{ij}^{e} = \frac{a_{ij} + 4m_{ij} + b_{ij}}{6}$$

$$\sigma^2(t_{ij}) = \left(\frac{b_{ij} - a_{ij}}{6}\right)^2$$

S rostoucí hodnotou rozptylu roste i pravděpodobnost, že se skutečná doba trvání činnosti bude více lišit od střední hodnoty. Cílem je výpočet středních hodnot a rozptylů všech činností a na jejich základě určení tzv. očekávané kritické cesty.[45]

Výpočet v této metodě probíhá obdobně jako u metody CPM s tím, že jsou zjišťovány také hodnoty $\sigma^2(t_j^0)$ a $\sigma^2(t_j^1)$, které jsou vypočteny postupným sčítáním rozptylů podél nejdelší cesty do uzlu *j*, resp. z uzlu *i* do koncového. Na základě výpočtu je možné stanovit očekávanou dobu trvání projektu, výsledný rozptyl této doby a také očekávanou kritickou cestu. Vzhledem k tomu, že veškeré veličiny jsou stochastické, existuje jistá pravděpodobnost, že projekt bude mít kratší nebo delší dobu trvání.

Ke zjištění této pravděpodobnosti *P*, že projekt bude dokončen předčasně, se použije vztah, ve kterém *F* charakterizuje distribuční funkci, ze které po úpravě získáme finální podobu vzorce:

$$P(T_n \leq T_s) = F\left(\frac{T_s - T_n^0}{\sqrt{\sigma^2(T_n^0)}}\right)$$

Ke zjištění pravděpodobnosti překročení daného termínu se používá upravený vzorec:

$$P(T_n > T_s) = 1 - P(T_n \leq T_s)$$

PERT je i v dnešní době relativně často používanou metodou. Jeho hlavní výhodou je stochastický přístup, který poskytuje spolehlivé informace s reálným nadhledem a současná jednoduchost a částečná kompatibilita s CPM.[46]

[45] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0. str. 27-28

[46] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0. str. 29-31

3.3.4 GERT

GERT (*Graphical Evaluation and Review Technique*) je další metodou, která ke svému grafickému zobrazení užívá hranově ohodnocený síťový graf (AOA). Vznikla v roce 1966, kdy ji prvně popsal *Dr. Alan B. Protsker* v dokumentech pro účely Národního úřadu pro letectví a kosmonautiku (NASA). Na rozdíl od metody PERT, která je stochastická ve svých činnostech je GERT metodou se stochastickou strukturou (topologií) grafu.

Metoda, která se zabývá především časovou analýzou projektu, má za cíl kromě nalezení kritické cesty také spočítat pravděpodobnost, za které se budou její jednotlivé činnosti realizovat a stanovit tak celkovou pravděpodobnost úspěšného dokončení celého projektu. Hrany, představující činnosti, mají přiřazenou podmíněnou pravděpodobnost p_{ij}, která označuje pravděpodobnost jejich realizace, pokud bude realizován i uzel, ze kterého daná hrana vystupuje. K tomu je třeba rozlišovat i jednotlivé vstupy a výstupy uzlu, které se řídí svou vlastní logikou. Jejich kombinace jsou zachyceny v následující tabulce č. 3.[47]

Tabulka 3: Tabulka možných kombinací uzlů metody GERT

Tabulka možných kombinací uzlů	KONJUNKTIVNÍ	DISJUNKTIVNÍ	INKLUZIVNÍ
DETERMINISTICKÝ	 Konjunktivně deterministický	 Disjunktivně deterministický	 Inkluzivně deterministický
STOCHASTICKÝ	 Konjunktivně stochastický	 Disjunktivně stochastický	 Inkluzivně stochastický

Zdroj: FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* 2004, str. 113

[47] ŠUBRT, Tomáš a Jan BARTOŠKA. *Projektové řízení: (měkké a pokročilé přístupy).* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2007. ISBN 978-80-213-1725-3. str. 4-6

Jak je v tabulce znázorněno, tyto tři možné vstupy a dva možné výstupy je možné vzájemně libovolně kombinovat. U zobecněného síťového grafu metody GERT tak vždy nastává jedna ze šesti možných kombinací. Význam jednotlivých typů uzlů včetně jeho charakteristiky je následující:

- **Konjunktivní vstup** - Uzel se realizuje právě tehdy, když se realizují veškeré vstupující činnosti.
- **Disjunktivní vstup** - Uzel se realizuje právě tehdy, když se realizuje právě jedna vstupující činnost.
- **Inkluzivní vstup** - Uzel se realizuje právě tehdy, když se realizuje alespoň jedna vstupující činnost.
- **Deterministický výstup** - Realizace uzlu vyvolá realizaci všech vystupujících činností.
- **Stochastický výstup** - Realizace uzlu vyvolá realizaci právě jedné vystupující činnost.[48]

Zajímavostí metody GERT je, že se na rozdíl od ostatních, výše zmíněných metod, stala metodou často modifikovanou a inovovanou. Výše uvedený zápis a forma aplikace metody tak není jediná ani původní, avšak pravděpodobně nejvíce rozšířená forma metody GERT. S rozvojem počítačové technologie na začátku 70. let 20. století obohatily metodu simulace. Pro výpočet metody GERT se začala používat simulační metoda Monte Carlo a také Markovovy řetězce s podporou simulací. I ve 21. století byl vyvíjen nový simulační přístup pro tuto metodu, který odpovídal aktuálním požadavkům trhu.[49]

Z předchozího odstavce je zřejmé, že metoda má ještě v dnešní době značný význam. Její přístupy se používají v oblasti řízení rizika, dále v oblasti dynamického plánování, nebo při rozhodování za nejistoty. Zajímavostí je, že Fisher a Goldstein v roce 1983 použili metodu GERT při analýze kognitivního chování. V roce 2004 použil Kosugi tuto metodu k vyhodnocení energetické účinnosti ve výzkumných projektech, které se zabývají technologií zachycení CO_2.[50]

---

[48] ŠUBRT, Tomáš a Jan BARTOŠKA. *Projektové řízení: (měkké a pokročilé přístupy).* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2007. ISBN 978-80-213-1725-3. str. 6

[49] NELSON, Richar Grahan a AZARON, Amir. *The use of a GERT based method to model concurrent product development processes.* European Journal of Operational Research. Elsevier, 2016, Vol. 250, Issue 2, (April, 2016), 566-578. str. 568

[50] ZHOU, Li. *Forecasting return of used products for remanufacturing using GERT.* International Journal of Production Economics, 2016, Vol. 181, part B, (November, 2016), 315-324., str. 316-317

3.3.5 VERT

Metoda VERT (*Venture Evaluation and Review Technique*) patří mezi méně známé a relativně nové metody pro řízení projektů. V roce 1972 ji prvně představil Gerard L. Moeller, který ji připravil na zakázku pro ministerstvo obrany Spojených států. Teprve koncem sedmdesátých let byla metoda veřejně představena a do poloviny osmdesátých let byla metoda ještě několikrát upravena a doplněna. Na těchto úpravách se podíleli především Lester A. Digman a Sang M. Lee.

VERT vychází ze známých a nejčastěji používaných metod CPM a PERT, se kterými se shoduje v hranové orientaci síťového grafu. Metoda vychází ze vzájemného vztahu mezi časem, náklady a výkonem, přičemž každá činnost je charakterizována právě těmito třemi faktory. Zatímco v případě času a nákladů je zcela logicky cílem minimalizace těchto faktorů, tak v případě výkonu se snažíme o maximalizaci. Výkon je v těchto případech modelován pomocí měřitelných jednotek nebo bezrozměrných indexů.

Schopnost metody spolehlivě reprezentovat „reálný svět“ v rámci projektu spočívá ve vysoké flexibilitě uzlů (rozhodovacích bodů), široké možnosti využití nejrůznějších statistických rozdělení pro charakterizování hran, opakovanému simulování projektu nebo již zmíněné kombinaci faktorů času, nákladů a výkonu pro vyjádření hodnoty jednotlivých činností. Metoda je tak velmi přesná a především díky využití simulací efektivní při poskytování časových odhadů k dokončení projektu. Na druhou stranu je nutné zmínit, že VERT je především složitý nástroj a sestavení modelu je časově velmi náročné. Z důvodu vysoké výpočetní složitosti a náročnosti při sběru dat se metoda nikdy nestala široce uznávaným nástrojem jako metody CPM a PERT, dokonce je v současnosti méně používaná než metoda GERT.[51]

Metodě VERT a jejím jednotlivým modifikacím je ponechán dostatečný prostor v *kapitole 4*, a proto se o ní na tomto místě autor této práce nebude více rozepisovat.

[51] CATES, G., R. *Improving project management with simulation and completion distribution functions*; Dostupné online z: http://etd.fcla.edu/CF/CFE0000209/Cates_Grant_R_200412_PhD.pdf, str. 29-30 [cit. 31.1.2017]

4 Metoda VERT

V této kapitole je detailně rozebrána metoda VERT včetně všech jejich známých modifikací. Pro lepší pochopení vývoje, zachycení časových odstupů a návazností na vývoj v počítačové technice je nejprve přiblížena historie této metody. V následující podkapitole je stručné srovnání s ostatními metodami z hlediska použití a pohledu na jednotlivé aspekty. Zbylá část této kapitoly zahrnuje samostatný popis metody VERT ve všech jejích známých modifikacích, přičemž největší rozsah informací je poskytnut u verze VERT2, na základě které je v kapitole 6 popsán vývoj aplikace.

4.1 Historie a vývoj metody VERT

Jak již bylo zmíněno v podkapitole 3.1, metoda VERT vznikla na základě požadavku armády Spojených Států v roce 1972. Autorem původní verze metody byl Gerald L. Moeller. Původní verze VERT (jako jedna z prvních metod) dala analytikovi možnost při modelování sítě projektu zohlednit čas, náklady a výkon na stejné úrovni, zatímco v předchozích metodách bylo obvyklé plánovat projekt čistě na základě času. Tato možnost realističtěji představovala skutečný svět a upozorňovala na skryté hrozby a nástrahy v projektu, které bylo doposud obtížné specifikovat. Novinkou byla také možnost definovat kterýkoliv parametr (čas, náklady, výkon) kterékoliv činnosti prostřednictvím matematického vztahu, jenž vychází z jiné činnosti. Nutno podotknout, že tato funkce byla tehdy ještě pro uživatele nepropracovaná.[52]

Metoda VERT2 byla sestrojena v roce 1979 Geraldem L. Moellerem s přispěním Lester A. Digmana. Významným posunem bylo omezení zadávání hran prostřednictvím matematických vztahů, které takřka kompletně nahradilo užitečnější zadávání pomocí statistických rozdělení. Novinkou bylo také rozšíření o 6 nových typů uzlů.[53]

VERT 3 vznikl v roce 1981 a společně s drobnou úpravou vznikl i zcela nový software pro řešení metody. Na úpravách se podílel především Sang M. Lee, avšak s podporou dvou původních autorů. Změna se nejvíce týkala typů uzlů vstupní a výstupní logiky, která

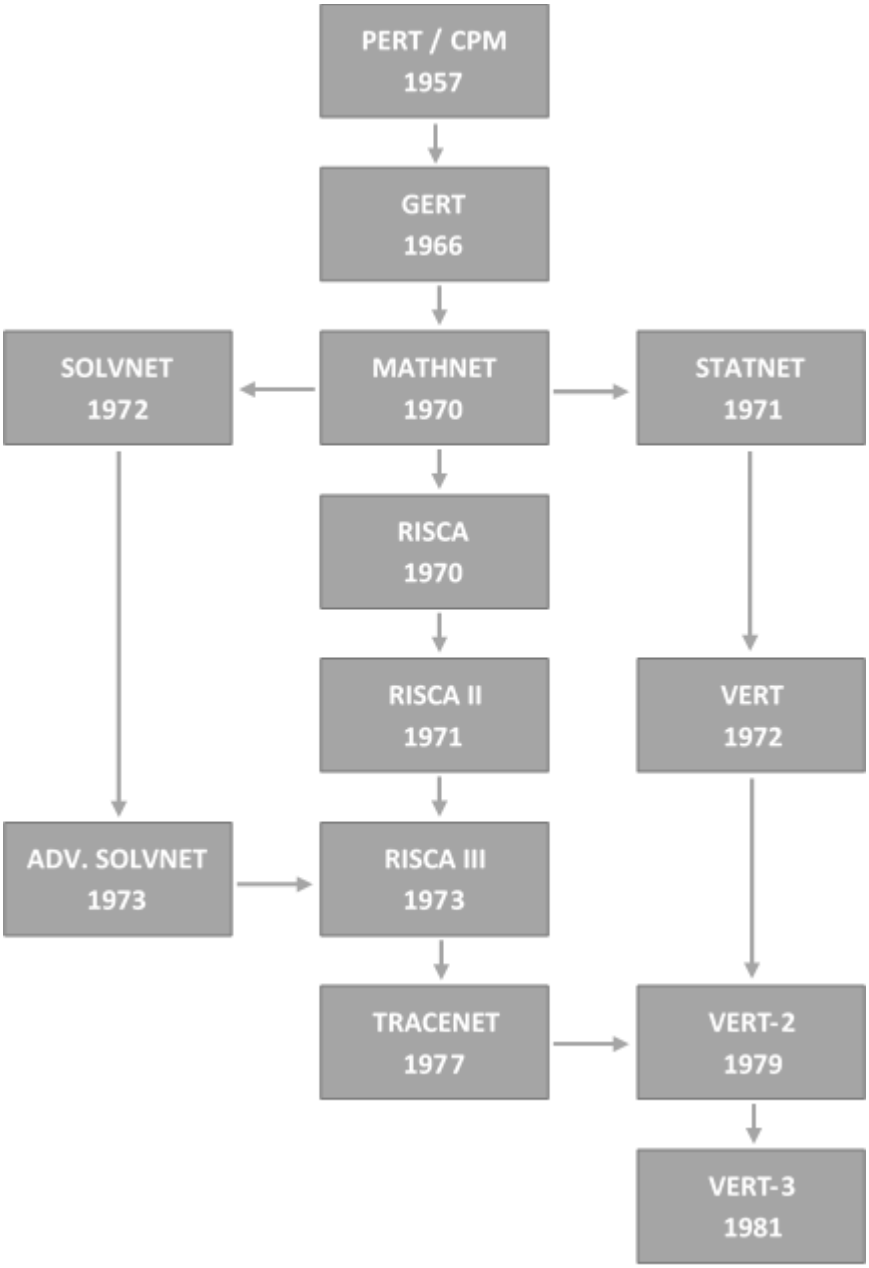
[52] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach.* Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 33

[53] MOELLER, Gerald L. a Lester A. DIGMAN. *Operations Planning with VERT.* Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 678-679

metodu učinila konzistentnější. Zásadní úpravou bylo definování padesáti matematických vztahů, které vyjadřují vzájemnou závislost mezi faktory činností, čímž se zjednodušila původní myšlenka stanovení libovolné hodnoty podle matematického vztahu.[54]

Historický vývoj metody VERT, ale i dalších matematicko-statistický přístupů, které vznikly zhruba do roku 1980, je zachycen pod tímto textem na obrázku č. 10.

Obrázek 10: Historický vývoj projektových metod



Zdroj: LEE, Sang M. a kol. *Network Analysis for Management Decision.* 1982, str. 29

[54] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach.* Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 33

4.2 Porovnání s ostatními metodami

Metoda VERT se pochopitelně nevyhnula srovnání s jinými metodami z různých hledisek. Nejčastěji je VERT porovnáván s metodou PERT a na toto srovnání se zaměřil i John B. Kidd, jehož závěrům je věnován první odstavec textu. Ve druhém odstavci je objektivnější porovnání s celou řadou dalších metod.

John B. Kidd ve svém článku porovnává aplikaci nejčastějších přístupů pro řešení čtyř náhodně vybraných síťových grafů. Důkladnou pozornost však věnuje porovnání metod PERT a VERT. Mírně preferuje metodu PERT především v úlohách, kde leží kritická cesta na dominantní cestě, které VERT nepřikládá dostatečný význam. Zároveň ale uznává výhodu simulací v řízení projektů. V diskuzi upozorňuje, že při zjišťování délky projektu metoda VERT podává na první pohled podobné výsledky jako PERT, ale pokud zohledníme délku trvání samotných činností, VERT je výrazně přesnější. V závěru autor přisuzuje velkou sílu metody VERT v možnosti výpočtu nákladů a výkonu projektu.[55]

Sang M. Lee se s ostatními autory rozhodl porovnat metodu VERT3 s metodami CPM, PERT, LOB[56] a GERT v řadě hledisek.[57] Z nich jasně vyplývá, že VERT (a částečně také GERT) vychází z metody PERT a mají tak velmi mnoho společného – definované rozdělení projektu na fáze, vyšší náklady na výpočet a aktualizace (modifikace) metod není jednoduchá. Samotný VERT proti ostatním metodám zkoumá hned tři parametry (čas, náklady a výkon) a je možné jej uplatnit i při analýze rizika. Vzhledem k užití simulací také pojme značnou většinu činností a výrazný počet různých režimů.[58]

4.3 VERT 1

Vznik metody VERT1 se váže k roku 1972, kdy ji na žádost americké armády sestrojil Gerald L. Moeller. VERT1 (tou dobou označovaný pouze jako VERT) se stal oblíbenou a často používanou pomůckou při plánování a řízení válečných operací, ale také zrychlil

[55] KIDD, John. *A Comparison Between the VERT Program and Other Methods of Project Duration Estimation.* Omega: The International Journal of Management Science. Pergamon Journals Ltd., 1987, Vol. 15, No. 2, 129-134. str. 131-133

[56] Line-of-Balance – Metoda plánování a řízení projektu, který má opakující se povahu a je tedy ve vybraných případech daleko užitečnější než klasické metody (např. CPM).

[57] Celá tabulka je k dispozici v příloze této práce.

[58] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach.* Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 41-43

rozvoj a výrobu zbraní (tanků, vrtulníků, stíhacích letounů, dělostřelectva, automatických houfnic a elektronických senzorů), zlepšil systém protivzdušné obrany a podílel se na řadě dalších souvisejících projektů.[59]

Přestože se jednalo o velmi přesnou a úspěšnou metodu, nebyla z výše uvedených důvodů v původní podobě do konce 70. let představena veřejnosti. Tou dobou začaly metodu používat i jiné státní instituce Spojených Států, např. pro plánování ochrany před povodněmi, pro analýzu zemětřesení a pro snížení znečištění řek.[60]

Metodu ve své původní podobě provázely velmi složité výpočty, které vycházely z definic jednotlivých činností. Ty byly nejčastěji vyjádřeny formou matematických vztahů, které se vázaly k jinému vybranému parametru. Moeller představil původně 37 základních matematických vztahů, které mohly být použity k vyjádření hodnoty parametru, ale zároveň doplnil, že je uživatel může dle vlastní potřeby upravit, čímž poskytl neomezenou škálu možností. Kromě toho mohly být tyto parametry činností vyjádřeny pomocí vybraných statistických rozdělení, což byla ale daleko méně preferovaná varianta.[61]

V roce 1979 byla metoda publikována zveřejněním vybraných původních materiálů v odborných článcích a uvedením nové (doplněné) metody VERT2.

4.4 VERT 2

Metoda VERT2 vznikla v roce 1979 a kromě původního autora, Gerald L. Moellera, se na jejím doplnění a inovaci podílel Lester A. Digman. Metoda byla sepsána především pro podnikatelskou sféru jako nástroj pro plánování a rozhodování při nedostatku informací.[62] V ohledu metod pro řízení projektů za daných podmínek nebyl v této době k dispozici lepší nástroj než VERT, který uměl modelovat i extrémně složité rozhodovací procesy. Největším rozdílem proti původní metodě bylo utlumení používání matematických vztahů pro stanovení parametrů činností. Ty byly takřka v celém rozsahu nahrazeny používáním třinácti statistických rozdělení. Pro stanovení parametru činnosti původní cestou tak nově

[59] MOELLER, Gerald L. a Lester A. DIGMAN. *Operations Planning with VERT.* Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 692-693

[60] MOELLER, Gerald L. *Venture Evaluation and Review Technique: Decision Models Directorate*. US Army Armament Material Readiness Command. Rock Island, Illinois, 1979. str. 1-4

[61] MOELLER, Gerald L. *Venture Evaluation and Review Technique: Decision Models Directorate*. US Army Armament Material Readiness Command. Rock Island, Illinois, 1979. str. 24-30

[62] Toto rozhodování za rizika efektivně řešily především simulace.

nebylo možné použít předdefinované vztahy, ale celý vzorec musel být zapsán uživatelem. Ti ale tuto změnu spíše uvítali.[63]

Vzhledem k tomu, že metoda byla detailně popsána až ve své první modifikaci (tedy VERT2) a tento popis byl shodou okolností také detailně cílený pro uživatele i z běžné obchodní sféry, rozhodl se autor této práce vysvětlit metodu právě na této verzi.

4.4.1 Síťový graf

Podobně jako u jiných projektových metod, tak i VERT využívá u grafického zobrazení hrany a uzly. Metoda VERT je hranově orientovaná metoda, což znamená, že uzly představují milníky nebo body pro rozhodování, zatímco hrany představují aktivity, neboli činnosti. Každá činnost je jednoduše charakterizována třemi parametry - spotřebovaným časem, náklady a generovaným výkonem při dokončení dané činnosti. Stejně jako u jiných projektových metod, je dokončení projektu v jeho grafickém modelu reprezentováno postupným tokem v síti a dosažením daných uzlů a hran.[64]

4.4.2 Vstupní a výstupní logika

Projektová metoda VERT umožňuje aplikaci dvou různých typů uzlů, přičemž každý z nich má svůj začátek, konec a specifický průběh během toku v síti. Tím jednodušším, a méně používaným, je uzel s „Jednoduchou logikou“ (Single-unit Logic). V rámci tohoto typu uzlu existují celkem čtyři různé možnosti průtoku, které mají každá svůj specifický počátek a konec. Jejich použití je celkem neobvyklé.

Mnohem zajímavější, a v případě VERT také používanější metodou, je uzel se „Složenou logikou“ (Split-node Logic). Ten nabízí využití hned čtyř možných vstupů činnosti do uzlu a šest různých výstupů činnosti z uzlu. Právě tento přístup, který věrohodně kopíruje různorodost reálného světa (jež je v jiných metodách problém zaznamenat) dělá metodu VERT extrémně přesnou, nicméně však velmi složitou při sběru a následném zadávání dat.

---

[63] MOELLER, Gerald L. a Lester A. DIGMAN. Operations Planning with VERT. Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 676-679

[64] MOELLER, Gerald L. a Lester A. DIGMAN. Operations Planning with VERT. Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 679

Při grafickém zobrazení se jedná o zobrazení grafu se stochastickou topologií, který je charakteristický tím, že neposkytuje zcela jednoznačné vazby mezi činnostmi (podobně jako třeba GERT).[65]

4.4.2.1 Jednoduchá logika

Jednoduchá logika je méně sofistikovaným přístupem při aplikaci metody VERT. Využívá se pouze při speciálních projektech, a to velmi vzácně. V jejím využití se setkáme se čtyřmi základními typy uzlů, kterými jsou POROVNÁVAJÍCÍ (compare), PREFERUJÍCÍ (preferred), ŘADÍCÍ (queue) a USPOŘÁDÁVAJÍCÍ (sort).

- **POROVNÁVAJÍCÍ (compare) logika** - Tato logika iniciuje začátek vystupujících hran podle funkce kumulovaných hodnot času, nákladů a výkonu, které charakterizují vstupující hrany.
- **PREFERUJÍCÍ (preferred) logika** - Tato logika iniciuje začátek vystupujících hran podle funkce uživatelem zvolených preferencí.
- **ŘADÍCÍ (queue) logika** - Tato logika simuluje prostředí front, ve kterém jeden nebo více systémů obsluhuje příchozí toky v závislosti na čase, nákladech nebo výkonu, které náleží vstupujícím hranám.
- **USPOŘÁDÁVAJÍCÍ (sort) logika** - Poslední logika přijímá tok v síti ze vstupujících hran a sekvenčním způsobem je postupně transformuje do výstupních hran, jako funkce uživatelem nastavených preferencí z hlediska času, nákladů a výkonu, nebo kombinací těchto faktorů.[66]

4.4.2.2 Složená logika

Mezi vstupní logiky, které se aplikují v uzlech se složenou logikou, patří POČÁTEČNÍ (Initial), A (and), ČÁST Z A (partial and) a NEBO (or).

- **POČÁTEČNÍ (Initial) vstupní logika** - Tato logika figuruje pouze na začátku projektu v prvním uzlu. Na rozdíl od jiných metod, VERT umožňuje zapojení více

[65] MOELLER, Gerald L. a Lester A. DIGMAN. Operations Planning with VERT. Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 680
[66] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 52

počátečních uzlů do projektu, přičemž všem je na začátku přidělen totožný čas, náklady a výkon. Mohou se však lišit v druhé (výstupní) části uzlu.

- **A (and) vstupní logika** - Vstupní logika vyžaduje, aby veškeré vstupující činnosti do uzlu byly dokončeny dříve, než se tok v rámci uzlu přesune do výstupní části a její logiky. Čas, který je uveden pro tento uzel, je vypočten jako jeden nejvyšší kumulovaný čas ze všech vstupujících činností od začátku projektu. Náklady jsou počítány jako součet všech nakumulovaných nákladů a výkon jako celková hodnota výkonu všech vstupujících hran. Z hlediska vstupní logiky si nemůžeme nevšimnout nápadné podobnosti s konjunktivním vstupem metody GERT.
- **ČÁST Z A (partial and) vstupní logika** - Logika je ve všech svých ohledech podobná předchozí vstupní logice s výjimkou, že pro realizaci daného uzlu je nezbytné, aby byla dokončena alespoň jedna z předchozích činností. Nicméně, před přechodem do výstupní části uzlu se bude čekat na dokončení všech nedokončených činností. Výpočet času, nákladů a výkonu je totožný s výpočtem u předchozí vstupní logiky. I v tomto případě vidíme jistou podobnost s metodou GERT, neboť vstupní logika odpovídá inkluzivnímu vstupu.
- **NEBO (or) vstupní logika** - Poslední vstupní logika vyžaduje dokončení alespoň jedné vstupující činnosti. Na rozdíl od předchozí logiky však v okamžiku jejího dokončení nečeká logika na dokončení ostatních činností a přesouvá se v rámci uzlu do jeho výstupní části. Čas, náklady a výkon, které charakterizují tento uzel, jsou vypočítány, jako součet všech nakumulovaných hodnot z předchozí dokončené činnosti. Náklady se vypočítají jakou součet všech kumulovaných nákladů na všech aktivních vstupujících činnostech (což znamená, že se při výpočtu počká na jejich dokončení). Stejně jako v předchozích případech, i tady existuje podobnost s disjunktivním vstupem metody GERT.

Cílem následující výstupní logiky uzlů se složenou logikou je dle potřeby rozdělit tok v rámci sítě do libovolného počtu následujících činností. Je dobré zdůraznit, že k hodnotám času, nákladů a výkonu výstupních činností se připočítávají kumulované hodnoty z předchozích činností, které vstoupily v předchozím kroku do daného uzlu.

Těchto šest výstupních logik, které se používají u uzlů se složenou logikou, se nazývají KONEČNÁ (terminal), VŠE (all), SIMULACE M.C. (monte carlo), FILTR 1 (filter 1), FILTR 2 (filter 2) a FILTR 3 (filter 3).

- **KONEČNÁ (terminal) výstupní logika** - Logika se používá v případě absence vystupujících činností z daného uzlu jako koncový bod v síti. Může jich být i více.
- **VŠE (all) výstupní logika** - Tato výstupní logika současně iniciuje začátek všech vystupujících činností z daného uzlu. I tady si můžeme všimnout (stejně jako ve vstupní logice) s podobností u výstupních uzlů metody GERT. Tato výstupní logika odpovídá deterministickému uzlu.
- **SIMULACE MONTE CARLO (monte carlo) výstupní logika** - Výstupní logika iniciuje zpracování jedné hrany vycházející z daného uzlu, která je vybrána za použití jediné iterace simulační metody Monte Carlo (náhodného čísla). Vystupující činnosti jsou ohodnoceny pravděpodobnostními vahami jejich realizace dle rozhodnutí uživatele, přičemž součet těchto vah je 1,0.
- **FILTR 1 (filter 1) výstupní logika** - Tato výstupní logika iniciuje začátek jedné nebo více vystupujících činností, které jsou vybrány na základě jejich individuální nebo kumulované hodnoty parametrů času, nákladů a výkonu. Ty jsou porovnány s omezujícími podmínkami (minimem a maximem) času, nákladů a výkonu, nastavenými pro danou výstupní logiku zkoumaného uzlu. Pokud náleží hodnota parametrů dané vystupující činnosti v těchto intervalech, dojde k její zpracování. Analogicky v opačném případě daná vystupující činnost realizovaná nebude. Omezující podmínky výstupní logiky mohou být následně děleny na průchozí, částečně se překrývající nebo neprůchozí.
- **FILTR 2 (filter 2) výstupní logika** - Výstupní logika se výrazně podobá předchozí (FILTR 1), od které se liší jen ve dvou věcech. Za prvé, logika sleduje pouze jedno omezení u vystupujících činností, které má svou horní a dolní hranici a které vychází z činností, jež vstupují do daného uzlu. Za druhé je logika omezena na spolupráci pouze s ČÁST Z A (inkluzivní) vstupní logikou.
- **FILTR 3 (filter 3) výstupní logika** - Poslední výstupní logika pro metodu VERT se využívá zřídka, a to především v situacích, kdy je třeba rozhodnout o dalším vývoji projektu na základě zcela odlišných dat. Na rozdíl od předchozích dvou filtrů, v tomto případě zde nefigurují intervalová omezení pro identifikaci vstupujících a vystupujících činností. Klíčové jsou činnosti, které vstupují do zkoumaného uzlu a kterým jsou přidělena znaménka plus (+) a mínus (-) na základě jejich významnosti pro další vývoj projektu. Potom počet vystupujících činností

z daného uzlu při této logice odpovídá (rovná se celkovému počtu vstupujících činností, které jsou realizovány a zároveň nesou označení plus (+)).[67]

Podle výše uvedených informací můžeme definovat celkem 18 různých typů uzlů, které mohou být v metodě VERT použity. Šest různých kombinací není možných na základě logiky nebo požadavků metody a daného typu uzlu. Tím je pochopitelně myšlena výstupní logika FILTR 2, která je omezena na kooperaci pouze se vstupní logikou ČÁST Z A, a nepřipouští tak možnost vytvoření dalších 3 typů uzlů. Dále je nezbytné vyloučit jakékoliv postavení filtru na začátku projektu, tedy uzly ve tvaru POČÁTEČNÍ – FILTR 1, FILTR 2 a FILTR 3. Logiku nemá ani kombinace uzlu POČÁTEČNÍ – KONEČNÁ. Veškeré možné kombinace jsou zachyceny v tabulce č. 4 pod tímto textem.

Tabulka 4: Možnosti rozhodovacích uzlů metody VERT2

Tabulka možných kombinací rozhodovacích uzlů	KONEČNÁ	VŠE	MONTE CARLO	FILTR 1	FILTR 2	FILTR 3
POČÁTEČNÍ	✗	✓	✓	✗	✗	✗
A	✓	✓	✓	✓	✗	✓
ČÁST Z A	✓	✓	✓	✓	✓	✓
NEBO	✓	✓	✓	✓	✗	✓

Zdroj: Vlastní zpracování

4.4.3 Zadávání parametrů činností

Zadávání parametrů jednotlivých činností (tedy času, nákladů a výkonu) v rámci VERT2 je možné několika způsoby. Nejvíce oblíbený je však v této metodě způsob pomocí statistických rozdělení. V případech, kdy není možné pro klasifikaci statistické funkce použít některé z výše uvedených statistických rozdělení, umožňuje VERT zadat údaje také prostřednictvím histogramu, který je novinkou první modifikace této metody. Možné je

[67] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 50-52

i definování matematického vztahu, který však musí být kompletně zadán uživatelem. Přímé zadání konstanty v této modifikaci nebylo překvapivě uživateli umožněno. Variantou tak je zadání prostřednictvím velmi jednoduchého matematického vztahu nebo jako maximální a minimální hodnota statistického rovnoměrného rozdělení.

