VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Odhady složitosti podnikových procesů

Business Processes Effort Estimation

2015 Peter Rafaj

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Peter Rafaj**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Odhady složitosti podnikových procesů
Business Processes Effort Estimation

Zásady pro vypracování:

Cílem práce je vytvoření nástroje pro odhadování složitosti (effort estimation) podnikových procesů na základě odlišných proměnných a KPI (key performance indicators) společně s modelem procesu vytvořeným buď v behaviorální, nebo ve funkcionální notaci (například UML diagramy aktivit nebo případy užití). Algoritmy odhadů složitosti student naimplementuje s pomocí umělé neuronové sítě, která bude jako vstupy přijímat proměnné, které budou určovat charakteristiku procesu a jeho instance v organizaci a na výstupu bude odhadovat složitost procesu reprezentovanou vhodnými veličinami (cena, čas, úsilí, rizikovost apod.).

Seznam doporučené odborné literatury:

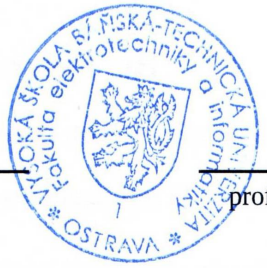
[1] AALST, Will van der. The Application of Petri Nets to Workflow Management. [online]. s. 53 [cit. 2012-07-26]. Dostupné z: http://wwwis.win.tue.nl/~wvdaalst/publications/p53.pdf
[2] AALST, Will van der. Workflow Patterns. [online]. s. 68 [cit. 2012-07-26]. Dostupné z: http://www.workflowpatterns.com/documentation/documents/wfs-pat-2002.pdf
[3] AALST, Will van der. Verification of Workflow Nets. [online]. [cit. 2012-07-26]. Dostupné z: http://wwwis.win.tue.nl/~wvdaalst/publications/p44
[4] JENSEN, Kurt a Lars Michael KRISTENSEN. Coloured Petri Nets: modelling and validation of concurrent systems. Dordrecht: Springer, c2009, xi, 384 s. ISBN 978-3-642-00283-0.
[5] VONDRÁK, Ivo. METODY BYZNYS MODELOVÁNÍ: pro kombinované a distanční studium. [online]. [cit. 2012-07-26]. Dostupné z: http://vondrak.cs.vsb.cz/download/Metody_byznys_modelovani.pdf
[6] VONDRÁK, Ivo. Neuronové sítě. [online]. [cit. 2012-07-26]. Dostupné z: http://vondrak.cs.vsb.cz/download/Neuronove_site.pdf
[7] HEATON, Jeff. Introduction to neural networks, ISBN: 1-60439-008-5

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Michael Alexander Košinár**

Datum zadání: 01.09.2014

Datum odevzdání: 07.05.2015



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 7. máj 2015 ..... Pekev Rafaj .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. máj 2015 ..... Pekev Rafaj .....

Rád by som sa na tomto mieste poďakoval za pomoc a rady všetkých, bez ktorých by táto práca nevznikla. Predovšetkým ďalujem vedúcemu diplomovej práce Ing.Michaelu Alexandrovi Košinárovi za pomoc a rady pri vytvárení tejto diplomovej práce.

Abstrakt

Počas vytvárania a implementácie podnikových procesov sa často krát dostávame do situácie kedy je potrebné určiť približný odhad úsilia, ktoré je potrebné vynaložiť na vytvorenie podnikového procesu. Ide o komplexnú úlohu pretože odhad ovplyvňuje množstvo faktorov. Pri odhadoch úsilia sa však často krát môžeme oprieť o známe vytvorené procesy a tak určitým spôsobom odvodiť komplexnosť procesu na základe už známych odhadov.

Diplomová práca sa zaoberá implementáciou nástroja pre odhad úsilia spojeného s vytvorením podnikového procesu. Program využíva výhody umelých neurónových sietí, UML diagram aktivít a grafovo orientované metriky podnikového procesu. Vybrané metriky budú implementované a použité pri odhadoch úsilia v spojení s umelou neurónovou sieťou.

Záver práce patrí zhodnoteniu výsledkov.

Klíčová slova: podnikový proces, model podnikového procesu, UML, XMI, Java, neuronové siete, odhad zložitosti, graf

Abstract

During the design and implementation of business processes are often in a situation where it is necessary to determine the approximate estimate of effort that must be undertaken to create a business process. This is a complex task because the estimates are subject to a number of factors. When estimating the effort, however, often times we can rely on the known processes and thus created in a way to derive the complexity of the process based on the already known estimates.

The thesis deals with the implementation of a tool for estimating the effort associated with creating a business process. The program takes advantage of artificial neural networks, UML activity diagram and graph-oriented business process metrics. Selected metrics are implemented and used in estimating the effort in conjunction with artificial neural networks.

The conclusion includes evaluation of results.

Keywords: business process,business process model, UML, XMI, Java, artificial neural network networ, effort estimation, graph

Seznam použitých zkratek a symbolů

UML	–	Unified Modeling Language
XMI	–	XML Metadata Interchange
AD	–	Activity diagram
EPC	–	Event-driven process
MOF	–	Meta-Object Facility
OMG	–	Object Management Group
BP	–	Business process
ANN	–	Artificial neural network
Java SE	–	Java Platform, Standard Edition
ARIS	–	Architecture of Integrated Information Systems
GUI	–	Graphical user interface

Obsah

Zoznam tabuliek

Zoznam obrázkov

1 Úvod

V dnešnom svete môžeme vnímať procesy všade okolo nás. Prakticky od narodenie sa riadime určitými zaužívanými procesmi, ktoré sme si osvojili, a ktoré pre nás predstavujú rutinu. Význam riadenie a údržby môžeme vnímať ešte radikálnejšie v prostredí podnikov. Ide o podnikové procesy, ktoré sú určené na riadenie všetkých podstatných činností. Takéto preddefinované a jasne stanovené podnikové procesy významnou mierou pomáhajú pri správnom fungovaní podniku ako samostatnej organizačnej jednotky.

Pred samotným nasadením podnikového procesu v organizácií je potreba tento proces správne navrhnúť a pri návrhu zohľadniť všetky možné aspekty a premenné ktoré môžu vstupovať do podnikového procesu a taktiež ako budú vypadať výstupy našeho procesu. Dynamické prostredie podnikových procesov a taktiež softvérových procesov typicky zahŕňa veľké množstvo technických, demografických, aspektov, ktoré môžu významnou mierou ovplyvniť výslednú štruktúru vytváraného procesu a taktiež jeho zložitosť, ktorá úzko súvisí so zložitosťou konečnej implementácie. Samotné vytvorenie podnikového procesu zahŕňa niekoľko hlavných fáz, ktoré musia byt splnené aby bol proces uskutočniteľný a bolo ho možné úspešne začleniť do organizácie. Jednou z hlavných úloh je korektne získať popis budúceho procesu a následne z neformálne špecifikácie vytvoriť model podnikového procesu. Model predstavuje kostru procesu s jeho hlavnými časťami. V dnešnej dobe existuje niekoľko grafických jazykov akými sú napríklad BPMN, UML, EPC, Petriho siete. Tieto jazyky sú v dnešnej dobe defakto štandard pre komunikáciu medzi jednotlivými členmi týmu, ktorí sa podieľajú na vytváranie procesu od jeho návrhu až po nasadzovanie a údržbu. Umožňujú taktiež spracovanie takto navrhnutého podnikového procesu softvérovými nástrojmi, ktoré tiež nazývame Workflow management system. Tvorba podnikových procesov je úzko spätá so softvérovým procesom. Ide vlastne o podmnožinu podnikových procesov, ktoré sú spojené s vývojom softvérových produktov a všetkých úzko súvisiacich častí. Takéto modely procesov by mali byť dobre štruktúrované a mali by byť ľahko udržiavateľné. Jednou z veľmi podstatných otázok pri návrhu procesu je otázka zložitosti podnikového procesu. V drvivej miere má zložitosť podnikového procesu významnú mieru na znížení alebo zvýšení nákladov, zdrojov na jeho implementáciu a v konečnom dôsledku aj na jeho samotné fungovanie. Taktiež nám zložitosť podnikového procesu môže odhaliť rôzne skryté vlastností o tom či návrh procesu je náchylný na chyby spojené s jeho vykonávaním. Pretože do návrhu podnikového procesu vstupuje veľké množstvo premenných, ktoré menia svoju váhu v danom kontext je veľmi zložité a praktický nemožné presne určiť zložitosť nami vytvoreného podnikového procesu. Pri návrhu modelu podnikového procesu sa návrhári často riadia skúsenosťami, ktoré nadobudli s predchádzajúcich projektov a procesov. To má za následok nemožnosť presne určiť zložitosť a preto môžeme hovoriť iba o odhade zložitosti, ktorý návrhárom môže priblížiť odhadovanú zložitosť z doposiaľ získaných informácií z predchádzajúcich projektov.

Diplomová práca sa zaoberá teoretickým popisom podstatných častí problematiky a taktiež implementačnou časťou. Diplomová práca je rozdelená do týchto základných častí:

1. Prvá teoretická časť sa zaoberá obecným popisom modelovania podnikových procesov s využitím UML diagramu aktivít. Popisuje jeho základné časti a taktiež stručne popisuje vytváranie modelov podnikových procesov. Kapitola začína od 2.1.

2. Druhá teoretická časť sa zaoberá popisom problematiky spojeným s matematickým ohodnotením zložitosti, komplexnosti podnikového procesu. Tejto téme sa venuje kapitola 2.2.

3. Tretia čas 2.5 sa venuje problematike umelých neurónových sietí, ich základným častiam ako aj ich využitiu pri odhadoch zložitosti.

4. Analýze a návrhu sa venuje 2.6. Po analýze nasleduje implementačná časť3. V poslednej časti práce zhodnotím dosiahnuté výsledky

2 Analýza

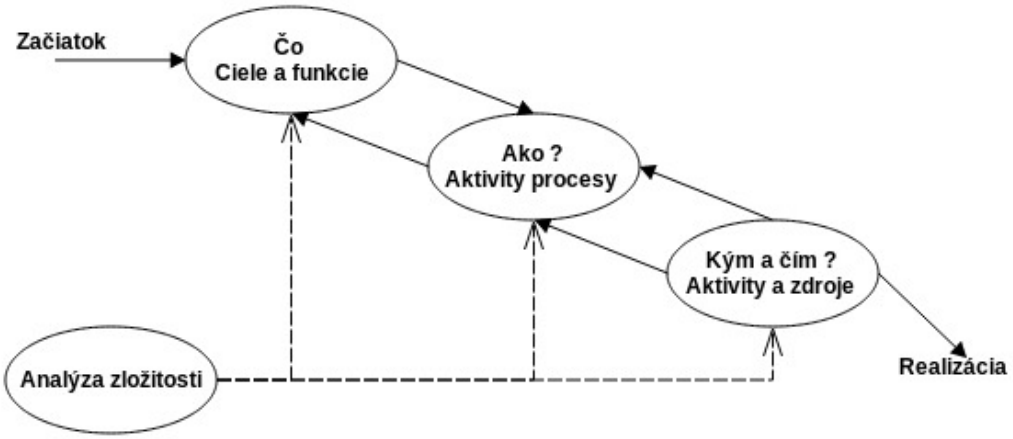
2.1 Modelovanie podnikových procesov

Každá organizácia alebo podnik určitým spôsobom vytvára alebo sa riadi podnikovými procesmi. Podnikové procesy významnou mierou zasahujú do činnosti organizácií a spoločností a v každom prípade pomáhajú k správnemu riadeniu podnikov ako organizačných jednotiek. Pri správnom zvládnutí podnikových procesov môžu zásadným spôsobom ovplyvniť fungovanie organizácie z pohľadu efektívneho využívania zdrojov, ktoré sa používajú pri jej činnosti ako aj zrýchlenia činností organizácie alebo podniku. Pred vytváraním a implementáciou podnikových procesov do organizácie je kľúčovým aspektom správne modelovanie konkrétnej časti BP s ohľadom na všetky jeho možné vstupy a časové postupnosti v danej organizácií.

Zmyslom BP modelovania je vytvorenie takej abstrakcie procesu, ktorá umožňuje pochopenie všetkých jeho aktivít, spojení, ktoré sú vytvárané medzi týmito aktivitami a rolami, ktoré priamo alebo nepriamo zasahujú do daného procesu a ovplyvňujú jeho chod. Business proces je teda po častiach usporiadaná množina procedúr a aktivít, ktoré spoločne realizujú podnikateľský alebo strategický cieľ v kontexte organizačnej štruktúry, ktoré definujú funkcie rolí a jejích vzťahy. Pojmom procedúra v kontexte BP rozumieme podproces obsiahnutý v danom procese. Pojmom po častiach usporiadaný množina vyjadrujeme fakt, že nie všetky aktivity a procedúry je možne zoradiť do jedinej postupnosti. Inými slovami ,takýchto postupností môže byť viac a môžu byť radené vedľa seba, môžu byť súbežné alebo paralelne uskutočniteľné[1].

Model podnikového procesu - Model BP je abstraktná reprezentácia BP a umožňuje jeho ďalšie spracovanie prevažne automatizovaným spôsobom s využitím softvérových nástrojov. Niekedy sa taktiež hovoríme o špecifikácií alebo definícií procesu. Táto špecifikácia môže byť neformálna alebo formálna. Do kategórie neformálnych špecifikácie môžeme zaradiť prirodzený jazyk, poprípade obrázky, tabuľky a ostatné popisné prvky, ktoré akýmkoľvek spôsobom umožňujú pochopiť modelovaný proces. Formálna špecifikácia je technika umožňujúca jednoznačne špecifikovať proces vďaka precízne určenej syntaxy a sémantike použitej metódy. Pri samotnom spracovaní BP softvérovými nástrojmi hovoríme o Workflow (ďalej už len WF). WF je v podstate automatizovaný BP. WF možno zjednodušene chápať ako postupnosť krokov, počas ktorej dokumenty, informácie alebo úlohy sú presúvané od jedného účastníka k druhému pre ďalšie spracovanie, napríklad podľa definovanej organizačnej štruktúry. Jednou zo základných častí BP je aktivita. Ide v podstate o popis činnosti, ktorá reprezentuje jeden krok vo vykonávaní procesu. Aktivita môže byť vykonávaná manuálne alebo automatický s využitím softvérových nástrojov.

Z uvedeného teda vyplýva, že proces môžeme chápať ako popis toho ako jeho jednotlivé prípady majú byť vykonané. Každá konkrétna implementácia má svoj začiatok a koniec a na jeho výstupe je zmysluplný produkt, služba. Obyčajne má jeden proces veľký počet prípadov, ktoré sú podľa takého predpisu vykonané.



Obr. 1: Process modelovania a vytvárania podnikového procesu a dôležitosť odhadov zložitosti

2.1.1 Návrh modelu

Účelom modelovania je vytvorenie takej abstrakcie procesu, ktorá umožňuje pochopenie všetkých jeho aktivít a vzťahov medzi aktivitami a rolami, ktoré reprezentujú schopnosti ľudí, zariadení a externých systémov, ktoré sú zapojené do daného procesu. V súčasnej dobre veľké množstvo metód postavených na rôznych technológiách, ktoré sú používané k vytváraniu modelov BP. Tieto metódy však majú spoločný abstraktný rámec, ktorý vyplýva z postupu návrhu BP. Najskôr je nutné identifikovať aké funkcie daná organizácia alebo podnik má plniť a za akým účelom. Hľadáme čo je vstupom a čo je výstupom týchto funkcií a ako sú tieto funkcie štruktúrované. Nasleduje ďalší krok popisujúci ako tieto funkcie budú zaisťovať transformáciu vstupov na výstupy pomocou k tomu určených aktivít a procesov. Nakoniec je potrebné definovať, čím konkrétne sú v predchádzajúcich krokoch definované dané toky a kto a čo bude realizovať špecifické aktivity. V podstate existujú tri základné prístupy, ktoré sa využívajú k modelovaniu procesov, a ktoré vychádzajú z troch základných typov použitej abstrakcie:

- Funkčný prístup zameraný predovšetkým na funkcie, ich štruktúrovane, vstupy a výstupy
- Prístup špecifikácie chovania je zameraný na riadiaci aspekt vykonávania procesu cestou stanovenia udalostí a podmienok za ktorých môžu byť jednotlivé aktivity vykonané.
- Štrukturálny prístup je zameraný na statický aspekt procesu.

Všetky moderné metódy modelovania BP používajú všetky z uvedených abstrakcií. Líšia sa iba v preferencií tej či onej stránky modelu. Existuje niekoľko všeobecne používaných

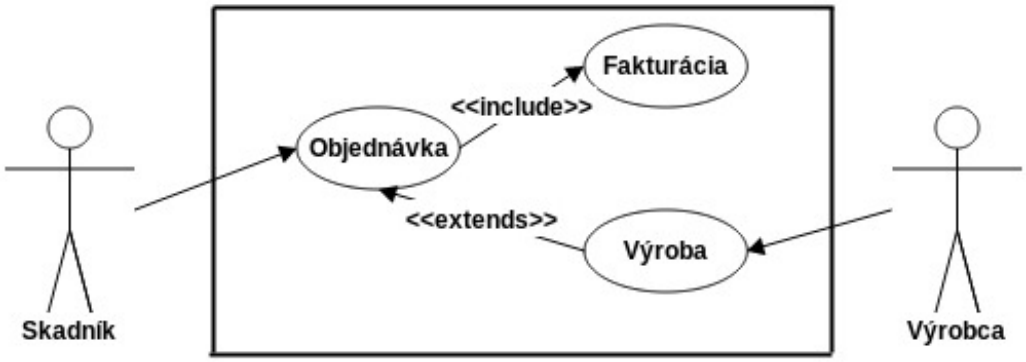
štandardov na vytváranie modelov BP. Ide napríklad o metódy ako sú IDEF, EPC, UML. V ďalších častiach sa budem zaoberať popisom UML jazyka, ktorý je jednou z alternatív, ktorú môžeme použiť pri modelovaní a analýze BP.