Při svém použití nabízí projektová metoda VERT řadu různých statistických rozdělení. Ta mohou být použita k částečnému nebo kompletnímu modelování délky trvání dané činnosti, vyčíslení nákladů nebo jejímu výkonu. Mezi těchto 13 rozdělení patří:

- Rovnoměrné rozdělení (*Uniform*)
- Trojúhelníkové rozdělení (*Triangular*)
- Normální rozdělení (*Normal*)
- Logaritmicko-normální rozdělení (*Lognormal*)
- Gama rozdělení (*Gamma*)
- Weibullovo rozdělení (*Weibull*)
- Erlangovo (exponenciální) rozdělení (*Erlang*)
- Chí-kvadrát rozdělení (*Chi square*)
- Beta rozdělení (*Beta*)
- Poissonovo rozdělení (*Poisson*)
- Pascalovo (geometrické) rozdělení (*Pascal*)
- Binomické rozdělení (*Binomial*)
- Hypergeometrické rozdělení (*Hypergeometrical*)

Zadávání parametrů činností prostřednictvím matematických vztahů nebylo u VERT2 příliš obvyklé a rozšíření (tedy přednastavené způsoby zadávání) se objevila až v metodě VERT3.[68]

4.4.4 Proces konstrukce sítě

V této části je naznačeno, jak probíhá konstrukce síťového grafu, který reprezentuje tok mezi činnostmi a uzly. Konstrukce síťového grafu, tak jako u každé jiné metody, vyžaduje znalost a úsudek na straně uživatele. Na následujících řádcích Vám autor této práce na

[68] MOELLER, Gerald L. a Lester A. DIGMAN. Operations Planning with VERT. Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 681-682

svém vlastním příkladu popíše, jak probíhá sestavení základního síťového grafu metody VERT.

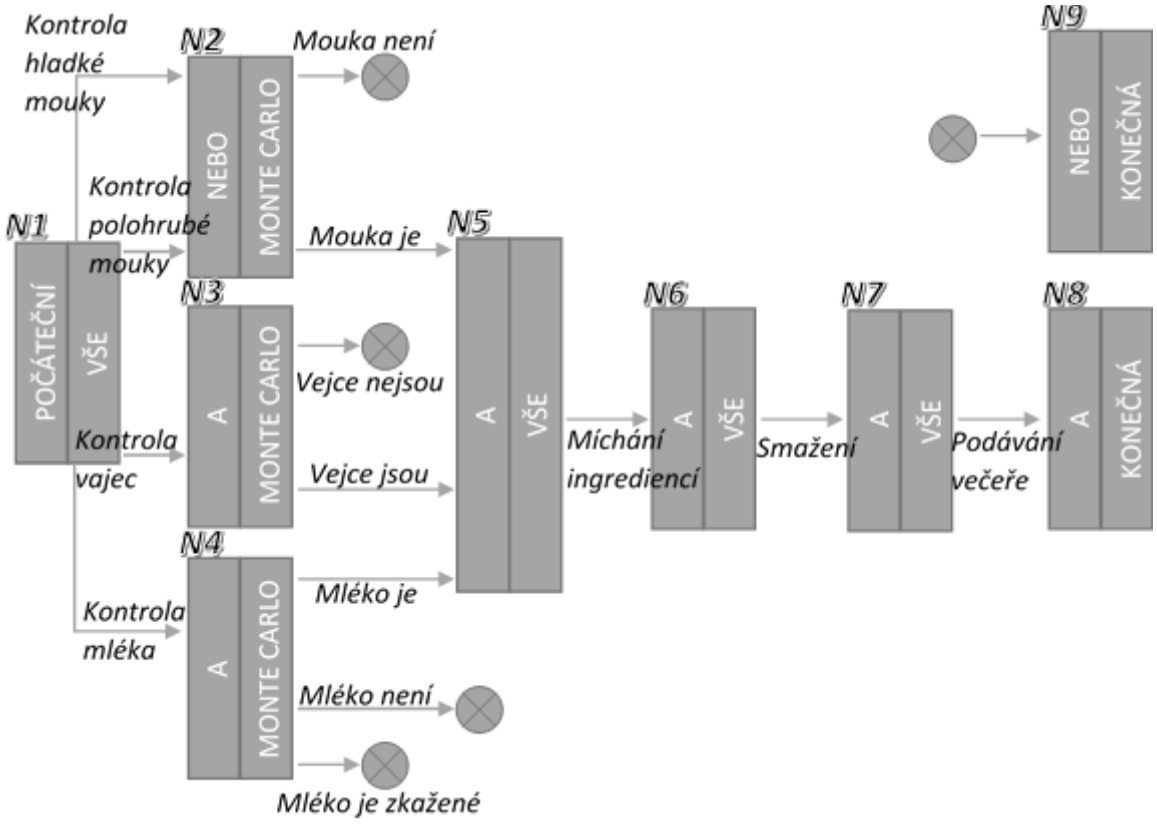
4.4.4.1 Ekonomický model

Konstrukce síťového grafu je vysvětlena na příkladu ženy, která dětem vaří večeři. Paní Eva se vrací večer z práce. Doma je vítána dětmi, které touží po tom, aby jim připravila oblíbené palačinky. Ona jim ráda vyhoví, ale není si jistá, zda má potřebné ingredience - mouku, vejce a mléko. Pokud ano, může pokračovat v dalších činnostech, kterými jsou míchání těsta, smažení a podávání jídla. Pokud bude takto postupovat, projekt bude úspěšně dokončen. Na druhou stranu tu stále je pravděpodobnost, že žena doma nebude mít polohrubou, ani hladkou mouku, žádná vejce a nebo bude postrádat mléko. Mléko se vzhledem ke své krátké trvanlivosti může ještě zkazit. Pokud některá z těchto možností nastane, nebude schopna připravit palačinky a tedy dokončit úspěšně tento projekt.

4.4.4.2 Matematický model

Síťový graf může být vytvořen tak, jak je uvedeno na obrázku č. 11 pod textem.

Obrázek 11: Konstrukce sítě daného problému



Zdroj: Vlastní zpracování

Obrázek zachycuje sled činností (hran) a rozhodovacích procesů (uzlů), které znázorňují výše popsaný „palačinkový“ problém. První uzel (N1), označený jako START má POČÁTEČNÍ vstupní logiku, která označuje začátek tohoto projektu a VŠE výstupní logiku, která iniciuje začátek všech vystupujících činností. Uzly N2, N3 a N4 mají výstupní logiku SIMULACE MONTE CARLO, která iniciuje začátek pouze jedné činnosti vystupující z daného uzlu, která vychází z pravděpodobnosti jednotlivých výstupů. Pro příklad uveďme možné situace v rozhodovacím uzlu N4:

Mléko je k dispozici	75 %
Mléko máme, ale je zkažené	10 %
Mléko není k dispozici	15 %

Ve chvíli, kdy je při metodě VERT spuštěn výpočetní proces, tak výstupní logika SIMULACE MONTE CARLO náhodně vybere jednu ze tří možností na základě pravděpodobnosti jejího výskytu. Podobná logika platí i pro uzly N2 a N3.

Za povšimnutí stojí uzel N2 i ve své vstupní logice NEBO - pro výrobu palačinek může být použita hladká ale i polohrubá mouka. Ke splnění projektu tak postačí, pokud bude domácnost vybavena alespoň jedním z těchto druhů mouky, přičemž při nalezení nebude druhý druh mouky vůbec hledán.

Ve chvíli, kdy jsou všechny ingredience k dispozici, může tok pokračovat do uzlu N5 se vstupní logikou A, a dále pak až do uzlu N8, který označuje úspěšné dokončení projektu. Pakliže by nezbytné ingredience v uzlech N2, N3 a N4 nebyly k dispozici nebo by mléko bylo zkažené, pokračuje tento tok (prostřednictvím symbolu) do alternativního konce N9, který charakterizuje neúspěšné dokončení projektu. Podobně jako u uzlu N2, tak i N9 má NEBO vstupní logiku, což znamená, že neúspěšný konec nastane vždy, když nebude splněna byť jen jedna podmínka v uzlech N2, N3 a N4.

Pokud bude tento problém opakovaně simulován, bude možné vzájemně poměřit procentuální poměr úspěšného a neúspěšného dokončení projektu, což je vhodný podklad k provedení analýzy rizika.

4.4.4.3 Řešení úlohy

Pokud by do tohoto síťového grafu byla přidána variabilní doba každé činnosti, můžeme snadno generovat graf rozložení dob dokončení projektu. Obstarání chybějících ingrediencí totiž prodlužuje délku trvání projektu, a pokud žena chce, aby děti večeřely v přesně určenou denní dobu, potřebuje si pro přípravy vyhradit větší množství času.

4.4.5 Simulace

Pokud chceme efektivně využívat veškeré možnosti, které projektová metoda VERT poskytuje, tak je vhodné modelovaný projekt ověřit. To se provádí prostřednictvím simulace, která informuje o stavu projektu v reálném světě. Spuštěním simulace projde sítí uzlů a hran, od jejího začátku až do konce, fiktivní tok, který zaznamená veškeré požadované údaje, včetně cesty daného toku. Tento simulační proces může být pochopitelně několikrát opakován (dle zadání uživatele), aby vytvořil dostatečně velký vzorek možných výsledků pozorování.
Informace o čase, nákladech a výkonu jednotlivých uzlů jsou uživateli podány podle následujících ukazatelů:

- Relativní četnost
- Kumulativní relativní četnost
- Průměr
- Standardní chyba (směrodatná odchylka vzorku)
- Variační koeficient
- Modus
- Medián
- Beta rozdělení - optimistický a pesimistický odhad
- Pearsonova míra šikmosti

Výše uvedené informace mohou být zobrazeny pro libovolný uzel v síti, ale také pro interval mezi dvěma vnitřními uzly. Simulace zároveň dokáže vzájemně kombinovat jednotlivé konečné uzly, čímž je možné získat informace o dokončení jak v rámci jednoho konečného uzlu, tak i celého projektu. Získané informace mohou být (dle požadavku uživatele) zobrazeny také jako vzniklé náklady a generovaný výkon v časovém intervalu.

Informace a poznatky tohoto charakteru jsou pak vhodné a užitečné při sestavování rozpočtu projektu, alokaci výdajů nebo porovnání investičních alternativ.[69]

4.4.6 Analýza rizika

Jedním z výstupů metody VERT je sloupcový graf indexu optimálního konečného uzlu. Právě tento graf slouží jako základní zdroj pro provedení analýzy rizika. Analýza rizika (v případě VERT spíše Analýza rozhodování) vychází z porovnání jednotlivých konečných uzlů, přičemž mezi sebou hodnotí alespoň jeden konečný uzel, který uzavírá úspěšný projekt s alespoň jedním konečným uzlem, který uzavírá neúspěšný projekt.

Uživatel porovnává tyto koncové uzle s celkovým počtem iterací v projektu, čímž získá pravděpodobnostní údaj o úspěchu či neúspěchu daného projektu. V případě, že má daný projekt více konečných uzlů, jejichž realizace skončí ve stejnou dobu, je třeba zvolit optimální konečný uzel. Ten se určí podle nejnižšího času dokončení, nejnižších nákladů a nebo největšího výkonu, případně váženou kombinací těchto faktorů dle požadavků uživatele.[70]

4.4.7 Kritická a optimální cesta

VERT ve své analýze vytváří pro jednotlivé hrany a uzly indexy kritických a optimálních cest, které ohodnocují, jak výrazně může daný uzel nebo hrana náležet do jedné či druhé množiny. Kritická cesta je v síti zvolená jako cesta s nejdelším trváním, nejvyššími náklady a nejnižším generovaným výkonem, případně váženou kombinací jednotlivých faktorů dle ohodnocení uživatelem. Optimální cesta je v podstatě opakem kritické cesty. Je to cesta s nejkratší dobou trvání, nejnižšími náklady nebo největším generovaným výkonem, případně nejméně výhodná cesta s váženou kombinací jednotlivých faktorů dle ohodnocení uživatele.

Vzhledem k tomu, že VERT je stochastická metoda využívající simulací, kritická a optimální cesta se může měnit podle každé iterace. Samotný program počítá podíl času

[69] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 55

[70] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 56

každé hrany na celkovém projektu a náležitost uzlu na kritické či optimální cestě, aby mohl souhrnnou informaci uvést ve svém výstupu a zobrazit ji ve sloupcovém grafu.[71]

4.4.8 Souhrn

Metoda VERT2 výrazně obohatila na konci 70. let svět projektových manažerů. Ačkoli si to většina z nich neuvědomuje, tak znalost této metody většinou znamená právě znalost této první modifikace. Stejně tak většina populace ani netuší, že právě tato metoda stála v 80. letech za množstvím rozsáhlých projektů (týkajících se převážně zajištění bezpečnosti a řešení krizových situací) ve Spojených Státech

Jedná se bezpochyby o užitečný nástroj, který však vyžaduje vysoké náklady na sběr a zadání dat. Zatímco na rozdíl u metody CPM odpovídá jedné hraně jediný údaj, kterým je ohodnocena (čas), tak u VERT se jedná o údaje tři (čas, náklady, výkon), které jsou zapsány prostřednictvím až čtyř možných způsobů. K tomu nabízí použití celkem 18 různých uzlů. Těžko říci, jestli je metoda v současné době stále natolik efektivní, aby se takto složitý sběr dat vyplatil.

Dle uvedených skutečností je tak vcelku překvapením, že nejvíce oblíbenou modifikací metody se stala až VERT3, která vznikla o další dva roky později.

4.5 VERT 3

VERT3 je modifikací původní metody VERT a doplněním a upřesněním metody VERT2. Na rozdíl od původní verze byl VERT3 detailně popsán ve všech svých původních funkcí a v plném rozsahu poskytnut k rukám uživatele.[72] Metoda byla popsána *Sang M. Leem,* s podporou dvojice autorů *Gerald L. Moellerem* a *Lester A. Digmanem* v publikaci *Network Analysis for Project Decision*, vydané v roce 1982. Sám hlavní autor tvrdí, že tato publikace je napsaná přímo pro management podniku. Metoda VERT3 na rozdíl od metody VERT2, dále rozvíjí univerzalitu ve všech svých ohledech a svým částečným

[71] MOELLER, Gerald L. a Lester A. DIGMAN. Operations Planning with VERT. Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697. str. 682

[72] Metoda VERT2 byla sice publikována veřejně již o dva roky dříve, ale v žádném případě v takovém rozsahu, aby byla dostupná široké komerční sféře.

návratem k původním kořenům VERT1 podporuje možnost svým komplexním nastavením plně představovat prostředí reálného světa.

K publikaci i k samotné metodě se vyjádřil *Duncan Boldy* z *University of Exeter* v magazínu *European Journal of Operational Research*. Ten hodnotí knihu jako vhodný manuál k pochopení funkcí, rozsahu a aplikace metody VERT (přesněji VERT3). Dle jeho názoru je kniha ideální pomůckou k zavedení daného přístupu ve firmě, nicméně svou strukturou ji ocení spíše analytici a specialisté, kteří budou muset přesvědčit management společnosti o vhodnosti metody VERT. K jejímu většímu rozsahu v budoucnosti je však skeptický, neboť metoda je příliš sofistikovaná a vyžaduje extrémně velké množství vstupních dat.[73]

4.5.1 Detailní specifikace rozdílů mezi VERT3 a VERT2

V tomto oddílu jsou postupně představeny všechny úpravy, které se týkaly metody VERT, resp. rozdílu v modifikacích VERT2 a VERT3. Tím pravděpodobně nejvýznamnějším je navrácení předdefinovaných matematických vztahů do procesu výpočtu, které byly definitivně upraveny a doplněny.[74]

4.5.1.1 Rozdíly ve výstupní logice FILTR 1 a FILTR 3

První z několika změn byla úprava nevyřešené (nestabilní) části metody VERT2 v logice výstupní části dvou složitějších typů uzlů – konkrétně FILTR 1 a FILTR 3, neboť často docházelo k uzavření projektu a nemožnosti pokračování toku právě v této části.

Z toho důvodu bylo zavedeno pro konzistenci dané sítě pravidlo, že musí n-1 (z celkového počtu n) vystupujících činností být definováno časem, náklady a výkonem a poslední (n-tá) vystupující činnost musí být bez jakýchkoliv omezení. Tato činnost bude zpracována pouze v případě, kdy žádná z vystupujících činností nebude realizována a zajistí tak „průtok“ projektu bez jakéhokoliv přerušení. V tomto úhlu pohledu upravoval přístup převážně podmínky výstupní logiky FILTR 1. Za menší úpravy bylo však možné tento přístup uplatnit i u výstupní logiky FILTR 3.

---

[73] BOLDY, Duncan. *Book review: Network Analysis for Management Decision: A Stochastic Approach.* European Journal of Operational Research. Fourth EURO IV special issue, 1982, Vol. 11, No. 3, (November, 1982), 303. str. 303

[74] Více v 4.5.1.3 – Vyjádření faktorů prostřednictvím matematických vztahů

Další vhodnou úpravou v tomto směru byla inovace, která v metodě VERT 3 za jistých podmínek umožňovala nechat realizovat činnost, která nesplňuje dílčí část z jejího omezení (např. splňuje podmínky pro realizaci v ohledu času, ale nikoli už v ohledu nákladů a výkonu), což bylo využito převážně u výstupní logiky FILTR 1. Tento krok však nebyl součástí výchozího nastavení a v případě použití musel uživatel sám rozhodnout o vhodnosti tohoto přístupu nastavením alternativních omezení.[75]

4.5.1.2 Aplikace uzlů jednoduché a složené logiky

Až v manuálu pro aplikaci metody VERT3 byla detailněji (pro potřeby uživatele) rozebrána problematika využití uzlů jednoduché logiky. Jak již bylo zmíněno v předchozí kapitole, existují celkem 4 typy uzlů, které v sobě zahrnují vstupní i výstupní logiku (z logického hlediska se jedná takřka pouze o výstupní logiku). Těmi jsou POROVNÁVAJÍCÍ (compare), PREFERUJÍCÍ (preferred), ŘADÍCÍ (queue) a USPOŘÁDÁVAJÍCÍ (sort). Jejich použití vyžaduje ohodnocení vstupujících hran podle kladných a záporných ukazatelů (+ a –) a vyhodnocení realizace následující hrany probíhá podobně jako u výstupní logiky FILTR 3 na základě splnění požadované logiky.
Metoda VERT3 dala těmto logikám větší rozsah, když umožnila použití obou typů uzlů (tedy jak jednoduché, tak složené logiky) v rámci jednoho projektu. Vzhledem ke konzistenci projektu je však vhodné alternativní logiky uzlů využívat pouze v dílčích subprojektech. Kromě toho, že zvyšují náročnost výpočetních operací, se negativně podepisují na stabilitě a reálné podobě projektu.[76]

4.5.1.3 Vyjádření faktorů prostřednictvím matematických vztahů

Další a asi nejvýznamnější inovací metody VERT3 je větší flexibilita zápisu jednotlivých faktorů náležících činnostem, tedy času, nákladů a výkonu. Činnosti mohou být opětovně (jako v případě VERT1) vyjádřeny funkcí jiných činností, a to nově použitím až padesáti (resp. sta) různých matematických vztahů. Tabulka č. 5 názorně ilustruje sedm náhodně vybraných matematických závislostí, které mohou být použity k modelování času, nákladů

[75] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 51-52
[76] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 70-73

nebo výkonu činnosti. Všechny ostatní z nabízených možností jsou k dispozici v příloze této práce.

Tabulka 5: Matematické vztahy k identifikaci činnosti v metodě VERT3

KÓD	MATEMATICKÝ VZTAH	PODMÍNKY
1	$X \cdot Y \cdot Z = R$	
2	$(X \cdot Y) / Z = R$	$Z \neq 0,0$
4	$1 / (X \cdot Y \cdot Z) = R$	$X \cdot Y \cdot Z \neq 0,0$
8	$-X - Y - Z = R$	
15	$X \cdot (LOG_{10}(Y \cdot Z)) = R$	$Y \cdot Z > 0,0$
18	$X \cdot (ArcTg(Y \cdot Z)) = R$	
37	$X / Y / Z = R$	$Y, Z \neq 0,0$

Zdroj: Vlastní zpracování

Sestavení matematického vztahu v rámci sítě VERT je možné ve třech následujících krocích:

- Dlouhý nebo komplikovaný matematický vztah je nezbytné zjednodušit do vztahu tří proměnných (tak, jak je uvedeno v tabulce č. 5).
- Je nezbytné definovat hodnoty tří proměnných (X, Y a Z). Tyto hodnoty mohou být získány z předchozí, příp. aktuální hrany nebo uzlu, dále jako přímo dosazená konstanta a nebo jako jedna z dříve vypočítaných transformací, která svým výpočtem předcházela aktuální řadě ke zjištění hodnoty času, nákladů nebo výkonu pro zkoumanou činnost.
- Získané výsledky jsou důležité pro zjištění hodnoty času, nákladů nebo výkonu zkoumané hrany. Po jejich výpočtu se můžeme rozhodnout, zda-li je zařadíme do celkového výpočtu jednotlivých faktorů dané hrany, nebo zda-li je vynecháme. Rozhodneme-li se vynechat některou z vypočtených jednotek, pak tato transformace slouží čistě jako etapa pro výpočet hodnoty dlouhého nebo komplikovaného matematického vztahu.[77]

4.5.1.4 Software pro metodu VERT3

Významným průlomem VERT3 byl i veřejně dostupný počítačový software, který uživateli umožňoval namodelovat náročný výpočet čítající řadu iterací. Celý zápis

[77] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 55

počítačového programu je k dispozici v již zmíněné knize "*Network Analysis for a Management Decision*" v rámci její poslední části na 90 stranách. Nicméně, výstup tohoto zápisu je zcela nekompatibilní se současným programovým vybavením osobních počítačů. Právě z toho důvodu je součástí této práce výroba obdobné aplikace.[78]

4.5.2 VERT3 a jeho pozice v 80. letech

VERT3 byl bezpochyby vrcholem své doby, co se týče vhodného přístupu pro řízení komplexních projektů. Modifikace byla úspěšná především z důvodu vyřešení několika původních problémů, ale také rozšířením a přizpůsobením původních přístupů pro takřka každého uživatele. Velkou výhodou byl i nový software.

Ačkoliv měl VERT3 nakročeno k tomu stát se univerzálním a přesným přístupem k řízení projektů, je dnes již takřka zapomenutý. První velkou překážkou je složité zadávání projektů, které vyžaduje detailní sběr dat a které jsou také časově velmi náročné. S tím souvisí také nemožnost jakéhokoliv použití v menších komerčních projektech, neboť se aplikace metody z finančního ani časového hlediska nevyplatí.

Posledním hřebíčkem se stal důvod, že řada uživatelů opouštěla tvrdé matematické přístupy a projektové řízení se začalo ubírat spíše k měkkým přístupům, které často vycházely z manažerských dovedností.

4.6 VERT 4

Metoda VERT byla modifikována ještě jednou, a to čistě pouze jedním z autorů - Sang M. Leem, který definoval VERT4 v roce 1983. Nicméně, není zcela korektní tvrdit, že metoda byla autorem modifikována, neboť se, ve své podstatě, nikterak nelišila od VERT3. Dle dostupných zdrojů se jednalo čistě o přeprogramování a vývoj nové aplikace pro řízení projektů prostřednictvím původní metody VERT3.[79]

[78] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. Network Analysis for Management Decision: A Stochastic Approach. Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7. str. 220-309

[79] KIDD, John. *A Comparison Between the VERT Program and Other Methods of Project Duration Estimation*. Omega: The International Journal of Management Science. Pergamon Journals Ltd., 1987, Vol. 15, No. 2, 129-134. str. 129 - 130

5 Simulace

Simulace jsou nedílnou součástí této práce, a proto jim je věnována samostatná kapitola. V kapitole jsou rozebrány základní výhody a nevýhody simulačních přístupů, dále je prostor ponechán nejznámější simulační metodě Monte Carlo a poslední část seznámí čtenáře s generováním náhodných čísel.

Základním pojmem, který je bezpochyby nutné definovat je pojem „simulace“. Ačkoli je možné si pod tímto pojmem představit řadu významů, můžeme jej chápat jako napodobení s cílem přiblížení se reálnému systému. Simulace je sama o sobě unikátním přístupem, při kterém se k dosažení výsledku neužívá vzorů ani optimalizací. Termín simulace tak můžeme volně přeložit jako „napodobení“.

Pojem simulace již definovala řada autorů. Shannon stojí za následujícím tvrzením: „*Simulace je proces tvorby modelu reálného systému a provádění experimentů s tímto modelem za účelem dosažení lepšího pochopení chování studovaného systému či za účelem posouzení různých variant činnosti systému.*“[80] Proti tomu Houška předkládá obsáhlejší definici: „*Simulace je numerická metoda, která spočívá v experimentování se speciálním matematickým modelem reálných systémů na počítači. Simulace se v tomto pojetí chápe jako postup, s jehož pomocí se zkoumaný proces, resp. jeho kroky v čase, generují na základě vlastností parametrů zobrazovaného systému.*“[81]

Poněkud stručněji jsou definovány následující termíny, které však znatelně souvisí se simulacemi a simulačními procesy:

- Systém – Část reálného světa, která se stává předmětem zkoumání.
- Model – Zjednodušení reálného systému, vyjádřeného pomocí matematických rovnic a nerovnic, případně nákresů a grafů.
- Simulační model – Model, který je sestaven odborným pracovníkem v simulačních jazycích za pomoci výpočetní techniky.
- Náhodné číslo – Nezávislé hodnoty rovnoměrného rozdělení na intervalu *(0;1)*, které jsou statisticky nezávislé.[82]

[80] SHANNON, Robert E. *Systems simulation: the art and science.* Englewood Cliffs, N.J.: Prentice-Hall, 1975. ISBN 0138818398. str. 9

[81] HOUŠKA, Milan. *Simulační modely I.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X. str. 7

[82] HOUŠKA, Milan. *Simulační modely I.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X. str. 11-16

5.1 Výhody a nevýhody simulací

Podobně jako každý jiný nástroj, i využití simulací má své výhody, stejně tak i nevýhody. Na následujících řádcích jsou uvedeny ty nejzákladnější z nich.

Pravděpodobně nejpodstatnější výhodou využití simulací je prověření všech aspektů bez vynaložení prostředků na realizaci, dále můžeme simulaci aplikovat i na velmi složité systémy. Umožňují změnit rychlost toku času a poskytují komplexnější pohled na studovaný systém. V neposlední řadě sledování simulačního modelu umožňuje lépe pochopit daný (zkoumaný) systém, což je výhoda relativně opomíjená.

Podstatnou nevýhodou simulačních modelů je nezbytnost odborných znalostí pro pochopení systému i pro užití sofistikovaného simulačního software. Sestavení modelu může být finančně i časově náročné a žádný takto sestavený model nemá obecnou platnost. Interpretace získaných výsledků bývá často velmi složitá a v případě simulačních modelů neznáme závislosti mezi vstupy a výstupy.[83]

5.2 Metoda Monte Carlo

Monte Carlo je jednou z nejstarších a pravděpodobně také nejznámější metodou v oblasti simulací. Díky své popularitě, která sahá až do 40. let 20. století, je často zaměňována i se samotnou simulací.

Základem metody Monte Carlo je mnohonásobné opakování definovaného stochastického procesu a následně jeho statistické vyhodnocení. Metoda pro tento proces využívá generátoru náhodných čísel, se kterým se čtenář seznámí dále v této práci.[84]

Prvně byla tato metoda využita pravděpodobně v Buffonově úloze, kdy se stejnojmenný matematik pokoušel odhadnout velikost Ludolfova čísla (π) opakovaným vhazováním jehly na linkovaný list.[85]

Jak již bylo uvedeno v kapitole 4, právě metoda VERT využívá ve svém výpočtu tuto simulační metodu.

---

[83] DLOUHÝ, Martin, Jan FÁBRY a Martina KUNCOVÁ. Simulace pro ekonomy. 2., upr. vyd. V Praze: Oeconomica, 2005. ISBN 80-245-0973-3. str. 5-6

[84] FÁBRY, Jan. *Matematické modelování*. Praha: Oeconomica, 2007. ISBN 978-80-245-1266-2. str.121

[85] GRAHAM, Carl a Denis TALAY. *Stochastic simulation and Monte Carlo methods mathematical foundations of stochastic simulation.* Berlin: Springer, 2013. ISBN 978-36-423-9363-1. str. 15-16

5.3 Generování náhodných čísel

Vzhledem k tomu, že náhodná čísla jsou modelem náhodných veličin, nelze v plném slova smyslu mluvit o náhodných veličinách, ale o pseudonáhodných veličinách, u kterých nelze předpokládat úplnou statistickou nezávislost.

Matematické generátory jsou založeny na speciálních algoritmech využívající rekurzivní funkce, a proto tvoří pouze pseudonáhodná čísla. V praxi se ale můžeme setkat i se „skutečnými" náhodnými čísly, které lze zjistit pomocí mechanických přístrojů či prostřednictvím fyzikálních nebo chemických operací a postupů. Zcela specifickým příkladem jsou pak tabulky náhodných čísel.[86]

5.3.1 Generátory náhodných čísel

Nejčastějšími generátory náhodných čísel jsou generátory mechanické, fyzikální a chemické.

Nejznámějším mechanickým generátorem je hod kostkou nebo hod mincí. V případech, kdy má výsledek značný význam jsou použity speciálně zkonstruované přístroje (např. Sazka). Bohužel, aplikace těchto generátorů pro simulaci na počítači je nemožná.

Fyzikální a chemické generátory náhodných čísel využívají postupů a principů, které mají náhodný charakter. Z chemických generátorů jsou nejčastěji používané poločas rozpadu prvků a interval mezi dopadem částic. Fyzikální generátory mohou využívat špičky v síti nebo pohyb počítačovou myší. Po dlouhou dobu byla za uznávaný a relativně schopný generátor považována i obyčejná lávová lampa.