2.1.2 UML

Jazyk UML je modelovací jazyk, ktorý slúži na grafické znázornenie vzťahov, častí a časových závislostí pri modelovaní softvérových produktov. Hlavným podnetom pre vytvorenie štandardizovaného jazyka, ktorý by bol univerzálny bolo zjednotenie rôznych prístupov pri modelovaní softvérových procesov. Tento jazyk sa stal jedným z používaných štandardom pri modelovaní a analýze softvérových procesov. Pretože softvérový proces je podmnožina podnikových procesov, stal sa jazyk UML používaným nástrojom taktiež na modelovanie BP. Umožňuje jednoznačne definovať model podnikového procesu a popísať jeho vlastnosti z určeného funkčného pohľadu. Z hľadiska použitia jazyka UML pre potreby modelovania BP nám jazyk UML ponúka predovšetkým nasledovné typy diagramov:

1. Use case diagram je určený k popisu a analýze funkcií modelovaného systému
2. Dynamický náhľad popisujúci chovanie je vyjadrený v diagrame aktivít
3. Logický náhľad je popísaný diagramom aktivít

Pre potreby ohodnocovania podnikového procesu využívam diagram aktivít. Pre úplnosť v nasledujúcej kapitole uvádzam taktiež use case diagramy. Funkčná špecifikácia či už v softvérovom systéme alebo podnikovom procese býva často krát vyjadrená pomocou diagramov use case (prípad užitia). Ide o pojem, ktorý je v rámci business modelovania definovaný nasledovným spôsobom.
Prípady použitia - Ide o postupnosť akcií, ktoré podnik či organizácia realizujú v interakcií so špecifickými aktérmi s cieľom vytvoriť výsledok požadovanej hodnoty. V podstate teda môžeme povedať, že use case je v kontexte zameniteľný pojem s pojmom podnikový proces.
Aktéri - Pod aktérom si môžeme predstaviť niekoho alebo niečo čo stojí mimo popisovaný podnikový proces. Use case diagram sa zobrazením externých častí podnikového procesu ideálne hodí na dokumentáciu a popis interakcie medzi službami, ktoré sú organizáciou poskytované a tými, ktorými sú tieto služby požadované. V podstate môže ísť o užívateľov, externé informačné systéme danej organizácie alebo úplne oddelené systémy, ktoré daná inštancia podnikového procesu využíva pri svojom behu. Okrem toho je možné medzi procesmi definovať relácie rozšírenia a použitia. Prvý z uvedených typov popisuje situáciu kedy jeden use case rozširuje iný a doplňuje tak scenár realizácie procesu o ďalšie aktivity. Naopak spojenie použitie nutne definuje zahrnutie aktivity do daného procesu.Obrázok 2 znázorňuje jednoduchý prípad užitia s aktérmi skladník a výrovbca a 3 prípadmi užitia a to objednávka, fakturácia, výroba.



Obr. 2: Príklad diagramu použitia

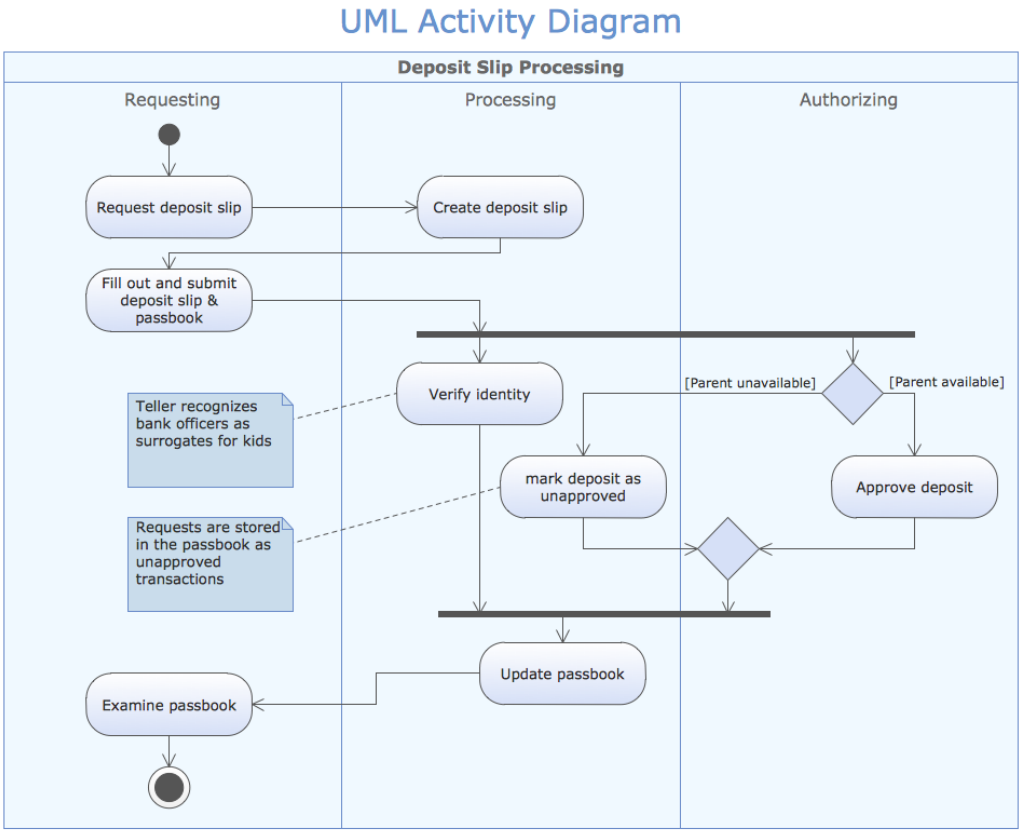
2.1.3 Diagram aktivít

Diagram aktivít popisuje toky činností, pomocou aktivít reprezentujúcich akčné stavy a prechody medzi nimi. Prechod medzi dvoma stavmi je realizovaný cestou ukončenia predchádzajúceho stavu. Prechod je teda realizovaný interným ukončením vykonávania prvej aktivity viazaný na daný stav a prechodom po hrane, ktorá spája ďalšiu aktivitu. Ďalším účelom diagramu aktivít je definovať, kto alebo ktorý objekt zodpovedá za danú aktivitu, prípadne aké objekty sú aktivitami vytvárané, spotrebované alebo modifikované. Týmto spôsobom môžu byť do jedného diagramu zaznamenané nielen toky, ktoré riadia podnikový proces ale taktiež dátové toky vytvárané počas prechádzania procesu. Obrázok 3 zázorňuje komplexný diagram aktivít.

Počiatky UML špecifikácie siahajú do roku 1997 kedy bola vydaná prvá verzia špecifikácia UML. Nasledujúci zoznam popisuje najdôležitejšie verzie a zameriava sa na tie časti, ktoré využívam v ďalšej analýze. Pre viac informácií [27].

- V roku 1997 bola prvý krát vydaná verzia s označením UML 1.1 organizáciou OMG.
- Verzie 1.3 a 1.4 priniesli drobné zmeny v metamodeli, sémantike a zmeny ktoré sa netýkali diagramu aktivít.
- V roku 2003 bola vydaná verzia 1.5, ktorá pridávala schopnosť zobrazenia akcií v diagrame aktivít.
- Zmeny pre diagram aktivít nastali s príchodom verzie 2.0, ktorá bola vydaná v roku 2005. Z pohľadu zmien išlo o prelomovú verziu. Pre diagram aktivít priniesla táto verzia prerobenie špecifikácie a použitie sémantiky používanej pri Petriho sieťach. Zmeny taktiež prispeli k možnosti zobrazovania hrán diagramu v oddieloch. Oddiely môžu byť hierarchické a viacrozmerné. Taktiež novinkou v tejto verzie bolo explicitné špecifikovanie toku objektov v diagrame aktivít.

- Verzie od 2.1 až po 2.3, ktoré boli vydané v rokoch 2006 až 2010 prinášali drobné zmeny.
- Aktuálne sa špecifikácia UML nachádza vo verzií 2.4.1 a z pohľadu zobrazovania diagramu aktivít so sebou nepriniesla výrazné zmeny v sémantike a špecifikácií DA.

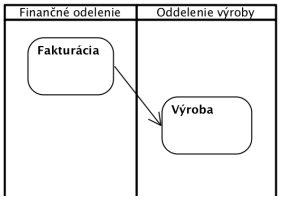


Obr. 3: Príklad modelovanie podnikového procesu s využitím diagramu aktivít. Dostupný z adresy

Diagram aktivít obsahuje niekoľko základných elementov, s ktorými budeme v nasledujúcom texte pracovať a ktoré je potrebné spomenúť vzhľadom k ich funkčnosti a ďalšej analýze [27]:

- **Aktivity** - Aktivita reprezentuje vykonanie časti podnikového procesu. V prípade že je aktivita štruktúrovaná do ďalšieho diagramu aktivít je symbol akčného stavu označený špeciálnou, k tomuto účelu, zadefinovanou ikonou. Ide o parametrizované správanie reprezentované ako koordinovaný tok činností. Tok činností je modelovaný ako postupnosť aktivít, ktoré sú medzi sebou prepojené hranami. Aktivita môže obsahovať taktiež parametre, ktoré vstupujú do aktivity. Aktivity môžu

obsahovať akcie, objekty a kontrolne elementy. Ako bolo spomenuté na začiatku kapitoly, aktivity, môžu byť organizované do tzv. swimlanes, ktoré zoskupujú skupinu aktivít, ktoré patria do spoločnej skupiny a oddeľujú sa od iných aktivít napríklad tým, že sú vykonávané určitým aktérom alebo inou časťou systému, ktorý sa snažíme popísať. Obrázok 4 znázorňuje jednoduchú aktivitu a jej zobrazenie v swimlanes.

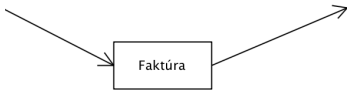


Obr. 4: Activita a jej zobrazenie vo swimlanes

- **Action** - Ide o pomenovaný prvok, ktorý predstavuje atomickú operáciu aktivity, 5 ďalší krok aktivity a teda ju už nie je možné rozložiť na menšie celky. V kontexte porovnania s aktivitou je aktivita správanie, ktoré sa skladá práve s jednotlivých akcií. Pri popise akcií ide prevažne o slovesá, ktoré vyjadrujú atomickú činnosť ako vyplnenie objednávky, validácia dokumentu, atď. Do akcie môže vstupovať niekoľko vstupných riadiacich hrán a taktiež z nej môže vystupovať niekoľko výstupných riadiacich hrán. Okrem bežných akcií existujú taktiež špecifické akcie ako akcie, ktoré odosielajú signál tzv. „Send Signal Action“. Jednou z vlastí, ktoré takéto aktivity majú je fakt, že odosielateľ nečaká na odpoveď ale okamžite pokračuje vo vykonávaní ďalších krokov riadiaceho toku. Ďalšími varianty, ktoré pre aktivity existujú sú „Accept Event Action“, „Accept Signal Action“, „Wait Time Action“. Ide o špecifické aktivity, ktoré znázorňujú asynchronné spracovanie akcií alebo znázorňujú periodicky sa opakujúce akcie.

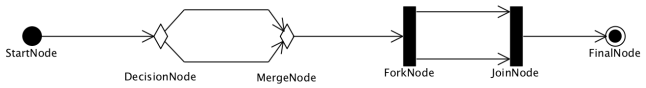
Obr. 5: Action a jej zobrazenie

- **Objekt** - Ide o abstraktný prvok aktivity, ktorý definuje tok objektov v aktivite. 6Definuje objekt, ktorý môže byť v určitom stave výstupom alebo vstupom do aktivity. S pojmom objekty UML špecifikácia zavádza taktiež tzv. „pin“, ktorý znázorňuje vstupné alebo výstupné objekty akcie.



Obr. 6: Zobrazenie objektu a jeho toku v diagrame aktivít

- **Kontrolné elementy** - Ide o elementy, ktoré slúžia na riadenie toku činností. 7Medzi tieto elementy patrí inicializačný element, ktorý jasne špecifikuje začiatok vykonávania podnikového procesu. Koncový element, ktorý definuje koniec aktivity alebo celého toku činností. Pri rozhodovaní toku činností UML špecifikuje rozhodovací element tzv. „Decision Node“ , ktorý slúži na rozdeľovanie niekoľkých tokov činností. Jeho hlavnou úlohou však je špecifikovanie rozhodovania pri vykonávaní toku činností. Spolu s definuje UML taktiež tzv. „Merge Node“, ktorý slúži na zlúčenie niekoľkých tokov činností do jedného. Podobnú funkcionalitu popisuje taktiež tzv. „Fork Node“ a „Join Node“, ktoré slúžia na rozdelenie alebo zlúčenie dvoch a viacerých rokov činností do nezávislých paralelných vetiev.



Obr. 7: Zobrazenie kontrolnych elementov diagramu aktivít

- **Hrany** - Znázorňujú tok činností od aktivít, akcií, rozhodovacích blokov. Ide o elementy, ktoré spájajú hlavné elementy do postupnosti a tak vytvárajú tok činností a výsledný model podnikového procesu. Okrem bežného toku činností hranou môžeme znázorniť taktiež tok objektov, ktorý vzniká medzi jednotlivými aktivitami.

2.2 Metriky podnikových procesov

Aby bolo možné pracovať s diagramom aktivít a sledovať jeho vlastnosti, je potrebné nad daným diagramom aktivít aplikovať určité metriky, ktoré vyjadria vlastnosti, ktoré budem následne skúmať a vyhodnocovať. Metriky v podstate vyjadrujú matematickou formou vlastnosti daného modelu. Pokiaľ hovoríme o metrikách týkajúcich sa obecných modelov podnikových procesov, či už ide o UML diagram aktivít alebo BPMN existuje veľké množstvo metrík, ktoré popisujú rôzne vlastnosti a taktiež sú určené na vyjadrenie chybovosti a iných vlastností úzko spojených s model BP. Aby bolo možné spracovať daný model BP je potreba nájsť vhodnú dátovú štruktúru pre reprezentáciu jednotlivých prechodov a aktivít v modeli. Z vlastností a taktiež grafického pohľadu môžeme povedať o diagrame aktivít, že ide v určitých vlastnostiach o dátovú štruktúru, ktorá je totožná s orientovaným grafom.

V nasledujúcej kapitole sa zameriam na definovanie metrík, ktoré využijem pri ohodnocovaní modelu BP. Tieto metriky sú prevažne založené na analýze grafov s využitím doplnkových informácií, ktoré nám diagram aktivít ponúka. Taktiež sa v tejto kapitole venuje definovaniu pojmu metrika a validácia metriky s ohľadom na relevantné informácie získané korektným ohodnotením modelu.

2.2.1 Orientovaný graf

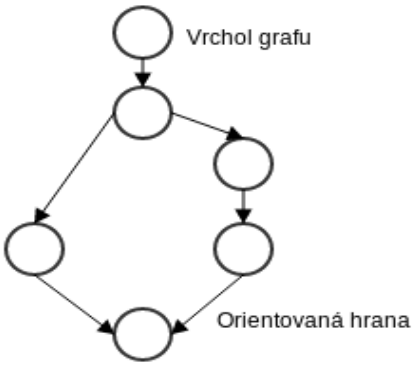
V základom rozdelení môžeme grafové štruktúry rozdeliť na neorientovaný graf a orientovaný graf. Ďalej sa už budeme zaoberať iba orientovaným grafom. Obecná definícia orientovaného grafu znie.

Definice 2.1 *Orientovaný graf je dvojica (V, E), kde V je množina vrcholov a je množina hrán. [2]*

Orientovaný graf sa od neorientovaného obecného grafu líši práve tým, že u každej jeho hrany záleží na poradí. Hrany preto môžeme považovať za šípky medzi jednotlivými dvojicami uzlov. Môžeme teda rozlíšiť počiatočný a koncový uzol danej hrany. Všimnime si, že hrany sú prvkami kartézskeho súčinu V x V, a teda usporiadané dvojice vrcholov. Orientovaný graf je vlastne totožný z binárnou reláciou na množine. Dvojicu (u,v) interpretujeme ako hranu, ktorá začína vo vrchole u a končí vo vrchole v. Pred samotnou analýzou modelu BP si musíme zadefinovať niekoľko pojmov, ktoré súvisia s teóriou grafov, a ktoré budeme využívať pri analýze modelu BP.