Veškeré výše uvedené přístupy podávají čísla skutečně náhodná, která však mají značnou nevýhodu. K provedení vybraného experimentu je vždy kromě počítače zapotřebí dalšího zařízení.[87]

Mezi generátory náhodných čísel můžeme počítat i *Tabulky náhodných čísel*, které mohou být vytvořeny i jednoúčelově pro řešení daného problému. K dispozici jsou však i obecné

[86] HOUŠKA, Milan. *Simulační modely I.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X. str. 16-17

[87] DLOUHÝ, Martin, Jan FÁBRY a Martina KUNCOVÁ. Simulace pro ekonomy. 2., upr. vyd. V Praze: Oeconomica, 2005. ISBN 80-245-0973-3. str. 17-19

tabulky, kterých byla vytvořena celá řada - např. tabulky RAND Corp. (1955), které obsahují milion náhodných čísel.[88]

5.3.2 Generátory pseudonáhodných čísel

Matematické generátory, známé také jako aritmetické generátory, jsou nejpoužívanějšími v oblasti počítačových simulací. Každé generované číslo se získává matematickou operací, která je provedena na čísle předchozím. Matematické generátory náhodných čísel vychází z rekurzivní funkce typu:

$$x_{n+1} = f(x_n, x_{n-1}, \ldots, x_{n-m}) \text{ pro } m \geq 0$$

V největším množství případů jsou následující generátory konstruovány pro získání náhodných čísel s rovnoměrným rozdělením $R(0;1)$, přičemž nejčastějšími generátory náhodných čísel jsou:

- Lineární kongruentní funkce (*smíšený generátor*)

$$c_{n+1} = (\lambda \cdot c_n + \mu)\ mod(p)$$

- Aditivní lineární kongruentní funkce (*aditivní generátor*)

$$c_{n+1} = (\lambda \cdot c_n)\ mod(p)$$

- Multiplikativní funkce (*multiplikativní generátor*)

$$c_{n+1} = (c_n + c_{n\cdot j})\ mod(p)$$

Obecně platí, že nejlepší výsledky by měla podávat lineární kongruentní funkce, neboť při generování čísla má jak aditivní, tak i multiplikativní část. Je však nezbytné si stále pamatovat, že se jedná pouze o pseudonáhodná čísla, jejichž řada je konečná a maximálně po *p* krocích se posloupnost čísel začíná opakovat od výchozí hodnoty. V dnešní době se uvádí, že perioda o délce 2^{64} je již nedostatečná.[89]

[88] HOUŠKA, Milan. *Simulační modely I.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X. str. 16
[89] HOUŠKA, Milan. *Simulační modely I.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X. str. 17-19

5.3.3 Příklad lineárně kongruenční funkce

Jak je zmíněno v předchozím oddíle, základní předpis smíšené lineární kongruentní funkce je:

$$c_{n+1} = (\lambda \cdot c_n + \mu)\ mod(p)$$

V tomto případě si autor práce zvolil následující parametry:

$\lambda = 4$; $\mu = 15$; $p = 17$; $c_n = 8$

Pomocí výše uvedeného předpisu se generují následující hodnoty:

$$c_1 = (4 \cdot 8 + 15)\ mod(17) = 13$$
$$c_2 = (4 \cdot 13 + 15)\ mod(17) = 16$$
$$c_3 = (4 \cdot 16 + 15)\ mod(17) = 11$$
$$c_4 = (4 \cdot 11 + 15)\ mod(17) = 8$$
$$c_5 = (4 \cdot 8 + 15)\ mod(17) = 13$$

Jak je z příkladu patrné, tak $c_1 = c_5 = 13$. Nevhodně zvolené parametry způsobily, že má generátor náhodných čísel malou periodu.

6 Vývoj aplikace

Hlavní část této práce zahrnuje stručný postup programování požadované aplikace. V první podkapitole jsou zmíněny základní požadavky a funkce, které výsledná aplikace bude mít. Druhá podkapitola představuje používané nástroje, vývojové prostředí a další součásti, bez kterých by aplikaci nebylo možné naprogramovat. Na tu navazuje samotný vývoj, který je doplněn řadou obrázků, kódů a informací z procesu programování. V závěru kapitoly je aplikace testována na jednoduchém problému s charakteristickými rysy tohoto programu.

6.1 Požadavky na aplikaci

Vývoj požadované aplikace probíhá tak, aby byl program ve své konečné podobě schopný řešit projekty podle metody PERT a pochopitelně také dle metody VERT. V druhém případě vychází program z první modifikace, známé jako VERT2.

Vzhledem k rozsahu práce a konzistence aplikace je nezbytné učinit omezující kroky, které se týkají aplikace v případě řešení projektů dle metody VERT.

Současná podoba metoda je omezená ve vybraných funkcích proti finálnímu plánu. Veškeré parametry činností je možné zadat pomocí libovolného z několika základních statistických rozdělení (rovnoměrné, normální, logaritmicko-normální, trojúhelníkové a beta rozdělení), prostřednictvím konstanty, výhledově (v budoucnu) pomocí šesti předem specifikovaných matematických vztahů. Aplikace také umožňuje práci výhradně s uzly složené logiky. V tomto případě bude software kompatibilní s celkem čtrnácti typy složených vstupních a výstupních uzlů, přičemž výstupní logiky typu FILTR 1, FILTR 2 a FILTR 3 jsou společně zahrnuty pod výstupní logiku FILTR, která vychází z první možnosti a limituje tak tok dle uživatelem nastavených omezení. V současné chvíli není možné zadat úlohu pomocí filtrační výstupní logiky. Samozřejmostí je řešení pomocí simulací, ze kterých je podán výstup ve formě tabulky se statistickými údaji. Ty může uživatel využít pro další výpočty, např. pro provedení analýzy rizika.

Aplikace je schopna v reálném čase řešit jednodušší úlohy (kolem 30 až 50 činností) dle obou projektových metodik, maximální limit je 99 činností a systém je přednastaven pro řešení celkem 100 simulací při jednom výpočtu, které poskytnou výsledky vhodně dělitelné na procentní body. Omezení se týká i uzlů, kde do každého uzlu může vstoupit maximálně 5 činností a stejný počet může také uzel opustit.

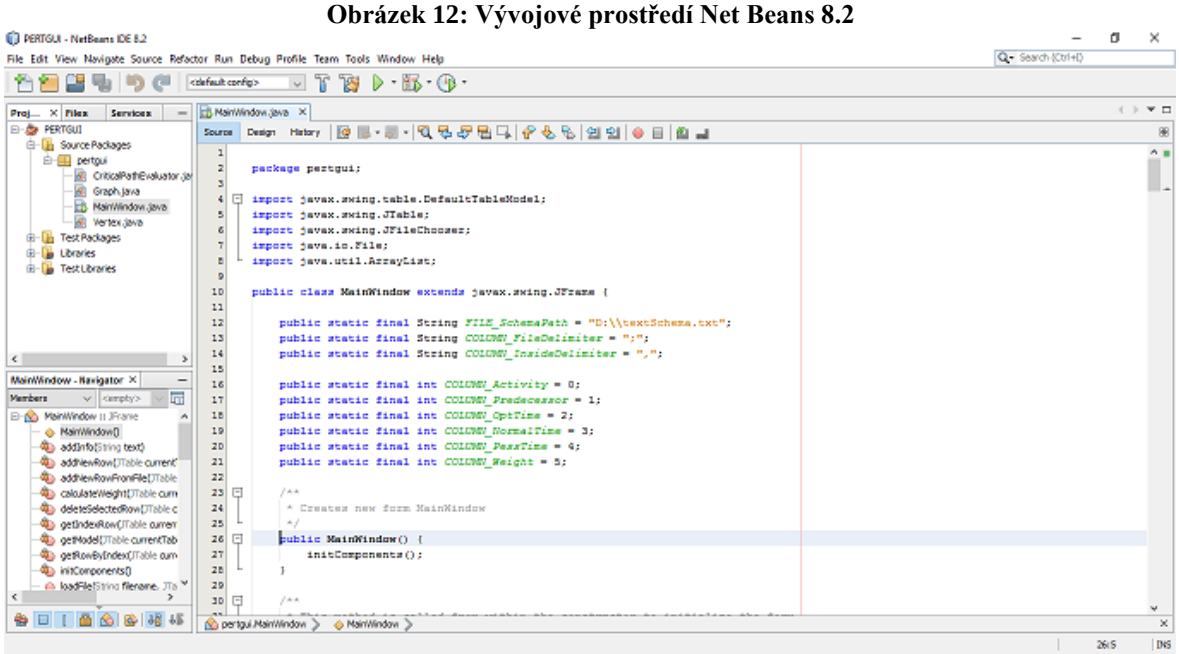
6.2 Softwarové vybavení pro vývoj aplikace

V této kapitole jsou popsány nástroje, vývojové prostředí a další nezbytné součásti, v jejichž prostředí programování této aplikace probíhá. Základní dvě součásti pro vývoj aplikace jsou vývojové prostředí *Net Beans* (verze 8.2) a *JAVA development kit*.

6.2.1 Net Beans

Net Beans 8.2[90] je integrované vývojové prostředí (IDE)[91]. Net Beans vlastní *Oracle Corporation*, která se podílí na jeho vývoji. Primárně toto vývojové prostředí slouží pro programovací jazyk JAVA, avšak jeho modulární softwarová architektura umožňuje práci i v odlišných programovacích jazycích. Net Beans je freeware[92] a obsahuje nástroj pro tvorbu grafického uživatelského rozhraní (GUI).[93] Pro snadnou tvorbu různých druhů dialogových a aplikačních oken má Net Beans integrovanou metodu „táhni a pusť.“[94]

Obrázek 12: Vývojové prostředí Net Beans 8.2



Zdroj: Vlastní zpracování

[90] Staženo ze stránky: https://netbeans.org/downloads/ (verze Java EE)
[91] Integrated Development Environment
[92] Software, který je distribuován zdarma.
[93] Graphical User Interface, více v podkapitole 6.2.3
[94] Drag and Drop

6.2.2 JAVA development kit

JAVA development kit[95] (zkráceně JDK) je soubor základních nástrojů, které slouží k vývoji aplikací pro platformu JAVA. JDK je (stejně jako Net Beans) produktem společnosti *Oracle Corporation* a stejně jako v předchozím případě se jedná o *freeware*. Vhodné je zmínit, že JDK je také rozšířenou verzí původního *Software development kit* (SDK).[96]

6.2.3 Grafické uživatelské rozhraní

Grafické uživatelské rozhraní je označení pro uživatelské rozhraní, které umožňuje uživateli ovládat počítač nebo samotný program pomocí grafických interaktivních ovládacích prvků. Takřka každý soudobý program využívá svého grafického rozhraní pro interakci s uživatelem. GUI tak můžeme označit za intuitivní a efektivní nástroj, jak získávat informace (data) od uživatele nebo mu je případně zobrazovat. Nejznámější vývojová prostředí (Net Beans, Eclipse, atd.) mají editor GUI jako součást svého vlastního programového vybavení.[97]

Obrázek 13: Jednoduché GUI (Grafické uživatelské rozhraní)

Zdroj: Vlastní zpracování

[95] Staženo ze stránky: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html (verze jdk-8u111-windows-x64)
[96] KOEGH, James. Java bez předchozích znalostí: průvodce pro samouky. Brno: Computer Press, 2005. ISBN 80-251-0839-2. str. 19-20
[97] KOEGH, James. Java bez předchozích znalostí: průvodce pro samouky. Brno: Computer Press, 2005. ISBN 80-251-0839-2. str. 192

6.3 Analýza a návrh aplikace

V této podkapitole je zběžně popsán proces programování aplikace včetně vytvoření jejího grafického uživatelského rozhraní GUI. V jednotlivých oddílech jsou zvlášť popsány jednotlivé části aplikace – orientace v nabídkách, výpočetní analýza PERT, výpočetní analýza VERT, dílčí akce k provedení výpočtů a podobně. Součástí tohoto popisu jsou i vybrané části kódů a popis jejich funkce v aplikaci, stejně tak jako zobrazení grafické podoby těchto dílčích částí.

6.3.1 Hlavní nabídka

Jednoduchá hlavní nabídka obsahuje čtyři základní tlačítka, která nasměrují uživatele do nového projektu, řešeného metodou PERT, do nového projektu řešeného metodou VERT, následně mohou zobrazit informace o projektu nebo aplikaci ukončit. Uprostřed menu se nachází jednoduché logo a název aplikace. Grafická podoba hlavního menu (bez loga) je zobrazena na obrázku č. 14 pod textem.

Obrázek 14: Grafická podoba hlavní nabídky

Zdroj: Vlastní zpracování

Pro otevření nového projektu je použit následující příkaz, který vyvolá otevření nového projektu, který bude řešen metodou PERT. Podobný příkaz je použitý, pokud chce uživatel spustit nový projekt, který bude řešen metodou VERT.

Kód 1: Spuštění výpočetní metody PERT z hlavní nabídky

```
private void btnPertActionPerformed(java.awt.event.ActionEvent evt) {
   java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new PertWindow().setVisible(true);
        }
    });
}
```

Zdroj: Vlastní zpracování

Informace o projektu se zobrazí po volání jednoduchého příkazu v novém dialogovém okně. Toto okno poskytne uživateli stručné informace o této práci i o jeho autorovi.[98] V případě ukončení aplikace volí uživatel možnost „*Konec*“, která volá následující příkaz. Aplikaci je možné ukončit standardně také jejím zavřením v pravém horním rohu.

Kód 2: Příkaz k ukončení aplikace z hlavní nabídky

```
private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}
```

Zdroj: Vlastní zpracování

6.3.2 Práce se souborem v aplikaci

Podobně jako v každém modernějším software, i tady je možné při práci nový soubor vytvářet (*new*), stávající ukládat (*save*) nebo dříve vytvořený nahrát (*load*). Pro všechny tyto operace je v horní liště menubar „*Soubor*“, který tyto operace zajišťuje. Samozřejmostí je i volba ukončení aplikace z tohoto menu.

Rozpracované projekty jsou ukládány do souborů v blíže nespecifikovaném formátu, který však ukládá tyto informace jako jednoduchý textový editor (poznámkový blok) - *.txt*, kde jsou jednotlivé činnosti vkládány na nový řádek a jejich parametry (hodnoty) jsou vzájemně odděleny.

[98] Více v oddílu 6.3.5 – Informace o projektu

Vzhledem k tomu, že jsou výše uvedené operace vcelku banální, není třeba zde uvádět jednotlivé kódy, které uložení, případně vyvolání uloženého souboru provedou.

Vhodným doplňkem v aplikaci jsou příkazy, které posouvají danou činnost (řádek) v tabulce výš, případně níž, a tím umisťují tuto činnost blíže k začátku, resp. ke konci projektu. Vhodné použití této funkce je nejčastěji v případech, kdy si uživatel po zadání celého projektu uvědomí, že je do projektu třeba dodat klíčovou činnost, která z logického hlediska patří na jeho začátek.

6.3.3 Metoda PERT

Výpočet PERT je jednodušším z obou výpočtů, neboť v aplikaci je jeho cílem pouze určení očekávané kritické cesty projektu, stanovení střední hodnoty (očekávané délky) trvání projektu a stanovení rozptylu této očekávané délky trvání. I z toho důvodu postačuje této metodě jednoduché grafické prostředí, ve kterém se takřka jen přidávají, editují a odstraňují činnosti a po výpočtu střední délky trvání a rozptylu přichází výpočet očekávané délky trvání projektu a dalších informací.

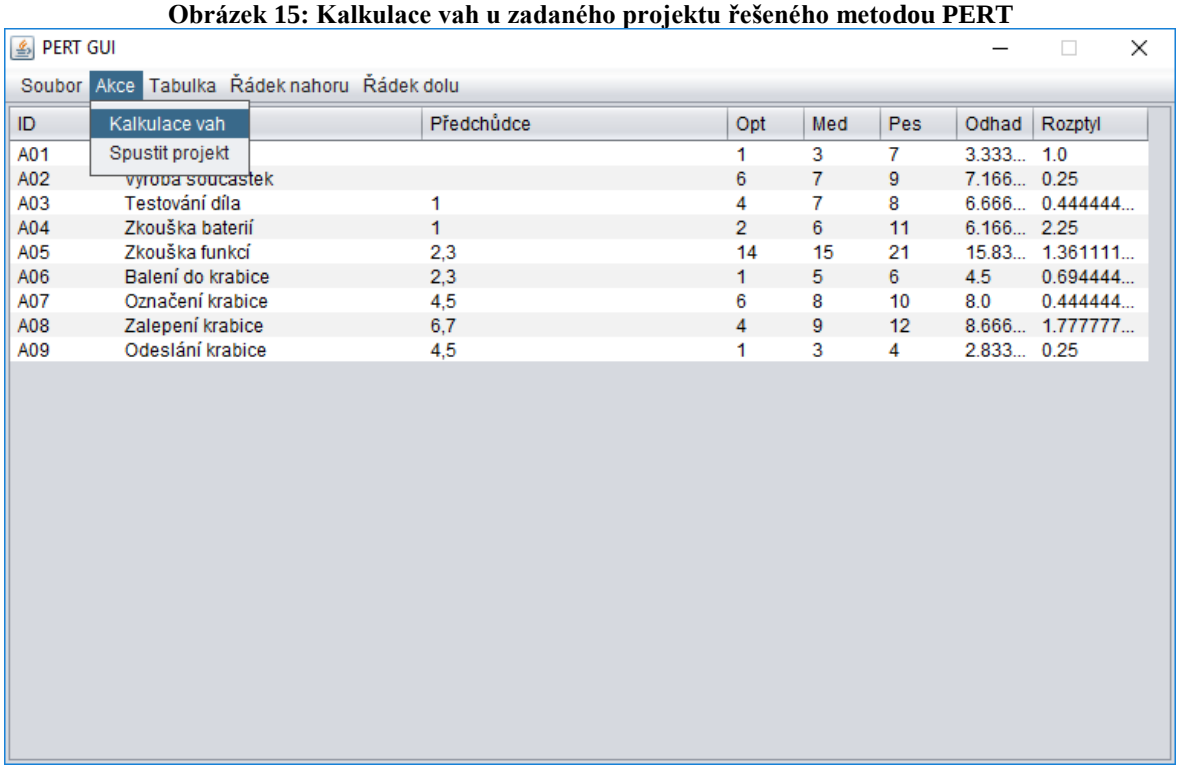
V horní části se tak nabízí celkem pět možností, ze kterých uživatel vybírá a které se váží k prováděným operacím:

- Soubor – Umožňuje základní operace se souborem, jako je jeho uložení, nahrání nebo ukončení v aplikaci. Již je popsáno v oddílu *6.3.2 – Práce se souborem v aplikaci.*
- Akce – V této nabídce má uživatel možnost dopočítat potřebné údaje pomocí příkazu „*Kalkulace vah*“, které spočítají hodnotu očekávané délky trvání a také rozptyl. Pomocí druhé možnosti „*Spustit projekt*“ se provede výpočet, který se uživateli zobrazí v novém dialogovém okně.
- Tabulka – Nabídka se týká editace činností z hlediska jejich přidávání či odebírání. Nový řádek se vygeneruje příkazem „*Přidat činnost*“ a smazání řádku se provádí příkazem „*Odebrat činnost*“.
- Řádek nahoru – Posune řádek výš (činnost blíže začátku)
- Řádek dolu – Posune řádek níž (činnost blíže konci)

6.3.3.1 Zadávání činností

Činnosti se zadávají přes vysouvací menubar příkazem „*Přidat činnost*“ (popsáno výše). Tím se v tabulce vygeneruje nový řádek, do kterého je možné vložit parametry dané činnosti – v tomto případě název, předchůdci a tři hodnoty délky trvání, které charakterizují Beta-rozdělení v metodě PERT (optimistický odhad, modální odhad a pesimistický odhad).

Obrázek 15: Kalkulace vah u zadaného projektu řešeného metodou PERT



ID		Předchůdce	Opt	Med	Pes	Odhad	Rozptyl
A01			1	3	7	3.333...	1.0
A02	Výroba součástek		6	7	9	7.166...	0.25
A03	Testování díla	1	4	7	8	6.666...	0.444444...
A04	Zkouška baterií	1	2	6	11	6.166...	2.25
A05	Zkouška funkcí	2,3	14	15	21	15.83...	1.361111...
A06	Balení do krabice	2,3	1	5	6	4.5	0.694444...
A07	Označení krabice	4,5	6	8	10	8.0	0.444444...
A08	Zalepení krabice	6,7	4	9	12	8.666...	1.777777...
A09	Odeslání krabice	4,5	1	3	4	2.833...	0.25

Zdroj: Vlastní zpracování

Chybějící hodnoty v posledních dvou sloupcích (odhad délky trvání a rozptyl) se dopočítávají podle níže uvedených vzorců, které jsou charakteristické pro výpočet v metodě PERT. Jejich výpočet se provede příkazem „*Kalkulace vah*“ v menubaru „*Akce*“.

$$\mu(t_{ij}) = t_{ij}^{s} = \frac{a_{ij} + 4m_{ij} + b_{ij}}{6}$$

$$\sigma^2(t_{ij}) = \left(\frac{b_{ij} - a_{ij}}{6}\right)^2$$

Kalkulace vah se provádí následujícím příkazem.

Kód 3: Kalkulace vah pro výpočet očekávané délky trvání a rozptylu

```
private boolean calculateValues(JTable currentTable) {

        try {

            DefaultTableModel model = (DefaultTableModel)
currentTable.getModel();
            int rowCount = model.getRowCount();
            int i;

            float optTime;
            float normalTime;
            float pessTime;

            float weight;
            double dispersion;

            for(i=0; i<rowCount;i++) {
                optTime = Float.parseFloat(String.valueOf(model.getValueAt(i,
Constants.COLUMN_OptTime)));
                normalTime =
Float.parseFloat(String.valueOf(model.getValueAt(i,
Constants.COLUMN_NormalTime)));
                pessTime = Float.parseFloat(String.valueOf(model.getValueAt(i,
Constants.COLUMN_PessTime)));

                dispersion = Math.pow(((pessTime - optTime) / 6), 2);

                weight = (optTime + (4 * normalTime) + pessTime ) / 6;

                currentTable.setValueAt(weight, i, Constants.COLUMN_Weight);
                currentTable.setValueAt(dispersion, i,
Constants.COLUMN_Dispersion);
            }

        }

        return true;
    }
```

Zdroj: Vlastní zpracování

Činnost, která nemá předchůdce, je činnost počáteční a vychází z prvního uzlu. Činnost, která není nikdy uvedená jako předchůdce, je činnost konečná a končí v posledním n-tém uzlu.

6.3.3.2 Výpočet

Pro výpočet v metodě PERT je nezbytné, aby byl celý projekt korektně zadán a také, aby byla provedena výše uvedená kalkulace vah, které jsou pro následný výpočet nezbytné. Část algoritmu, která počítá všechny cesty z počátečního ke každému uzlu, je uvedena pod

tímto textem (kód č. 4). Celý početní algoritmus pro výpočet je k dispozici v příloze této práce.

Kód 4: Výpočet maximální hodnoty uzlu

```
Vertex maximum = this.findMaximuWeight();
        while(maximum != null) {
            visited.add(maximum);
            for(Vertex neighbour : maximum.neighbours) {
                if(weights.get(maximum) + neighbour.duration >
weights.get(neighbour)){
                    weights.put(neighbour, weights.get(maximum) +
neighbour.duration);
                    precedessors.put(neighbour, maximum);
                }
            }

            maximum = this.findMaximuWeight();
        }
```

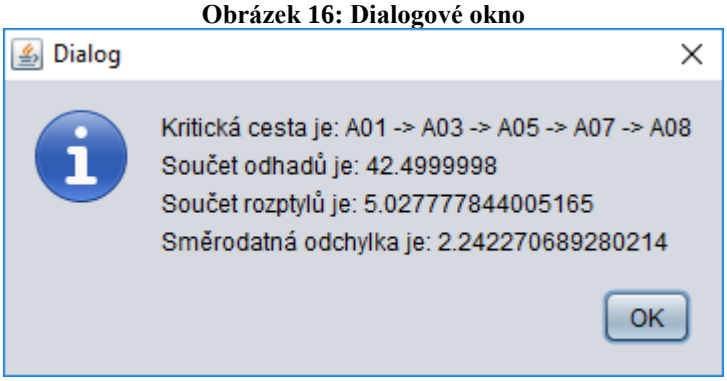
Zdroj: Vlastní zpracování

Výstupem vypočteného projektu v této aplikaci je dialogové okno na obrázku č. 16, které sumarizuje vypočtené hodnoty:

- Zobrazení kritické cesty (seznam činností).
- Očekávaná délka trvání, která je vypočítaná, jako součet délky trvání všech činností, ležících na kritické cestě.
- Rozptyl, který je vypočítán jako součet rozptylů všech činností, ležících na kritické cestě.
- Směrodatná odchylka, která je vypočítaná jako odmocnina z rozptylu.

Výstupem je jednoduché dialogové okno.

Obrázek 16: Dialogové okno



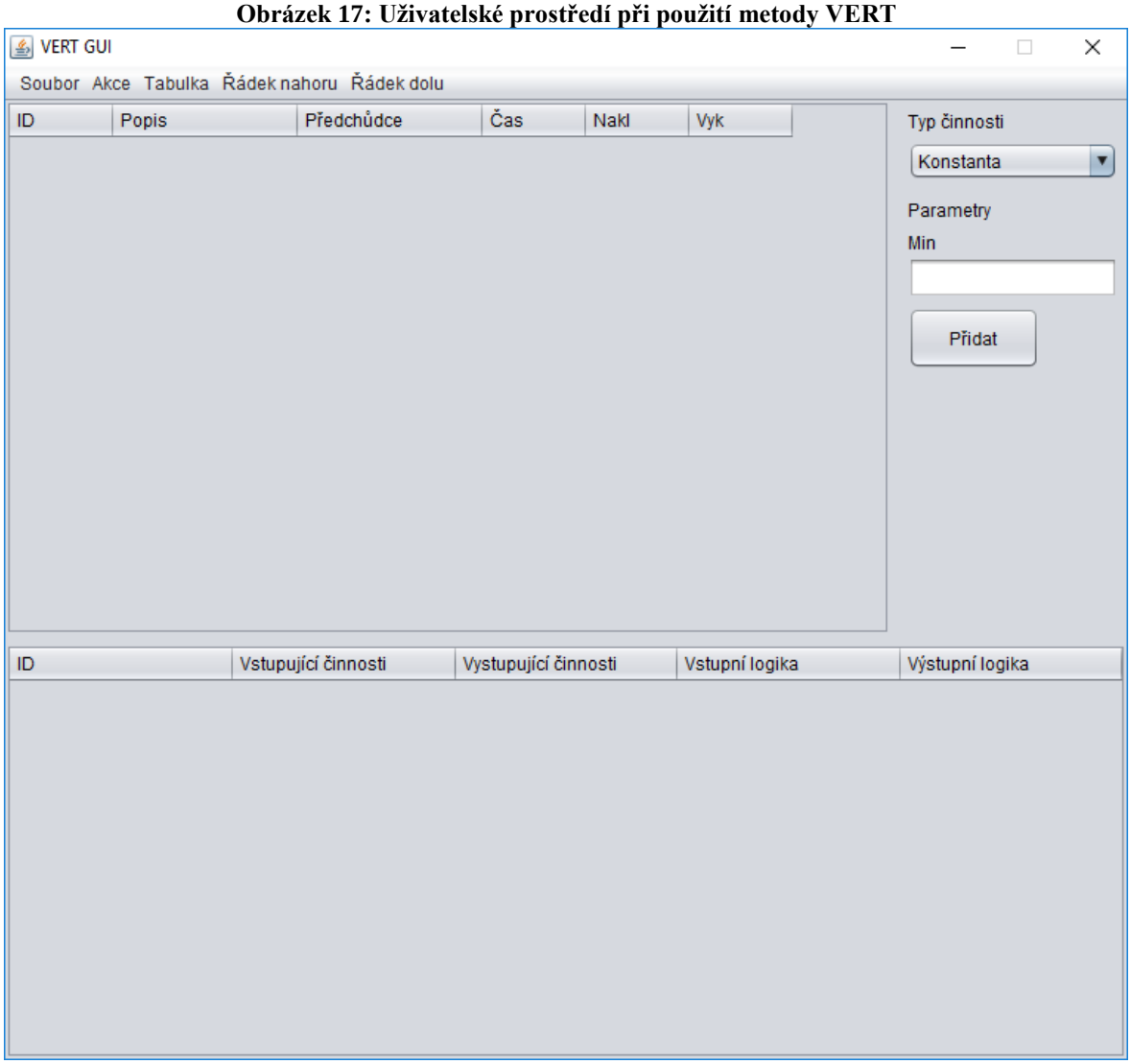
Zdroj: Vlastní zpracování

6.3.4 Metoda VERT

VERT je druhou a složitější metodou, kterou tato aplikace řeší. Její grafické prostředí vychází z dříve specifikované metody PERT, které je ale v tomto případě doplněno o řadu dalších funkcí v takřka každé fázi výpočtu.

Hlavním rozdílem je rozdělení uživatelského rozhraní do dvou separovaných částí, přičemž horní část zabírá zhruba ⅔ obrazovky a slouží pro zadání a editaci činností a spodní část, která zabírá zbývající ⅓ obrazovky a její funkcí je generování a editace rozhodovacích uzlů.

Obrázek 17: Uživatelské prostředí při použití metody VERT



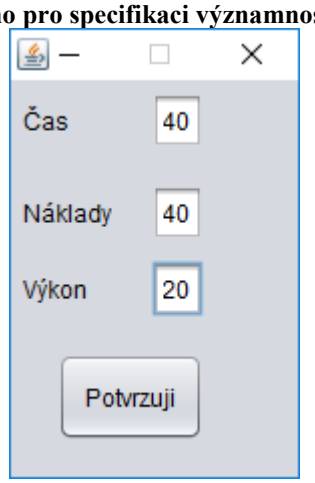
Zdroj: Vlastní zpracování

Dalším rozdílem je nově přidaný panel po pravé straně v horní části aplikace, prostřednictvím kterého probíhá editace činností – konkrétně specifikace jejich délky trvání, nákladů a generovaného výkonu. Z podobného důvodu se ve spodní části (na stejné straně) nachází volný sloupec, jehož cílem je vytvoření prostoru pro specifičtější definování výstupní logiky rozhodovacích uzlů se simulační výstupní logikou nebo filtrační výstupní logikou.

Pokud bychom se měli zabývat vyloženě výpočetním algoritmem, tak je metoda takřka stejná (resp. velmi podobná) s výchozím algoritmem výpočtu metody PERT, která je však v tomto případě mnohokrát opakována pro získání řady simulovaných výsledků, které jsou následně statisticky vyhodnoceny.

I pro tyto účely je pro uživatele předpřipraveno níže uvedené dialogové okno (obrázek č. 18), které od uživatele očekává vstupy, v nichž definuje význam jednotlivých parametrů, a to vyjádřený v procentech.

Obrázek 18: Dialogové okno pro specifikaci významnosti jednotlivých parametrů



Zdroj: Vlastní zpracování

Po potvrzení výše uvedených hodnot je uživatel přesměrován do jádra (uživatelského rozhraní) aplikace pro výpočet VERT. Přesto se může k těmto vahám ještě vrátit a přenastavit je pomocí vysouvacího menubar v záložce „*Akce*“.

6.3.4.1 Zadávání činností

Proces zadávání se takřka neliší od metody PERT s rozdílem, že na místo optimistické, modální a pesimistické varianty jsou dosazovány hodnoty délky trvání, nákladů a generovaného výkonu. Zadávání také neprobíhá vpisováním hodnot do řádků v tabulce,

ale pomocí postranního panelu, kde je možné zadat hodnoty pomocí různých specifikací. Aktivity je možné zadat dvojím poklepáním na danou činnost. Kód pro provedení tohoto procesu se nachází pod tímto textem.