Slučka - Hrana sa nazýva slučka ak je tvaru (v,v) a teda počiatočná aj koncový bod je rovnaký. Z toho vyplýva že orientovaný graf môže obsahovať slučky. Graf sa nazýva jednoduchý ak neobsahuje slučky a násobné hrany.
Násobné hrany - Ide o skupinu hrán, ktoré existujú medzi dvoma rovnakými vrcholmi v jednom smere. V d'alšej analýze budeme predpokladať, že tento stav nie je povolený a násobné hrany v našom orientovanom grafe nesmú existovať. Graf, ktorý obsahuje slučky alebo násobné hrany sa nazýva multigraf.
Stupeň vrcholu - Pre jednotlivé vrcholy grafu definujeme výstupný stupeň a vstupný

stupeň. Výstupný stupeň je počet hrán grafu, ktorých počiatočných uzlom je práve daný vrchol. Vstupný stupeň je počet hrán grafu, ktorých koncovým uzlom je práve daný uzol grafu.
List - Ide o vrchol so výstupným stupňom 0. Z pohľadu dosiahnuteľnosti ide teda o koncový element grafu.
Koreň - Ide o vrchol so vstupným stupňom 0. Z naše ho pohľadu ide teda o začiatok grafu.
Cesta v grafe - je postupnosť orientovaných hrán, pri ktorých vždy nasledujúce hrany začínajú v uzle, v ktorom končí predchádzajúce hrana.
Súvislý graf - Graf G = (V, H) sa nazýva súvislým ak pre každú dvojicu vrcholov u, v (u sa nerovná v) existuje cesta, ktorá ich spája.
Acyklický graf - je graf, ktorý neobsahuje žiadne cykly.



Obr. 8: Orientovaný graf

2.3 Tvorba metrík

V nasledujúcej časti sa venujem definovaniu metrík, ktoré slúžia na matematický opis modelu podnikového procesu. V prvom rade musíme opísať prečo vznikla potreba vytvárania takýchto metrík. Aké výhody nám prináša matematické vyjadrenie modelu z ohľadom na všetky jeho vlastnosti. S obecnými vlastnosťami modelu podnikového procesu je spojené taktiež vlastnosti, ktoré definujú chybovosť modelu podnikového procesu. Obecne existuje niekoľko výskumov, ktoré popisujú chyby v reálnych modeloch podnikových procesov. Ako sa uvádza Mendling v [5], pri ľuďoch dochádza k obmedzeniu kognitívnych schopností a to nás privádza k záveru, že ľudia konajú racionálne len do určitej miery. Pokiaľ ide o chyby pri modelovaní procesov môžeme usudzovať že procesní inžinieri pri rozsiahlych modeloch podnikových procesov strácajú prehľad vo vzájomných vzťahoch jednotlivých častí kvôli ich obmedzenej kognitívnej schopnosti

pri modelovaní. Existujú štúdie, ktoré skúmajú do akej miery má zložitosť a komplexita podnikového procesu vplyv na chyby, ktoré vznikajú v danom procese.

Pri meraní ide obecne o proces, kedy prirad'ujeme čísla alebo symboly atribútom objektom reálneho sveta. Meranie vlastností daného objektu alebo činnosti nám prináša 3 základné výhody. Ide o schopnosť porozumieť danému objektu, kontrolovať daný objekt a vo výsledku schopnosť zlepšenia danej vlastnosti alebo procesu. V kontexte ohodnocovania modelu podnikového procesu sa musíme jednoznačne zamerať na schopnosť porovnať vlastností dvoch odlišných procesov s cieľom zistiť rozdiely, ktoré môžu mať vplyv na zložitosť a komplexnosť. Schopnosť porozumieť odmeraným vlastnostiam daného procesu nám umožňuje lepšie pochopiť vzťahy medzi jednotlivými entitami daného procesu a taktiež prinášajú lepšie pochopenie BP. Pri spracovávaní modelu však nie je vždy na prvý pohľad jasné čo na danom modeli je možné odmerať a aký vplyv môže mať hodnota nameraného atribútu pri d'alšom spracovaní.

2.3.1 Teória merania

Teória popisuje ako empiricky popísať a ohodnotiť prvky reálneho sveta. V kontexte ohodnocovania modelov BP ide o matematické ohodnotenie modelu. Pri meraniach sa stretávame s tromi základnými problémami. Prvým s problémov je problém zastúpenia. Týka sa podmienky, že ohodnocovanie by malo zachovať vzťahy, ktoré existujú v reálnom svete. Po druhé jedinečnosť problému môže viesť k výrazným rozdielom pri nameraných hodnotách. Tretím problémom je schopnosť vyvodiť zmysluplné prehlásenie o danej vlastnosti pri porovnaní dvoch nameraných hodnôt na rovnakej stupnici. Existuje nominálna, postupná, intervalová, pomerová mierka.

- **Nominálna** - Hodnota tejto mierky môže byť reprezentovaná iba ako jedinečný identifikátor. Príkladom môže byť ID procesu ako napríklad číslo 1 alebo 7. Jediné čo v danom kontexte môžeme povedať o daných procesoch je nie sú rovnaké. Akékoľvek transformácie môžu byť vykonané iba ak neberieme do úvahy jedinečnosť danej hodnoty.

- **Postupná** - Ide napríklad o hodnoty, ktorými môže mapovať komplexnosť daného BP. Môžeme napríklad povedať že BP s hodnotou 1 je triviálny, 2 je jednoduchý, 3 je komplexný a 4 je neuskutočniteľný.

- **Intervalová** - Intervalová stupnica je konštantná a zachováva relatívnu vzdialenosť. Príkladom môže byť čas kedy boli dva podnikové procesy navrhnuté. Následne môžeme vyjadriť hodnoty ako dni od kedy je projekt modelovaný.

- **Pomerová** - Ide o mierku, ktorú budeme využívať najčastejšie. Pomerová mierka s hodnotou 0 reprezentuje absenciu danej vlastnosti. Príkladom takejto mierky môže byť počet paralelných blokov v modeli podnikového procesu.

Pred samotným prehlásením toho, že danú nameranú vlastnosť môžeme považovať za platnú a môžeme z nej usudzovať určité vlastnosti modelu BP musíme podrobiť vlastnosť určitej validácií. V podstate môžeme povedať, že validácia odpovedá na otázku či

sú merania skutočne presné a či namerané hodnoty skutočne odpovedajú hodnotám, o ktoré máme záujem. Pri prehlásení, že ide o platné meranie, musíme zodpovedať tri základné otázky. Obsah platnosti, kritéria platnosti a poslednou z nich je ako bude platnosť danej vlastnosti vytvorená. Ak meranie nie je validné nemôžeme prehlásiť naše zistenia za validné.

- **Obsah platnosti** - Tento typ platnosti sa vzťahuje na schopnosť meranie reprezentovať celú škálu významov spojených so skúmaným empirickým javom.
- **Kritéria platnosti** - Tento typ platnosti poukazuje na pragmatickú hodnotu meraní a teda ako blízko sú spojené namerané hodnoty s reálnymi. Predpokladajme, že skúmame komplexitu modelu. Ak sa napríklad rozhodneme porovnávať komplexitu na základe počtu aktivít v modeli, môžeme sa dostať do situácie kedy dva modeli s rovnakým počtom aktivít budú mať v skutočnosti rozdielnu komplexitu pretože pôjde v prvom príklade napríklad o sekvenčný model a v druhom o model s veľkým počtom cyklov. Preto by počet aktivít ako kritérium ktoré vyjadruje komplexitu modelu mohlo mať problém s kritériami platnosti pretože reálne sa výrazne líši od skutočnej komplexnosti.
- **Konštrukcia platnosti** - Tento typ platnosti poukazuje na teoretickú úvahu spojenú s otázkou či je daná vlastnosť primeraná. Príkladom môže byť označovanie modelu BP určitými číslami. Je jasné že takúto vlastnosť musíme jasne zamietnuť pretože neexistuje teoretické spojenie medzi číslom modelu a napríklad ohodnotením komplexnosti modelu.

Za spoľahlivé merania považujeme tie, ktoré sú konzistentné nad subjektmi a časom a teda dávajú rovnaké výsledky pre rovnaké subjekty pri zmene času kedy boli merania vykonávané. Kým stupnice môžu byť validné a nie platné, nespoľahlivé metódy nemôžu byť validné.

2.4 Analýza metrík podnikových procesov

V nasledujúcej kapitole sa budem venovať definovaniu metrík, ktoré je možne merať nad vytvoreným modelom podnikového procesu. Prevažná časť popisovaných metrík vychádza z analýzy sietí, ktoré môžeme považovať za orientovaný graf s hranami a vrcholmi, na ktoré môžeme aplikovať vlastnosti a skúmať vzťahy medzi jednotlivými prvkami takéhoto grafu. Ide o aplikovanie teórie grafov. Tvorba sietí a grafov má široké zastúpenie v našom sociálnom živote a taktiež našla široké uplatnenie vo vedeckým oblastiach akými sú napríklad skúmanie elektrických obvodov, hľadanie vzťahov v sociálnych väzbách ľudí ale taktiež v oblasti medicíny na skúmanie šírenia epidémií. Ako vyplýva z doposiaľ prezentovaných vlastností modelu podnikového procesu, ide o špeciálny typ orientovaného grafu. Preto je prirodzené že tvorba metrík bude zameraná na sieťovú analýzu modelu podnikového procesu.

Analýza sietí sa obecne zaoberá tromi základnými oblasťami. Prvou kategóriou je skúmanie prvkov danej siete z kvantitatívneho hľadiska. Druhou kategóriou je skúmanie špecifickej skupiny prvkov daného grafu. Z pohľadu reálnej analýzy môže ísť o skúmanie vzťahov ľudí v určitej skupine. Poslednou kategóriou je skúmanie a ohodnocovanie celej siete. Pre analýzu podnikových procesov sa budeme zameriavať na poslednú kategóriu sieťovej analýzy, pretože sa na model podnikového procesu musíme pozerať vždy ako celok a teda nie je možné hodnotiť vlastnosti procesu na základe jeho častí.

Pred popisom jednotlivých metrík, ktoré môžu byť použité na meranie vlastností modelu BP, je nutné definovať o aký typ grafu ide, popísať a definovať jeho základné elementy. Ako som už spomenul model podnikového procesu je špeciálny typ grafu $G = (N, A)$ s najmenej tromi typmi uzlov N, o ktorých môžeme povedať že $N = T \cup S \cup J$ kde T reprezentuje aktivitu podnikového procesu a teda nedeliteľnú jednotku práca, S rozdeľovací blok a teda v UML ide o AND, OR alebo XOR element, ktorý rozdeľuje tok činností do dvoch a viacerých vetiev a J, ktorý reprezentuje elementy podnikového procesu, ktoré spájajú tok činností dvoch alebo viacerých vetiev. Preto teda môže taktiež ísť o AND, OR alebo XOR elementy modelu. Ďalej je graf zložený z hrán $A \subseteq N \times N$, ktoré tieto uzly spájajú. V nasledujúcich analýzach tieto hrany nijako neohodnocujeme. Jedinú vlastnosť, ktorú hrana má je jej smer. V ďalšom texte uvádzam pre lepší prehľad taktiež anglický názov metriky.

2.4.1 Veľkosť modelu (Size)

Ide o základnú a ľahko merateľnú metriku, ktorá nám podáva základný pohľad o obecnej veľkosti modelu bez ohľadu a to ako zložito sú jednotlivé elementy modelu medzi sebou prepojené. Táto metrika je široko zastúpená v materiáloch, ktoré sa venujú obecným vlastnostiam modelov BP. Metriku popisuje taktiež[5][6][9]. Metrika je podobná s metrikou NOA (Number of Activity), ktorá však ráta iba s počtom aktivít.

Symbol: S_N

$$S_N(G) = |N|$$

Definícia: Počet vrcholov grafu G modelu podnikového procesu

Dôvod: Veľký počet podnikových procesov, ktorých metrika S nadobúda veľké hodnoty môže obsahovať chyby spojené s veľkosťou modelu. Z veľkosti a chybovosti môžeme teda usudzovať, že proces bude komplexnejší a bude ho ťažké zrealizovať. Tým pádom rastie zložitosť a celkový časť strávený implementáciou.

Obmedzenia: Existuje však veľký počet modelov, kedy aj vysoká hodnota metriky S nemá vplyv na komplexnosť modelu a teda pri ňom nedochádza k zvýšeniu zložitosti. Takýto proces si môžeme predstaviť ako sekvenčné vykonávanie aktivít, kedy jednotlivé aktivity na seba nadväzujú v jednoduchom poradí bez zložitejšieho prepojenia či dokonca rozdelenia toku.

Hypotéza: Zvýšenie metriky S môže mať za následok zvýšenie úsilia na implementáciu daného procesu.

2.4.2 Najdlhšia cesta (Diam)

Ide o metriku, ktorá popisuje najdlhšiu cestu od začiatku modelu diagramu aktivít až do jeho konca. Pretože diagram aktivít môže obsahovať niekoľko koncov, rátame so všetkými možnosťami. Číslo udáva počet hrán, ktoré definujú danú cestu. Túto metriku je možné vypočítať na základe matice vzdialeností grafu alebo taktiež na algoritme najkratšej cesty.[5]

Symbol: Diam

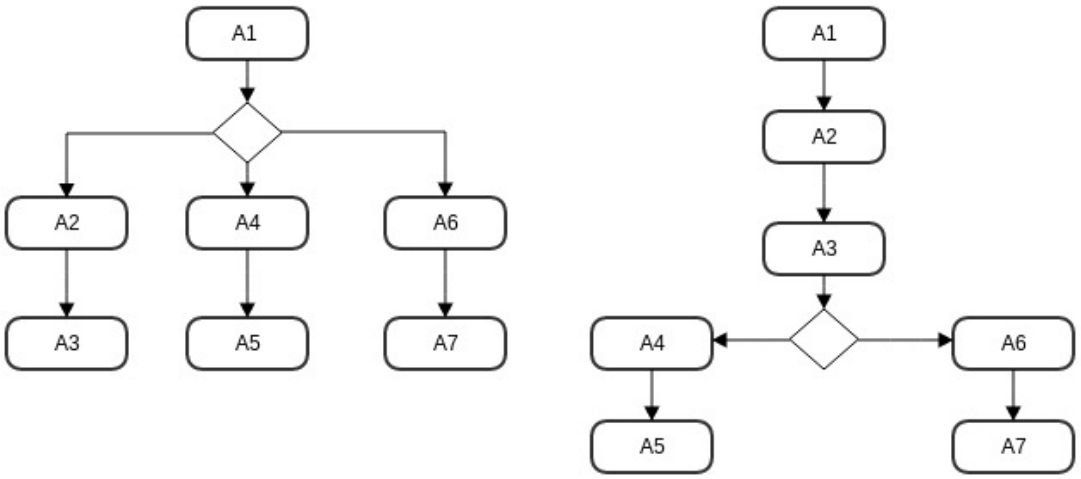
Definícia: Metrika určuje najdlhšiu cestu z vrcholu definovaného ako začiatok modelu až do vrcholu, ktorý je označený ako koniec modelu BP.

Dôvod: Rozsiahle podnikové procesy môžu obsahovať veľmi dlhé cesty grafom od počiatku modelu až po jeho koniec. To má za následok zvýšenie zložitosti a možnosti vzniku chyby. Z toho vyplýva, že náročnosť a implementácia modelu s vysokou hodnotou tejto metriky sa môže razantne zvýšiť.

Obmedzenia: Ako aj v predchádzajúcej metrike tak aj v metrike najdlhšej cesty môžeme naraziť na problém, kedy hodnota bude vysoká a teda bude indikovať komplexne zložitý proces, reálne môže však ísť o sekvenčné zoskupenie aktivít podnikového procesu.

Hypotéza: Zvýšenie metriky diam môže mať za následok zvýšenie komplexity a tým zvýšenie nákladov na implementáciu.

Na obrázku 9 môžeme vidieť nedostatky jednoduchej metriky veľkosti podnikového procesu. Ľavý aj pravý model má rovnakú veľkosť, pritom sa líšia v najdlhšej ceste. Veľkosť týchto modelov je 8. Hodnota najdlhšej cesty je však pre ľavý model 3 a pre pravý model 5.



Obr. 9: Porovnanie Size a Diam. Veľkosť týchto modelov je 8. Líšia sa však v najdlhšej ceste.

2.4.3 Hustota (Density)

V nasledujúcom kontexte budeme používať pojem hustota v spojení s definíciou a vyjadrením počtu elementov v modeli a počtu hrán, ktoré tieto elementy spájajú. Existuje niekoľko metrík, ktoré skúmajú počty prvkov v danom modeli.[5]

Symbol: $D_{\triangle}$

$$D_{\triangle}(G) = \frac{|A|}{|N| \cdot (|N| - 1)}$$

Definícia: Hustota modelového grafu je vyjadrená ako počet hrán grafu vydelený maximálnym počtom vrcholov.

Dôvod: Podnikový proces s veľkou hustotou by mal byť náchylnejší k tvorbe chýb a taktiež by mal byť zložitejší na implementáciu v porovnaní s modelom, ktorý má rovnaký počet spočitateľných vrcholov.

Obmedzenia: Hustotu v podmienkach rozdielu je ťažké porovnať pri modeloch s rozdielnym počtom vrcholov. Väčšie modely s rovnakým priemerom spojenia majú menšiu hustoty, pretože maximálny možný počet spojení rastie druhou mocninou počtu elementov N. Z toho môže vyplývať že väčšie modely môžu mať menšiu hustotu na základe prirodzenosti grafov.

Hypotéza: Zvýšenie hustoty modelu grafu môže mať za následok zvýšenie komplexnosti, väčšiu chybovosť a zdĺhavejšiu implementáciu.

2.4.4 Koeficient prepojenia (CNC)

Symbol: CNC

$$CNC(G) = \frac{|A|}{|N|}$$

Definícia: Koeficient prepojenia popisuje pomer hrán grafu ku vrcholom grafu.

Dôvod: Z pohľadu hustoty prepojenia hrán medzi jednotlivými vrcholmi grafu v kontexte CNC môžeme povedať, že zvýšenie tejto hodnoty má za následok zvýšenie komplexnosti daného modelu. To má taktiež za následok zvýšenie chybovosti pri implementácií.

Obmedzenia: Hustotu v podmienkach rozdielu je ťažké porovnať pri modeloch s rozdielnym počtom vrcholov. Väčšie modely s rovnakým priemerom spojenia majú menšiu hustoty, pretože maximálny možný počet spojení rastie druhou mocninou počtu elementov N. Z toho môže vyplývať že väčšie modely môžu mať menšiu hustotu na základe prirodzenosti grafov.

Hypotéza: Zvýšenie metriky CNC má za následok zvýšenie chybovosti a celkovej zložitosti modelu BP. Podľa môjho názoru hodnoty tejto metriky > 2 môžu mať za následok razantne odzrkadľovanie komplexnosti modelu. Z tohto pohľadu ide teda o relevantnú metriku.

2.4.5 Priemer prepojenia konektorov (Avarege degree)

Táto metrika sa zameriava na rozdeľovacie, rozhodovacia a spájacie prvky modelu BP. V podstate hovorí o priemernej zložitosti logických blokov, ktoré rozhodujú o toku činností v danom modeli.

Symbol: DC

$$DC(G) = \frac{1}{|C|} \sum_{c \in C} d(c)$$

Definícia: Ide o priemernú hodnotu, ktorá udáva počet vrcholov do ktorých je daný rozhodovací prvok pripojený.

Dôvod: Model pri ktorom bude dosahovať dc metrika vysoké hodnoty bude pravdepodobne náchylný na chyby a ťažšie zrealizovateľný ako v priemere bežný podnikový proces, ktorý má priemernú hodnotu tejto metriky 2.

Obmedzenia: Rozdelenie a celkové prepojenie konektorov môže byť rozdielne. Príkladom môže byť model, ktorý ma 5 konektorov s hodnotou prepojenia 2 a jeden, ktorý obsahuje 5 takýchto prepojení. Takýto razantný výkyv môže výraznou mierou ovplyvniť výslednú vierohodnosť metriky. V reálnych modeloch vidíme zriedkavo aby konektor spájal 5 možných ciest.

Hypotéza: Z implementačného hľadiska nám hodnota hovorí všeobecne o rozhodovacích blokoch a ich zložitosti. Zložitosť týchto prvkov môže mať za následok výslednú chybovosť a časovú náročnosť na implementáciu.

2.4.6 Maximálne prepojenia konektorov (Maximal avarage degree)

Symbol: DC_{max}

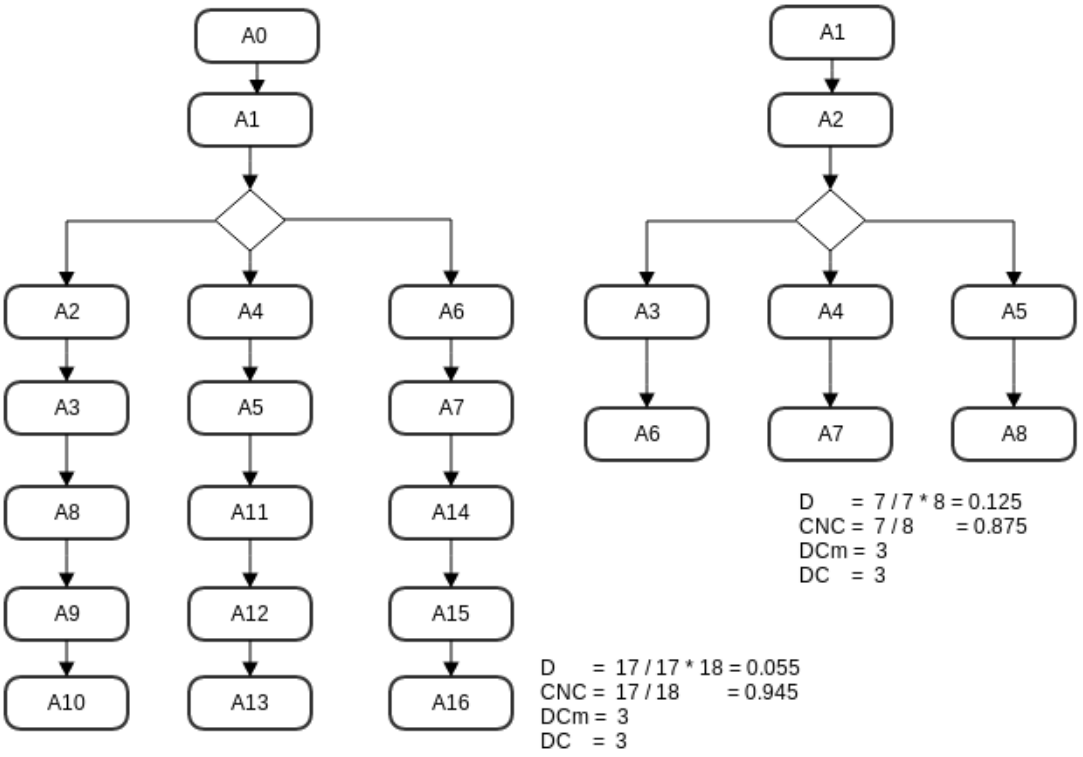
$$DC_{max}(G) = max\{d(c)|c \in C\}$$

Definícia: Ide o totožnú vlastnosť modelu BP s hodnotou DCA. Rozdiel je iba v tom že udáva maximálnu hodnotu vlastnosti, ktorá bola s pozorovaná na danom modeli.

Dôvod: Vysoká hodnota tejto metriky môže mať za následok zvýšenie komplexnosti. Ak sa napríklad pozrieme na hodnoty > 4. Môže indikovať zle navrhnutý model, ktorý bude časovo náročné implementovať a teda sa naňho spotrebuje viac zdrojov. Môže ísť ísť taktiež o bod, ktorý významnou mierou zmení výslednú zložitosť modelu.

Obmedzenia: Môže existovať model, ktorý bude mať vysokú hodnotu tejto metriky ale zvyšné hodnoty konektorov budú nízke. Myšlienka tejto metriky je podobná s metrikou uvádzanou v, ktorá hovorí o identifikovaní komponenty, ktorá je najťažšie pochopiteľná pretože obsahuje množstvo vstupov a výstupov.

Obrázok10 znázorňuje rovnaké a rozdielne vlastnosti metrík, ktoré popisujú hustotu modelu dvoch relatívne odlišných modelov. Model na ľavej strane obsahuje 18 elementov a 17 hrán a model na pravej strane obsahuje 8 elementov a 7 hrán. Hustota modelu na pravej strane je viac ako o polovicu väčšia oproti ľavej strane. Tento fakt potvrdzuje to, že je ťažké porovnávať zložitosť dvoch rozdiele veľkých modelov iba na základe hustoty. Hodnota CNC nám však ale hovorí, že modely sú si podobné pretože pre ľavý je táto metrika rovná číslu 0.945 a pre pravý hodnote 0.875. Priemerná a maximálna hodnota prepojenia konektorov sa pre oba modely rovná číslu 3. Z toho môžeme usudzovať že modely sú si podobné.



Obr. 10: Porovnanie metrík $D_{\triangle}$, CNC, DC a DC_{max}

2.4.7 Oddeliteľnosť (Separability)

Nasledujúca metrika sa zaoberá obecným pohľadom skupiny objektov v grafe voči celému grafu. Oddeliteľnosť je úzko spojená s grafovou analýzou a popisuje vrcholy, ktoré ak sú z danej grafovej štruktúry vymazané, model je rozdelený do dvoch nezávislých grafických modelov a teda nie je možné medzi nimi nájsť spojenie hranou. Tieto vrcholy tiež nazývame *articulation point*

Symbol: Π

$$\Pi(G) = \frac{|\{n \in N | n = articulationpoint\}|}{|N| - 2}$$

Definícia: Metrika definuje pomer vrcholov, ktoré po odstránení rozdelia model na dva nezávislé modely a celkového počtu vrcholov v grafe.

Dôvod: Model s vysokým pomerom takýchto vrcholov by mal pravdepodobne obsahovať menej chýb a malo by byť jednoduchšie takýto model implementovať ako model s nízkou hodnotou. Ak sa pozrieme napríklad na sekvenčný model

procesu. Odstránením ľubovoľnej aktivity rozdelíme model na dva nezávislé. Preto je pre nás každá aktivita rozdeľovacím vrcholom a preto je tento model jednoduché zrealizovať.

Obmedzenia: Oddeliteľnosť môže byť vysoká napríklad ak model obsahuje dva sekvenčné časti. Odstránením ľubovolnej aktivity z tohto modelu nezískame dva oddelené modely. V ďalších častiach môže byť model zložitý a my to nie sme schopný z danej vlastnosti zistiť.

Hypotéza: Zvýšenie oddeliteľnosti by malo mať za následok zníženie chybovosti a času stráveného implementáciou.

2.4.8 Súvislosť modelu (Sequentiality)

Súvislosť popisuje fakt, že sekvencia aktivít alebo udalostí je najjednoduchšia časť podnikového procesu. Súvislosť teda popisuje pomer hrán, ktoré sa nachádzajú v sekvencií voči celkovému počtu hrán v modeli BP.

Symbol: S_q

$$S_q(G) = \frac{|A \cap (T \times T)|}{|A|}$$

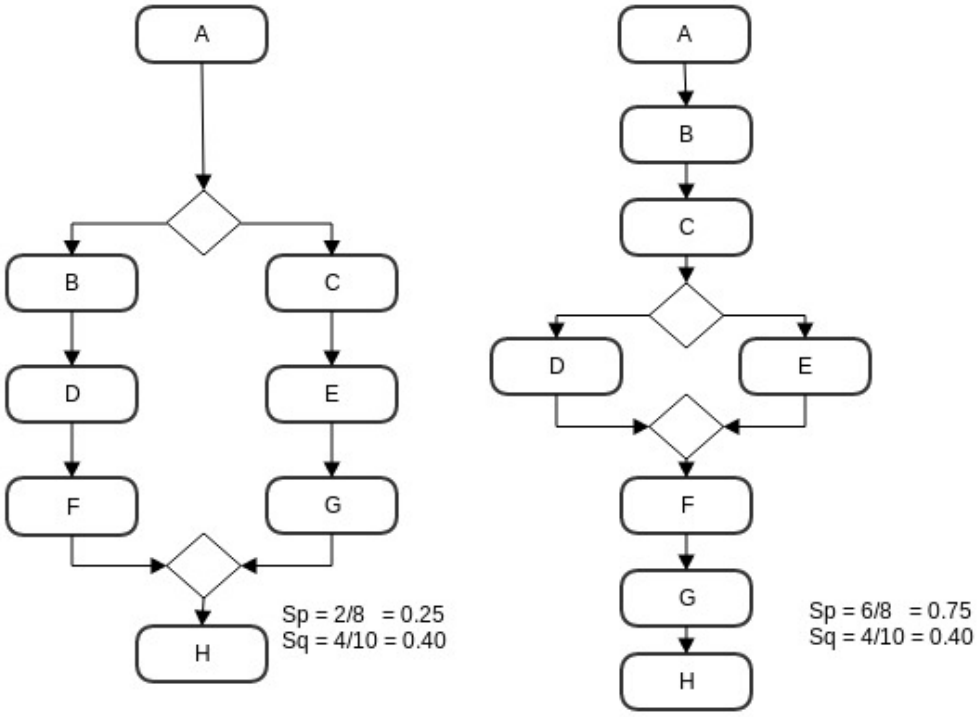
Definícia: Pomer hrán, ktoré sa nachádzajú v sekvencií voči celkovému počtu hrán skúmaného modelu.

Dôvod: Táto metrika jasne popisuje celkovú stavbu modelu podnikového procesu. Proces, ktorého väčšina častí je tvorená sekvenčným vykonávaním aktivít by nemal byť náchylný na chyby a mal by byť ľahko implementovateľný. V porovnaní do metrikou Oddeliteľnosť táto metrika berie do úvahy taktiež sekvencie vykonávané paralelne alebo výlučným spôsobom. AK model obsahuje iba jednu sekvenciu, táto metrika bude nadobúdať hodnotu 1.

Obmedzenia: Dva modely môžu mať rovnakú hodnotu tejto metriky ale vlastnosti a prepojiteľnosť konektorov daného modelu bude iná.

Hypotéza: Zvýšenie tejto hodnoty indikuje sekvenčnejší podnikový proces a teda jednoduchšiu a časovo menej náročnú implementáciu.

Obrázok 11 znázorňuje rozdiel medzi oddeliteľnosťou a súvislosťou modelu. Model na ľavej strane obsahuje dva oddeliteľné vrcholy (XOR-join a XOR-split), kým model na pravej strane navyše obsahuje oddeliteľné vrcholy B,C,F a G. I keď počet vrcholov je pri oboch modeloch rovnaký, oddeliteľnosť je u modelu na ľavej strane 0.25 a modelu na pravej strane 0.75. Súvislosť týchto modelov je však rovnaká. Tá je pre model na ľavej strane medzi vrcholmi B a F, C a G. A pre model na pravej strane je to sekvencia medzi A a C a taktiež F a H. Hodnota tejto vlastnosti je 0.4.



Obr. 11: Porovnanie metrík *Separability* Π a *Sequentiality* S_q

2.4.9 Hĺbka modelu (Depth)

Hĺbka modelu popisuje maximálny počet vnorených štrukturálnych blokov v procese. V našom prípade ide o množinu konektorov, ktoré rozdeľujú tok daného procesu. Pred samotnou definíciou musíme popísať algoritmus, ktorým popíšeme kroky potrebné k získaniu danej vlastnosti. Algoritmus musí počítať do hĺbky v (n) uzla n vzhľadom k jeho predchodcovi n*. Všetky uzly sú inicializované hodnotou 0. Algoritmus následne prejde všetky cesty. Začína z počiatočného uzlu a končí buď v koncovom uzle alebo v uzle ktorý navštívil v predchádzajúcom kroku. Pri každom navštívení uzla zvýši hodnotu tak že pripočíta jedna k hodnote, ktorá bola nastavená v predchádzajúcom uzle.

Symbol: D_p

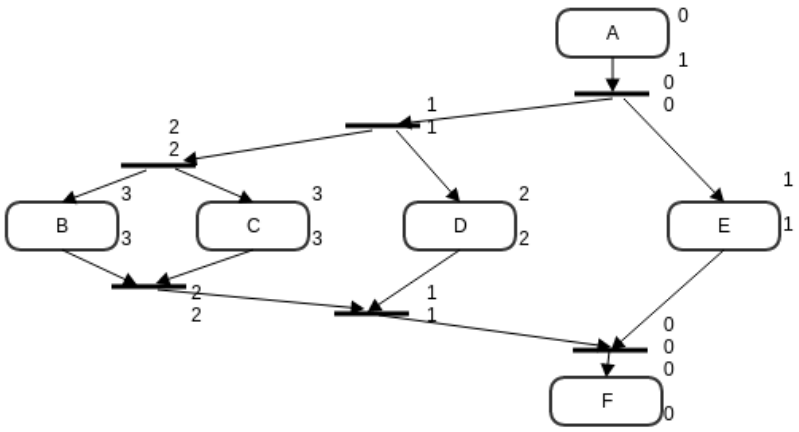
$$D_p = max\,\{\lambda(n)|n \in N\}$$

Definícia: Hĺbka vyjadruje maximálne hĺbku všetkých elementov. Ide teda o číslo, ktoré vyjadruje najhlbší možný výskyt rozdeľovacích elementov.

Dôvod: Procesný model s vysokou hodnotou tejto metriky je pravdepodobne náchylnejší k chybám a je ho teda zložitejšie zrealizovať. Vyžaduje viac úsilia na implementáciu.

Obmedzenia: Existujú procesné modely, ktoré majú identickú hodnotu tejto vlastnosti ale sú rozmerovo diametrálne odlišné.

Hypotéza: Vyššia hodnota tejto metriky má za následok vznik chýb a časovú náročnosť na implementáciu.



Obr. 12: Algoritmus Hlbky modelu D_p

Obrázok 12 znázorňuje implementáciu algoritmu, ktorý zisťuje hĺbku navštívených konektorov.

2.4.10 Zhoda rozhodovacích blokov (Connector Interplay)

Táto metrika sa zameriava na skupinu rozhodovacích a rozdeľovacích elementov podnikového procesu. Popisuje fakt, že pokiaľ je model dobre štruktúrovaný existuje ku každému rozhododovaciemu a rozdeľovaciemu elementu element spájací rovnakého typu. Nekorektné umiestnenie a použitie rozhodovacích a spájacích elementov môže viesť ku chybám. Celková spojitá súhra je súčet všetkých typov konektorov.