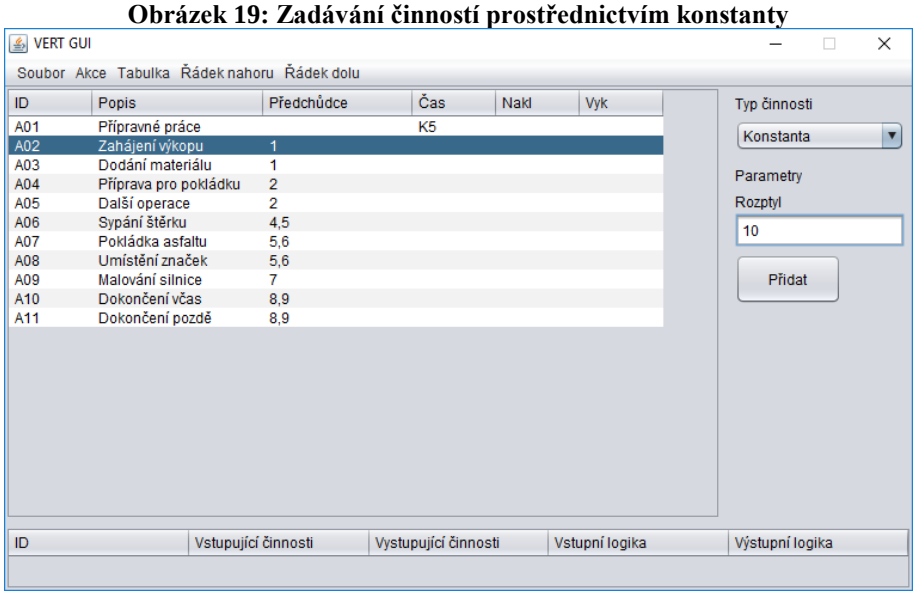
Kód 5: Označení činnosti pro její editaci v metodě VERT

```
jTable1.addMouseListener(new MouseAdapter() {
          public void mousePressed(MouseEvent me) {
            JTable table =(JTable) me.getSource();
            Point p = me.getPoint();
            if (me.getClickCount() == 2) {

                currentActivityRowIndex = table.rowAtPoint(p);
                currentActivityColumnIndex = table.columnAtPoint(p);

                prepareActivityPanel();
                setupActivityPanelCombos();
```

Zdroj: Vlastní zpracování

Po označení dané činnosti je možné ji zadat jednoduše konstantou, dále generováním náhodné hodnoty na základě statistického rozdělení a výhledově matematickým vztahem.
V prvním případě je možné zadat hodnotu konstantou, tedy první z možností. Daná konstanta je zadána do přednastaveného textového pole a následně zaznamenána pod kódovým označením *Kx* (kde x představuje hodnotu) do tabulky.

Obrázek 19: Zadávání činností prostřednictvím konstanty

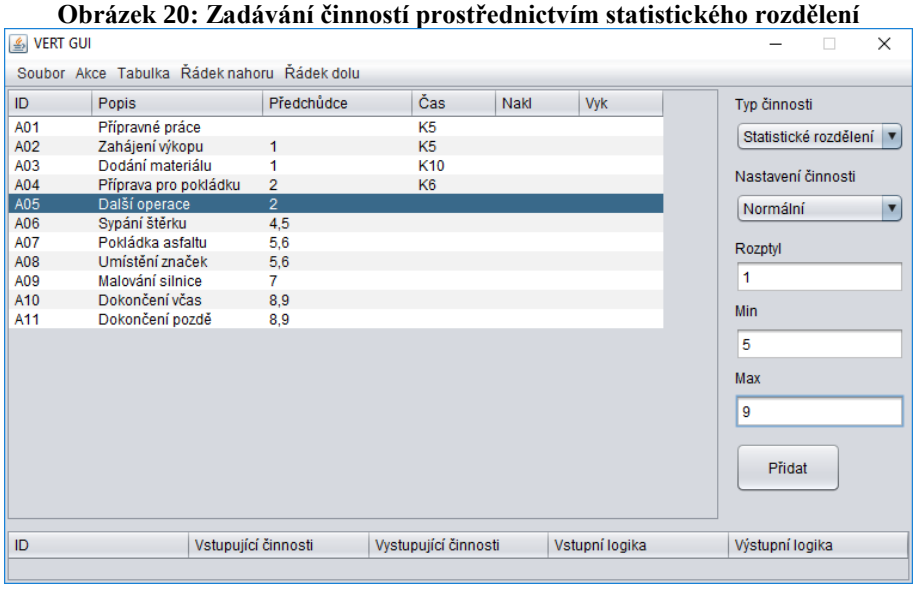


Zdroj: Vlastní zpracování

Další možností je zadání dat podle statistického rozdělení, kde se uživatel může rozhodnout pro použití libovolného z celkem pěti různých možností rozdělení (normální,

logaritmicko-normální, rovnoměrné, trojúhelníkové a beta). Každé z těchto rozdělení má předdefinované vlastní pole, do kterého je zapotřebí zadat specifické informace. Rozdělení je zadáno do tabulky pod označením daného typu a uvedením potřebných specifikací v závorce. Ukázka panelu je na obrázku č. 20 pod textem.

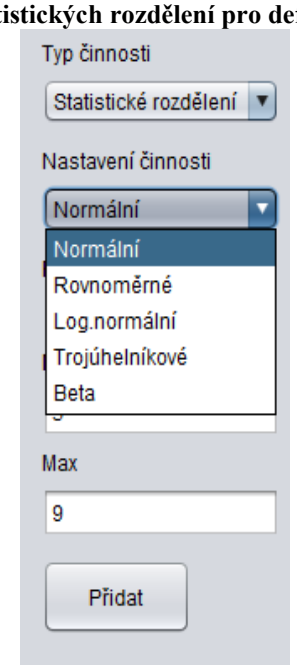
Obrázek 20: Zadávání činností prostřednictvím statistického rozdělení



Zdroj: Vlastní zpracování

Ukázka výběru jednotlivých rozdělení je na obrázku č. 21 pod textem.

Obrázek 21: Nabídka statistických rozdělení pro definování činností v aplikaci

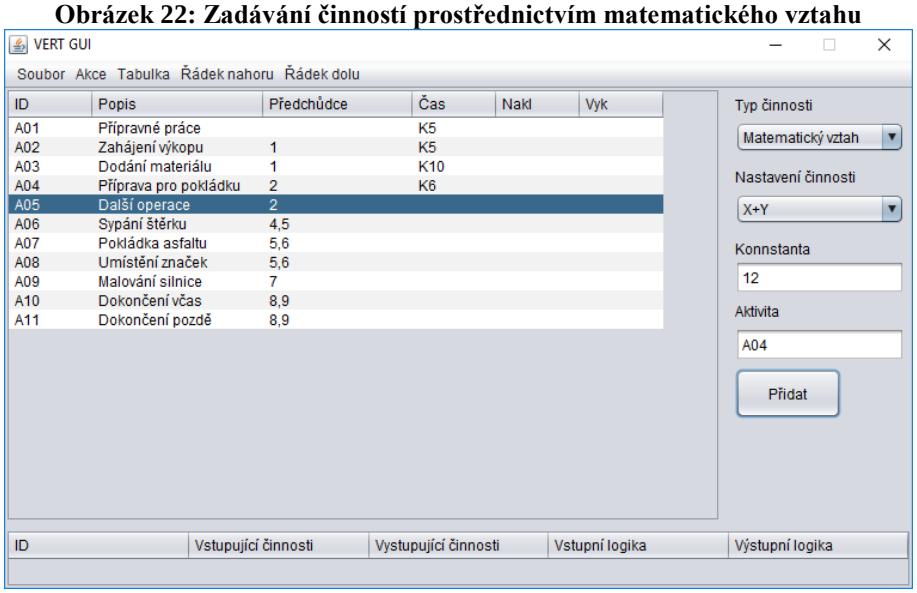


Zdroj: Vlastní zpracování

Pro generování náhodných čísel pomocí uvedených rozhodnutí se autor rozhodl pro použití předpřipravené knihovny statistických rozdělení ve staženém balíčku „*JDistlib – Java Statistical Distribution Libary*“[99]. Více informací o používaných statistických rozdělení je k dispozici v příloze této práce.

Poslední možností zadání bude pomocí matematického vztahu, který vychází ze dvou parametrů – činnosti vybrané předcházející činnosti a libovolné celočíselné konstanty.

Obrázek 22: Zadávání činností prostřednictvím matematického vztahu



Zdroj: Vlastní zpracování

Na rozdíl od původní podoby metody VERT, kde fungovala užší závislost mezi třemi parametry tohoto projektu, tomu tak v této aplikaci není, a proto je možné definovat činnost matematickým vztahem pouze pomocí původního ze tří parametrů (pokud je tak brána za klíčovou např. činnost A04, automaticky se při ohodnocení délky trvání činnosti A05 vychází z délky trvání činnosti A04).

Při definování činností není (na rozdíl od jiných metod) nezbytné vyplnit všechna volná pole, neboť ne každou činnost je možné definovat pomocí všech parametrů. Provedení libovolného rozhodnutí (které je mj. v případě VERT velmi časté) většinou není klasifikované časem a výkonem, ale pouze náklady, které vyjadřují „cenu“ plynoucí z provedeného rozhodnutí.

[99] Staženo ze stránky: http://jdistlib.sourceforge.net/ (verze 0.4.5)

6.3.4.2 Zadávání uzlů

Proces zadávání uzlů začíná ve chvíli dokončení projektu, a to v závislosti na preferencích uživatele. Ten může veškeré uzly postupně začít vypisovat do přiřazené tabulky. Ta umožňuje v této modifikované a zjednodušené verzi uživateli vybírat ze čtrnácti různých typů uzlů, které jsou zachyceny v tabulce pod textem.

Pro zjednodušení se autor práce rozhodl zavést pro dané kombinace uzlů grafické označení a názvosloví, přičemž řada typů i názvů uzlů byla převzatá ze stochastické metody GERT. Vstupní části uzlů se nazývají:

- *Iniciativní* (POČÁTEČNÍ) – Vyskytuje se pouze na začátku projektu.
- *Konjunktivní* (A) – Realizuje se pouze ve chvíli, kdy se realizují všechny vstupující činnosti.
- *Disjunktivní* (NEBO) – Realizuje se ve chvíli, kdy se realizuje první vstupující činnost.
- *Inkluzivní* (ČÁST Z A) – Realizuje se ve chvíli, kdy se realizuje alespoň jedna ze vstupujících činností, čeká však na dokončení ostatních vstupujících činností.

Názvosloví vzniklo i pro výstupní části uzlů, které se nazývají:

- *Finalizační* (KONEČNÁ) – Vyskytuje se pouze na konci projektu. V případě metody VERT může existovat více uzlů s touto výstupní logikou, přičemž každý může označovat odlišný konec – úspěšné dokončení projektu a neúspěšné dokončení projektu.
- *Deterministický* (VŠE) – Dokončení vstupní části uzlu iniciuje začátek realizace všech vystupujících činností z daného uzlu.
- *Simulační* (resp. *Stochastický* – SIMULACE MONTE CARLO) – Simulační výstupní logika má za cíl vybrat na základě metody *Monte Carlo* jedinou ze všech vystupujících činností, která se bude realizovat, přičemž všem vystupujícím činnostem jsou přiřazeny pravděpodobnosti realizace, jejichž součet je 100 %. Existuje tady podoba se Stochastickou výstupní logikou metody GERT, ale vzhledem k rozdílné funkci s metodou GERT, která v tomto případě vychází ze simulací, je uzel přejmenován.
- *Filtrační* (resp. *Redukční* – FILTR 1 / 2 / 3) – Filtrační výstupní logika iniciuje začátek vybraných vystupujících činností na základě předdefinovaných omezujících podmínek času, nákladů a výkonu.

Není však možné vzájemně kombinovat všechny možné typy uzlů. Z hlediska logiky je nemožné vytvořit kombinaci uzlu *Iniciativní* a *Finalizační* logiky, neboť by projekt neměl žádnou délku trvání. Další neexistující možností je kombinace uzlu *Iniciativní* a *Filtrační* logiky, protože k jejímu vyhodnocení je nezbytné započítat hodnoty vstupujících činností, které v tomto případě neexistují.

Obrázek 23: Typy možných kombinací uzlů zjednodušené metody VERT

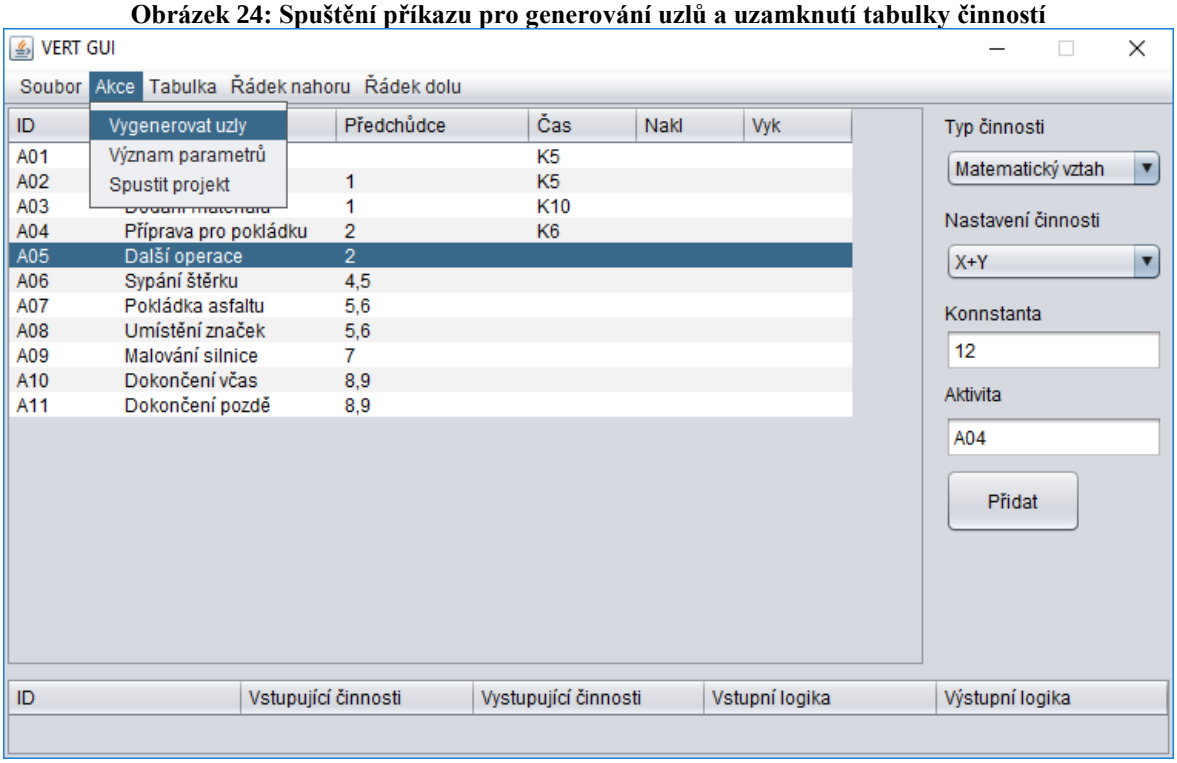
Tabulka možných kombinací uzlů	INICIATIVNÍ	KONJUNKTIVNÍ	DISJUNKTIVNÍ	INKLUZIVNÍ
FINALIZAČNÍ	Neexistuje	Konjunktivně finalizační	Disjunktivně finalizační	Inkluzivně finalizační
DETERMINISTICKÝ	Iniciativně deterministický	Konjunktivně deterministický	Disjunktivně deterministický	Inkluzivně deterministický
SIMULAČNÍ (STOCHASTICKÝ)	Iniciativně simulační	Konjunktivně simulační	Disjunktivně simulační	Inkluzivně simulační
REDUKČNÍ (FILTRAČNÍ)	Neexistuje	Konjunktivně redukční	Disjunktivně redukční	Inkluzivně redukční

Zdroj: Vlastní zpracování

Pravděpodobně proti očekávání se uživatel s grafickou podobou výše uvedených uzlů v samotné aplikaci nikdy nesetká. Naproti tomu jsou v programu uzly pojmenovány přesně podle výše uvedeného názvosloví.

Uzly se generují po volání daného procesu uživatelem, kdy se podle první tabulky tvoří skupiny činností, které mají stejné předchůdce. Pak můžeme všechny tyto uvedené předchůdce považovat za vstupující činnosti a všechny ID těchto hran za vystupující činnosti z daného uzlu. Při generování uzlů dochází k uzamčení horní tabulky, kterou od této chvíle není možné editovat. Tento proces je zobrazen na obrázku č. 24.

Obrázek 24: Spuštění příkazu pro generování uzlů a uzamknutí tabulky činností



Zdroj: Vlastní zpracování

Část algoritmu, který zajišťuje tento proces generování uzlů, se nachází pod tímto textem, označený jako kód č. 6. Celý algoritmus je pak v příloze této práce.

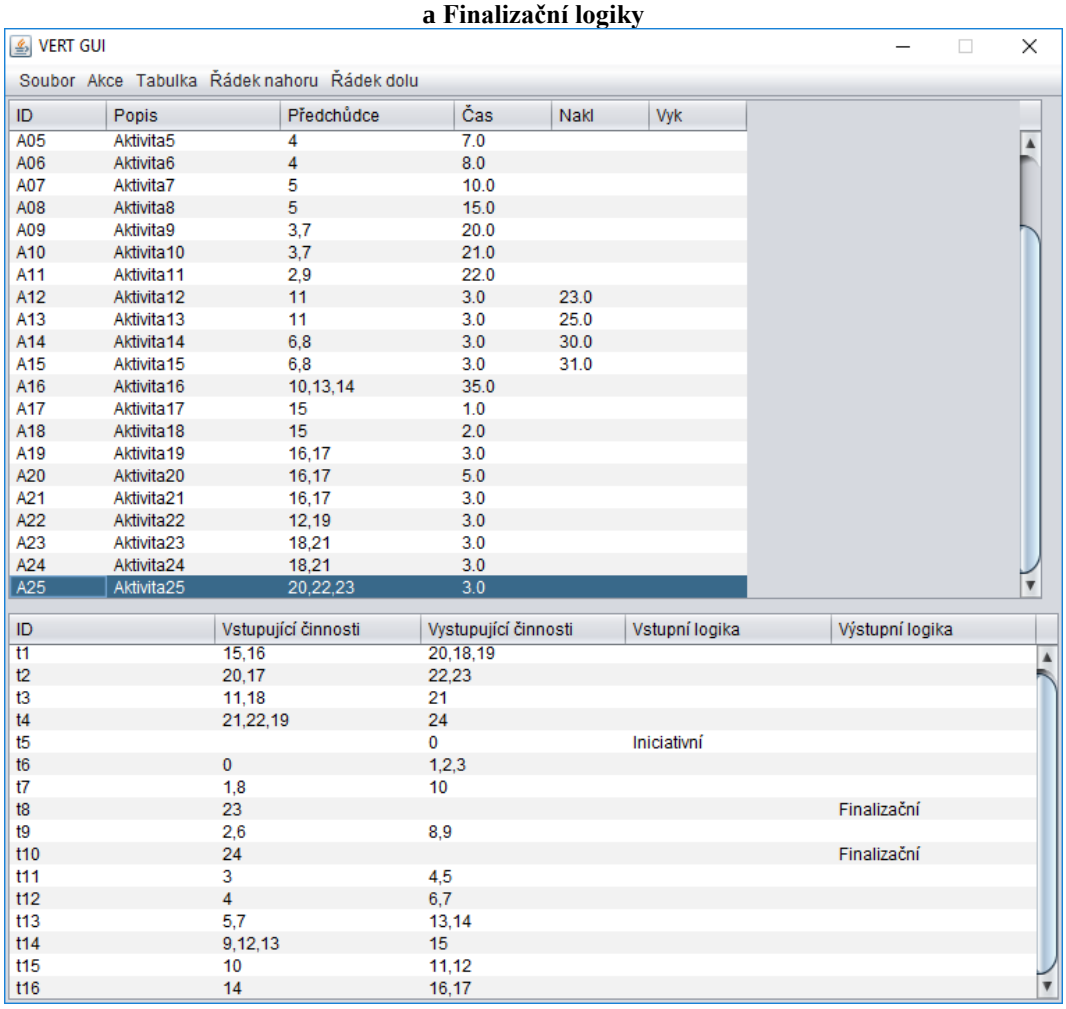
Kód 6: Generování uzlů na základě předchůdců činností

```
        public String getActivitiesString() {
        String res = "";
        for (Object o : this.activities) {
            String item = o.toString();
            res = res + item + ",";
        }
        res = res.substring(0, res.length()-1);
        return res;
    }
    public String getPredecessor() {
        String res = "";
        for (Object item : this.pre.keySet()) {
            res = res + item.toString() + ",";
        }
```

Zdroj: Vlastní zpracování

Specifickou podobu mají uzly, které se vyskytují na začátku, případně na konci projektu. Ty totiž nemají definovaného žádného předchůdce, resp. nikdy se jako předchůdci nevyskytují. O těchto uzlech je tak možné okamžitě rozhodnout, že mají iniciativní vstupní nebo finalizační výstupní logiku. Na druhou stranu můžeme s jistotou říci, že žádný jiný uzel nemůže mít tuto vstupní nebo výstupní logiku, a proto jsou v obou případech tyto uzly dopředu označeny již se spuštěním procesu generování, jak je zachyceno na obrázku č. 25.

Obrázek 25: Generované uzly při výpočtu metody VERT včetně automatické specifikace Iniciativní a Finalizační logiky



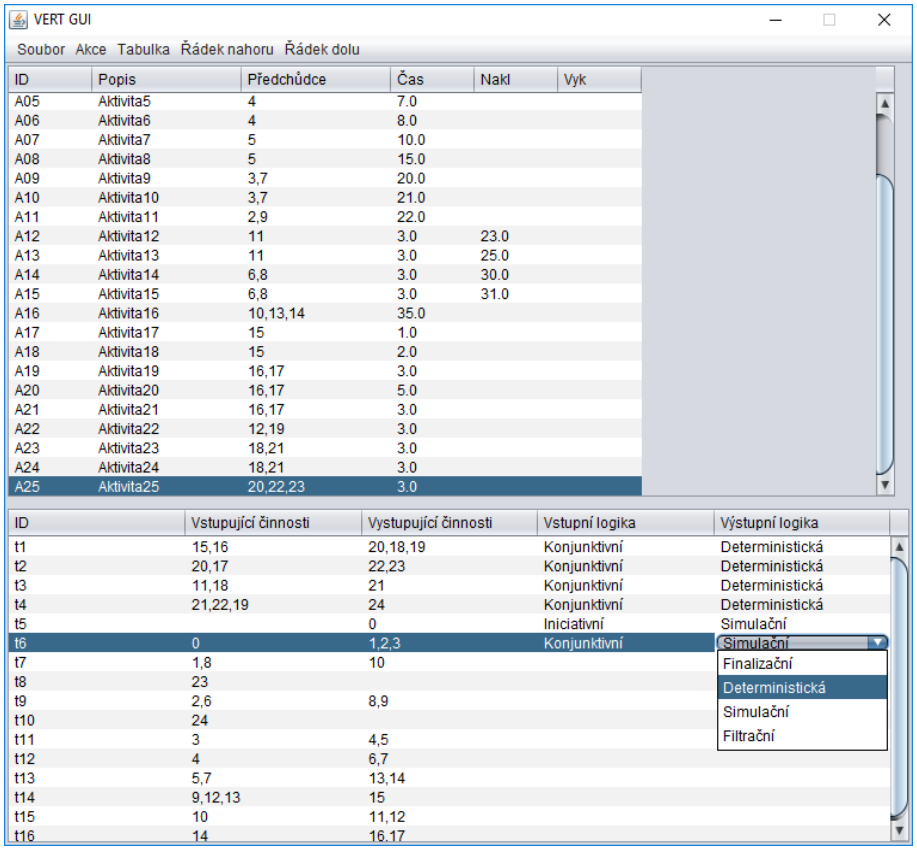
ID	Popis	Předchůdce	Čas	Nakl	Vyk
A05	Aktivita5	4	7.0		
A06	Aktivita6	4	8.0		
A07	Aktivita7	5	10.0		
A08	Aktivita8	5	15.0		
A09	Aktivita9	3,7	20.0		
A10	Aktivita10	3,7	21.0		
A11	Aktivita11	2,9	22.0		
A12	Aktivita12	11	3.0	23.0	
A13	Aktivita13	11	3.0	25.0	
A14	Aktivita14	6,8	3.0	30.0	
A15	Aktivita15	6,8	3.0	31.0	
A16	Aktivita16	10,13,14	35.0		
A17	Aktivita17	15	1.0		
A18	Aktivita18	15	2.0		
A19	Aktivita19	16,17	3.0		
A20	Aktivita20	16,17	5.0		
A21	Aktivita21	16,17	3.0		
A22	Aktivita22	12,19	3.0		
A23	Aktivita23	18,21	3.0		
A24	Aktivita24	18,21	3.0		
A25	Aktivita25	20,22,23	3.0		

ID	Vstupující činnosti	Vystupující činnosti	Vstupní logika	Výstupní logika
t1	15,16	20,18,19		
t2	20,17	22,23		
t3	11,18	21		
t4	21,22,19	24		
t5		0	Iniciativní	
t6	0	1,2,3		
t7	1,8	10		
t8	23			Finalizační
t9	2,6	8,9		
t10	24			Finalizační
t11	3	4,5		
t12	4	6,7		
t13	5,7	13,14		
t14	9,12,13	15		
t15	10	11,12		
t16	14	16,17		

Zdroj: Vlastní zpracování

Ve chvíli, kdy jsou uživateli vygenerovány uzly, je možné se přesunout do další části, a tedy zadávání jejich logiky. Ta probíhá výběrem z několika možností (z comboboxu) přímo v řádku tabulky, a může být doplněna dalšími vstupními daty (viz. příloha), jak je naznačeno na obrázku č. 26.

Obrázek 26: Ohodnocování uzlů



Zdroj: Vlastní zpracování

Po zadání vstupní a výstupní logiky uzlů je projekt připraven pro další zpracování a provedení simulace. Ta se spouští z horní nabídky „Akce“ výběrem možnosti „Spustit projekt“.

6.3.4.3 Simulace

Simulace se v aplikaci vyskytuje ve třech případech. V prvním případě je metoda Monte Carlo využita ve chvíli, kdy se spustí celkový výpočet. Ten pošle skrz síťový graf (dle výchozího nastavení) stokrát tok, který je následně analyzován a vyhodnocen z hlediska četnosti a délky trvání. Ve druhém případě je simulace použita při rozhodování v uzlech (ve výstupní logice) u simulačních uzlů. Z nich je vždy vybrána jediná následující činnost právě na základě provedené simulace a přiřazené pravděpodobnosti. Třetí případ nastává při stanovení parametrů činnosti (délky trvání, nákladů a výkonu), přičemž simulace je využita v případě definování zmíněných ukazatelů pomocí statistických rozdělení.

Vzhledem k tomu, že simulace je pouze operace na pozadí aplikace, není ji možné na tomto místě blíže popsat nebo ji graficky zachytit.

6.3.4.4 Výpočet

Samotný výpočet metody VERT se vcelku neliší od metody PERT, neboť obě metody využívají pro samotný výpočet stejný algoritmus. Nicméně v tomto případě je zapotřebí doplnit, že kromě samotného algoritmu je nezbytné provést několik dalších, níže představených operací.

Prvním krokem je ohodnocení hran, které mohou být statické v případě definování hrany konstantou, případně dynamické, pokud je tato hrana definována statistickým rozdělením, které vychází z generovaného náhodného čísla.

Druhou operací je tzv. eliminace hran, kdy na základě stanovené logiky vstupních a výstupních uzlů je rozhodnuto, které hrany budou odstraněny a které z nich zůstanou jako součást projektu. K eliminaci hran dochází při simulační nebo filtrační výstupní logice (v tomto případě mají svůj význam i výše uvedené hodnoty hran), případně při konjunktivní vstupní logice, která odstraní celý uzel a s ním i všechny vystupující činnosti.

Po provedení druhé operace nastává třetí fáze výpočtu, kterou je výpočet kritické cesty podle standardní logiky výpočtu, vycházející z „přehodnoceného“ projektu.

Celý výše popsaný proces je velmi náročný a je možné jej nalézt v příloze této práce. Jeho dílčí část je uvedena v rámečku pod tímto textem.