Symbol: MM

$$MM_l = \left| \sum_{c \in S_l} d(c) - \sum_{c \in J_l} d(c) \right|$$

$$MM(G) = MM_{or} + MM_{xor} + MM_{and}$$

Definícia: Súhra rozhodovacích blokov je súčet nezhôd pre všetky typy konektorov v danom modeli podnikového procesu.

Dôvod: Procesný model ktorý ma vysokú mieru nezhody v rozhodovacích blokoch je pravdepodobne zle navrhnutý a celkovo takýto proces bude ťažké zrealizovať a to hlavne kvôli zlej synchronizácií paralelných tokov. Takýto proces môže veľmi ľahko skončiť v zablokovanom stave.

Obmedzenia: UML aktivity diagram dovoľuje ľubovolný počet konečných elementov a teda ľubovoľný počet procesných koncov. To môže mať za následok zvýšenie tejto metriky pričom model môže byť dobre štruktúrovaný a relatívne jednoduchý.

2.4.11 Počet možných ciest (CFC – Control Flow Complexity)

Meria celkový počet ciest, ktorými môžeme prejsť model grafu. Meria všetky možné stavy, ktoré môžu nastať pri rozdelení toku činností riadiacim elementom. Ide o jednu z kľúčových metrík [5][6][9][10].

Symbol: CFC

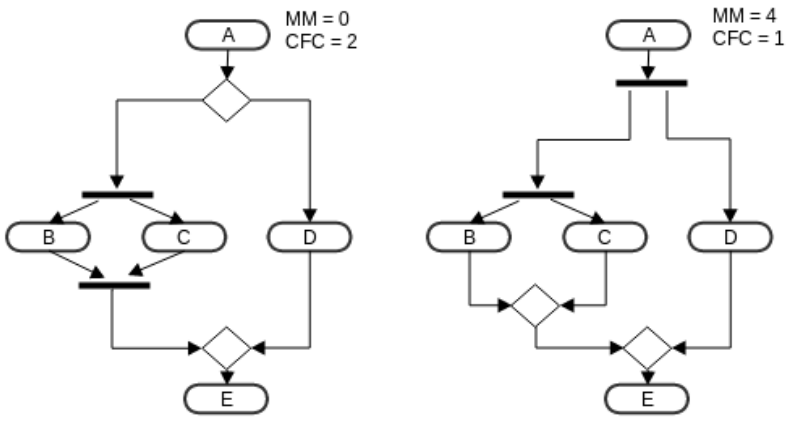
$$CFC(G) = \sum_{c \in S_{and}} 1 + \sum_{c \in S_{and}} |c_{xor}\bullet| + \sum_{c \in S_{and}} 2^{|c_{or}\bullet|} - 1$$

Definícia: CFC je celkový súčet pre všetky typy konektorov vyjadrujúci celkový počet kombinácií, ktoré môžu nastať po rozdelení toku činností špecifickým konektorom.

Dôvod: Ako bolo spomínané už pri mnohých metrikách so zvyšujúcim sa počtom konektorov rôznych typov musí merateľných spôsobom rásť komplexnosť a chybovosť procesného modelu.

Obmedzenia: Modely s rovnakou štruktúrou ale s rozdielnymi typmi konektorov môžu mať rozdielnu metriku CFC a pritom budu jednoducho pochopiteľné a a dobre štruktúrované.

Hypotéza: Zvýšenie CFC môže mať za následok celková zvýšenie chybovosti a komlexnosti pri realizácií takého modelu.



Obr. 13: Metriky MM a CFC

Obrázok 13 znázorňuje 2 typy metrík, ktoré sú spojené s konektormi modelu podnikového procesu. Proces na ľavej strane má 1 AND konektor a 1 OR konektor, s ktorých každý s ukončením v podobe spojovacieho konektoru. Nezhoda rozhodovacích blokov je teda 0. CFC metrika je 2 pretože OR reprezentuje rozhodovací blok s výberom dvoch ciest. Model na pravej strane má rovnakú štruktúre ale rozdielny počet konektorov. V modeli sa nachádzajú dva AND rozdeľovacie konektory a 2 OR spojovacie konektory. Tieto konektory však nie sú párové. To vedie k nezhode rozhodovacích blokov na hodnotu 4. Hodnota metriky CFC pre druhý model je 1 a to kvôli AND konektoru, ktorý rozdeľuje tok na jednu alternatívnu cestu.

2.4.12 Cyklicita (Cyclicity)

Cyklicita modelu definuje pomer vrcholov, ktoré sú zahrnuté v nájdených cykloch modelu podnikové procesu ku celkovému počtu vrcholov celého modelu.[5]

Symbol: CYC

$$CYC = \frac{|N_C|}{|N|}$$

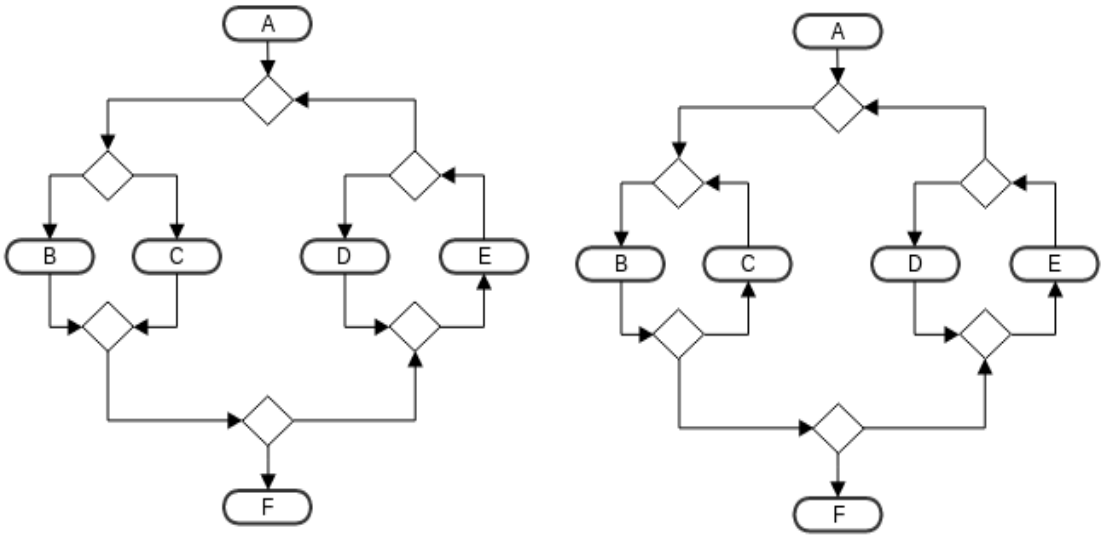
Definícia: Pomer celkového počtu vrcholov v nájdených cykloch ku celkovému počtu vrcholov modelu.

Dôvod: Proces s vysokou mierou cyklicity bude pravdepodobne obtiažnejšie realizovateľný hlavné kvôli potrebe kontrolovať vykonávanie takýchto slučiek, čo má za následok zvýšenie časovej náročnosti a celkovej chybovosti modelu.

Obmedzenia: Existujú modely s rovnakou mierou cyklicity ale rozdielnou mierou zrozumiteľnosti ak je jeden z nich väčší.

Hypotéza: Zvýšenie tejto hodnoty má za následok zvýšenie komplexnosti a časovej náročnosti

Obrázok 14 znázorňuje dva podobné modely BP. Majú rovnakú cyklicitu pretože majú rovnaký počet uzlov obsiahnutý v slučkách. Rozdiel je v tom že model na pravej strane ma dva cykly, ktoré sú vnorené, zatiaľ čo model na pravej strane má dva vnorené XOR bloky do jednej slučky. Vzhľadom na to že cyklicita popisuje iba celkový počet vrcholov obsiahnutých v cykle, nemôžeme rozlišovať medzi modelmi s rôznym počtom vnorených cyklov.



Obr. 14: Dva modely s rovnakou hodnotou cyklicity *CYC* ale rozdielnou veľkosťou cyklov.

2.4.13 Počet rolí, aktérov (Role size)

Ide o jednoduchú metriku, ktorá nám hovorí koľko aktérov, zasahuje do skúmaného podnikového procesu [6].

Symbol: R_s

Definícia: Celkový počet aktérov vstupujúcich do podnikového procesu. Aktérov môžeme chápať ako samostatné role modelovaného systému ako aj externé systémy.

Dôvod: Veľký počet aktérov, ktorý vstupujú do systému môže mať za následok zvýšenie komplexnosti a nutnosti riadiť podnikový proces na základe vstupov od množstva aktérov.

Obmedzenia: Existujú modely, ktoré majú rovnaký počet aktérov ale ich interakcia so systémom môže byť rôzna a tým pádom zložitosť a komplexnosť takéhoto modelu môže byť rozdielna. Tým pádom nemožno rozhodnúť komplexnosť iba na základe počtu aktérov vstupujúcich do podnikového procesu.

Hypotéza: Zvýšenie počtu aktérov a rolí má za následok zvýšenie času stráveného implementáciou, kvôli nutnosti riadenia týchto aktérov.

Pri modelovaní podnikových procesov môžeme využiť niekoľko štandardov, ktorými sú napríklad BPMN, IDEF, ARIS. Každý z týchto štandardov zo sebou prináša množinu metrík, ktorými je možné merať vlastnosti procesu ako komplexnosť, zložitosť, chybovosť alebo kvalitu modelu podnikového procesu. Tieto metriky teda môžeme ešte rozdeliť do množín a to podľa toho na aké vlastnosti je ich vhodné použiť. Niektoré z momentálne spomenutých metrík sú taktiež využívané v iných modeloch alebo som ich priamo prebral a upravil pre potreby modelovania pomocou UML diagramu aktivít. Pre jednotlivé štandardy modelovania podnikových procesov poznáme ešte niekoľko metrík:

BPMN
Pri modelovaní procesu pomocou štandardu BPMM popisujú metriky, ktoré úž boli spomenuté v predchádzajúcej kapitole ako napríklad CFC, NOAC(Size)[21],[23] ale taktiež nové metriky ako [21] Halstead-based (HPC), IC. Pri hodnotení kvality sa taktiež zameriava [21] na ohodnotenie vlastností akými sú spojitosť (coupling) a súdržnosť (cohesion). Tieto pojmy pochádzajú z návrhu softvérových systémov. Ide o vlastnosti, ktoré nám hovoria o vzájomnom prepojení softvérových komponentov. Autor sa snaží použiť vlastnosti softvérových systémov na modely podnikových procesov. Metrikami ako NOA, NOAC alebo NOAJS sa taktiež zaoberá [22]. Ide o metriky ktoré rátajú celkový výskyt elementov v danom modeli. O metrikách ako DSM a PSM hovorí [22]. Ide v podstate o metriky, ktoré vyjadrujú pomerné množstvo hlavných elementov. Implementácií metrík využívaných pri skúmaní softvérových procesov do prostredia podnikových procesov s využitím BPMN sa venuje [24]. Ide o veľké množstvo metrík, ktoré je zamerané na súčet všetkých elementov, ktoré sa nachádzajú v modeli. Aplikovanie metódy COSMIC sa zaoberá [25].

IDEF
Ide o staršiu metódu, ktorá zahŕňa metodiku modelovania pre popisovanie výrobných funkcií, funkcionálny modelovací jazyk pre analýzu, vývoj, integráciu s informačnými systémami, modelovanie podnikových procesov alebo softvérových procesov. Jej počiatky siahajú do sedemdesiatych rokov minulého storočia, kedy našla uplatnenie počas integrácie programu vzdušných síl. Je rozdelená do niekoľkých modulov. Modelovaniu podnikových procesov sa venuje najmä modul IDEF0. Štandard popisuje niekoľko metrík, ktoré sú rozdelené do kategórií popisujúce veľkosť modelu, objem a typ spracovávaných dát ako aj komplexnosť výsledného modelu. Taktiež pre IDEF model BP existujú metriky, ktoré popisujú štrukturálne vlastnosti diagramu [26].

ARIS
Ide o štandard, ktorý ako modelovací jazyk využíva EPC diagram. Prakticky, všetky metriky, ktoré som spomenul ako vhodné na použitie v spojení s UML diagramom aktivít majú spojitosť s EPC diagramom. Z veľkej časti som vychádzal práve z metrík vytvorených pre tento štandard. Vo veľkej miere sa týmto metrikám venuje [5]. Medzi metriky, ktoré neboli spomenuté patrí napríklad metrika, ktorá vyjadruje súbežnosť modelu.

2.5 Umelé neurónové siete

Nasledujúca časť sa venuje stručnému popisu problematiky umelých neurónových sietí. Existuje niekoľko výskumov, ktoré sa zaoberajú využitím neurónových sietí pri odhade úsilia a zložitosti či už softvérových alebo podnikových procesov.

Umelá neurónové sieť je výpočtový model, ktorým sa snažíme v určitej miere napodobniť niektoré vlastnosti biologických nervových systémov. Ide v podstate o simuláciu veľmi malej časti ľudského mozgu. V dnešnej dobe počítače dokážu vykonávať milióny operácií za sekundu omnoho rýchlejšie ako ľudský mozog. Dôležité je však podotknúť že rýchlejšie neznamená vždy lepšie pre riešenie problému. Existuje veľké množstvo úloh pri ktorých ľudský mozog jasne dominuje v porovnaní s počítačmi. Príkladom môže byť rozpoznanie predmetu na obrázku. Takáto úloha je pre človeka prakticky triviálna avšak pre počítačový systém môže byť značne komplikovaná. Ako uvávdza prof. Ing. Ivo Vondrák [11], špecifické vlastnosti umelých neurónových sietí môžeme zhrnúť do týchto základných bodov.

1. Ako už bolo spomenuté, NS sú inšpirované biologickými neurónovými sieťami. Táto vlastnosť by ich mala predurčovať k tomu, že by sa mali chovať rovnako alebo veľmi podobne ako biologické vzorky. Ako už bolo povedané, vytvorenie umelého ľudského mozgu je či už z hľadiska kvantity neurónov či spôsobu prepojenia týchto neurónov je veľmi komplikované, v dnešnej dobe bohužiaľ zatiaľ nemožné. Môžeme sa pomocou umelých neurónových sietí aspoň pokúsiť simulovať určité malé časti ľudského myslenia a tieto potom implementovať.

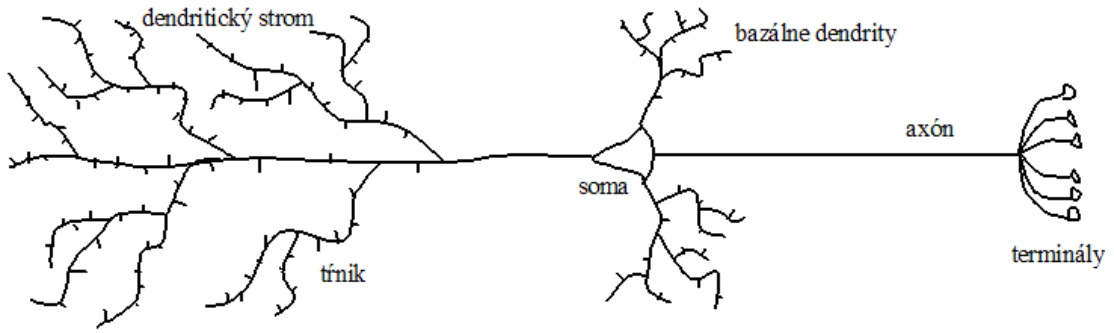
2. Neurónové siete využívajú distribuované, paralelné spracovanie informácií pri vykonávaní výpočtov. Inými slovami ukladanie, spracovanie predávanie informácií prebieha prostredníctvom celej neurónovej siete, než pomocou určitých pamäťových miest. V podstate teda pamäť, ktorá je vytváraná učením NN je globálneho charakteru.

3. Znalosť alebo schopnosť rozpoznať danú informáciu je pri NS uložená hlavne prostredníctvom sily väzieb medzi jednotlivými neurónmi. Väzby, ktoré indikujú správny výsledok sú posilňované a väzby, ktoré vedú k nesprávnym odpovediam sú oslabované pomocou opakovanej expozície príkladov, ktoré popisujú reálny problém.

4. Učenie je základná a podstatná vlastnosť NN. Tento fakt je zjavne novinkou ak porovnáme doposiaľ spomenuté algoritmické metódy, ktoré nijakým spôsobom nepracovali s pamäťou alebo inak nevyužívali už doposiaľ spracované hodnoty. Ide teda o základný rozdiel medzi bežným použitím počítača a použitím počítača na báze NS. Pokiaľ sme doposiaľ vytvárali algoritmy, ktoré transformujú vstupnú množinu parametrov na výstupnú množinu parametrov, pri NS tento prístup neplatí. Spôsob transformácie vstupných premenných na výstupné premenné sa v NS deje práve procesom učenia, kedy je neurónovej sieti predkladaná množina dobre známych dvojíc vstupov a výstupov a teda v prostrední NS ide o trénovaciu mno-

žinu. Z toho vyplýva fakt, že pri riešení problému pomocou NS s časti odpadá algoritmizácie úlohy nakoľko táto fáza je nahradená procesom učenia.