Kód 7: Dílčí část algoritmu pro výpočet metody VERT

```
public class VertAlgorithm {

    public static void main(String[] args) {
        try {

            Graph g = new Graph("data/input1.in");
            if (g.getTransitions().length == 0) {
                g.generateTransitions();
            }
            Simulation s = new Simulation(g);
            Activity[] result = s.run();

            FileProcessing();
            CreateGraphProgrammatically();

        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Zdroj: Vlastní zpracování

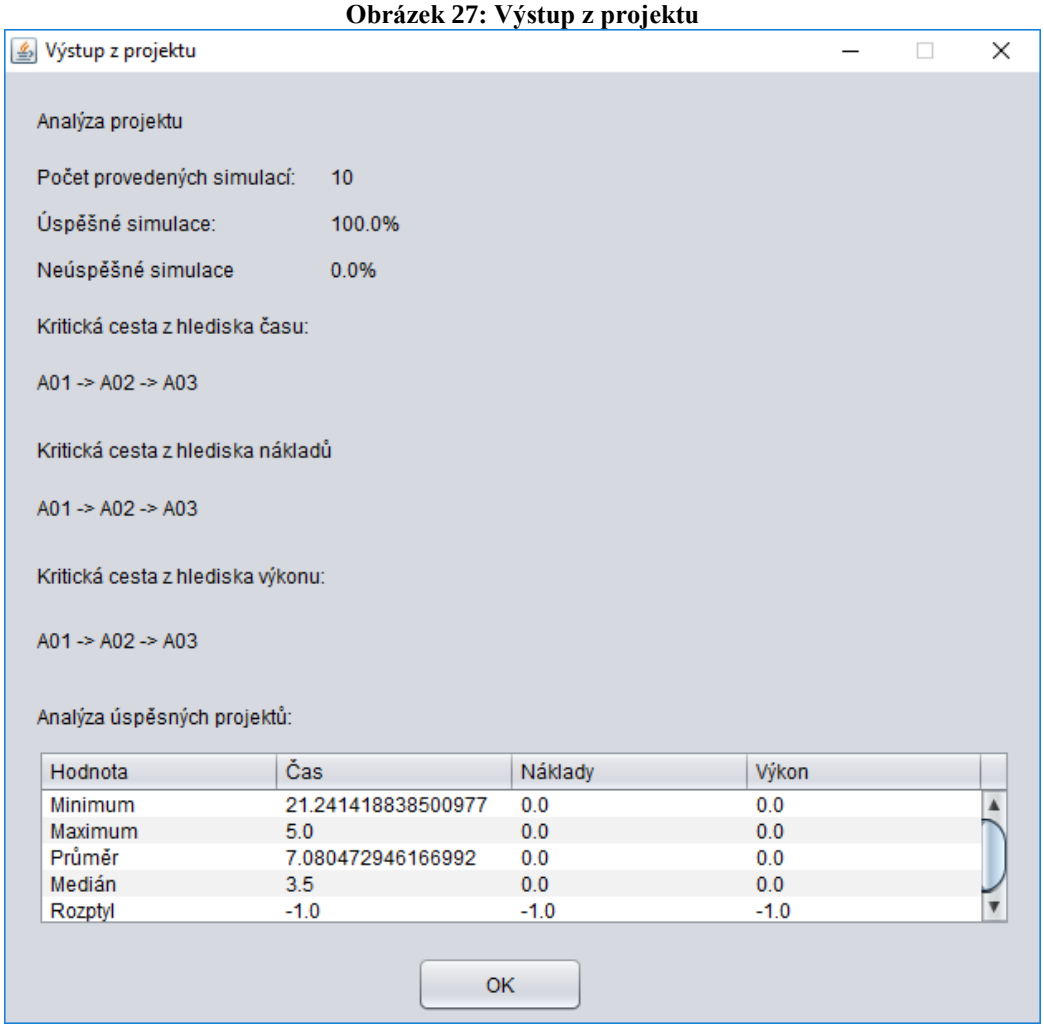
Na závěr tohoto výpočtu je dobré říct, že celý výše uvedený výpočet je pouze jednou z několika desítek či stovek simulací, které projektem prochází. Zaznamenané kritické

cesty se ukládají do přiloženého textového souboru a jejich statistické vyhodnocení se zobrazuje ve výstupním dialogovém okně.

6.3.4.5 Statistické ukazatele

Výstup z aplikace je v tomto případě daleko sofistikovanější než v případě metody PERT. Dialogové okno je rozděleno na dvě části – horní a spodní. Horní část informuje o projektu jako celku, zobrazuje celkový počet simulací, podíl úspěšných a neúspěšných projektů na celku a níže je pak zobrazena kritická cesta z hlediska času, nákladů a výkonu. Další informace jsou zobrazeny ve spodní části tabulky. Z hlediska času, nákladů i výkonu je vypočítána nejdelší a nejkratší zaznamenaná délka trvání. Následně průměrná hodnota, medián a také rozptyl získaných hodnot. Ukázka takové tabulky je zachycena pod tímto textem na obrázku č. 27 (uvedené údaje jsou smyšlené).

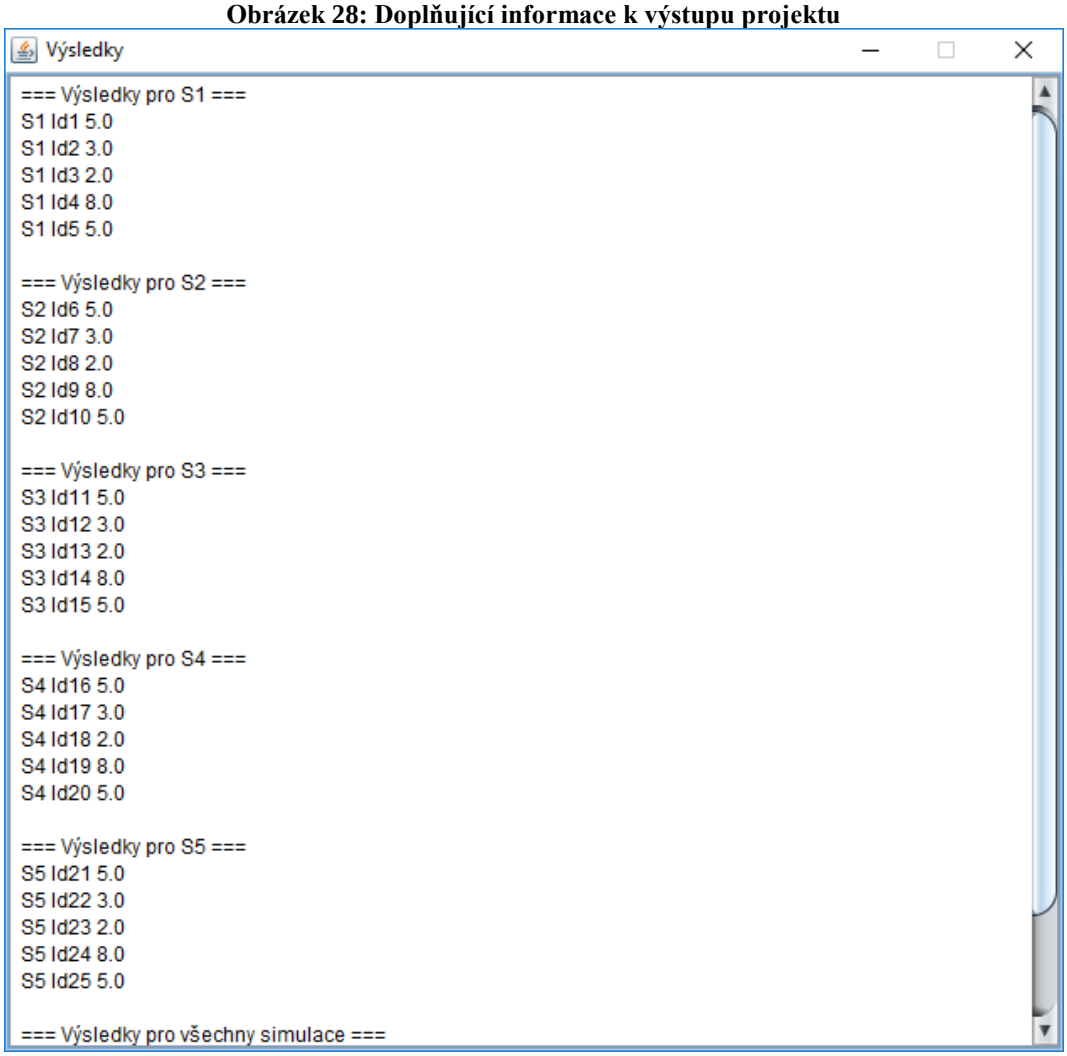
Obrázek 27: Výstup z projektu



Zdroj: Vlastní zpracování

Pokud by chtěl uživatel tato data verifikovat a zjistit jednotlivé kritické cesty, pak je tento seznam dostupný po vyžádání dalších informací v okně výstupu. Podoba tohoto doplňku k výstupu je na obrázku č. 28.

Obrázek 28: Doplňující informace k výstupu projektu



Zdroj: Vlastní zpracování

6.3.5 Informace o projektu

Informace o projektu je obecná statická stránka, seznamující uživatele s aplikací. Uživatel tady může zjistit základní informace týkající se tohoto projektu vývoje aplikace, informace o metodě VERT, případně o metodě PERT a také základní informace o autorovi této práce i samotné aplikace. Přepis tohoto textu je k dispozici v příloze této práce.

6.4 Testování aplikace

V rámci tohoto testování je aplikace zkoušena z funkčního hlediska pro ověření funkčnosti všech náležitostí. Vzhledem k jednoduchosti projektu a menšímu množství dat slouží dané výsledky pouze k verifikaci této aplikace, resp. výsledných dat.

6.4.1 Zadání úlohy

Se zadáním úlohy byl čtenář seznámen již v kapitole 5. Jedná se o uvedený „*Palačinkový problém,*“ kdy žena, vracející se z práce, připravuje svým dětem večeři. Tato úloha již má definovanou logiku jednotlivých uzlů.

Pro reálný výpočet je nyní nezbytné doplnit chybějící informace – délky trvání činností a pravděpodobnosti simulačních uzlů. Pro začátek a zjednodušení úlohy postačí, když bude sledováno pouze časové hledisko této úlohy. Délky trvání jednotlivých činností jsou uvedeny v tabulce pod textem.

Tabulka 6: Hodnoty "Palačinkového problému"

KÓD	NÁZEV ČINNOSTI	ZADÁNÍ	HODNOTA
A.01	*Kontrola hladké mouky*	*Konstanta*	*1*
A.02	*Kontrola polohrubé mouky*	*Konstanta*	*1*
A.03	*Kontrola mléka*	*Konstanta*	*1*
A.04	*Kontrola vajec*	*Konstanta*	*2*
A.05	*Surovina není/Surovina je*	*Konstanta*	*0*
A.06	*Míchání*	*St. rozdělení*	*Rovnoměrné, MIN = 2; MAX = 4*
A.07	*Smažení*	*Mat. vztah*	*A.07 = A.06 · 3*
A.08	*Podávání večeře*	*Konstanta*	*2*

Zdroj: Vlastní zpracování

Činnost *A.06* je definována rovnoměrným statistickým rozdělením, protože nemůžeme rozhodnout, jak dlouho bude tento proces trvat. Nemůžeme totiž rozhodnout, jestli přijde domu unavená nebo ne, případně jestli jí děti budou chtít pomoci a nebo jestli bude připravovat malou či velkou porci palačinek.

Následující činnost *A.07* vychází z matematického vztahu, který je definován následovně:

$$A.07 = A.06 \times 3$$

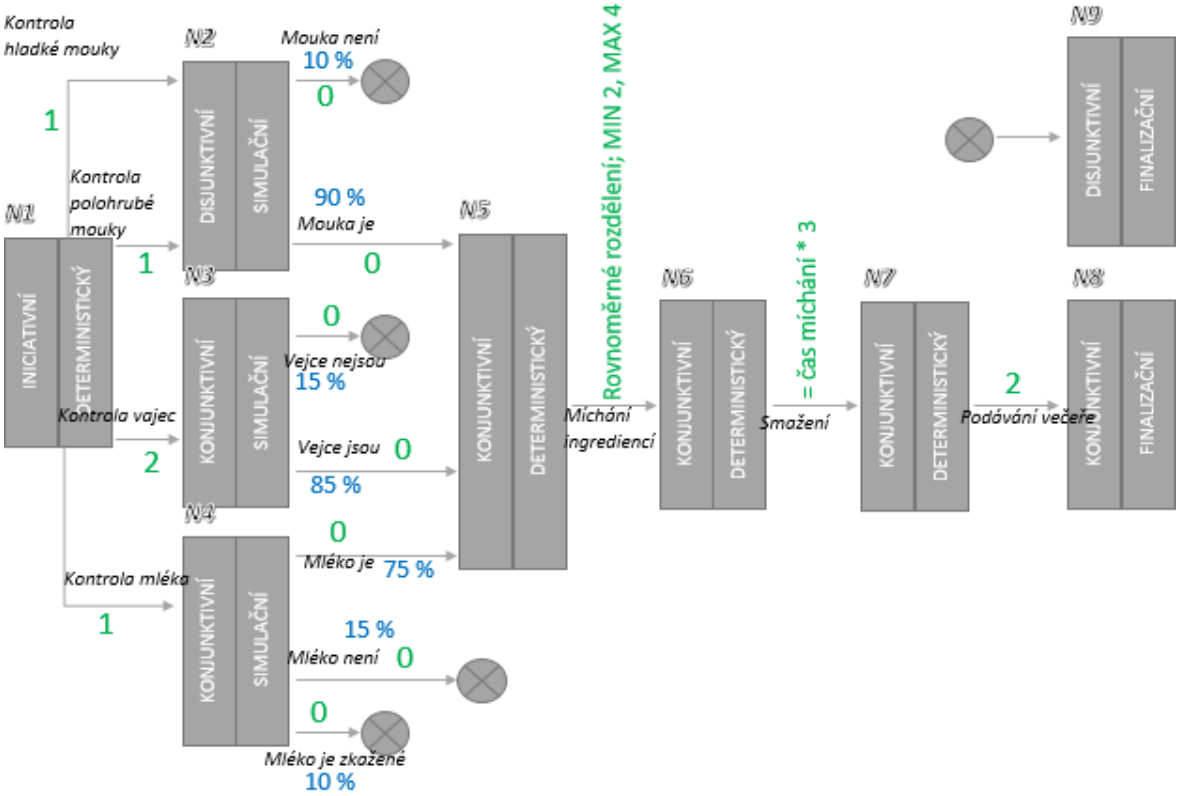
To znamená, že délka trvání smažení přímo závisí na délce trvání míchání. Situaci můžeme vysvětlit tak, že pokud žena připravuje větší porci a míchá jí tedy delší dobu, pak ji bude déle i smažit.

Pro zkompletování zadání doplňme ještě pravděpodobnosti simulačních uzlů:

- Simulační výstup N2 (mouka)
    - Mouka je: 90 % (náhodné číslo je 0,0 - 0,90)
    - Mouka není: 10 % (náhodné číslo je 0,91 - 1,0)
- Simulační výstup N3 (vejce)
    - Vejce jsou: 85 %
    - Vejce nejsou: 15 %
- Simulační výstup N4 (mléko)
    - Mléko je: 75 %
    - Mléko není: 15 %
    - Mléko je, ale zkažené: 10 %

Nyní již nic nebrání sestavení grafické podoby této úlohy, která je zachycené na obrázku č. 29 pod textem.

Obrázek 29: Palačinkový problém doplněný hodnotami a nastavenými pravděpodobnostmi



Zdroj: Vlastní zpracování

6.4.2 Řešení úlohy

Řešení této úlohy je zaznamenáno ve formě několika statistických hodnot. Je ale dobré si uvědomit, co tyto hodnoty v sobě skrývají a co znamenají. Dále je vysvětleno, jakých hodnot může tato úloha nabývat a jak by tedy mělo vypadat korektní řešení.

Z hlediska realizace tohoto projektu je nezbytné, aby byly k dispozici všechny potřebné suroviny – tedy mléko, mouka i vejce. Pokud alespoň jedna z těchto podmínek není splněna, projekt není možné dokončit. Pod tímto textem je procentuálně vyjádřena pravděpodobnost úspěšného a neúspěšného dokončení projektu.

Úspěšné dokončení	*0,57375 ≈ 57,4 %*
	(= 0,9 · 0,85 · 0,75)
Neúspěšné dokončení	*0,42625 ≈ 42,6 %*
	(= 1 – 0,57375)

Délka trvání projektu vychází z jednotlivých, na sebe navzájem navazujících činností, které leží na kritické cestě. Kritická cesta je v případě metody VERT určena jako cesta, která měla v jednotlivých simulacích nejdelší délku trvání. V tomto případě leží na následujících činnostech.

Kontrola vajec	*2*
Vejce jsou	*0*
Míchání ingrediencí	*Rovnoměrné rozdělení, MIN = 2; MAX = 4*
Smažení	*Matematický vztah, (3 · $t_{míchání\ ingrediencí}$)*
Podávání večeře	*2*

Projekt tak může trvat od *12 časových jednotek* do *20 časových jednotek* v závislosti na generované hodnotě rovnoměrného rozdělení u míchání ingrediencí. Zároveň můžeme tvrdit, že v polovině až ⅔ simulací by měl být projekt úspěšně dokončen.

Nyní zkusme tedy porovnat tyto očekávané hodnoty se skutečně získanými na základě výpočtu v aplikaci.

Výsledky uvedené v tabulce č. 7 pod tímto textem jednoznačně potvrzují výše očekávané závěry. Úspěšně bylo dokončeno celkem 59 % simulací při očekávání zhruba 57,4 %,

rozdíl je tedy méně než dva procentní body. Získaná minimální i maximální délka trvání se odchyluje od původního očekávaného odhadu o necelou jednotku.

Tabulka 7: Výsledky získané simulací v aplikaci

VÝSTUP Z APLIKACE	HODNOTA
Úspěšně dokončeno	*59 % (59 simulací ze 100)*
Neúspěšně dokončeno	*41 % (41 ze 100)*
Nejkratší délka trvání	*12,84*
Nejdelší délka trvání	*19,71*

Zdroj: Vlastní zpracování

6.5 Charakteristiky aplikace

V této části jsou doplněny technické informace a charakteristiky, které se vážou k vytvořené aplikaci. Nejprve jsou uvedeny základní informace, kde jsou uvedeny nezbytné prvky, které je třeba mít v osobním počítači pro spuštění aplikace, uvedena je velikost nebo očekávaná rychlost této aplikace. V další části je komentován název aplikace, v následné části logo a grafické prvky. Poslední dva oddíly jsou věnovány dostupnosti aplikace a vznikajícím webovým stránkám o aplikaci.

6.5.1 Základní informace

Pro spuštění finální aplikace je nezbytné mít v PC nainstalovaný JVM[100] (*Java Virtual Machine*), který je dnes však již součástí každého moderního počítače jako součást operačního systému. JVM a API[101] (*Application Programming Interface*) společně tvoří celek, který je poskytován jako *Java Runtime Environment*[102] (JRE).

Cílem je, aby konečná velikost aplikace byla maximálně 10 MB a je k této práci v omezené verzi přiložena na CD, které se nachází na zadních deskách. Aplikaci je možné jak spustit z tohoto kompaktního disku, tak je možné ji i přesunout do počítače. Aplikace však vyžaduje, aby do místa, kde je aktuálně umístěn „*jar*“ soubor, byl povolen zápis. K pracovním souborům je vhodné dodat, že se ukládají do dokumentů.

[100] Java Virtual Machine (JVM) je sada počítačových programů a datových struktur, která využívá modul virtuálního stroje ke spuštění dalších počítačových programů a skriptů vytvořených v jazyce Java. Úkolem tohoto modulu je zpracovat pouze tzv. mezikód, který je v Javě označován jako Java bytecode.
[101] Sada standardních knihoven
[102] Ke stažení z: http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html

6.5.2 Název aplikace

Ačkoliv název aplikace není klíčový, je vhodné pro tyto potřeby značku vytvořit. V tomto případě je název „*PERT & VERT Solver*“. Užívanějším označením je však zkratka původního názvu - jednoduše „*PEVESO*.“ Ta je však ve většině případů psaná pomocí tzv. *CamelCase* ve stylu „*PeVeSo*“ pro snazší pochopení jednotlivých slabik.

6.5.3 Logo a grafické prvky

Doplňkem a nezbytnou součástí aplikace jsou její grafické prvky, primárně logo. To je uživateli zobrazeno v hlavní nabídce po spuštění aplikace společně s dalšími grafickými prvky. Logo má černobílé provedení s černým křížem uprostřed a ve vzniklých kvadrantech jsou umístěny jednotlivé slabiky názvu aplikace. Grafická podoba loga je zobrazena na obrázku č. 30 pod textem.

Obrázek 30: Logo aplikace

Zdroj: Vlastní zpracování

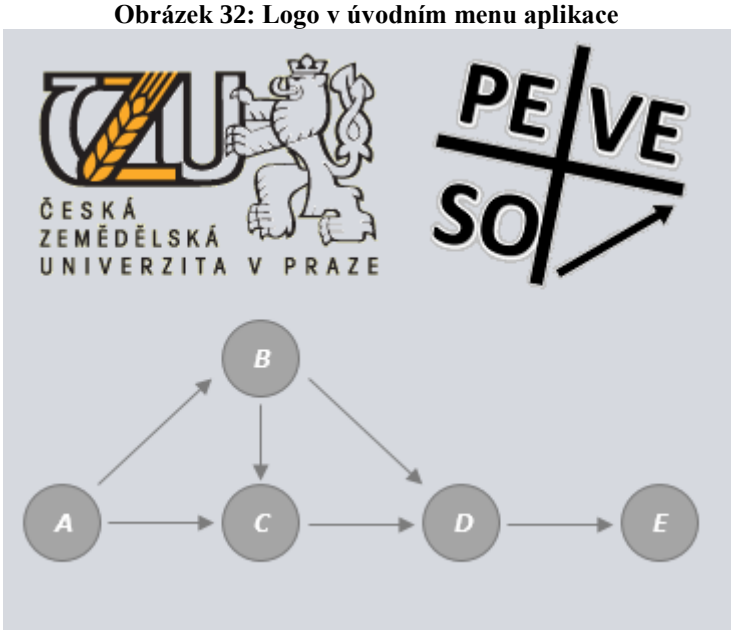
Pro spuštění aplikace v prostředí Microsoft Windows je nezbytné vytvořit ikonu (obrázek č. 31). Ta vychází z grafické podoby původního loga a je pouze jeho zmenšeninou. Spuštění aplikace pak probíhá standardním dvojklikem. Ikona bude součástí první aktualizace této aplikace.

Obrázek 31: Ikona (vedoucí ke spuštění aplikace)

Zdroj: Vlastní zpracování

V úvodním menu aplikace je zobrazen odlišný typ loga, které vychází z výše uvedené grafiky a který je dále doplněn logem univerzity v omezené barevné úpravě. Poslední součástí tohoto grafického objektu je v dolní části umístěný fiktivní projekt, sestavený pomocí několika neohodnocených uzlů a hran. Grafická podoba tohoto objektu je umístěna na obrázku č. 32.

Obrázek 32: Logo v úvodním menu aplikace



Zdroj: Vlastní zpracování

6.5.4 Dostupnost aplikace na trhu

Aplikace je ve své současné omezené i budoucí finální podobě na trhu k dispozici jako freeware, tedy zcela zdarma se všemi svými funkcemi a bez časového omezení. Přesto není vyloučeno, že pozdější (aktualizované) verze budou dostupné pouze jako shareware.[103]

[103] Označení pro software, který je chráněn autorským právem, ale je možné jej volně distribuovat. Uživatel má možnost software po určitou dobu zkoušet, a tedy zjistit zda mu vyhovuje nebo ne. Pokud ho ale nadále používá, je povinen se řídit podle autorovy licence a zpravidla zaplatit cenu za používání programu.

V současné době, kdy je k dispozici omezená verze aplikace a probíhá dokončování první aktualizace, probíhá také jednání s několika servery o možné distribuci programu jako freeware (*www.stahuj.centrum.cz*, *www.slunecnice.cz*, apod). K dispozici však bude aplikace v každém případě na *www.uloz.to* a na jednoduchých webových stránkách s doménou třetího řádu.

6.5.5 Webové stránky

Vzhledem ke složitému procesu dohledávání informací o dané aplikaci, metodě a jejímu dalšímu vývoji jsou vytvořeny stránky s doménou třetího řádu (*www.metodavert.xf.cz*), kde se nachází souhrnné informace o metodě, aktualizace a odkaz ke stažení software. Grafická podoba webových stránek je zobrazena na obrázku č. 33.

Obrázek 33: Grafická podoba webové stránky v prostředí internetového prohlížeče Google Chrome

Zdroj: Vlastní zpracování

7 Použití aplikace

Pro ověření spolehlivosti aplikace se autor práce rozhodl podrobit výsledný program testu. Ten spočívá v řešení běžné úlohy řízení projektů pomocí metody VERT. V této kapitole je postupně prezentováno zadání jak v ekonomickém, tak matematickém modelu. Stručně je rozebráno i očekávané řešení této úlohy. Součástí práce jsou výstupy z aplikace, jejich stručná analýza a také diskuze nad procesem výpočtu aplikace a uživatelské přívětivosti aplikace.

Následující úloha se zabývá přípravou a organizací badmintonového turnaje v Plzni. V rámci projektového managementu můžeme tento projekt kategorizovat do Event managementu, který je specifický především tím, že výsledné projekty mají neúměrně krátkou délku trvání vzhledem k délce trvání příprav.

Tento projekt je vybrán záměrně, neboť je to ideální příklad projektu, pro který je tato aplikace stvořena. Projekt je charakterizován zhruba 30 činnostmi a všemi možnými typy uzlů, které mají maximálně tři vstupující a tři vystupující hrany. Pro charakteristiku jednotlivých činností jsou parametry stanoveny svým časem, náklady a ve vybraných případech i výkonem, a to za použití všech možností zadávání.

7.1 Ekonomický model

Příprava tohoto turnaje začíná vytvořením nezbytné a oficiální žádosti u zprostředkovatele (města nebo badmintonového klubu). V tomto případě sice využíváme prostor vybraného badmintonového klubu, nicméně z finančního a organizačního hlediska je vhodnější podat žádost přímo na městském úřadě. Ten ji může přijmout nebo zamítnout.

Pokud je tato žádost zamítnuta, projekt končí bez zisků nebo vysokých ztrát. V opačném případě je projekt schválen a může se pokračovat tím, že dojde na sestavení projektového týmu, který ponese za jeho úspěšné dokončení odpovědnost. Schválením také dojde k přislíbení části finančních prostředků od města. Akce je následně připravována ve třech, méně závislých *streamech*, kdy cílem prvního týmu je zajištění finančních prostředků, cílem druhého týmu je marketingová propagace události a cílem posledního týmu je bezchybná organizace celé akce, včetně zajištění doprovodného programu.

Cílem marketingového týmu je dostatečná propagace této události, a to pro potenciální soutěžící a stejně tak i pro diváky. Na základě stanoveného rozpočtu musí tento tým

sestavit dostatečně vhodný marketingový plán, který zajistí dostatečnou poptávku po turnaji. Očekávaná je inzerce v lokálních tiskových materiálech nebo využití plakátovacích ploch. Vzhledem k velikosti zasaženého okolí a rozpočtu dané akce je optimální spustit tuto propagaci dva týdny před začátkem turnaje.

Tým, který se zabývá organizací této akce, se soustředí především na výběr vhodné místa konání a na sestavení ideálního doprovodného programu a zajištění cateringové společnosti. Pro pořádání akce jsou předběžně vybrána tři místa ve městě, a tedy hala „*Branka*“, dále hala „*Na Ostrově*“ a poslední je sportovní hala „*v Základní škole*.“ Hala „*Branka*“ je po všech stránkách nejvhodnější, neboť se běžně používá pro trénování badmintonu a má i vlastní tribunu. Nájem je ale obvykle vyšší, což je ale takřka jediná nevýhoda. Hala „*Na Ostrově*“ je multifunkční víceúčelový prostor, který je ale často obsazený pro pořádání jiných akcí. Nájem je však nižší. Poslední možností je pořádat akci v hale „*v Základní škole*,“ kde je nájem minimální, nicméně pravidla pro pořádání turnaje podléhají pravidlům základní školy, což omezuje délku trvání akce do nočních hodin nebo reguluje prodej alkoholických nápojů.

Klíčovou částí je však získání finančních prostředků pro pořádání této akce. Jak je zmíněno výše, část příspěvků přichází od města, které za touto akcí částečně stojí. Další prostředky mají původ u různých zdrojů. Mohou pocházet od sponzorů ze soukromé sféry ve formě daru a nebo jako cena za umístění reklamy v den konání akce. Dále jako příspěvek na pořádání této akce od kraje, což je další státní instituce, avšak plně nezávislá na městě. Posledním zdrojem finančních prostředků jsou další veřejné zdroje a v případě trochu představivosti u tohoto projektu to může být i Evropská Unie. Pro úspěšné dokončení tohoto projektu – přípravy turnaje, je nezbytné získat dostatečně velké množství těchto finančních prostředků. Pokud jich bude méně než stanovené minimum, nebude možné projekt realizovat, a to povede k jeho neúspěšnému ukončení.

Pro pořádání projektu je nezbytné úspěšné dokončení všech výše uvedených streamů. Není tedy možné spustit projekt s detailně připravenou marketingovou kampaní a vybranou halou, ale bez dostatečného množství finančních prostředků, neboť by byl takový projekt ztrátový a pro pořadatele by byla jeho realizace zcela iracionální.

7.2 Matematický model

Výše popsané zadání představuje ekonomický model úlohy, čili to, jak jej vnímá veřejnost nebo nekvalifikovaný organizační tým. Pro potřeby pořadatele (projektového manažera) a sestavení matematického modelu je nezbytné doplnit výše uvedené informace o číselné hodnoty. Dále definovat v projektu tzv. činnosti a na jejich základě sestavit síťový graf, který bude ohodnocen délkou trvání, vzniklými náklady a generovaným výkonem.

Takřka všechny činnosti jsou charakterizovány svou délkou trvání a také náklady na realizaci. Vybrané činnosti, které charakterizují získávání finančních prostředků, mají definovaný i svůj výkon.

O schválení a případném zamítnutí projektu se rozhoduje na základě simulace Monte Carlo, tedy generováním náhodného čísla. Jednotlivé možnosti tak mají stanovenou pravděpodobnost realizace.

Projekt bude schválen	85 %
Projekt bude zamítnut	15 %

Možnost pořádání akce na výše uvedených lokacích je ohodnocena pravděpodobností, přičemž o výběru daného místa rozhodne při výpočtu simulace Monte Carlo. Samozřejmě existuje i možnost, že se pořadatelé nedohodnou a projekt bude zrušen.

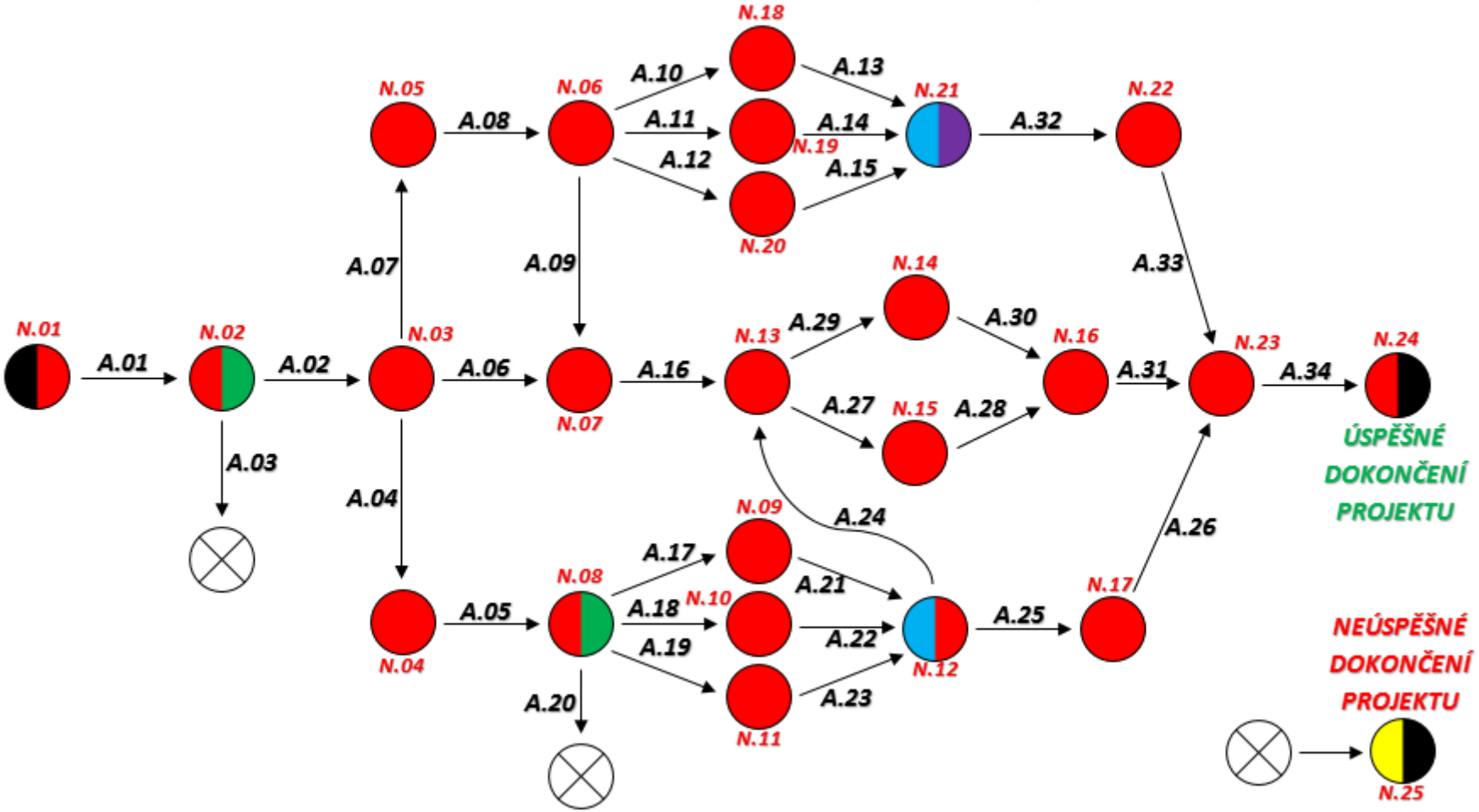
Hala „*Branka*“	45 %
Hala „*Na Ostrově*“	30 %
Hala „*v Základní škole*“	20 %
Zrušení projektu	5 %

Pro vyhodnocení získání dostatečného množství finančních prostředků je v této úloze zařazen uzel s filtračním výstupem. Ten porovnává generovaný *výkon* u jednotlivých činností, které představují získávání finančních prostředků od stanovených zdrojů pro realizaci této akce, přičemž hodnota stanoveného výkonu odpovídá množství získaných financí v tisících.[104]

Grafické znázornění daného problémů je uvedeno na obrázku č. 34 (na str. 99), který znázorňuje síťový graf.

[104] Pokud se tedy podaří získat ze tří uvedených zdrojů dostatečné množství finančních prostředků, projeví se to kladně na výkonu, který filtrační uzel vyhodnotí jako dostačující a schválí pokračování v tomto projektu. Pokud se však z těchto zdrojů nepodaří získat dostatek finančních prostředků, nebude výkon dostatečný a projekt bude neúspěšně ukončen.

Obrázek 34: Praktická aplikace metody VERT – Pořádání badmintonového turnaje



Zdroj: Vlastní zpracování

Tabulka č. 8 pod tímto textem zachycuje klíčové údaje, které se týkají tohoto projektu, konkrétně zadané činnosti všech tří parametrů – času, nákladů a výkonu. Parametry jsou nejčastěji definovány podle konstanty, ale vyskytuje se zde také Rovnoměrné rozdělení (značené RR), Normální rozdělení (NR) a činnosti zadané Matematickým vztahem (MV). Je vhodné doplnit, že hodnoty času v tomto projektu jsou vyjádřeny ve dnech, náklady v českých korunách a výkon v tisících českých korun.

Tabulka 8: Seznam činností projektu včetně jejich ohodnocení

KÓD	ČINNOST	PŘEDCH.	ČAS	NÁKLADY	VÝKON
A.01	Vytvoření žádosti	x	1	300	0
A.02	Schválení žádosti	1	0	0	0
A.03	Zamítnutí žádosti	1	0	0	0
A.04	Vytvoření org. týmu	2	1	0	0
A.05	Analýza vybraných míst	4	4	500	0
A.06	Vytvoření mkt. týmu	2	1	0	0
A.07	Vytvoření fin. týmu	2	1	0	0
A.08	Příprava rozpočtu na akci	7	6	2000	0
A.09	Uvolnění financí na mkt.	8	1	0	0
A.10	Obcházení sponzorů	8	10	3000	RR (15;35)
A.11	Odeslání žádosti na kraj	8	2	600	25
A.12	Žádost o EU dotaci	8	5	1500	NR (1;10;20)
A.13	Finance od sponzorů	10	0	100	0
A.14	Finance od kraje	11	0	100	0
A.15	Finance od EU	12	0	100	0
A.16	Příprava mkt. Plánu	6,9	2	250	0
A.17	Komunikace „Branka“	5	RR (3;5)	1000	0
A.18	Komunikace „Ostrov“	5	RR (3;5)	1000	0
A.19	Komunikace „ZŠ“	5	RR (3;5)	1000	0
A.20	Místo nenalezeno	5	0	0	0
A.21	Dojednání detailů	17	MV = A.17 * 2	0	0
A.22	Dojednání detailů	18	MV = A.18 * 2	0	0
A.23	Dojednání detailů	19	MV = A.19 * 2	0	0
A.24	Předání informací	21,22,23	0	0	0
A.25	Podpis smlouvy o nájmu	21,22,23	1	250	0
A.26	Zapůjčení vybavení	25	3	8000	0
A.27	Příprava plakátů	16,24	RR (2;10)	2500	0
A.28	Vyvěšení plakátů	27	MV = 20 - A.27	1500	0
A.29	Příprava inzerce	16,24	RR (5;15)	500	0
A.30	Objednání inzerce	29	MV = 30 - A.29	4500	0
A.31	Vyhodnocení	28,30	NR (1;2;3)	500	0
A.32	Získání financí od města	13,14,15	2	0	10
A.33	Financování projektu	32	1	0	0
A.34	Dokončovací práce	26,31,33	3	3000	0

Zdroj: Vlastní zpracování