Základnou časťou, z ktorej je neurónová sieť tvorená je neurón. Tieto základné stavebné prvky neurónových sietí sú navzájom medzi sebou prepojené a tak vytvárajú komplexný systém, ktorý dokáže spracovávať dáta a na základe naučených vlastností rozhodnúť o výstupnej množine. *„Odhaduje sa, že v ľudskom mozgu sa nachádza rádovo* 10^{11} *nervových buniek (neurónov). Dve tretiny neurónov tvoria 4-6 mm hrubú mozgovú kôru, ktorá tvorí jeho silne zvrásnený povrch. Predpokladá sa, že mozgová kôra je sídlom poznávacích (kognitívnych) procesov, ako sú myslenie, vnímanie a pamäť.V neurónoch prebiehajú zložité biochemické deje, ktoré zabezpečujú to, že neuróny môžu spracúvať signály z iných neurónov a vysielať k nim svoje vlastné signály. Signály môžeme reprezentovať ako reálne čísla vyjadrujúce intenzitu (veľkosť) prijímaných a vysielaných signálov, ktoré majú v skutočnosti elektrickú a chemickú povahu. Spoj medzi dvoma neurónmi (miesto prenosu signálu z jedného neurónu na druhý) sa nazýva synapsa. V synapse sa môže ten istý signál buď zosilniť alebo zoslabiť. Silu pôsobenia synapsy určuje váha synapsy. Jeden neurón môže mať na svojom povrchu rádovo* 10^3 *až* 10^5 *synáps. Vstupný povrch neurónu pozostáva z dendritov a tela (somy) neurónu. Tisícky dendritov tvoria bohato rozvetvený strom.“* [12]



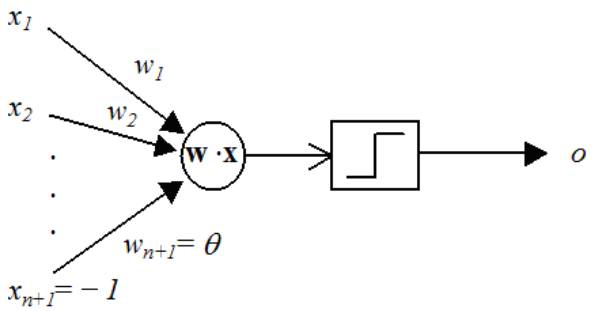
Obr. 15: Neurón a jeho základné časti

- Dendrity reprezentujú miesto vstupu signálu do tela neurónu.
- Telo bunky sčíta signály, ktoré vstupujú do neurónu. Takto stanovený vnútorný potenciál vedie k vybudeniu neurónu.
- Axonové vlákno prenáša signál daný stupňom vybudenia k synapsiám
- Synapsia tvorí výstupné zariadenie neurónu, ktoré signál zosilňujú alebo zoslabujú a predávajú ďalším neurónom.

Pre modelovanie umelých neurónových sietí vznikol model ktorý simuluje biologické vlastnosti neurónu. Pre takýto matematický model zavádzame pojem Perceptron.

Perceptron

Dodnes patrí medzi najdôležitejšie modely. Prijma vstupné signály $\bar{x} = (x_1, x_2, x_3, .., x_{n+1})$ cez synaptické váhy tvoriace váhový vektor $\bar{w} = (w_1, w_2, w_3, .., w_{n+1})$. Vstupný vektor $\bar{x}$ sa nazýva vzor alebo obrazec. Zložky vstupného vektora môžu nadobúdať reálne alebo binárne hodnoty. Príkladom môže byť obrázok (písmeno, odtlačok prsta atď.) zakódovaný ako pole (vektor) hodnôt. Zložky váhového vektora sú reálne čísla.



Obr. 16: Matematický model perceptrónu

Výstup perceptrónu *o* je daný vzťahom:

$$o = f(net) = f(\bar{w} \cdot \bar{x}) = f\left(\sum_{j=1}^{n+1} w_j x_j\right) = f\left(\sum_{j=1}^{n} w_j x_j - \theta\right)$$

kde premenná net označuje váhovanú sumu vstupov, t.j. skalárny (zložkový) súčin váhového a vstupného vektora. Funkcia f sa volá **aktivačná funkcia** perceptrónu. V tejto notácií predpokladáme, že perceptrón má *n + 1* vstupov. Hodnota(n + 1)-tého vstupu je vždy -1 a $w_{n+1} = \theta$, čo je hodnota prahu excitácie perceptrónu (angl. threshold). V roku 1958 Rosenblatt zaviedol diskrétny perceptrón s bipolárnou binárnou aktivačnou funkciou (funkcia signum, znamienko):

$$f(net) = sign(net) = \begin{cases} +1 & net \geq 0 \Leftrightarrow \sum_{j=1}^{n} w_j x_j \geq \theta \\ -1 & net < 0 \Leftrightarrow \sum_{j=1}^{n} w_j x_j < \theta \end{cases}$$

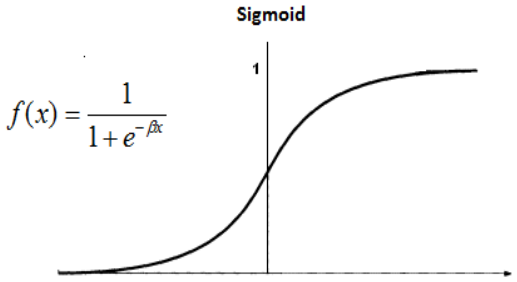
Rovnica:

$$\sum_{j=1}^{n} w_j x_j - \theta = 0$$

je rovnicou nadroviny v n -rozmernom priestore. Napr. v 2-rozmernom priestore je to rovnica priamky: $w_1 x_1 + w_2 x_2 - \theta = 0$. Váhy perceptrónu predstavujú koeficienty deliacej priamky (nadroviny), ktorá rozdeľuje priestor vzorov na dva podpriestory. Perceptrón teda dokáže klasifikovať, t.j. zatriedʼovať vzory, do dvoch tried, ktoré sú lineárne oddelené deliacou hranicou. Inými slovami, perceptrón dokáže riešiť iba tzv. lineárne

separovateľné problémy.

Spojitý perceptrón



Obr. 17: Sigmoid, aktivačná funkcia spojitého perceptrónu

Ako bolo spomenuté pri popise obecného perceptrónu, hodnoty vybudenia neurónu sú dané tzv. aktivačnou funkciou neurónu. Na obrázku 17 má funkcia tvar sigmoidy. Aktivačná funkcia môže byť ľubovolná funkcia pre ktorú môžeme nájsť jej deriváciu. Pretože sigmoida sa najviac podobá reálnemu vybudeniu neuóna, je jednou z najčastejších aktivačných funkcií využívaných v neurónových sieťach. Formálne je možné túto funkciu definová nasledujúcim spôsobom:

$$y = S(z) = \frac{1}{1 + e^{-\lambda z}}$$

$$z = \sum_{i=0}^{n} w_i x_i$$

S(z) - Aktivačný funkcia
z - vnútorný potenciál neurónu
λ - strmost sigmoidu

Z priebehu sigmoidu môžeme usudzovať nasledujúce tvrdenia:

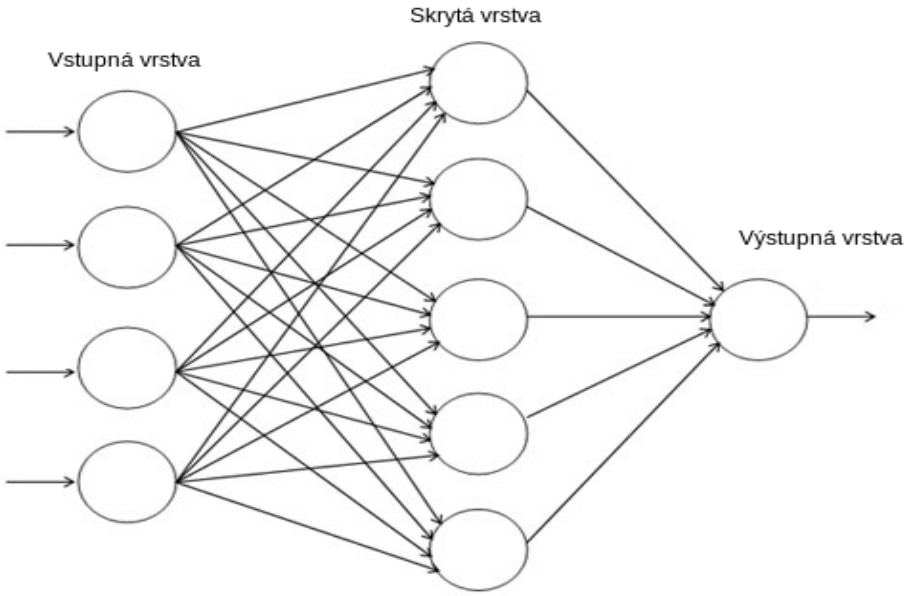
1. Vybudenie neurónu sa pohybuje v rozmedzí od 0 po 1, kde 1 znamená plné vybudenie a 0 úplny útlm.
2. V prípade ze sa vnútorný potenciál blíži k hodnote $+\infty$, potom dochádza k plnému vybudeniu $x = (+\infty) = 1$
3. V prípade ze sa vnútorný potenciál blíži k hodnote $-\infty$, potom dochádza k úplnemu útlmu $x = (-\infty) = 0$

Až správnym prepojením niekoľko perceptronom možno vytvoriť umelú neurónovú sieť, ktorá je schopná simulovať malú časť ľudského mozgu. Existuje niekoľko prístupov pri vytváraní a zoskupovaní perceptronov do umelej siete. Každý z týchto modelov

sa svojimi vlastnosťami hodí na riešenie špecifických problémov. Jedným zo základných a najjednoduchších princípov spájanie neurónov je Hopfield NN. Ide o sieť kedy každý neurón danej siete je medzi sebou prepojený. Takáto sieť sa napríklad používa na rozpoznanie veľmi malej skupiny objektov ako napríklad písmen v abecede. Jedným zo široko využívaných modelov neurónových sietí sú viacvrstvové dopredné neurónové siete.

2.5.1 Viacvrstvová dopredná neurónová sieť

Väčšina reálnych problémov má nelineárny charakter. To znamená, že sa nedajú vyriešiť sčítaním lineárnych hraníc. Problém s obmedzenou výpočtovou schopnosťou perceptrónov sa vyriešil až v roku 1986, keď Rumelhart, Hinton a Williams (1986) zaviedli pravidlo trénovania nazvané metóda spätného šírenia chýb (angl. error backpropagation) pre dopredné neurónové siete so skrytými neurónmi. Tzv. viacvrstvové dopredné umelé neurónové siete (angl. multilayer feed- forward ANN), ktoré sa trénujú týmto pravidlom sú schopné riešiť aj nelineárne problémy. Dopredné neurónové siete sa vyznačujú tým, že v nich existujú iba dopredné spojenia medzi neurónmi. Každý neurón jednej vrstvy vysiela signály na každý neurón nasledujúcej vrstvy. Spojenia do predchádzajúcej vrstvy ani v rámci jednej vrstvy neexistujú.[12] Obrázok 18 znázorňuje schému doprednej neurónovej siete, ktorá obsahuje vstupnú vrstvu, výstupnú vrstvu a jednu skrytú vnútornú vrstvu. Čo je však pre sieť kľúčové je správne určenie všetkých synoptických váh Najpoužívanejší typ umelej neurónovej siete je tvorený minimálne z 3 vrstiev. Pr-



Obr. 18: Schéma doprednej neurónovej siete

vou z nich je vstupná vrstva, ktorá prijíma vstupný vektor. Druhou vrstvou je skrytá vrstva, ktorá spája vstupnú a výstupnú vrstvu. Teoreticky môže takýto typ siete obsahovať ľubovolný počet skrytých vrstiev. Pri definovaní vnútornej vrstvy siete je potrebné vyriešiť 2 základné otázky. Prvou z nich je počet skrytých vrstiev a druhou z nich je počet skrytých neurónov v danej vrstve. Z matematického hľadiska môže sieť z dvomi skrytými vrstvami simulovať funkciu ľubovolného tvaru. Momentálne neexistuje dôvod prečo by sme mali vytvárať siete s viac ako jednou skrytou vrstvou. Pri definovaní počtu neurónov skrytej vrstvy sa pre dosiahnutie najoptimálnejšej siete riadime základnými pravidlami:

- počet neurónov by mal byt medzi počtom vstupných a výstupných neurónov
- počet neurónov skrytej vrstvy by mal byť 2/3 veľkosti vstupných neurónov plus počet neurónov výstupnej vrstvy
- počet neurónov skrytej vrstvy by mal byt menší ako dvojnásobok neurónov vstupnej vrstvy

Pokiaľ zvolíme príliš málo alebo príliš veľa neurónov skrytej vrstvy, môže to mať za následok výrazné zvýšenie času, ktorý je potrebný na naučenie siete alebo to taktiež môže vieš ku nekorektným výsledok a vysokej celkovej chybe neurónovej siete. Poslednou vrstvou je výstupná vrstva, ktorá už obsahuje reálny výsledok vybudenia umelej neurónovej siete. Medzi jednotlivými vrstvami sa nachádza tzv. úplné prepojenie neurónov. Ide o prepojenie kedy každý neurón nižšej vrstvy je prepojený so všetkými neurónmi vyššej vrstvy. V takejto sieti smeruje šírenie signálu od vstupnej vrstvy smerom k výstupnej vrstve a nazýva sa dopredné šírenie. Má nasledujúci priebeh:

1. Neurónová sieť príjme skúmaný vstupný vektor. Tento vektor je ohodnotený neurónmi vstupnej vrstvy a na základe aktivačnej funkcie je signál z každého vstupného neurónu šírený do všetkých neurónov vyššej vrstvy, ktorou je v našom prípade jedna skrytá vrstva.
2. Každý neurón vyššej vrstvy uskutoční sumarizáciu všetkých vstupných signálov a na základe aktivačnej funkcie je vybudený odpovedajúcou hodnotou. Túto hodnotu ďalej predáva do vyšších vrstiev.
3. Tento proces sa opakuje až do výstupnej vrstvy, ktorá zobrazí výsledné hodnoty spracované umelou neurónovou sieťou.

Takýmto spôsobom sme dostali výsledné hodnoty, ktoré spracovala umelá neurónová sieť. Presne takýmto spôsobom sú spracované signály v našom mozgu. Napríklad ak vidíme červené svetlo na križovatke. Tento signál je najprv spracovaný ľudským okom a vo forme elektrických signálov je následne spracovaný veľkým množstvom neurónov až vo výsledku mozog rozpozná že ide o červené svetlo na semafore.

2.5.2 Proces učenia

Jedným z hlavných problémov pri vytváraní umelých neurónových sieti je proces korektného stanovenia synaptických váh pre všetky väzby neurónovej siete. Tento proces je spojený s pojmom učenia alebo taktiež adaptácie siete. Pokiaľ je umelá neurónová sieť správne naučená a teda synaptické váhy odpovedajú korektným výstupom sieť má schopnosť generalizácie čo je v podstate schopnosť odhadovať javy, ktoré neboli súčasťou učenia. Proces učenia je kľúčový proces, ktorý vedie k použiteľnosti neurónových sietí pri riešení nelineárnych problémov. Pri procese učenia NN pracujeme s tzv. trénovacou množinou, kto reprezentuje riešenú problematiku vo forme vyriešených príkladov a procesom kedy túto trénovaciu množinu transformujeme do korektného nastavenia synaptických váh. Trénovaciu množinu môžeme reprezentovať ako usporiadanu množinu dvojíc vektorov. Prvý prvok dvojice reprezentuje vstupné vlastnosti riešeného príkladu. Druhý prvok dvojice reprezentuje správne výstupné vlastnosti riešeného problému. Ide vlastne o nelineárnu transformáciu. Metóda, ktorá umožňuje adaptáciu neurónovej siete nad danou trénovacou množinou sa nazýva *backpropagation*, čo môžeme preložiť ako metóda spätného šírenia. Ako názov tejto metódy napovedá, táto metóda pracuje opačným smerom akým je smer spracovania doprednej neurónovej siete. Ide o proces kedy sa snažíme nájsť minimálnu chybu siete a na základe tejto chyby upraviť synaptické váhy jednotlivých spojov neurónovej siete nasledovným spôsobom:

1. Postupne spracujeme každý vektor i-tého prvku trénovacej množiny. Tento vektor privedieme na vstup neurónovej siete a následne šírime vybudenia vstupnej vrstvy až po výstupnú vrstvu.

2. Porovnáme získané hodnoty s hodnotami i-tého prvku trénovacej množiny s hodnotami, ktoré sme získali s reálneho vybudenia výstupnej vrstvy.

3. Rozdiel medzi skutočnými hodnotami a požadovanými definuje chybu siete. Takto vypočítanú chybu následne v určitom pomere vnášame do synaptických váh siete. Snažíme sa docieliť toho aby pri ďalšom prvku trenovacej množiny bola táto výsledna chyba siete čo najmenšia.

4. Pokiaľ sme použili všetky prvky trenovacej množiny, je potrebné vyhodnotiť celkovú chybu vytvorenej neurónovej siete. Pokiaľ je táto chyba väčšia ako požadujeme, proces učenia s využitím trenovacej množiny opakujeme. Tu je potrebné podotknúť, že pri riešení neurónových sietí s vysokým počtom vstupných hodnôt je často veľmi náročné získať relatívne malú celkovú chybu neurónovej siete. Na celkovú chybu a čas potrebný na správne naučenie siete má taktiež veľký vplyv správne a relevantné hodnoty z trénovacej množiny. Preto pri využívaní zložitých sietí musíme pri záveroch vždy rátať s určitou chybou.

2.6 Odhad zložitosti a úsilia

V nasledujúcej kapitole sa pokúsim navrhnúť model odhadu zložitosti a úsilia podnikového procesu na základe modelu vytvoreného pomocou diagramu aktivít UML s využitím umelých neurónových sietí.

Pri obecnom odhade zložitosti a úsilia činnosti ide o proces, kedy sa snažíme na základe určitých vstupných vlastností odhadnúť hodnoty a vlastnosti, ktoré by nám napríklad povedali aký čas je nutné stráviť za oberaním sa touto činnosťou aby sme ju úspešne dokončili. S takýmto typom odhadov sa obecne stretávame v mnoho odvetviach a teda nielen pri odhade zložitosti podnikových alebo softvérových projektov. Odhady môžeme nájsť v stavebníctve, vede, medicíne. Pri odhadoch môžeme skúmať rôzne výsledné vlastnosti takýchto odhadov. Takýmito vlastnosťami môžu byť celkový čas potrebný na dokončenie danej činnosti. Objem obecných zdrojov, ktoré bude potrebné spotrebovať na dokončenie alebo môže ísť o celkové úsilie, ktoré môže zahŕňať všetky aspekty pri vyhodnocovaní a vykonávaní.

Pri realizácií a vytváraní podnikových procesov je pri návrhu kľúčovým aspektom správny návrh, ktorý je možné znázorniť pomocou modelu BP. Takýto model je kľúčový pri implementácií a taktiež pri orientácií v podnikovom procese. Pri samotnej implementácií v dnešnej dobe je využívané veľké množstvo podporných softvérových nástrojov, ktoré umožňujú zautomatizovať vykonávanie podnikového procesu na základe spracovania daného modelu BP. Odhad zložitosti s využitím zautomatizovaných nástrojov pre spracovanie modelu vyžaduje rozdielny prístup pretože takéto procesy sú hlavne založené na modeli BP a teda odhad zložitosti sa zameriava hlavne na tento model. V predchádzajúcej časti som definoval niekoľko metrík, ktoré sú založené na skúmaní vlastností modelu podnikového procesu znázornené ho UML diagramom aktivít. Ide v podstate o popis vlastností modelu podnikového procesu, ktoré môžu byť následne využité a spracované pri ďalšej analýze.

Existuje niekoľko metód, ktoré boli alebo sú využívané na odhad zložitosti či už softvérových alebo podnikových procesov. Pretože softvérový proces je špecifický typ podnikového procesu, v ďalšom texte budem taktiež uvádzať závery prevzaté z výskumov, ktoré boli vykonávané na softvérových procesoch. Metódy na odhad zložitosti a úsilia môžeme rozdeliť do niekoľko základných kategórií:

- **Algoritmické** – Tieto metódy určujú celkové úsilie na implementáciu pomocou expertných algoritmov, ktoré vychádzajú zo skúseností a znalostí nadobudnutých počas dlhoročných implementácií softvérových alebo podnikových procesov.
- **Štatistické** – Pri týchto metódach ide o vytvorenie štatistického modelu. Na vytvorenie modelu potrebuje vhodný súbor nadobudnutých dát, z ktorých štatistickými metódami získame informácie, ktoré z vlastností modelu majú význam pri určovaní zložitosti, a ktoré môžeme z analýzy vyškrtnúť. Po takto vytriedenom dátovom súbore je možné vytvoriť rovnicu, ktorá popisuje vlastnosti medzi závislými premennými dátového súboru a výsledným odhadom zložitosti a úsilia pri implementácií.

- **Expertné** – Ide o skupinu, ktorá využíva techniky akými sú umelé neurónové siete, generické algoritmy a techniky, ktoré nie je možné zaradiť do algoritmickýh metód.

Výber správnej metódy je jednou z kľúčových aspektov pre dosiahnutie hodnôt, ktoré sa čo najvernejšie môžu približovať realite.

2.6.1 Porovnanie metód

Existuje niekoľko porovnaní, ktoré sa snažia na určitej vzorke dátového súboru porovnať skupiny spomenutých metód využívaných pri odhadoch. Taktiež existujú niektoré hybridné metódy, ktoré spájajú prvky z algoritmických metód a využívajú ich pri implementácií neurónových sietí. Odhadom úsilia v podnikových procesoch sa venuje [6], ktorý použil niekoľko metrík typicky použitých pri analýze zložitosti modelu podnikového procesu a využil regresnú analýzu na zistenie závislých a nezávislých premenných. Dospel k záverom kedy bolo možné časť metrík vynechať z ďalšej analýzi na získanie najlepších výsledkov. Pri vytváraní tohto modelu využíval štatistické metódy. Anupama Kaushik [18] vytvoril predikčnú umelú neurónovú sieť a pri učení tejto siete využil výstupy z algoritmickej metódy COCOMO, ktorá je jednou z najpoužívanejších na určenie odhadov pre softvérové projekty. Výsledky boli uspokojivé a potvrdili, že umelá neurónová sieť môže slúžiť ako predikčný model odhadu úsilia. K podobným výsledok sa taktiež dopracoval Ali Idri, Taghi M. Khoshgoftaar, Alain Abran [16], ktorý využil rovnakú metodiku ako [18]. Výsledky porovnania umelých neurónových sietí, regresného modelu a COCOMO algoritmického prístupu [17] pri odhade úsilia poukázali na to že umelé neurónové siete nijako nezaostávali za ostatnými metódami. Mamoona Humayun and Cui Gang [15] pri porovnávaní zhodnotil aplikovanie umelých neurónových sietí za uspokojivé. Taktiež poukázal na to, že žiadna z metód výrazne nezaostáva. Podotkol taktiež že vytváranie odhadov zložitosti pre softvérový proces je zložitý proces pretože rýchlosť vývoja závisí od veľkého počtu faktorov. Obecne môžeme tento záver aplikovať aj na odhady podnikových procesov. Kľúčový fakt, ktorý je potrebné zdôrazniť pri využívaní umelých n. sietí je mať použiteľnú množinu dát, ktorá je kľúčová pre správne naučenie siete. Podobnému porovnávaniu sa taktiež venujú [19],[20]. Hlavné výhody využitia umelých neurónových sietí ako nástroja na predikčné odhady zložitosti môžeme zhrnúť do niekoľko hlavných bodov:

- Schopnosť naučenia sa vykonávať určitú úlohu na základe vzoriek daného problému.
- Samostatne organizovateľné. Ide o schopnosť neurónových sietí vytvoriť svoju vlastnú štruktúru, ktorá odpovedá riešenému problému. Toto vytváranie sa deje počas fáze učenia sa.
- Pracujú v reálnom čase. Neurónové siete je možné vytvárať v paralelnom prostredí a tak dosiahnuť rýchlejšiu odozvu. V dnešnej dobre taktiež existujú špecifické hardvérové nástroje umožňujúce simulovanie takýchto sietí veľmi rýchlo.

- Univerzálny aproximátor. Neurónová sieť funguje ako aproximátor spojitej funkcie.

Použitie umelých neurónových sietí zo sebou taktiež prináša niektoré nevýhody:

- Jednou z vlastností pre ktoré boli kritizované bol fakt, že môžu byť užitočné pri predpovediach. Problém však je, že nedokážu porozumieť modelu predpovedi.
- V starších implementáciách boli NN brané ako čierna skrinka bez možnosti presnej definície takéhoto predikčného modelu. S novými nástrojmi sa táto nevýhoda stráca.
- Neurónové siete sú náchylné k pretrénovaniu. Ak vytvoríme sieť s veľkou schopnosťou učenia a sieť naučíme malou množinou dát, môže to viesť k nesprávnym výsledkom kedy nie je možné z naučených vzorkou generalizovať riešený problém.
- Rýchlosť učenia pri veľkých modeloch neurónových sietí pre dosiahnutie ideálnej celkovej chyby siete môže byť často časovo náročné.

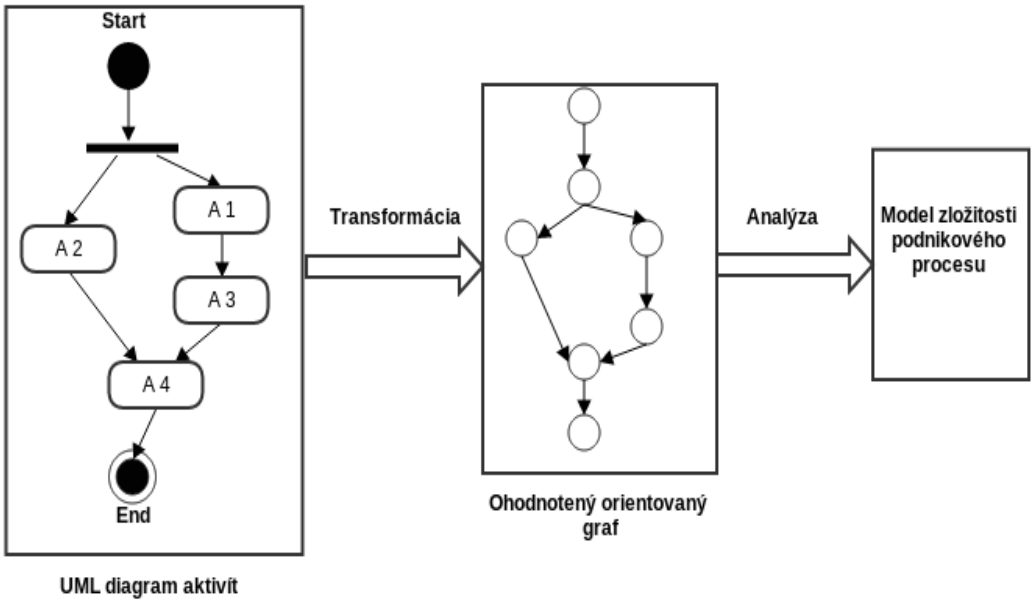
2.6.2 Výber metódy odhadov

Ako bolo spomenuté v kapitol 2.6.1, existuje niekoľko prístupov ako je možné implementovať predikčný algoritmus na výpočet odhadu úsilia potrebného pre vytvorenie a implementáciu podnikového procesu na základe stanoveného modelu. Preto môžeme vysloviť hypotézu.

Definice 2.2 *Odhad úsilia a zložitosti pri implementácií podnikových procesov s využitím umelých neurónových sietí a grafovo orientovaných metrík môže priniesť pozitívne výsledky, ktoré môžu mať za následok zníženie chybovosti odhadu pokiaľ ide o porovnanie s realitou, čo je prakticky kľúčový parameter pri porovnávaní metód, ktoré sú úzko spojené s odhadom úsilia.*

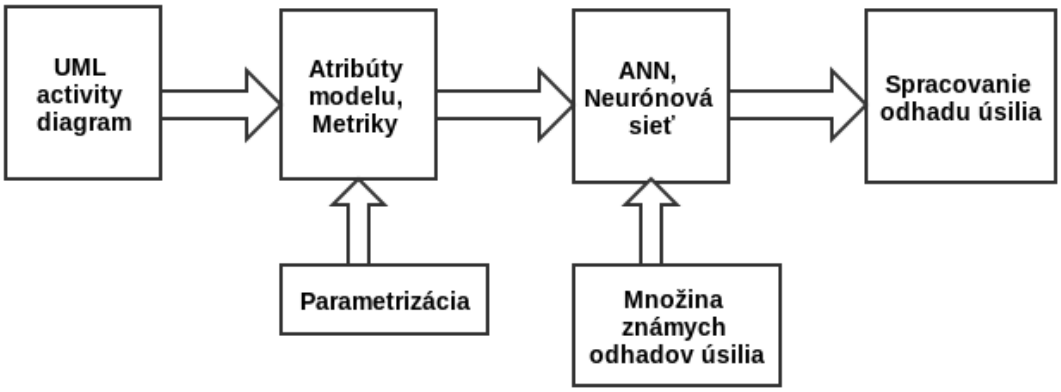
Ako bolo spomenuté v predchádzajúcich kapitolách, hlavným vstupom do programu pred odhad úsilia bude model podnikového procesu s využitím UML diagramu aktivít. Aby bolo možné zautomatizovať odhady zložitosti na základe modelu podnikového procesu vytvoreného prostredníctvom diagramu aktivít, je potreba matematicky vyjadriť tie vlastnosti modelu, ktoré by mohli mať zásadný vplyv na ohodnocovanie odhadov. Preto je potreba transformovať model BP do vhodnej dátovej štruktúry, ktorá je ideálna na spracovanie a analýzu. Tento proces transformácie môžeme rozdeliť do troch základých krokov:

1. Prvým krokom je vytvorenie dátovej štruktúry pre následné spracovanie. Touto štruktúrou je v našom prípade špeciálny typ orientovaného grafu.
2. Aplikovanie kritérií nad grafovou štruktúrou a vytvorenie množiny vstupov, ktoré vyjadrujú dôležité vlastnosti modelu BP prostredníctvom reálnych čísel.
3. Aplikovanie týchto vlastností nad umelo vytvorenou neurónovou sieťou s cieľom aplikovania nadobudnutých znalostí z už známych a presných odhadov zložitosti.



Obr. 19: Proces transformácie a vytvorenia modelu zložitosti podnikového procesu

Pri definovaní modelu pre odhad zložitosti a úsilia započnem analýzu výberom vhodných metrík, ktoré je možné získať z modelu podnikového procesu vytvoreného diagramom aktivít UML. Väčšina metrík, ktoré boli prezentované v predchádzajúcej kapitole sú založené na grafovo orientovanej analýze, kedy sa snažíme matematicky vyjadriť určitú vlastnosť modelu, ktorá môže prispieť k lepšiemu odhadu komplexnosti modelu. Tieto metriky vznikajú transformáciou modelu diagramu aktivít na orientovaný graf. Časť metrík preto nepotrebuje inú parametrizácia. Po transformácií modelu vieme okamžite rozpoznať hodnoty týchto metrík. Druhou skupinou sú metriky, ktoré potrebujú okrem modelu podnikového procesu d'alšiu parametrizácia. Príkladom môže byť počet aktérov, ktorí zasahujú do podnikového procesu. Proces spracovania modelu do použiteľného formátu skupiny metrík, ktoré následne môže spracovať umelá neurónová sieť znázorňuje obrázok20. Z predchádzajúcej kapitole sme zistili, že pre vyjadrenie komplexnosti podnikového procesu existuje viac ako 12 metrík, ktoré vyjadrujú rôzne vlastnosti procesu. Pri analýze týchto metrík bolo zrejmé, že niektoré metriky sa na seba značne podobajú a teda vyjadrujú podobné alebo skoro totožné vlastnosti podnikového procesu. Je potrebné analyzovať metriky z pohľadu efektívneho vyjadrenia komplexnosti a schopnosti metriky vyjadriť zložitosť a tým pádom aj časovú náročnosť. Táto analýza je potrebná kvôli definovaniu výslednej skupiny vlastností, ktoré budú vstupovať ako parametre do neurónovej siete. Pre potreby učenia siete by sme mohli posielať prakticky všetky spomenuté metriky. Malo by to však za následok spomalenie učenia a taktiež by bol potrebný väčší počet prvkov trénovacej množiny.



Obr. 20: Schématické znázornenie modelu odhadu úsilia s využitím ANN

2.7 Výber metrík

Po popise metrík, ktoré môžu byť použité v implementácií je potrebné vybrať tie, ktoré budú vhodné na aplikovanie a využitie ako vstup do ANN. V nasledujúcej kapitole vyberám vhodné metriky a pridávam taktiež doplnkové metriky, ktoré môžu slúžiť pre lepšiu parametrizíciu odhadu.

2.7.1 Príklad spracovania

Obrázok 21 znázorňuje reálny podnikový proces a jeho transformáciu do orientovaného grafu. Každý z vrcholov grafu predstavuje element procesu. Model rozdeľuje tieto typy vrcholov: START, END, ACTIVITY, OBJECT, OR_SPLIT, OR_JOIN, AND_SPLIT, and_JOIN, XOR_SPLIT, XOR_JOIN. Ako finálnu skupinu parametrov pre vstup programu som vybral nasledovnú skupinu metrík.