V následující tabulce č. 9 se nacházejí další informace, které tentokrát charakterizují vstupní a výstupní logiku jednotlivých uzlů. Dále také informují o vstupujících činnostech, vystupujících činnostech a dalších informacích, které se týkají vybraných výstupních logik. Odpovídá tak klasickému zadání v aplikaci.

Tabulka 9: Seznam uzlů projektu včetně jejich vstupní a výstupní logiky

KÓD	VSTUP. Č.	VYSTUP. Č.	VS. LOGIKA	VÝS. LOGIKA	DALŠÍ INFO
N.01	x	1	*Iniciativní*	*Determin.*	
N.02	1	2,3	*Konjunktivní*	*Simulační*	85 % / 15 %
N.03	2	4,6,7	*Konjunktivní*	*Determin.*	
N.04	4	5	*Konjunktivní*	*Determin.*	
N.05	7	8	*Konjunktivní*	*Determin.*	
N.06	8	9,10,11,12	*Konjunktivní*	*Determin.*	
N.07	6,9	16	*Konjunktivní*	*Determin.*	
N.08	5	17,18,19,20	*Konjunktivní*	*Simulační*	45% / 30% / 20% / 5%
N.09	17	21	*Konjunktivní*	*Determin.*	
N.10	18	22	*Konjunktivní*	*Determin.*	
N.11	19	23	*Konjunktivní*	*Determin.*	
N.12	21,22,23	24,25	*Inkluzivní*	*Determin.*	
N.13	16	27,29	*Konjunktivní*	*Determin.*	
N.14	29	30	*Konjunktivní*	*Determin.*	
N.15	27	28	*Konjunktivní*	*Determin.*	
N.16	28,30	31	*Konjunktivní*	*Determin.*	
N.17	25	26	*Konjunktivní*	*Determin.*	
N.18	10	13	*Konjunktivní*	*Determin.*	
N.19	11	14	*Konjunktivní*	*Determin.*	
N.20	12	15	*Konjunktivní*	*Determin.*	
N.21	13,14,15	32	*Inkluzivní*	*Filtrační*	Výkon = MIN 60
N.22	32	33	*Konjunktivní*	*Determin.*	
N.23	26,31,33	34	*Konjunktivní*	*Determin.*	
N.24	34	x	*Konjunktivní*	*Finalizační*	Úspěšný konec
N.25	3,20	x	*Disjunktivní*	*Finalizační*	Neúspěšný konec

Zdroj: Vlastní zpracování

Veškeré výše uvedené informace v podkapitole *7.2 – Matematický model* odpovídají údajům, které uživatel musí znát pro zadání projektu do aplikace. Zároveň je nezbytné zmínit, že data nemusí přesně odpovídat tomu, jak je kvalifikuje aplikace – vzhledem k přednastavenému algoritmu tak může mít většina činností a uzlů odlišné kódové označení.

7.3 Parametry úlohy

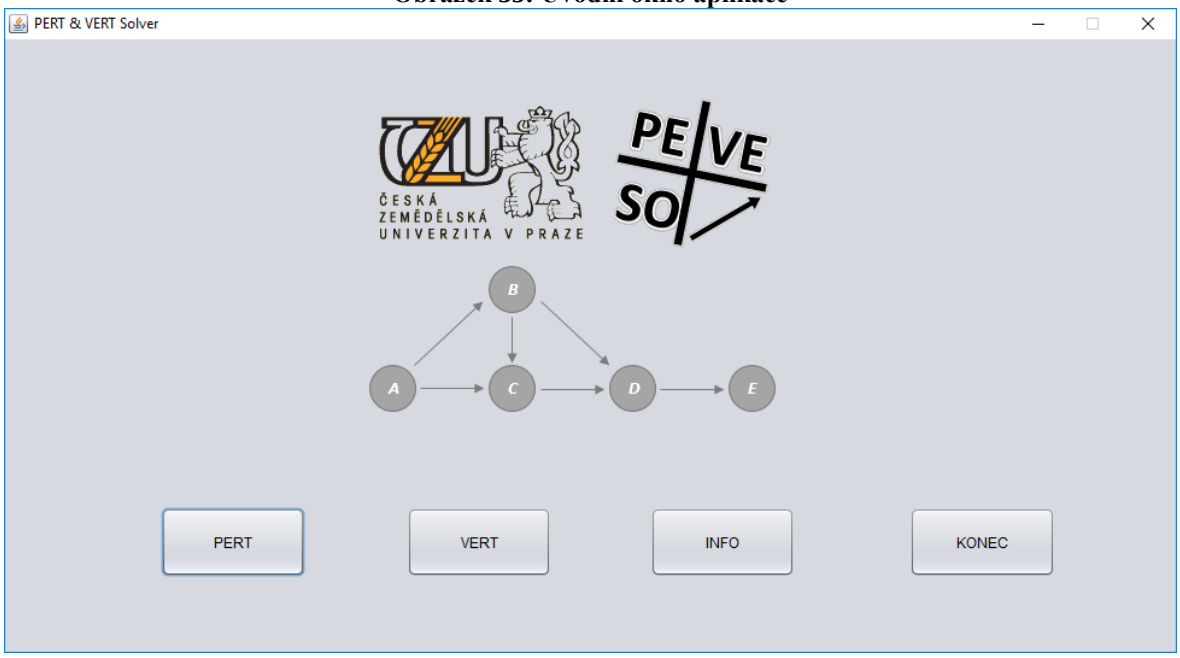
V úloze se vyskytuje celkem 34 činností, které jsou stanoveny převážně konstantou. Vybrané hodnoty jsou také zadány pomocí statistického rozdělení, konkrétně normálního a rovnoměrného rozdělení a nebo jsou charakterizovány matematickým vztahem. V projektu je také celkem 26 uzlů, nejčastěji s konjunktivně-deterministickou logikou. V úloze jsou však využity všechny možné vstupní a výstupní logiky, které aplikace umožňuje použít. Následně je úloha stokrát simulována.

Na základě výše popsané, a v celé kapitole charakterizované úlohy, můžeme hodnotit minimálně fakt, že úloha testuje skutečně všechny možnosti této aplikace. Následující podkapitoly uvádějí číselné řešení úlohy.

7.4 Řešení úlohy v aplikaci

Pro řešení úlohy pomocí metody VERT je nezbytné nejprve spustit aplikaci pro výpočet. V hlavním menu volí uživatel možnost „*VERT*“, čímž spouští složitější z výpočtů a je přesměrován do okna dané metody (obrázek č. 35).

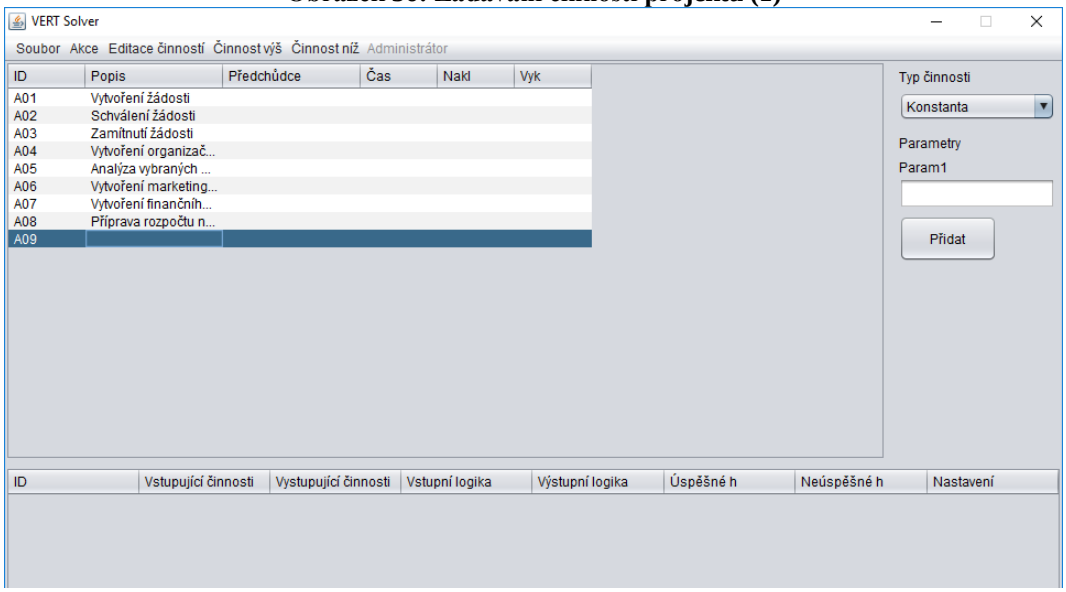
Obrázek 35: Úvodní okno aplikace



Zdroj: Vlastní zpracování

Prostřednictvím uživatelského rozhraní na obrázku č. 36 se do aplikace postupně zadávají činnosti tohoto projektu.

Obrázek 36: Zadávání činností projektu (1)



Zdroj: Vlastní zpracování

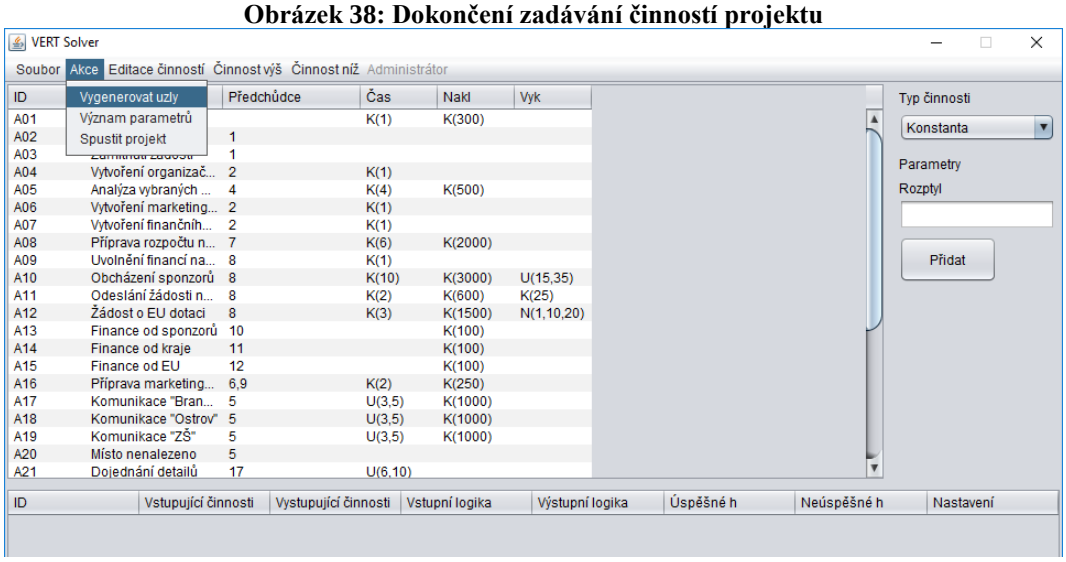
Činnosti je nezbytné ohodnotit jejich parametry a určit jejich předchůdce (obrázek č. 37).

Obrázek 37: Zadávání činností projektu (2)

Zdroj: Vlastní zpracování

Pokud jsou činnosti projektu ohodnoceny, vypadá tabulka jako na obrázku č. 38 pod textem.

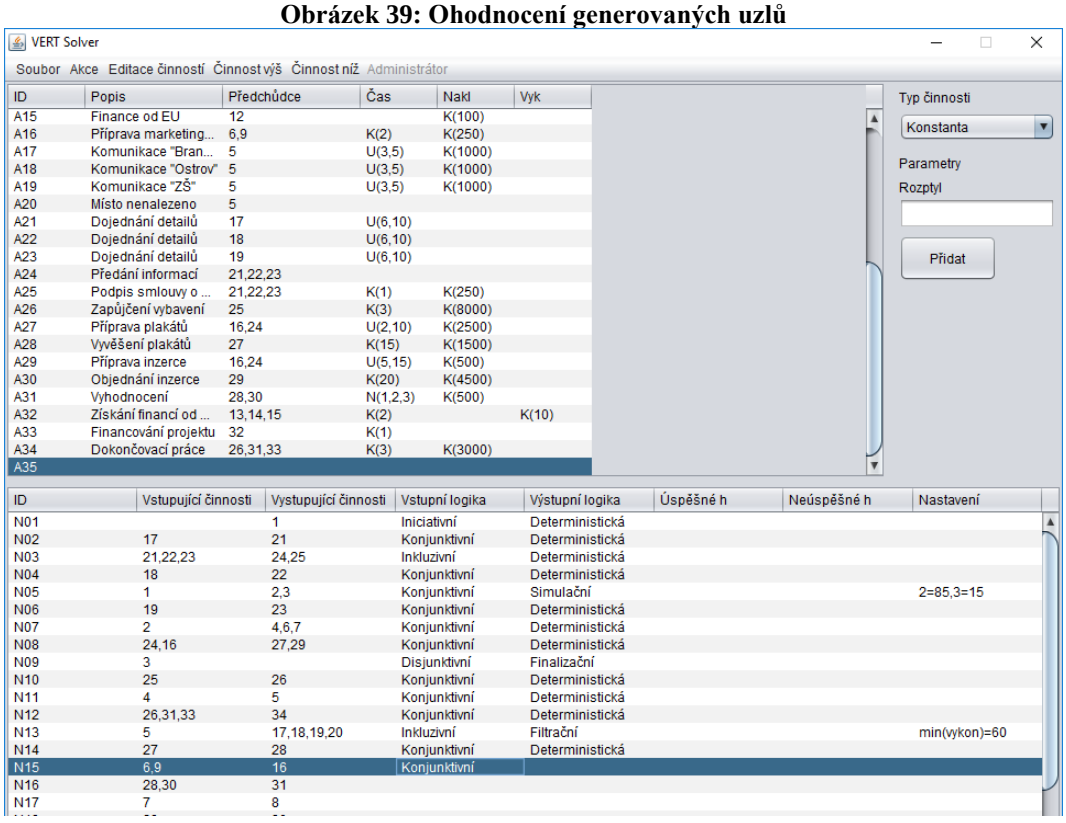
Obrázek 38: Dokončení zadávání činností projektu



Zdroj: Vlastní zpracování

Po zadání činností je aplikace připravena pro další použití. Jak je patrné z obrázku 38, jsou prostřednictvím příkazu „*Vygenerovat uzly*“ vytvořena nová data ve spodní tabulce, zabývající se uzlovou logikou projektu.

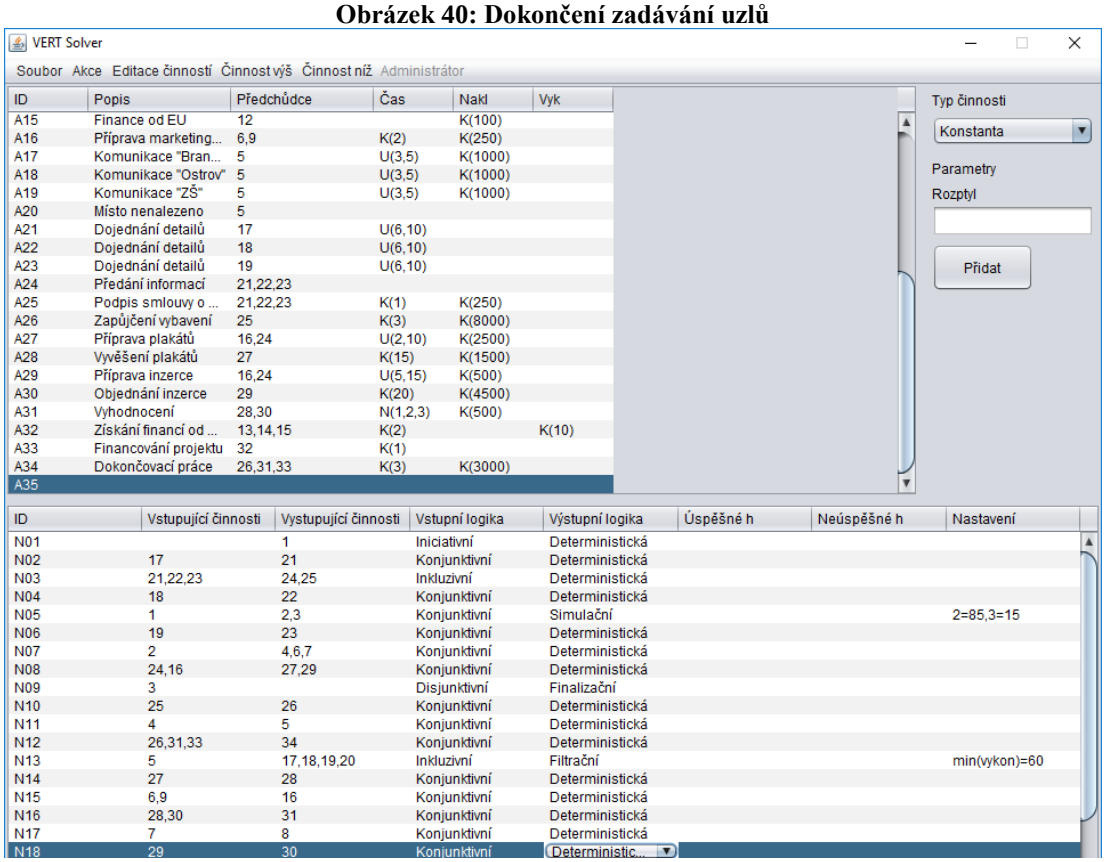
Obrázek 39: Ohodnocení generovaných uzlů



Zdroj: Vlastní zpracování

Za povšimnutí stojí na obrázku č. 39 mimo jiné uzel *N05* a *N13*, které mají simulační, resp. filtrační výstupní logiku. Tyto uzly nestačí zadat výběrem z několika možností, ale je třeba definovat i doplňující podmínky, za kterých se realizují. V tomto případě bude tok z uzlu *N05* pokračovat do činnosti *A02* v 85 % případů, zatímco do *A03* pouze v 15 % případů. Pro realizaci uzlu *N13* je nezbytné, aby výkon dosáhl minimální výše 60.

Obrázek 40: Dokončení zadávání uzlů



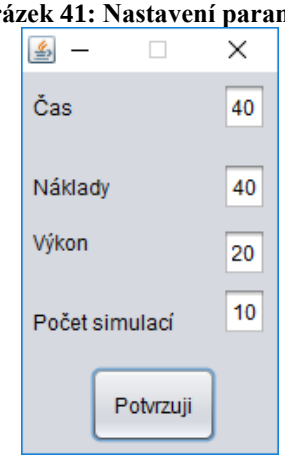
ID	Popis	Předchůdce	Čas	Nakl	Vyk
A15	Finance od EU	12		K(100)	
A16	Příprava marketing...	6,9	K(2)	K(250)	
A17	Komunikace "Bran...	5	U(3,5)	K(1000)	
A18	Komunikace "Ostrov"	5	U(3,5)	K(1000)	
A19	Komunikace "ZŠ"	5	U(3,5)	K(1000)	
A20	Místo nenalezeno	5			
A21	Dojednání detailů	17	U(6,10)		
A22	Dojednání detailů	18	U(6,10)		
A23	Dojednání detailů	19	U(6,10)		
A24	Předání informací	21,22,23			
A25	Podpis smlouvy o ...	21,22,23	K(1)	K(250)	
A26	Zapůjčení vybavení	25	K(3)	K(8000)	
A27	Příprava plakátů	16,24	U(2,10)	K(2500)	
A28	Vyvěšení plakátů	27	K(15)	K(1500)	
A29	Příprava inzerce	16,24	U(5,15)	K(500)	
A30	Objednání inzerce	29	K(20)	K(4500)	
A31	Vyhodnocení	28,30	N(1,2,3)	K(500)	
A32	Získání financí od ...	13,14,15	K(2)		K(10)
A33	Financování projektu	32	K(1)		
A34	Dokončovací práce	26,31,33	K(3)	K(3000)	
A35					

ID	Vstupující činnosti	Vystupující činnosti	Vstupní logika	Výstupní logika	Úspěšné h	Neúspěšné h	Nastavení
N01		1	Iniciativní	Deterministická			
N02	17	21	Konjunktivní	Deterministická			
N03	21,22,23	24,25	Inkluzivní	Deterministická			
N04	18	22	Konjunktivní	Deterministická			
N05	1	2,3	Konjunktivní	Simulační			2=85,3=15
N06	19	23	Konjunktivní	Deterministická			
N07	2	4,6,7	Konjunktivní	Deterministická			
N08	24,16	27,29	Konjunktivní	Deterministická			
N09	3		Disjunktivní	Finalizační			
N10	25	26	Konjunktivní	Deterministická			
N11	4	5	Konjunktivní	Deterministická			
N12	26,31,33	34	Konjunktivní	Deterministická			
N13	5	17,18,19,20	Inkluzivní	Filtrační			min(vykon)=60
N14	27	28	Konjunktivní	Deterministická			
N15	6,9	16	Konjunktivní	Deterministická			
N16	28,30	31	Konjunktivní	Deterministická			
N17	7	8	Konjunktivní	Deterministická			
N18	29	30	Konjunktivní	Deterministic...			

Zdroj: Vlastní zpracování

Po dokončení zadávání vstupní a výstupní logiky uzlů (obrázek č. 40) a nastavení doplňujících informací se přechází k dalšímu kroku – nastavení významnosti parametrů a nastavení celkového počtu simulací. Ve výchozím nastavení jsou parametry rozloženy po 33 procentech pro každý (čas, náklady a výkon) a počet simulací je stanoven na 50. Nastavení probíhá v dialogovém okně na obrázku č. 41.

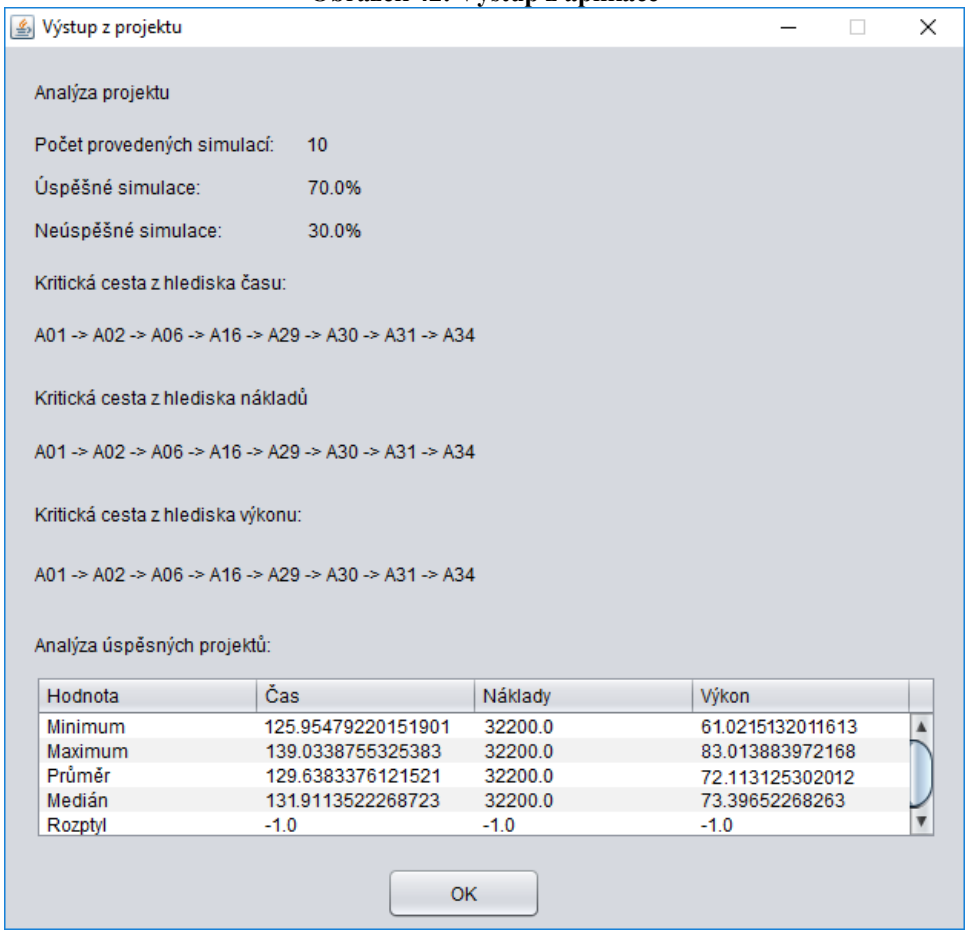
Obrázek 41: Nastavení parametrů



Zdroj: Vlastní zpracování

Potvrzením zadaných údajů se uživatelovi preference uloží do aplikace. Dalším krokem je už pouze spuštění samotného projektu a získání klíčového výstupu (obrázek č. 42), který je zhodnocen v následující podkapitole.

Obrázek 42: Výstup z aplikace



Zdroj: Vlastní zpracování

7.5 Analýza výsledků a ověření

Na základě výsledků uvedených v kapitole 7.4 můžeme nyní provést dvě analýzy, které nejprve zhodnotí dosažené výsledky a následně ověří správné fungování aplikace, když dojde na porovnání s dosažitelnými daty v tomto projektu. V prvním oddíle jsou analyzovány právě zmíněné výsledky, druhý oddíl se zabývá porovnáním získaných dat.

7.5.1 Analýza výsledků

Na základě výše uvedených výsledků je možné hodnotit rizikovost nebo očekávaný vývoj projektu při jeho spuštění v reálném světě. Dle tabulky je známo, že sedm z deseti projektů skončilo úspěšně, zatímco zbylých 30 % nikoliv. Kritická cesta z hlediska času, nákladů i výkonu se nejčastěji vyskytovala v marketingovém streamu, pravděpodobně z důvodu dlouhých příprav propagace.

Délka trvání tohoto projektu se nejčastěji pohybovala kolem 130 dní a celkové náklady vzrostly na 32.200 Kč. Generovaný výkon, který v tomto specifickém případě charakterizuje příjmy pořadatelů ze státních i soukromých sektorů, dosáhl průměrné výše mezi 70.000 až 75.000 Kč.

7.5.2 Ověření výsledků a fungování aplikace

Tato analýza se zabývá verifikací získaných dat na základě měření. V následujících třech odstavcích jsou shrnuty jednotlivé parametry metody VERT a jejich možná hodnota při výpočtu.

Většina údaje o spotřebovaném čase při pořádání tohoto projektu vychází ze stanovené konstanty, či stanovenou konstantu tvoří (pomocí matematického vztahu – ať už propagace v marketingovém oddělení nebo při vyjednávání o místě konání). To dává časovému hledisku nízkou variabilitu, ve které může nabývat hodnot v intervalu (123; 142). Veškeré hodnoty, které jsou zaznamenány ve výsledcích, náleží tomuto intervalu, tudíž z hlediska času neexistuje podezření na chybu.

Pozastavíme-li se čtenář nad náklady, brzy zjistí, že všechny činností jsou definovány pevnou konstantní hodnotou, což vede k jedinému řešení při realizaci projektu, v tomto případě 32.200, což je hodnota, kterou náklady vždy nabývají.

Posledním sledovaným parametrem je výkon, který se vyskytuje pouze ve čtyřech činnostech jako měrná jednotka příjmů. Výkon může nabývat hodnoty od 50 do 90, avšak málokdy dosáhne této hranice, vzhledem k vysoké variabilitě činností. Na základě zadání (filtrační výstupní logiky uzlu) je však projekt úspěšný pouze v případě, kdy celkový generovaný výkon nabude hodnoty minimálně 60. Pravděpodobně i z tohoto důvodu je minimální zaznamenaná hodnota výkonu 61.

7.6 Hodnocení aplikace

V této kapitole je sepsané kompletní hodnocení klíčového produktu této práce – finální podoby připravované aplikace. Nejprve je o aplikaci provedena obecná diskuze, která hodnotí její základní parametry – chybovost, přesnost, nedostatky, rychlost nebo intuitivnost aplikace. Následně jsou shrnuty vstupy a výstupy aplikace, přičemž jak vstupy, tak výstupy jsou popsány již více stroze, především z důvodu dřívějšího detailního popisu v této a předchozích kapitolách. V dalším oddíle jsou popsány výhody a nevýhody této aplikace a poslední oddíl poskytuje stručný výhled do budoucna ve spojitosti s vývojem tohoto programu.

7.6.1 Obecná diskuze

Po vyzkoušení aplikace na výše uvedeném příkladu můžeme přejít k jejímu hodnocení z nejrůznějších hledisek, které uživatel primárně vnímá. Ačkoliv je prostředí aplikace pro uživatele vcelku intuitivní, některé procesy (např. zadávání činnosti) mohou působit složitě. Velmi nepříjemný je vynaložený čas na zadávání úlohy, avšak to není chybou aplikace, protože projekty řešené metodou VERT vyžadují sběr velkého množství dat.
Co se týče editace stávajícího projektu, je ve vybraných případech lepší začít od začátku, neboť projekt se tak snadno může stát „křehkým“ a nekonzistentním. Velmi dobrá je na druhou stranu navigace v menu - ať už v hlavním menu či při editaci projektu, neboť aplikace neobsahuje matoucí a zbytečná tlačítka, ale vede uživatele přímo k cíli (k vyřešení projektu). Po spuštění je i větší projekt, např. výše uvedený, vyřešen při celkovém počtu deseti simulací za méně než 1 vteřinu.

Největším nedostatkem současně vyvíjené aplikace jsou její omezené funkce, které neumožňují zjištění kompletního potenciálu této metody. Dále můžeme u metody stále evidovat relativně vysokou chybovost při výpočtu (především u výstupních logik uzlů). Ta způsobuje, že rozsáhlejší problémy projektového řízení mohou za použití této aplikace řešit pouze uživatele znalí základního programování v jazyce JAVA, kteří si mohou požadované funkce přednastavit.

7.6.2 Vstupy do aplikace

Vstupy do aplikace jsou data, která jsou, jak již bylo několikrát zmíněno, zadávána v několika formách. Celý proces zadávání ze strany uživatele by bylo možné charakterizovat v několika krocích, které v jisté posloupnosti vedou ke kompletní charakteristice úlohy pomocí (na první pohled) různorodých a nezávislých hodnot.

Prvním krokem je zadání činností projektu. Ty mohou být zadány několika způsoby na základě rozhodnutí uživatele a možností metody VERT. Po vyhodnocení tohoto prvního kroku je v druhé fázi nezbytné pokračovat se zadáním uzlů projektu, které jsou automaticky generovány. Uzlům je přiřazeno několik různých typů vstupní a výstupní logiky, které jsou pro uživatele v aplikaci předdefinovány. Každý typ uzlu má svůj specifický význam, na základě kterého probíhá jeho realizace.

Kromě výše uvedených možností jsou součástí aplikace ale i další preference, jako například význam jednotlivých parametrů v daném projektu nebo počet iterací daného simulačního procesu.

7.6.3 Výstupy z aplikace

Klíčovým výstupem z aplikace je jednoduché dialogové okno, ve kterém jsou uváděny informace, jež charakterizují finální výpočet a získané hodnoty z projektu. Ve výstupu jsou uvedeny tři kritické cesty – kritická cesta z hlediska času, nákladů a výkonu. Další součástí je podíl úspěšných a neúspěšných projektů, které jsou vyjádřeny v procentech.

Pod tímto textem se nachází tabulka pod názvem „*Analýza úspěšných simulací*“, která sumarizuje statistický výstup. Ve čtyřech sloupcích jsou postupně uvedeny hodnoty pro jednotlivé parametry – čas, náklady a výkon a v posledním sloupci jsou hodnoty pro kombinaci těchto faktorů. V řádcích se nachází postupně optimální hodnoty projektu,

pesimální hodnoty, průměrné hodnoty a medián. V posledním řádku je uveden rozptyl naměřených hodnot. Je vhodné zmínit, že tato čísla vycházejí pouze z úspěšně dokončených projektů, neboť z hlediska logiky by zahrnutí nedokončených projektů výrazně znehodnocovalo hodnoty a mystifikovalo uživatele o dosažených výsledcích.
Na závěr popsaných výstupů z aplikace je vhodné upozornit ještě na jednu nezbytnou zajímavost, kterou si většina nezkušených uživatelů neuvědomí. V důsledku toho, že se jedná o simulační metodu (vyhodnocenou podle statistických přístupů) může být každé řešení této úlohy zcela odlišné.

7.6.4 Výhody a nevýhody aplikace

V tomto oddíle jsou specifikovány hlavní klady a zápory celé aplikace, která je k dispozici v uváděné formě. V prvním odstavci jsou shrnuty výhody aplikace, v druhém její nevýhody. V posledním odstavci dochází ke konfrontaci těchto pohledů.
Na prvním místě je důležité zmínit, že se jedná o první dostupný počítačový program v českém jazyce, který je schopen řešit projekty pomocí této simulační úlohy projektového řízení. Jiné metody jsou buď nedostupné, nebo neaktuální pro současné potřeby. Další velkou předností tohoto programu je jeho jednoduchost. Aplikace nemá velké množství prázdných míst a uživatel se s tímto software seznámí velmi rychle a následně je velmi brzy schopný řešit sofistikované projekty pomocí metody VERT. Další značnou výhodou je velikost a „rozsah“ aplikace, neboť na rozdíl od většiny používaných programů pro řízení projektů, které disponují řadou obtížných funkcí a přístupů, je tato aplikace malá, a tedy i relativně rychlá, protože má pouze dvě základní výpočetní funkce. Dalším pozitivem na dané aplikaci je i snadnost aktualizace, neboť je program napsán vskutku obecně a navíc v jednoduchém jazyce JAVA – s aktualizací si tak poradí takřka každý zkušený programátor. V poslední řadě stojí za zmínku dostupnost, která je (jak je již uvedeno výše) neomezená, neboť software je autorem distribuován jako freeware.
Proti tomu je primární nevýhodou jistá křehkost programu, která pramení z velkého množství operací, jež má program vykonávat, a to v přesně stanoveném pořadí. V důsledku toho vznikají v aplikaci menší chyby, které mohou nezkušeného uživatele mást. Další menší nepříjemností, která stojí za zmínku, je jistá zmatečnost a složitost v ovládání aplikace (např. dvojklik pro označení činnosti). Poslední nevýhodou je pak především fakt,

že aplikace je stavěna pro použití pouze v akademické sféře a nedisponuje tak prostředím pro řízení komplexnějších projektů v komerční sféře.

Klady po všech stránkách převyšují negativní aspekty, které v souvislosti s touto aplikací vznikly. Určitě by ze strany autora nebylo správné tvrdit, že aplikace je v současné podobě v nejlepším možném provedení. Na druhou stranu ale program disponuje všemi potřebnými a požadovanými funkcemi a splňuje tak očekávání, která u programů podobného typu mohou vzniknout. Autor je přesvědčený, že aplikace je při nejmenším vhodným nástrojem pro vysvětlení použití simulací v projektech, což je svým způsobem dílčí cíl této práce.

7.6.5 Výhledy do budoucna

Aplikace je již v současné době velmi efektivní nástroj pro testování složitějších projektů, a to především v akademické sféře. Přesto se dá spekulovat o různých možnostech, jak by bylo možné aplikaci upgradovat, aby byla vhodnější a aby se případně do budoucna stala vhodným nástrojem i v komerční sféře.

Prioritou v dalším vývoji je bezpochyby doplnění zbývajících matematických vztahů, které poskytnou aplikaci další rozměr v řešení rozsáhlých projektů. Velký potenciál se dá očekávat při zařazení logaritmicko-normálního rozdělení nebo rozdělení exponenciálního. Svůj význam však v aplikaci můžou mít i rozdělení diskrétní (Binomické, Poissonovo,…).

Na druhou stranu asi nemůžeme příliš očekávat využití dalších uzlů s filtrační výstupní logikou, a proto tyto postupy již nadále nebudou autorem připravovány. S podobnými typy uzlové logiky se v dnešních metodách již příliš nesetkáme, neboť nemají výraznou možnost využití.[105] Naposledy tomu tak bylo koncem 90. let.

V neposlední řadě bude složitým, ale velmi efektivním nástrojem případné doplnění dalšími matematickými vztahy. Především ve chvíli, kdy nebude vyžadováno vzájemně kombinovat vždy jednu konstantu a údaj jedné (předchozí) činnosti. Pak bude například generovaný výkon (získání finančních prostředků) možné použít jako vstupní údaj pro náklady alokované jednotlivým skupinám, které mohou mít mezi sebou ještě další stanovené závislosti.

[105] Podobný přístup se již využíval v metodě LOB, který sloužil jako metoda na bázi funkčního produkčního systému.

Veškeré úpravy a aktualizace této metody budou nadále veřejně dostupné, a to na dříve uvedených internetových stránkách *www.metodavert.xf.cz*. Tyto stránky by měly nadále fungovat jako hlavní zdroj informací a aktualit, které navazují na tuto práci.

Závěr

Tato práce je koncipována tak, aby poskytovala ucelený přehled o projektovém řízení se zaměřením na projektové metody, především pak metodu VERT. Čtenáři nabízí hloubkovou analýzu zmíněného matematického přístupu, který je bezpochyby metodou, jež přes veškeré jedinečné možnosti nedostala prostor na světovém trhu.

Pokud se pozastavíme nad tím, proč k tomu nikdy nedošlo, dojdeme k několika důvodům. Pravděpodobně je to způsobeno přechodem z tvrdých přístupů (řízení projektů) na měkké přístupy (projektové řízení) v období 80. let, kdy byla metoda po sestrojení software VERT3 na svém vrcholu. Dále můžeme spekulovat o možné obtížnosti zadávání. Přeci jen, sběr dat pro provedení efektivního výpočtu je extrémně složitý a tedy velmi nákladný, což odradilo pravděpodobně řadu firem od používání této metody. Na základě provedeného měření je možné tvrdit, že zadání identického projektu prostřednictvím metody VERT zabere více než desetinásobek času, než v případě metody CPM. Posledním důvodem, o kterém můžeme však pouze polemizovat, je přílišná orientace metody na specifické potřeby armády, pro kterou byla ve své původní podobě připravena.

Je tak těžké predikovat, jestli se v dnešní době situace obrátí a potřeba podobných metod pro projektové řízení, jako je například VERT, vzroste. Ačkoliv se jedná o výpočetně velmi náročné systémy, podávají uživateli velmi přesné výsledky a pomáhají při rozhodování, často i v rizikových situacích. Právě této možnosti bylo využito při plánování vojenských operací a výrobě zbraní ve Spojených Státech, kde byla metoda uplatněna pravděpodobně nejvíce ve své historii. Pomyslným *„světlem na konci tunelu“* pro využití této nebo jiné metody i v dnešním světě tak může být skeptický pohled na schopnosti dnešních projektových manažerů. Ti často bez zkušeností a potřebné kvalifikace zastávají významné pozice v nadnárodních korporacích.

Pro ověření možnosti využití metody VERT se autor práce rozhodl sestrojit jednoduchou aplikaci v programovacím jazyce JAVA. Ta svým rozsahem sice nepokrývá všechny možnosti, které původní verze VERT nabízela, nicméně disponuje v omezené míře takřka všemi původními funkcemi. K dispozici je celkem čtrnáct typů rozhodovacích uzlů, které jsou charakterizovány pomocí všech tří původních parametrů – času, nákladu i výkonu. Ty je možné zadat prostřednictvím konstanty, statistického rozdělení nebo matematickým vztahem, kde je hodnota zkoumané činnosti vyjádřena prostřednictvím jiné, libovolné předcházející činnosti. Při spuštění projektu probíhá simulace – projekt je tedy opakovaně

zkoušen a průběžné výsledky jsou ukládány. Výstupem je tabulka hodnot, které vypovídají o průběhu projektu ve statistických ukazatelích.

Je více než jasné, že v této podobě nemůže aplikace sloužit v komerční sféře jako klíčový nástroj pro řízení projektů. Je však dostačující pro použití v akademické sféře pro vysvětlení nebo lepší pochopení metody VERT. V této podobě může sloužit také jako zkušební nástroj, který lze v omezené míře implementovat do firmy pro analýzu dílčí části projektu. To je v současné době asi jediný způsob, jak zjistit potenciál této metody v obchodním světě.

Vzhledem ke svému stáří není vhodné tuto metodu posuzovat jako nepoužitelnou, ale spíše se zamýšlet nad tím, jak ji „aktualizovat“ nebo případně doplnit pro použití v současné době. Autor v tomto směru shledává velký potenciál například v parametrech, které jsou přiřazené činnostem, konkrétně pak především ve výkonu. Ten byl zaveden se záměrem kvantifikace výstupu ze splnění dané činnosti a nejčastěji byl následně převeden na finanční ukazatel. Dnes by bylo daleko užitečnější *kvantifikovat* tímto způsobem (třetím parametrem) například *riziko*, a to na základě jeho hodnoty (pravděpodobnosti a dopadu). Potom by metoda disponovala velmi užitečným podkladem pro provedení užitečnější analýzy rizika, než kterou metoda obsahuje nyní. Dalším užitečným rozvojem metody by mohlo být zavedení tzv. *multi-simulační výstupní logiky* zkoumaného uzlu. Ta by mohla rozhodnout o realizaci většího množství vystupujících činností na základě generování náhodných čísel. Jednou z dalších inovací by mohlo být i zavedení vhodnější a *sofistikovanější simulační metody*, než je (v dnešní době již překonané) Monte Carlo. V neposlední řadě by bylo dobré se zamyslet, především z důvodu zjednodušení zadávání, nad sloučením logiky činností a uzlů, které by v kombinované formě výrazně ulehčily proces zadávání dat a zjednodušily i pozdější editaci projektu.

Ačkoliv se to může zdát nemožné, autor práce si dokáže představit použití této metody (v původní a případně i inovované úpravě) v dnešním světě. Své uplatnění by tato metoda mohla najít při zefektivnění výrobních procesů, zavádění nových systémů ale třeba i v obchodně orientovaných společnostech. Autor však zastává názor, že klíčová by pro metodu byla implementace v logistice či distribuci. Tři parametry by zde ideálně fungovaly jako nástroj pro vyvažování jednotlivých variant, přičemž výše zmíněný třetí parametr (výkon) by mohl sloužit jako pomyslný žolík a zastávat v jednotlivých projektech různé funkce, často i v podobě např. binárních proměnných.

Každopádně je vhodné, že jsou tyto materiály zpracované a veřejně dostupné pro případné následující bádání. Zpracovaná aplikace i detailně vysvětlená metoda odkrývá spoustu možností v řízení projektů a zároveň slouží jako vhodný vstupní materiál pro další výzkumnou činnost. Bylo by příjemné, pokud by tato práce nezůstala ležet v archívech, ale dostala prostor k hlubšímu prostudování a verifikování uvedených závěrů.

Seznam použitých zdrojů

[1] BOLDY, Duncan. *Book review: Network Analysis for Management Decision: A Stochastic Approach.* European Journal of Operational Research. Fourth EURO IV special issue, 1982, Vol. 11, No. 3, (November, 1982), 303.

[2] CATES, G., R. Improving project management with simulation and completion distribution functions; Dostupné online z: http://etd.fcla.edu/CF/CFE0000209/Cates_Grant_R_200412_PhD.pdf

[3] DAWSON, C. W. a R. J. DAWSON. Clarification of Node representation in Generalized Activity Networks for Practical Project Management. International Journal of Project Management. Elsevier, 1994, Vol. 12, No. 2, 81-88.

[4] DLOUHÝ, Martin, Jan FÁBRY a Martina KUNCOVÁ. *Simulace pro ekonomy.* 2., upr. vyd. V Praze: Oeconomica, 2005. ISBN 80-245-0973-3.

[5] FÁBRY, Jan. *Matematické modelování.* Praha: Oeconomica, 2007. ISBN 978-80-245-1266-2.

[6] GRAHAM, Carl a Denis TALAY. *Stochastic simulation and Monte Carlo methods mathematical foundations of stochastic simulation.* Berlin: Springer, 2013. ISBN 978-36-423-9363-1.

[7] FIALA, Petr. *Projektové řízení: modely, metody, analýzy.* Praha: Professional Publishing, 2004. ISBN 80-86419-24-X.

[8] FIALA, Petr. *Řízení projektů.* Vyd. 2., přeprac. Praha: Oeconomica, 2008. ISBN 978-80-245-1413-0.

[9] HOUŠKA, Milan. *Simulační modely I.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2005. ISBN 80-213-1334-X.

[10] KERZNER, Harold. *Applied project management: best practices on implementation.* New York: Wiley, c2000. ISBN 0-471-36352-9.

[11] KIDD, John. *A Comparison Between the VERT Program and Other Methods of Project Duration Estimation.* Omega: The International Journal of Management Science. Pergamon Journals Ltd., 1987, Vol. 15, No. 2, 129-134.

[12] KIDD, John. *Project Analysis Today: The End Users´ Disquiet?* Omega: The International Journal of Management Science. Pergamon Press, 1989, Vol. 17, No. 2, 103-111.

[13] KOEGH, James. *Java bez předchozích znalostí: průvodce pro samouky.* Brno: Computer Press, 2005. ISBN 80-251-0839-2.

[14] LANGROVÁ, Pavlína a Tomáš ŠUBRT. *Projektové řízení II: softwarová podpora.* Vyd. 2. V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2010. ISBN 978-80-213-2075-8.

[15] LEE, Sang M., Gerald L. MOELLER a Lester A. DIGMAN. *Network Analysis for Management Decision: A Stochastic Approach.* Boston: Nijhoff Publishing, 1982. ISBN 978-94-009-8173-7.

[16] LOCK, Dennis. *Project management.* Vyd. 9. Burlington: Gower Publishing, c2007. ISBN 0-566-08772-3.

[17] LURIE, Philip M. A Handbook of Cost Risk Analysis Methods. Institute for Defense Analyses, 1993.

[18] MARGOT, Note. *Project management for information professionals.* Vyd. 1. Chandos Publishing, 2015. ISBN 978-0-08-100127-1.

[19] MARSH, Edward R. *The Harmonogram of Karol Adamiecki.* The Academy of Management Journal. Academy of Management, 1975, Vol. 18, No. 2, 358-364.

[20] MOELLER, Gerald L. a Lester A. DIGMAN. *Operations Planning with VERT.* Operations Research: INFORMS: Institute for Operations Research and the Management Science. Operations Research Society in America, 1981, Vol. 29, no. 4, Operations Management (Jul. - Aug., 1981), 676-697.

[21] MOELLER, Gerald L. *Venture Evaluation and Review Technique: Decision Models Directorate*, US Army Armament Material Readiness Command. Rock Island, Illinois, 1979.

[22] NELSON, Richar Grahan a AZARON, Amir. *The use of a GERT based method to model concurrent product development processes.* European Journal of Operational Research. Elsevier, 2016, Vol. 250, Issue 2, (April, 2016), 566-578.

[23] SHANNON, Robert E. *Systems simulation: the art and science*. Englewood Cliffs, N.J.: Prentice-Hall, 1975. ISBN 0138818398.

[24] SVOZILOVÁ, Alena. *Projektový management.* Praha: Grada, 2006. Expert (Grada). ISBN 80-247-1501-5.

[25] ŠUBRT, Tomáš a Pavlína LANGROVÁ. *Projektové řízení: (základy a matematické metody).* Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 80-213-1194-0.

[26] ŠUBRT, Tomáš a Jan BARTOŠKA. *Projektové řízení: (měkké a pokročilé přístupy).* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2007. ISBN 978-80-213-1725-3.

[27] TAVARES, L. Valadares. *Advanced Models for Project Management.* New York: Science & Business Media, 1999. ISBN 978-1-4613-4649-4.

[28] ZHOU, Li. *Forecasting return of used products for remanufacturing using Graphical Evaluation and Review Technique (GERT).* International Journal of Production Economics, 2016, Vol. 181, part B, (November, 2016), 315-324.

Seznam obrázků

Seznam tabulek

Seznam kódů

Seznam příloh

A. Matematické vztahy VERT

V kapitole 4 je zmíněno, že v metodě VERT je možné k určení parametrů činnosti využít matematického vztahu. Na následujících řádcích je uvedeno všech 50 (resp. 100) vztahů, pomocí kterých je možné činnost definovat. Libovolná proměnná X, Y, Z pak představuje číselnou hodnotu nebo parametr jiné činnosti.

Jednotlivé rovnosti je možné částečně rozlišit. Například, zatímco matematické vztahy 1-50 vracejí uživateli pouze celočíselné hodnoty, vztahy 51-100 mohou vrátit hodnotu R ve formě desetinného čísla. Dále, matematické vztahy 38-44 a 88-94 využívají tabulkového podkladu, ve kterém vyhledávají požadovanou hodnotu. Vztahy 45-50 a 95-100 vycházejí z předdefinovaných podprogramů, které jsou dopředu nastaveny uživatelem.

Číslo vztahu	Matematický vztah	Výsledek	Omezení
1, resp. 51	X*Y*Z	= R	
2, resp. 52	(X*Y)/Z	= R	$Z \neq 0$
3, resp. 53	X/(Y*Z)	= R	$Y*Z \neq 0$
4, resp. 54	1/(X*Y*Z)	= R	$(X*Y*Z) \neq 0$
5, resp. 55	X+Y+Z	= R	
6, resp. 56	X+Y–Z	= R	
7, resp. 57	X–Y–Z	= R	
8, resp. 58	–X–Y–Z	= R	
9, resp. 59	X*(Y+Z)	= R	
10, resp. 60	X*(Y–Z)	= R	
11, resp. 61	X/(Y+Z)	= R	
12, resp. 62	X/(Y–Z)	= R	
13, resp. 63	$X*(Y)^Z$	= R	$Y > 0$
14, resp. 64	$X*(LOG_E(LOG(Y*))$	= R	$Y*Z > 0$
15, resp. 65	$X*(LOG_{10}(Y*Z))$	= R	$Y*Z > 0$
16, resp. 66	X*(sin(Y*Z))	= R	
17, resp. 67	X*(cos(Y*Z))	= R	
18, resp. 68	X*(actg(Y*Z))	= R	
19, resp. 69	$X_{19 \text{ nebo } 69} \geq$		
	$Y_{19 \text{ nebo } 69} \geq X_{20 \text{ nebo } 70}$	$= R_{20 \text{ nebo } 70}$	

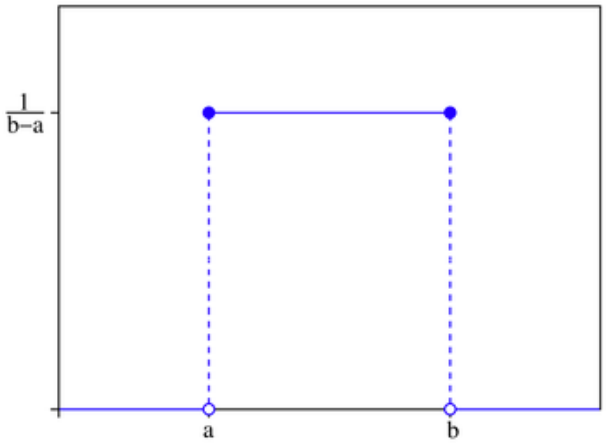
Matematický vztah 19 nebo 69 musí být následován operací 20 nebo 70

20, resp. 70	$X_{19 \text{ nebo } 69} \leq$		
	$Y_{19 \text{ nebo } 69} \leq Y_{20 \text{ nebo } 70}$	$= R_{20 \text{ nebo } 70}$	
21, resp. 71	$X \geq Y; Z$	= R	

	$X \leq Y; X$	= R	
22, resp. 72	$X \geq Y; X$	= R	
	$X \leq Y; Z$	= R	
23, resp. 73	(X*Y)+Z	= R	
24, resp. 74	(X*Y)–Z	= R	
25, resp. 75	(X/Y)+Z	= R	$Y \neq 0$
26, resp. 76	(X/Y)–Z	= R	$Y \neq 0$
27, resp. 77	(X+Y)*Z	= R	
28, resp. 78	(X+Y)/Z	= R	$Z \neq 0$
29, resp. 79	(X–Y)*Z	= R	
30, resp. 80	(X–Y)/Z	= R	$Z \neq 0$
31, resp. 81	X+(Y*Z)	= R	
32, resp. 82	X–(Y*Z)	= R	
33, resp. 83	X+(Y/Z)	= R	$Z \neq 0$
34, resp. 84	X–(Y/Z)	= R	$Z \neq 0$
35, resp. 85	–X–Y+Z	= R	
36, resp. 86	–X+Y+Z	= R	
37, resp. 87	X/Y/Z	= R	$Y \neq 0; Z \neq 0$
38, resp. 88	X, Y, Z f(tab1)	= R	$X, Y, Z \subset$ tab1
39, resp. 89	X, Y, Z f(tab2)	= R	$X, Y, Z \subset$ tab2
40, resp. 90	X, Y, Z f(tab3)	= R	$X, Y, Z \subset$ tab3
41, resp. 91	X, Y, Z f(tab4)	= R	$X, Y, Z \subset$ tab4
42, resp. 92	X, Y, Z f(tab5)	= R	$X, Y, Z \subset$ tab5
43, resp. 93	X, Y, Z f(tab6)	= R	$X, Y, Z \subset$ tab6
44, resp. 94	X, Y, Z f(tab7)	= R	$X, Y, Z \subset$ tab7
45, resp. 95	X, Y, Z f(sub1)	= R	
46, resp. 96	X, Y, Z f(sub2)	= R	
47, resp. 97	X, Y, Z f(sub3)	= R	
48, resp. 98	X, Y, Z f(sub4)	= R	
49, resp. 99	X, Y, Z f(sub5)	= R	
50, resp. 100	X, Y, Z f(sub6)	= R	

B. Statistická rozdělení použitelná v aplikaci

Rovnoměrné rozdělení

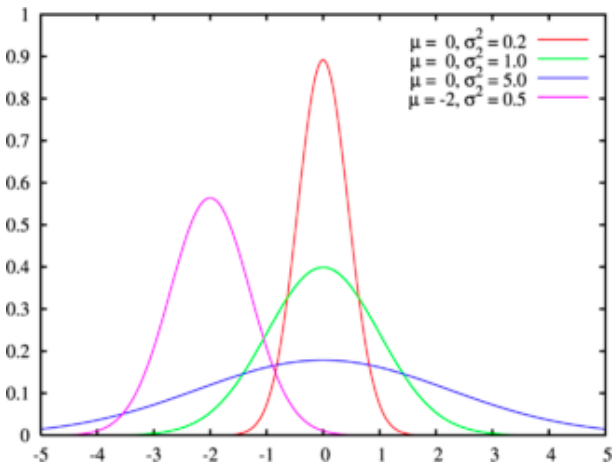


Rovnoměrné rozdělení pravděpodobnosti přiřazuje všem hodnotám náhodné veličiny stejnou pravděpodobnost. Rovnoměrné rozdělení na intervalu **(a, b)**, kde $-\infty < \mathbf{a} < \mathbf{b} < \infty$, má ve všech bodech daného intervalu konstantní hustotu pravděpodobnosti. Mimo tento daný interval je tedy hustota pravděpodobnosti nulová.

Vstupní údaje, které je v aplikaci nezbytné zadat pro definici intervalu, jsou hodnoty:

- **MIN** (minimální hodnota ≈ **a**)
- **MAX** (maximální hodnota ≈ **b**)

Gaussovo normální rozdělení



Normální (nebo Gaussovo) rozdělení pravděpodobnosti je jedno z nejdůležitějších rozdělení pravděpodobnosti spojité náhodné veličiny. Tímto rozdělením pravděpodobnosti se sice přesně řídí jen málo náhodných veličin, ale jeho význam spočívá v tom, že za určitých podmínek dobře aproximuje řadu jiných pravděpodobnostních rozdělení. Normální rozdělení pravděpodobnosti s parametry $\boldsymbol{\mu}$ a $\boldsymbol{\sigma^2}$, pro $-\infty < \boldsymbol{\mu} < \infty$ a $\mathbf{0} < \boldsymbol{\sigma^2}$, je pro $-\infty < \mathbf{x} < \infty$ definováno hustotou pravděpodobnosti ve tvaru Gaussovy funkce

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Vstupní údaje, které je v aplikaci nezbytné zadat, jsou hodnoty:

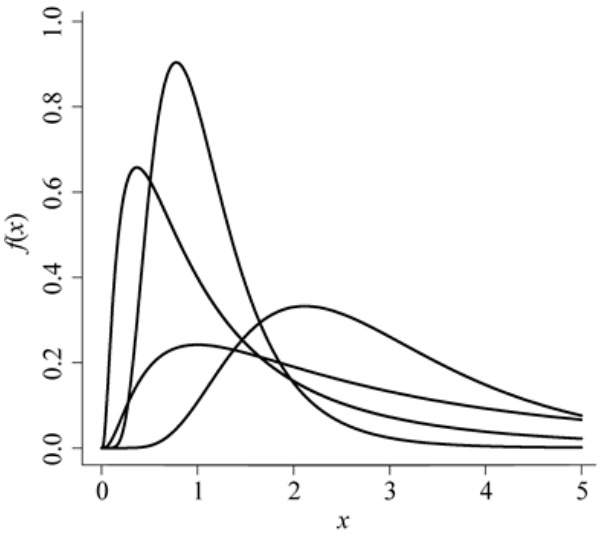
- **MIN** (minimální hodnota ≈ **a**)
- **MAX** (maximální hodnota ≈ **b**)
- **SMĚRODATNÁ ODCHYLKA** (σ)

Nezadává se **střední hodnota** (μ), která se automaticky dopočítává jako průměr minimální hodnoty **a** a maximální hodnoty **b**. Pokud je následně generovaná hodnota mimo stanovený interval **(a, b)**, generování se opakuje. Tímto způsobem je interval „ořezán“, čímž je zajištěno mimo jiné to, aby byly generované hodnoty vždy kladné.

Logaritmicko-normální rozdělení

Logaritmicko-normální rozdělení (také log-normální rozdělení) s parametry **μ** a **σ**, označované **LN (μ, σ)**, je spojité rozdělení pravděpodobnosti jednorozměrné reálné náhodné veličiny **X** takové, že náhodná veličina **ln (X)** má normální rozdělení se střední hodnotou **μ** a směrodatnou odchylkou **σ**.

$$fx(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}} \quad ,x > 0$$



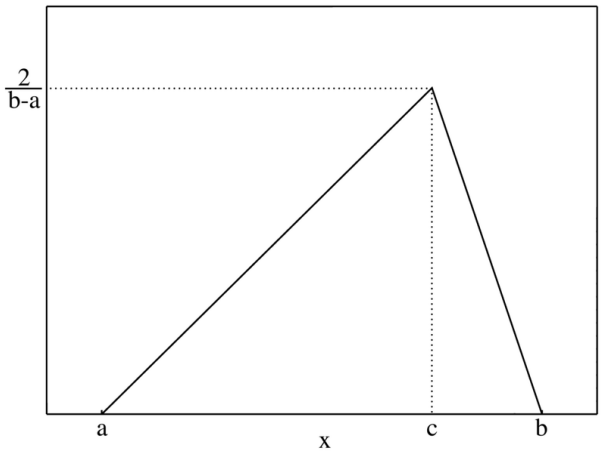
Vstupní údaje, které je v aplikaci nezbytné zadat, jsou hodnoty:

- **STŘEDNÍ HODNOTA** (μ)
- **SMĚRODATNÁ ODCHYLKA** (σ)

V této aplikaci není možné zadat logaritmicko-normální rozdělení jako **tříparametrické**.

Trojúhelníkové rozdělení

Trojúhelníkové rozdělení je v teorii pravděpodobnosti a matematické statistice spojité rozdělení pravděpodobnosti. Zadává se jednoduše podle symbolického zápisu **X ~ Tri (a, b, c)**. Spojitá náhodná veličina X má trojúhelníkové rozdělení s parametry **a, b, c**, kde **a < b** a **a ≤ c ≤ b**.



Vstupní údaje, které je v aplikaci nezbytné zadat, jsou hodnoty:

- **MIN** (Minimální hodnota ≈ **a**)

- **MAX** (Maximální hodnota ≈ **b**)
- **MOD** (Modální hodnota ≈ **m**)

Beta (PERT) rozdělení

Na rozdíl od výše uvedených statistických rozdělení vychází toto rozdělení ze tří odhadů, stejně jako je tomu u metody PERT. Ke generování hodnoty tak není potřebné použití náhodného čísla, jako u výše uvedených rozdělení.

Vstupní údaje, které je v aplikaci nezbytné zadat, jsou hodnoty:

- **OPT** (Optimistický odhad ≈ **a**)
- **PES** (Pesimistický odhad ≈ **b**)
- **MOD** (Modální odhad ≈ **m**)

Na základě těchto hodnot je vypočítán odhad délky trvání, který je možné vyjádřit následujícím vzorcem:

$$\mu(t_{ij}) = t_{ij}^{e} = \frac{a_{ij} + 4m_{ij} + b_{ij}}{6}$$

C. Představení autorů metody VERT

Gerald L. Moeller

Gerald L. Moeller (*1939) je vědecký pracovníkem, který od konce 60. let pracoval pro statní složky Spojených Států amerických. Pochází ze *Sheridanu* ve státe *Iowa*, dnes žije ve městě *Bettendorf*, též v *Iowa*. Během svého života se podílel na několika výzkumech projektech pod záštitou USA, například v Arizoně nebo Coloradu.

Lester A. Digmann

Lester A. Digmann (*1940) je bývalý univerzitní profesor na *College of Business Administration, University od Nebraska – Lincoln* kde zároveň působí jako garant graduálních a postgraduálních studijních programů. Zabývá se primárně strategickým managementem, rozhodováním za nejistoty, mezinárodním managementem a technologickým managementem. Během své kariéry publikoval v několika desítkách uznávaných časopisů, např. *Harvard Business Review, Organizational Dynamics* nebo *Operational Research*.

Sang M. Lee

Sang M. Lee (*1939) je stejně jako L. A. Digmann významný univerzitní profesor na *College of Business Administration, University od Nebraska – Lincoln* a výkonný ředitel podnikatelského centra v Nebrasce. Dr. Lee je mezinárodně uznávaným odborníkem v oblasti teorie rozhodování, řízení produktivity a mezinárodního obchodu. Dodnes získal řadu ocenění včetně *Valley Forge Freedoms Foundation Leavery* za mimořádnou výzkumnou činnost a několik cen na *University of Nebraska*.

D. Porovnání a hodnocení síťových řídících přístupů

V tabulce pod textem je zachyceno souhrnné porovnání pěti základních síťových technik, které jsou hodnoceny podle devíti klíčových kritérií, významných pro manažery.

KRITÉRIA	PERT/CPM	PERT/COST	LOB	GERT	VET-3
Vhodné typy projektů	Návrh & Vývoj	Návrh & Vývoj	Produkce	Koncepční, Návrh & Vývoj, Produkční	Koncepční, Návrh & Vývoj, Produkční, Operační
Sledované parametry	Čas	Čas, náklady (omezeně)	Čas	Čas, náklady (omezeně), riziko	Čas, náklady, výkon, riziko
Náročnost přípravy	Lehce obtížná	Značně obtížná	Jednoduchá	Středně až značně obtížná	Značně obtížná
Databáze vstupních dat	Střední	Velká	Malá	Střední až velká	Velká
Náklady na provoz systému	Střední	Vysoké	Velmi nízké	Střední až vysoké	Vysoké
Obsáhlost	Pouze jednorázové činnosti	Pouze jednorázové činnosti	Pouze opakující se činnosti	Jednoduché i opakující se	Jednoduché i opakující se
Flexibilita	Deterministická	Deterministická	Deterministická	Stochastická	Stochastická
Náročnost inovace	Lehce obtížná	Obtížná	Snadná	Velmi obtížné	Velmi obtížné
Výstup z výpočtu	Velmi dobrý	Výborný	Velmi dobrý	Vynikající	Vynikající

E. Porovnání a hodnocení hlavních přístupů a metod projektového řízení

V tabulce pod textem je zachycen tentokrát detailnější pohled na danou problematiku.

KRITÉRIA	PERT/CPM	PERT/COST	LOB	GERT	VERT-3
Vhodné typy projektů	Primárně se soustředí na aplikaci v jednorázových projektech, dále také v návrhu & vývoji	Stejné jako v případě PERT/CPM	Pouze produkční	Nejužitečnější v koncepční fázi, ale univerzálně použitelné i v jiných fázích, ve stejném rozsahu	Stejné jako v případě GERT
Sledované parametry	Časově orientované; zachází s náklady a s výkonem jako se svými cíli nebo omezujícími podmínkami	Přidává plánování nákladů a řídící funkce k PERT/CPM	Čas orientované (cílem rozvrhování & řízení množství)	Časově orientované s přidanou funkcí nákladů; analyzuje čas a hodnotu rizika	Plně se soustředí na čas, náklad a různé podoby výkonu; provádí analýzu rizika
Náročnost přípravy	Důležitý korektní odhad sítě a délky trvání činností	Odhad nákladů pomocí pracovního balíčku nebo činnosti; součást metody, která by měla být vždy provedena	Hlavním požadavkem je sestavení grafu, který zachycuje produkci a časy cyklů	Sestavení projektové sítě vyžaduje specifické znalosti; Větší množství funkcí se projevuje na komplexnosti	Podobné jako v případě GERT, vhodné je použití dalších volitelných funkcí
Databáze vstupních dat	Vzniká jako vedlejší produkt při přípravě, vychází z kvality dat	Sledování a řízení nákladů vyžaduje větší rozsah vstupních dat	Vyžaduje pouze délky trvání a hodnoty množství ve stanovených řídících bodech	Podle počtu vybraných funkcí metody, avšak s rostoucí tendencí při plném využití	Stejné jako v případě GERT
Náklady na provoz systému	Snadno ovladatelné, největším nákladem je příprava a aktualizace dat	Rozšiřuje původní PERT, ale největší náklady jsou stále na přípravu dat	Prakticky neexistují	Vyšší náklady na vstupní data a na několik běhů simulačního procesu	Stejné jako v případě GERT, použití dalších nástrojů zvyšuje náklady na provoz
Obsáhlost	Omezeno časovým parametrem a nemožností opakovat činnosti	Omezeno časovým a nákladovým parametrem s nemožností opakovat činnosti	Omezeno použitím pouze na opakující se činnosti	Pracuje s většinou typů činností	Pracuje s většinou typů činností, přičemž umožňuje použití různých způsobů jejich zadávání

KRITÉRIA	PERT/CPM	PERT/COST	LOB	GERT	VERT-3
Flexibilita	Řeší pouze deterministické situace, nemá žádnou schopnost přizpůsobit se rozhodovacím alternativám nebo náhodným událostem	Řeší pouze deterministické situace, stejné jako v případě PERT/CPM	Řeší pouze deterministické situace, dále vyžaduje určitý cyklus; Zkušenostní křivka vyjadřuje problémy	Řeší deterministické i stochastické činnosti	Výrazně více možností vstupních a výstupních funkcí než v případě GERT; Má schopnost sestavit rozpočet
Náročnost inovace	Relativně jednoduchá, avšak vyžaduje úsudek uživatele pro přehodnocení nadcházející činnosti; Délky trvání představují menší problémy	Teoreticky snadná, avšak hlavní snaha v ohodnocení *„skutečných"* nákladů a konstantních změn jsou problematické	Vyžaduje pouze konkrétní počet kumulativní produkce	Obtížné, je snazší vytvořit nový přístup	Stejné jako v případě GERT
Výstup z výpočtu	Upozorňuje na kritické činnosti v problémových oblastech, což je silnou stránkou, předpovídá finální stav	Přidává k PERT/CPM možnost sledovat alokované náklady na zdroje	Upozorňuje na potenciálně vznikající problémy v pravidelně plánovaných dodávkách	Analýza rizika soustředí pozornost na to, co může nastat a vyjadřuje pravděpodobnost tohoto nežádoucího stavu	Analyzuje a upozorňuje na výstupy z hlediska času, nákladů a výkonu; Přístup může být použit i na plánování strategie mimo řízení projektů

F. Statický text v aplikaci „O projektu“

Tato aplikace byla vytvořena jako součást diplomové práce pod názvem „*Vývoj aplikace pro řešení metod PERT a VERT*“ na Provozně-ekonomické fakultě, ČZU v Praze.

Jedná se o metodickou aplikaci, která slouží jako nástroj pro řešení jednoduchých projektů a to za pomoci projektových metod PERT (Program Evalution and Review Technique) a VERT (Venture Evalution and Review Technique).

Více informací na www.metodavert.xf.cz.

Autor práce: Daniel Chamrada (*daniel.chamrada@gmail.com*)
Konzultant: Pavel Malý

G. Vybrané kódy z programování – Výpočet kritické cesty metody PERT

```
public class CriticalPathEvaluator {

    private Vertex start;
    private Vertex end;
    private Vertex[] vertices;

    private Set visited;
    private HashMap<Vertex, Vertex> precedessors;
    private HashMap<Vertex, Float> weights;

    public CriticalPathEvaluator(Vertex start, Vertex end, Vertex[]
vertices) {
       this.start = start;
       this.end = end;
       this.vertices = vertices;
    }

    private Vertex findMaximuWeight() {

        float maximum = -1;
        Vertex result = null;

        for(Vertex item : this.vertices) {
           if (!this.visited.contains(item)) {

               if (this.weights.get(item) > maximum ) {
                  result = item;
                  maximum = this.weights.get(item);
               }
           }
        }

        return result;
    }

    public java.util.ArrayList<Vertex> evaluate() {

         this.visited = new HashSet<Vertex>();
         this.precedessors = new HashMap<Vertex, Vertex>();
         this.weights = new HashMap<Vertex, Float>();

         precedessors.put(this.start, null);
         weights.put(this.end, Float.parseFloat("0"));

        for(Vertex vertex : this.vertices) {
            this.weights.put(vertex, Float.parseFloat("0"));
        }

        Vertex maximum = this.findMaximuWeight();
        while(maximum != null) {
            visited.add(maximum);
            for(Vertex neighbour : maximum.neighbours) {
```

```
                if(weights.get(maximum) + neighbour.duration >
weights.get(neighbour)){
                    weights.put(neighbour, weights.get(maximum) +
neighbour.duration);
                    precedessors.put(neighbour, maximum);
                }
            }

            maximum = this.findMaximuWeight();
        }


        System.out.println("Graph created");

        Vertex tmp = this.end;
        java.util.List<Vertex> verticesList =
java.util.Arrays.asList(vertices);

        java.util.ArrayList<Vertex> criticalPath = new
java.util.ArrayList<Vertex>();

        while(tmp!=null) {
            criticalPath.add(tmp);

            tmp = precedessors.get(tmp);
        }

       java.util.Collections.reverse(criticalPath);

       return criticalPath;
    }

}
```

H. Vybrané kódy z programování – Kalkulace výstupu metody PERT

```
private String createSummary(JTable currentTable, ArrayList
criticalPath) {

        String res = "";
        double duration = 0.0;
        double sumDispersion = 0.0;
        double stdDeviation = 0.0;
        int rowIndex = 0;

        for (Object obj : criticalPath) {
            Vertex item = (Vertex) obj;

            rowIndex =
ComponentHelper.getInstance().getIndexRowByColumn(currentTable,
Constants.COLUMN_Activity, item.name);

            duration = duration +
Double.parseDouble(currentTable.getValueAt(rowIndex,
Constants.COLUMN_Weight).toString());
            sumDispersion = sumDispersion +
Double.parseDouble(currentTable.getValueAt(rowIndex,
Constants.COLUMN_Dispersion).toString());
            stdDeviation = Math.sqrt(sumDispersion);
        }

        res = res + "Kritická cesta je: " +
Graph.ArrayListToString(criticalPath) + "\n";
        res = res + "Součet odhadů je: " + duration + "\n";
        res = res + "Součet rozptylů je: " + sumDispersion + "\n";
        res = res + "Směrodatná odchylka je: " + stdDeviation;

        return res;
    }
```

I. Vybrané kódy z programování – Editace činností

Přidání činnosti

```
public void addNewRow(JTable currentTable, int mode) {

        Object[] rowData = new
Object[currentTable.getColumnCount()];
        int i;
        for (i=0; i<=currentTable.getColumnCount()-1; i++) {
            rowData[i] = "";
        }
    rowData[Constants.COLUMN_Activity] =
SimplePertIDCreator.getInstance().getNewID();

        DefaultTableModel model =
ComponentHelper.getInstance().getTableModel(currentTable);
        model.addRow(rowData);

    }
```

Odebrání činnosti

```
public void deleteSelectedRow(JTable currentTable) {

        DefaultTableModel model =
ComponentHelper.getInstance().getTableModel(currentTable);
        int searchRowIndex =  currentTable.getSelectedRow();

        if (searchRowIndex!=-1) {

ComponentHelper.getInstance().deleteApropriatePredecessor(currentTab
le, searchRowIndex);

           String ID = currentTable.getValueAt(searchRowIndex,
Constants.COLUMN_Activity).toString();
           SimplePertIDCreator.getInstance().removeID(ID);

           model.removeRow(searchRowIndex);
        }
    }
```

Posun činnost nahoru (zpět)

```
public void moveUp(JTable currentTable){
        DefaultTableModel model = this.getTableModel(currentTable);
        int[] rows = currentTable.getSelectedRows();

        if (rows.length > 0) {
            model.moveRow(rows[0],rows[rows.length-1],rows[0]-1);
            currentTable.setRowSelectionInterval(rows[0]-1,
rows[rows.length-1]-1);
        }
    }
```

Posun činnosti dolu (vpřed)

```
public void moveDown(JTable currentTable){
        DefaultTableModel model = this.getTableModel(currentTable);
        int[] rows = currentTable.getSelectedRows();

        if (rows.length > 0) {
            model.moveRow(rows[0],rows[rows.length-1],rows[0]+1);
            currentTable.setRowSelectionInterval(rows[0]+1,
rows[rows.length-1]+1);
        }
    }
```

J. Návod na zadání projektu VERT ze strany uživatele

Proces zadávání projektu VERT ze strany uživatele

1. Pro korektní zadání, bezchybnost výpočtu a především dostatečný přehled o projektu je vhodné si daný projekt nejprve nanečisto specifikovat a „*předkreslit*“ na papír, zvláště pokud uživatel nemá dostatečné znalosti v oblasti řízení projektů.
2. Uživatel vybere v hlavním menu možnost „VERT“ (*Venture Evaluation and Review Technique*)
3. Při výběru se otevře dialogové okno, které se uživatel dotáže na významnost zkoumaných parametrů. Na tomto místě uživatel ohodnotí alespoň jeden z uvedených parametrů (čas, náklady a výkon) hodnotou větší než 0, čímž zadá její význam. Hodnotu je možné zadávat pouze celočíselně.
    a. Příklad 1 – *Čas 60 %, Náklady 30 %, Výkon 10 %*
    b. Příklad 2 – *Čas 50 %, Výkon 50 %*
    c. Příklad 3 – *Náklady 100 %*
4. Po nastavení hodnot uživatel potvrdí svou volbu v okně prostřednictvím stisknutí „OK“ a je přesměrován do hlavního uživatelského rozhraní.
5. Nyní postupně vkládá činnosti daného projektu. Každá nová činnost je generována s unikátním kódem „*Axx*“[106], přičemž uživatel doplňuje jméno činnosti, jejího předchůdce a následně hodnoty předem vybraných parametrů (čas, náklady, výkon). Hodnoty parametrů se zadávají pomocí postranního panelu. Tímto způsobem přidává uživatel činnost za činností.
6. Výše uvedené zadávání parametru činnosti probíhá v postranním panelu a každý parametr je zvlášť ohodnocený jednou ze tří možností:
    a. **Konstantou** – celým číslem, dle rozhodnutí uživatele
    b. **Statistickým rozdělením** – Na výběr je několik možností
        i. *Normální* – pomocí tří parametrů – rozptyl, min, max
        ii. *Rovnoměrné* – pomocí dvou parametrů – min, max
        iii. *Beta (viz. PERT)* – pomocí tří parametrů – min, max, modální
    c. **Matematickým vztahem** – V tomto případě má uživatel na výběr celkem šest různých možností, kde „*X*“ je vždy uživatelem stanovená konstanta (ta může být celočíselná nebo desetinná) a „*Y*“ je vždy parametr, který je výslednou

[106] Kde „xx“ je pořadové číslo vytvořené činnosti.

hodnotou jiné (předcházející) činnosti. Matematickým vztahem nemůže být nikdy definována činnost, která vychází z uzlu s iniciativní vstupní logikou.

   i. $X + Y$
   ii. $X - Y$
   iii. $Y - X$
   iv. $X \cdot Y$
   v. X / Y
   vi. Y / X

7. Během tohoto zadávání může uživatel kdykoliv přehodnotit své priority z kroku č. 3 a pomocí menu „**Akce → Význam parametrů**“ může přenastavit význam těchto jednotlivých parametrů.
8. Ve chvíli, kdy má uživatel zadané všechny činnosti, vybere možnost v horním menu „**Akce → Vygenerovat uzly**“. V této fázi je upozorněn, že tímto krokem uzamkne horní tabulku činností pro jakoukoli další práci. Pokud se přesto bude chtít vrátit do horní tabulky, dvakrát do ní poklepe myší, přičemž se aplikace dotáže na smazání vygenerovaných uzlů a návrat k práci s činnostmi. Uživatel volí mezi možnostmi ANO/NE.
9. Vygenerované uzly se zobrazí v tabulce o šesti sloupcích, přičemž první tři sloupce si vygenerují vlastní hodnoty - postupně „**Kód uzlu**“ (podobně jako činnosti ve formě N01, N02,…), „**Vstupující činnosti**“ (označené jako A01, A02,…), „**Vystupující činnosti**“ (označené stejným způsobem). Další sloupce tabulky jsou „**Vstupní logika**“, „**Výstupní logika**“ a „**Dodatečné informace**“. Pokud uzel nemá vstupující činnost, pak je vstupní logika vždy „*Iniciativní*“ – což se propíše u těchto uzlů do sloupce „Vstupní logika“ bez možnosti změny uživatelem. Pokud uzel nemá vystupující činnost, pak je výstupní logika vždy „*Finalizační*“ – což se propíše u těchto uzlů do sloupce „Výstupní logika“ bez možnosti změny uživatelem. Další uzly nemůžeme nikdy označit za „*Iniciativní*“ nebo „*Finalizační*“, protože mají vstupující, resp. vystupující činnost. Ve všech ostatních případech má uživatel možnost výběru z několika možností. Sloupec „**Dodatečné informace**“ mají svůj vlastní, specifický význam, který je zmíněn v bodě číslo 13.
10. Uživatel zadává uzly tak, že z uvedených možností vybere pro daný uzel specifickou vstupní a výstupní logiku. Iniciativní a Finalizační logika je generována pouze automaticky.
11. Možné typy vstupní logiky jsou:
    a. *Iniciativní*,

b. *Konjunktivní*,
c. *Disjunktivní*,
d. *Inkluzivní*

12. Možné typy výstupní logiky jsou
    a. *Finalizační*,
    b. *Deterministická*,
    c. *Simulační*,
    d. *Filtrační*
13. K *Iniciativní* vstupní logice nikdy nesmí být napojena *Finalizační* nebo *Filtrační* výstupní logika. Sloupec „**Dodatečné informace**“ je přednastaven pro další údaje k výpočtu *Simulační*, *Filtrační* nebo *Finalizační* výstupní logiky.
    a. *Simulační* – doplňuje se podmíněná pravděpodobnost realizace vystupujících činností. Jejich součet je roven 100 %. O realizaci jediné vybrané činnosti rozhodne náhodné číslo.
    b. *Filtrační* – doplňuje se omezení průtoku dle jednotlivých parametrů ve formě MAX a MIN. Vystupující činnosti jsou realizovány na základě hodnot jejich parametrů a těchto přednastavených omezení.
    c. *Finalizační* – doplňuje se informace o úspěšném, případně neúspěšném dokončení projektu. Činnosti bez následníka vždy dosahují jedné z těchto možností.
14. Po definování vygenerovaných uzlů uživatel vybere v horním menu „**Akce → Spustit projekt**“. V dialogovém okně je dotázán, kolikrát má být projekt simulován. Tady má na výběr z několika možností od 10 do 100 simulací, alternativně lze vyplnit vlastní hodnotou v intervalu <1; 100>
15. Uživatel získá výstup v dialogovém okně.

K. Specifikace zadávání dodatečných informací při výpočtu VERT

Tento text obsahuje základní informace, jak stanovit pomocí aplikace dodatečné informace u výpočtu uzlu s filtrační nebo simulační výstupní logikou.

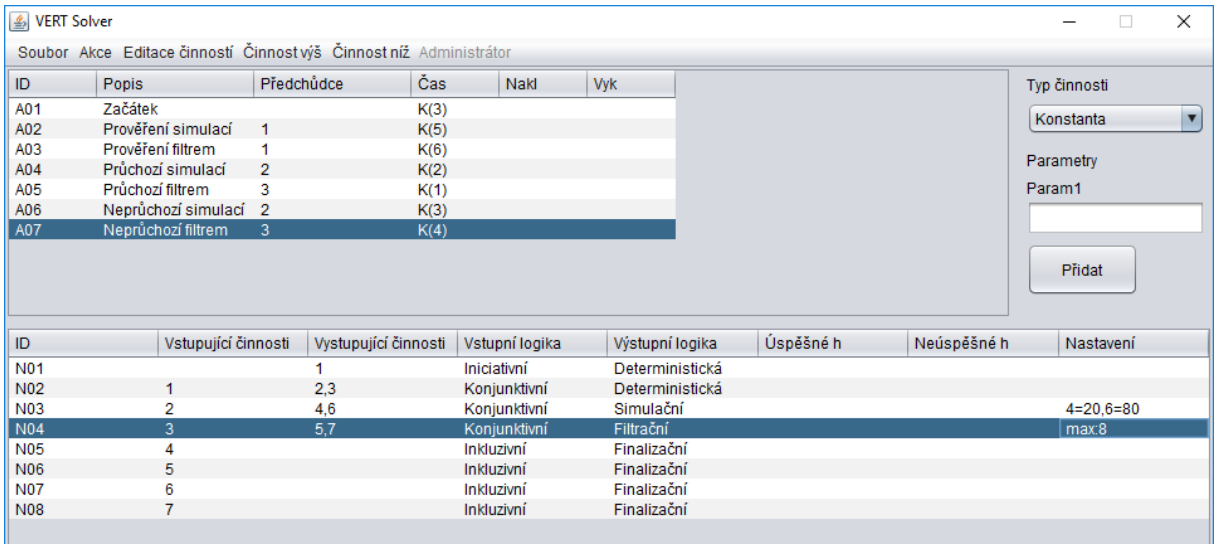
Do políčka zvýrazněného na této stránce se zadávají informace prostřednictvím specifického kódu, který se započítá pouze v případě, když je za výstupní logiku uzlu nastaven typ „Filtrační“ nebo „Simulační“.

V případě filtrační výstupní logiky se údaje zadávají následovně (viz. *N04*):

- Maximální hodnota 8 „max:8“
- Minimální hodnota 10 „min:10“

V případě simulační výstupní logiky se údaje zadávají následovně (viz. *N03*):

- Realizace hrany 4 je 20 % „4=20“
- Realizace hrany 6 je 80 % „6=80“
- Údaje jsou od sebe odděleny čárkou (,) bez mezery



L. *Vybrané kódy z programování – Zadávání činností statistickým rozdělením*

```
public class Distributions {

    public static double getNormalValue(double param1, double
param2, double param3) {

       Random rand = new Random();
       double r = rand.nextGaussian()* param1 + param3;

       return r;
    }

    public static double getUniformValue(double min, double max) {
        double r = Uniform.random(min, max, new MersenneTwister());
        return r;
    }

    public static double getLogNormalValue(double param1, double
param2) {
        double r = LogNormal.random(param1, param2, new
MersenneTwister());
        return r;
    }

    public static double getTriangualValue(double param1, double
param2) {
         double a = param1; // lower bound, inclusive
         double b = param2; // upper bound, exclusive
         java.util.Random rand = new java.util.Random();
         double weightedRand = Math.sqrt(rand.nextDouble()); // use
triangular distribution
         weightedRand = 1.0 - weightedRand; // invert the
distribution (greater density at bottom)
         double result = (int) Math.floor((b-a) * weightedRand);
         result += a; // offset by lower bound
         if(result >= b) result = a; // handle the edge case
         return result;
    }

    public static double getBetaValue(double param1, double param2,
double param3) {
        double res=0;
        res = (param1 + (4 * param2) + param3) / 6;
        return res;
    }



}
```

M. Vybrané kódy z programování – Přechod

```
public class Transition {

    private final ArrayList<Activity> predecessors;
    private final ArrayList<Activity> successors;
    protected String id;
    private InputType inputType;
    private OutputType outputType;
    private static int idCouter = 1;

    public void addPredecessor(Activity activity) {
        if (!predecessors.contains(activity)) {
            predecessors.add(activity);
        }
    }

    public void addSuccessor(Activity activity) {
        if(!successors.add(activity)) {
            successors.add(activity);
        }
    }

    public Collection<Activity> getPredecessors() {
        return predecessors;
    }

    public Collection<Activity> getSuccessors() {
        return successors;
    }

    public InputType getInputType() {
        return inputType;
    }

    public void setInputType(InputType inputType) {
        this.inputType = inputType;
    }

    public OutputType getOutputType() {
        return outputType;
    }

    public void setOutputType(OutputType outputType) {
        this.outputType = outputType;
    }

    public Transition() {
        this.id = "t" + idCouter++;
        predecessors = new ArrayList<>();
        successors = new ArrayList<>();
        inputType = new ConjuctionType();
        outputType = new DeterministicType();
    }
```

```
    public Transition(String id) {
        this.id = id;
        predecessors = new ArrayList<>();
        successors = new ArrayList<>();
        inputType = new ConjuctionType();
        outputType = new DeterministicType();
    }

    public String getId() {
        return this.id;
    }
}
```

N. Vybrané kódy z programování – Generování uzlů metody VERT

```
public class VertTableNode {
    private String name;
    private HashMap<Integer, Integer> pre;
    private HashMap<Integer, Integer> suc;
    private ArrayList activities;
    public VertTableNode() {
        this.pre = new HashMap<Integer, Integer>();
        this.suc = new HashMap<Integer, Integer>();
        this.activities = new ArrayList();
    }
        public void setName(String value) {
        this.name = value;
    }
        public String getName() {
        return this.name;
    }
        public boolean hasPredecessor(int value) {
        return this.pre.containsKey(value);
    }
        public void addPredecessor(int value) {
        this.pre.put(value, value);
    }
        public boolean hasSuccessor(int value){
        return this.suc.containsKey(value);
    }
        public void addSuccessor(int value){
        this.suc.put(value, value);
    }
        public void addActivity(int lineNumber) {
        this.activities.add(lineNumber);
    }
        public ArrayList getActivities() {
        return this.activities;
    }
        public String getActivitiesString() {
        String res = "";

        for (Object o : this.activities) {
           String item = o.toString();
           res = res + item + ",";
       }
        res = res.substring(0, res.length()-1);

       return res;
    }
    public String getPredecessor() {
       String res = "";

       for (Object item : this.pre.keySet()) {
           res = res + item.toString() + ",";
       }
       return res;
    }
}
```

O. *Vybrané kódy z programování – Výpočet uzlu s filtrační výstupní logikou*

```
private boolean reductionComparison(ActivityValue completeValue,
double requiredValue, MinMax minMax) throws java.lang.Exception {
        double tmpVal = 0;
        switch (partialValueType) {
            case TIME:
                tmpVal = completeValue.getTime();
                break;
            case RESOURCES:
                tmpVal = completeValue.getResources();
                break;
            case PERFORMANCE:
                tmpVal = completeValue.getPerformance();
                break;
            default:
                throw new Exception("Invalid comparasion");
        }

        if (minMax == FilterType.MinMax.MAXIMAL) {
            return tmpVal <= requiredValue;
        } else if (minMax == FilterType.MinMax.MINIMAL) {
            return tmpVal >= requiredValue;
        } else {
            throw new Exception();
        }
    }
```

P. Vybrané kódy z programování – Odstranění „přebývajících“ hran

```
private void removeActivity(Activity activity,
Collection<Transition> ignoredTransitions) {
        for (Transition t : this.workingGraph.getTransitions()) {
            if (t.getSuccessors().contains(activity)) {
                t.getSuccessors().remove(activity);
            }

            if (t.getPredecessors().contains(activity)) {
                t.getPredecessors().remove(activity);
                if (t.getInputType() instanceof ConjuctionType) {
                    if (!ignoredTransitions.contains(t)) {
                        ignoredTransitions.add(t);
                    }

                    for (Activity a : t.getSuccessors()) {
                        removeActivity(a, ignoredTransitions);
                    }
                }
            }
        }

        for (Activity a : this.workingGraph.getActivities()) {
            a.removePredecessor(activity);
        }
    }

    private boolean reductionComparison(ActivityValue completeValue,
double requiredValue, MinMax minMax) throws java.lang.Exception {
        double tmpVal = 0;
        switch (partialValueType) {
            case TIME:
                tmpVal = completeValue.getTime();
                break;
            case RESOURCES:
                tmpVal = completeValue.getResources();
                break;
            case PERFORMANCE:
                tmpVal = completeValue.getPerformance();
                break;
            default:
                throw new Exception("Invalid comparasion");
        }

        if (minMax == FilterType.MinMax.MAXIMAL) {
            return tmpVal <= requiredValue;
        } else if (minMax == FilterType.MinMax.MINIMAL) {
            return tmpVal >= requiredValue;
        } else {
            throw new Exception();
        }
    }


            if (t.getOutputType() instanceof SimulationType) {
```

Q. Vybrané kódy z programování – Algoritmus eliminace simulačních hran

```
SimulationType type = (SimulationType) t.getOutputType();

                int rndNum = ThreadLocalRandom.current().nextInt(0,
101);
                int acc = 0;
                Collection<Activity> successors = t.getSuccessors();
                Activity[] staticSuccessors = successors.toArray(new
Activity[0]);
                Activity winner = null;

                for (Activity a : successors) {
                    int prob = type.getProbability(a);
                    if (rndNum <= prob + acc) {
                        winner = a;
                        break;
                    } else {
                        acc += prob;
                    }
                }

                if (winner == null) {
                    throw new Exception("Simulation type has
probably invalid options");
                }

                Collection<Transition> ignoredTransitions = new
ArrayList<>();
                for (Activity a : staticSuccessors) {
                    if (a != winner) {
                        removeActivity(a, ignoredTransitions);
                    }
                }
            }
```

R. *Vybrané kódy z programování – Algoritmus eliminace filtračních hran*

```
            else if (t.getOutputType() instanceof FilterType) {
                FilterType filterType = (FilterType)
t.getOutputType();
                Collection<Activity> successors = t.getSuccessors();
                Activity[] staticSuccessors = successors.toArray(new
Activity[0]);

                ActivityValue precedessorsValue = new
ActivityValue(0, 0, 0);
                for (Activity a : t.getPredecessors()) {
                    precedessorsValue =
precedessorsValue.sum(a.getValue());
                }

                Collection<Transition> ignoredTransitions = new
ArrayList<>();
                for (Activity a : staticSuccessors) {
                    if
(!reductionComparison(precedessorsValue.sum(a.getValue()),
filterType.getValue(), filterType.getMinMax())) {
                        removeActivity(a, ignoredTransitions);
                    }
                }
            }
        }

        if (finalTransition != null) {
            ArrayList<Activity> result = new ArrayList<>();
            while (finalTransition != null) {
                Activity lastActivity =
pathPredecessors.get(finalTransition);
                if (lastActivity == null) {
                    finalTransition = null;
                } else {
                    finalTransition =
lastActivity.getInputTransition();
                    result.add(lastActivity);
                }
            }

            Collections.reverse(result);
            return result.toArray(new Activity[0]);
        } else {
            throw new Exception("There is no final activity with
positive ranking");
        }
    }

    public Activity[] run() throws Exception {
        reset();
        Iterable<String> strOrderedTransitions = topologicalOrder();
        reset();

        Map<String, Transition> transitions = new HashMap<>();
```

```
        for (Transition t : workingGraph.getTransitions()) {
            transitions.put(t.getId(), t);
        }

        Collection<Transition> orderedTransitions = StreamSupport
                .stream(strOrderedTransitions.spliterator(), false)
                .map(id -> transitions.get(id))
                .collect(Collectors.toList());

        return findTheLongestPath(orderedTransitions);
    }
}
```