- **Size** - Počet všetkých vrcholov grafu. Ide o jednu z najpoužívanejších metrík

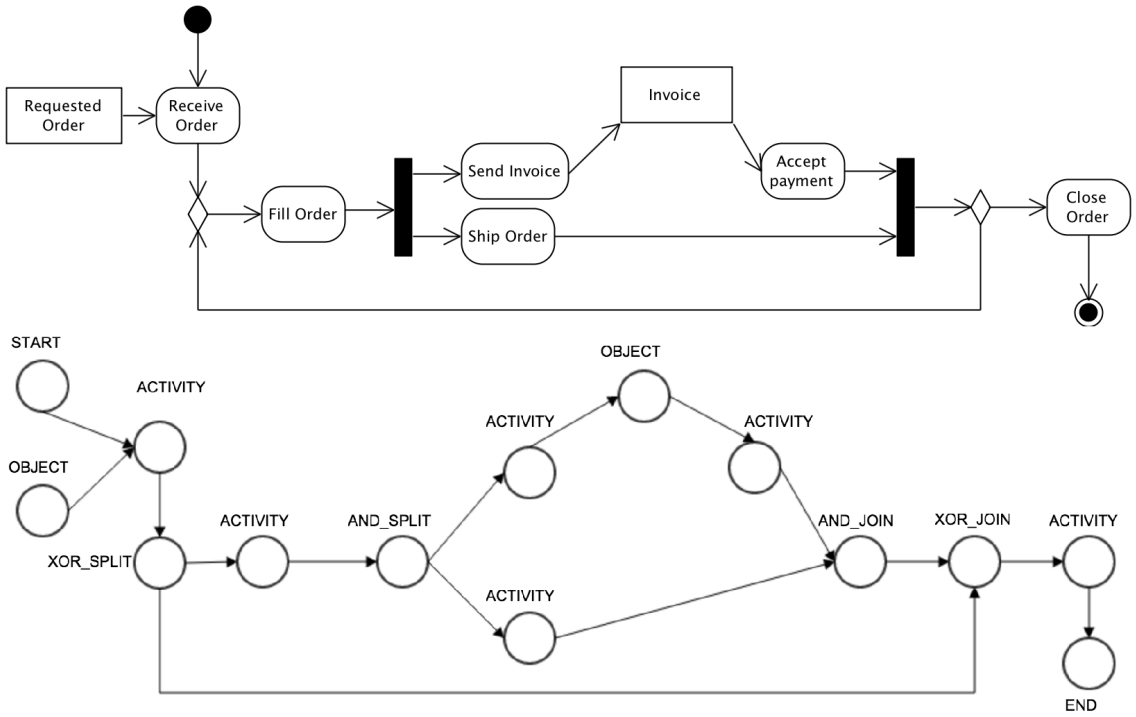
$$S_N(G) = 14$$

- **Density** – Vyjadruje hustotu prepojení medzi jednotlivými vrcholmi grafu.

$$D_{\triangle}(G) = \frac{|15|}{|14| \cdot (|13| - 1)} = 0.0824$$

- **CNC** – Ide o podobnú vlastnosť ktorú vyjadruje hustota

$$CNC(G) = \frac{|15|}{|14|} = 1.0714$$



Obr. 21: Transformácia BP do orientovaného grafu

- **Avarage degree** – Priemer prepojenia konektorov. Z obrázka vieme zistiť, že process obsahuje 4 rozdeľovacie vrcholy, z ktorých každý spája alebo rozdeľuje 2 toky. Po dosadení nam priemer prepojenia konektorov vychádza ako:

$$DC(G) = \frac{|8|}{|4|} = 2$$

- **Max Avarage degree** – Maximálna hodnota prepojenia konektorov je v našom testovacom modeli rovná hodnote 2

- **Separability** – Oddeliteľnosť v skratke hovorí o takých vrcholoch grafu, ktoré keď odstránime získame dva samostatné grafy, a teda také grafy, ktoré nie sú medzi sebou prepojené hranou. Z príkladu vyplýva že articulation point sú elementy: XOR_SPLIT, XOR_JOIN, Receive Order a Close Order.

$$\Pi(G) = \frac{|4|}{|4| - 2} = 2$$

- **Sequentiality** – Sekvenčnosť diagramu vyjadruje pomer hrán, ktoré sa nachádzajú v sekvencií elementov ako ACTIVITY alebo OBJECT voči všetkým hranám grafu.

Hrany, ktoré vyhovujú pravidlu sú: START -> ACTIVITY(Receive Order), OBJECT(Requested Order) -> ACTIVITY(Receive order), ACTIVITY (Send Invoice) -> OBJECT (Invoice), OBJECT(Invoice) -> ACTIVITY(Accept Payment), ACTIVITY(Close Order) -> END.

$$S_q(G) = \frac{|5|}{|15|} = 0.33333$$

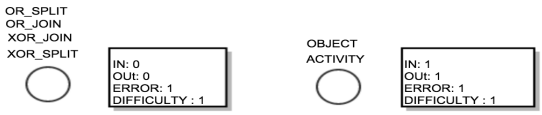
- **Deep** – Vyjadruje hĺbku rozvetvenia rozhodovacích blokov. V príklade je najhblšíe zanorenie úrovne 2. Prvý rozhodovací blok OR pridáva úroveň jedna a prvý AND blok pridáva úroveň 2.
- **Control Flow Complexity** – Ide o jednu z dôležitých metrík. Vyjadruje počet možných ciest BP. Skúmaný proces obsahuje jeden rozhodovací XOR element a teda výsledný počet ciest je rovný 2.

$$CFC(G) = 2$$

- **Cyklicita** – Vyjadruje pomer elementov, nachádzajúcich sa v cykle. Z pohľadu cyklicity graf neobsahuje žiadne cykly preto je hodnota tejto metriky rovná 0.
- **Role size** – Počet rolí, aktérov. V našom príklade existuje iba jedna rola a teda hodnota metriky sa rovná 1.

Aby bolo možné taktiež špecifikovať komplexnosť konkrétnych elementov BP, rozhodol som sa pridať pre každý element BP doplnkové metriky, ktoré môžu byť parametrizované. Obrázok 22 znázorňuje pridelenie hodnôt doplnkových metrík pre jednotlivé elementy.

- **Error propability** – Výskyt chyby elementu. Je možné ho parametrizovať v rozmedzí od 1-10.
- **Element complexity** – Vyjadrenie komplexnosti elementu v rozmedzí 1-10.
- **IN Object** - Počet vstupných objektov elementu. Túto metriku je možné špecifikovať v ľubovolnom rozsahu
- **OUT Object** – Počet výstupných objektov elementu. Možná parametrizácia v ľubovolnom rozsahu.



Obr. 22: Doplnkové metriky

2.7.2 Požiadavky

Aby bolo možné analyzovať podnikový proces pomocou metrík čo najideálnejšie. Je potrebné aby tvorca podnikovhé procesu dodržal nieľko zásad.

1. Je potrebné aby dodržal rovnakú mieru granurality všetkých aktivít a akcií podnikovhé procesu.

2. Pokiaľ je aktivita rozsiahla. Je potrebné zvážiť premenu tejto aktivity na samostatný proces a tento proces následne integrovať do pôvodného procesu. Program dokáže spracovať podproces, ktorý je súčasťou aktivity pôvodného procesu.

3. Pre rozíšenie zložitosti a komplexnosti je možné parametrizovať každý jeden element procesu s využitím metrík ako je error rate a difficulty.

4. Proces by mal byť dobre štrukturovaný a nemal by obsahovať nedosiahnuteľné vetvy. Nekorektne navrhnutý proces bude mať tendenciu obsahovať viac chýb a v konečnom dôsledku by mohol ovplyvniť výsledny odhad času a finančných prostriedkov.

]=

2.8 Návrh ANN

Ďalším podstaným aspektom, ktorý je potrebné vyriešiť pri návrhu modelu odhadu zložitosti a úsilia je stanovenie výsledných vlastností, ktorými budeme vyjadrovať výsledky odhadov. Existuje niekoľko atribútov, ktorými je možné merať celkovú náročnosť a zložitosti BP. Ja som sa rozhodol zachytiť nasledovné atribúty:

1. **Overall price (Celková cena)** - Ide o atribút, ktorým vyjadrujeme celkovú cenu podnikového procesu. Zahrňuje všetky náklady, ktoré boli vynaložené počas analýzy a implentácie podnikovhé procesu. Prevačná časť tejto hodnoty zahrňuje celkovú cenu práce, ktorá bola spotrebovaná na vytvorenie podnikovhé procesu. Taktiež môže obsahovať sekundárne položky ako licencie softvéru využitého pri analýze a vývoji a podobné výdavky, ktoré úzko súvisia s vývojom podnikových procesov.

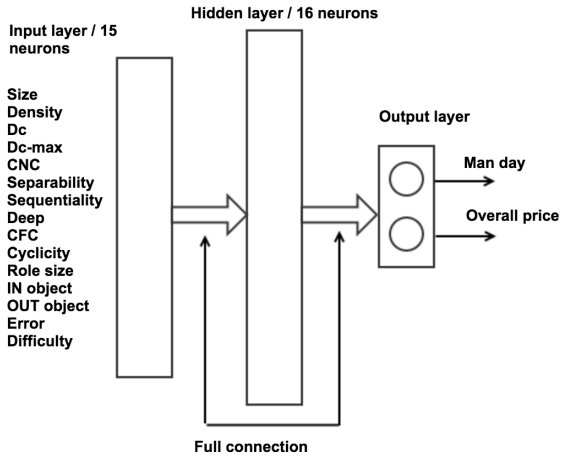
2. **Man-day (Jednotkový čas práce)** - Ide o pojem, ktorým sa vyjadruje časová náročnosť napríklad softvérového projektu z pohľadu obsadenia ľudí a celkového času stráveného na projekte. Ide teda o počet dní, ktoré daný vývojár strávi na danom projekte. Pôjde o reálne číslo.

Poslednou časťou, ktorá je kľúčová pri implementácií modelu odhadu úsilia je správny návrh umelej neurónovej siete. Ako znázorňuje obrázok 23 neurónová sieť je navrhnutá nasledovným spôsobom:

1. **Vstupná vrstva** - Je tvorená 15 neurónmi a ako aktivačnú funkciu som použil funkciu sigmoid. Každý neurón bude spracovávať hodnoty 15 spomenutých metrík, ktoré som určil ako smerodajne pre určenie ideálneho odhadu úsilia.

2. **Skrytá vrstva** - Pri návrhu ANN je jednou z najkontroverznejších otázok práve počet neurónov vnútornej vrstvy. Príliš málo neurónov môže mať za následok nekorektné naučenie a generalizáciu skúmaného problému a taktiež príliš veľa neurónov skrytej vrstvy môže mať za následok pretrénovanie siete a nemožnosť dostať celkovú chybu siete pod nami stanovenú hodnotu. V práci teda vytvorím niekoľko neurónových sietí a pokúsim sa nájsť najvhodnejší model siete. Testom ANN sa venujem v kapitole 4. Na začiatok zvolím počet vnútorných neurónov o jeden viac ako je počet neurónov vstupnej vrstvy. Postupne vytvorím niekoľko ANN a vyberie najvhodnejšie rozdelenie ANN. Ako aktivačnú funkciu vnútornej siete som zvolil funciu sigmoid.

3. **Výstupná vrstva** - Obsahuje 2 neuróny, z ktorých každý reprezentuje skumáné vlastnosti odhadu zložitosti. Výstupné neuróny Man day a Overall prices využívajú ako aktivačnú funkciu sigmoid.



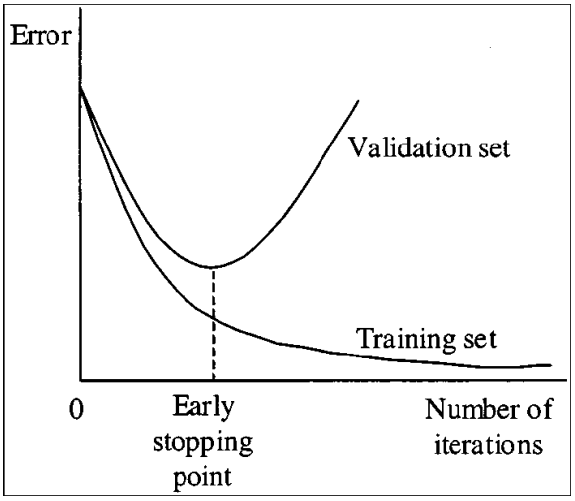
Obr. 23: Návrh neurónovej siete pre odhad úsilia podnikového procesu

2.8.1 Trénovacia množina

V predchádzajúcom texte som popísal vstupné parametre neurónovej siete a taktiež som popísal vnútornú štruktúru navrhnutej umelej neurónovej siete. Je potrebné určiť trénovaciu množinu. Pri riešení problémov s využitím umelých neurónových sietí je jednou z kľúčových predispozícii pre korektné a použiteľné výsledky vhodná trénovacia množina dát. Hlavnou úlohou trénovania siete je nájsť najvhodnejšie vnútorné nastavenia siete pre čo nelepšie výsledky. Tento krok zväčša rozhoduje ako dobre bude nami vytvorená sieť generalizovať skúmaný problém. Generalizácia je meranie či sa daná sieť správa korektné pri aplikovaní vzorky dát, ktorá nebola súčasťou trénovacej množiny.

Generalizácia je v podstate používaná na rozhodovanie či si sieť zapamätala vstupné dáta korektne. Pokiaľ chceme usudzovať na javy alebo vlastnosti, ktoré nie sú súčasťou vytvorenej množiny dát, je potrebné aby dáta reálne odzrkadľovali skúmaný problém a čo najvernejšie popisovali všetky možné skúmané vlastnosti. Ide o jeden z kľúčových aspektov pri implementácií umelých neurónových sietí s ohľadom na ich výslednú chybovosť. Nekorektné spracovaná trénovacia množina môže so sebou priniesť taktiež zdĺhavé učenie umelej neurónovej siete. V niektorých prípadoch to môže viesť taktiež k nemožnosti dostať výslednú chybovosť siete pod určitú nami požadovanú hodnotu. Pri vytváraní a validácií trénovacej množiny môžeme rozdeliť trénovacie vzorky do 3 kategórií:

- **Trénovacia množina (Training set)**- Ide o najdôležitejšou množinu a slúži na naučenie neurónovej siete nami skúmanými hodnotami. Hlavnou úlohou je korektné nastavenie váh všetkých neurónov neurónovej siete.
- **Validačná množina (Validation set)** - Ide o trénovaciu množinu pri ktorej sa snažíme predísť neželanému efektu „overfitting“ neurónovej siete. Nie je určená na upravenie vnútorných váh neurónov. Pri tejto množine sa snažíme overiť či akékoľvek zvýšenie presnosti trénovacej množiny má za následok zvýšenie celkovej



Obr. 24: Graf procesu učenia a zobrazenie trénovacej množiny.

presnosti neurónovej siete. Je známe že viacvrstvové dopredné NN s určitým algoritmom učenia sa učia v niekoľkých fázach. Pohybujú sa od realizácie jednoduchých k zložitejším mapovacím funkciám. To má za následok znižovanie celkovej chyby siete. Ako bolo spomenuté na začiatku, pri generalizácie sa snažíme docieliť čo najlepšej generalizácie. Je však ťažké rozhodnúť kedy je najlepšie zastaviť trénovací algoritmus a tak predísť efektu „overfitting". Obrázok 24 znázorňuje trénovaciu množinu a validačnú množinu vo vzťahu k celkovej chybe a taktiež stav kedy by malo byť trénovanie zastavené.

- **Testovacia množina (Testing set)** - Množina dát na otestovanie výsledného riešenia.

Pre potreby trénovania umelej siete som využil 28 vzoriek podnikových procesov s ohodnotením. Trénovacia množina obsahuje diagramy reálnych firemných podnikových procesov a preto nie je možné v tejto práci uviesť taktiež grafické znázornenie podnikového procesu s využitím UML diagramu aktivít. Odhady boli vytvorené mnou na základe odhadu a mnou nadobudnutých skúsenosti počas implementácie niektorých podnikových procesov. Taktiež som sa pri práci opieral o reálne odhady z [6].

2.8.2 Normalizácia trénovacej množiny

Pred samotným spracovaním trénovacej množiny je potrebné normalizovať jednotlivé parametre trénovacej množny. Ide o proces transformácie všetkých vlastností, ktoré sú vstupom a výstupom neurónovej siete na požadavané hodnoty, ktoré sú ideálne pre vstup aktivačnej funkcie. Ide teda o transformáciu množiny parametrov na hodnoty z požadovaného intervalu.

min_{new} = minimálna hodnota nového intervalu

max_{new} =maximálna hodnota nového intervalu
min =minimálna hodnota konrétneho atribútu trénovacej množiny
max =maximálna hodnota konrétneho atribútu trénovacej množiny
v =aktuálna hodnota
v' =nová normalizovaná hodnota

$$v' = \frac{v - min}{max - min}(max_{new} - min_{new}) + min_{new}$$

3 Implementácia

3.1 Výber technológie

Nasledujúca kapitola sa zaoberá implementáciou a použitými technológiami. Z pohľadu prístupu užívateľov som sa rozhodol navrhnúť aplikáciu ako desktopovú. Umožní to efektívnejšie využívanie aplikácie pretože samotné učenie neurónovej siete je časovo veľmi náročné a pri niektorých testovacích nastaveniach môže trvať učenie ANN aj niekoľko hodín v závislosť od nastavenia kľúčových parametrov a použitej trénovacej množiny. Ďalšou požiadavkou bola aby bol program multiplatformový z pohľadu behu na rôznych platformách. Výber technológií začnem výberom vhodného formátu pre načítaní podnikového procesu v štandardizovanom formáte UML activity diagram. Pre načítanie diagramu aktivít som zvolil formát XMI:

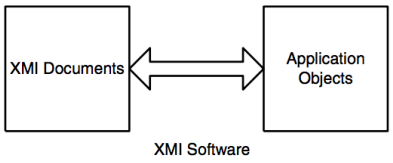
3.1.1 XMI

Pri masívnom rozširovaní a používaní UML štandardu nastal problém kompatibility medzi jednotlivými nástrojmi na vytváranie a modelovanie UML diagramov. Vznikla potreba vytvoriť štandard, ktorý by zachytával vzťahy a modely vytvorené v UML a dokázal ich transformovať do všeobecne akceptovanej podoby, ktorá následne môže byť transformovaná a využitá iným nástrojom na spracovanie a vytváranie UML diagramov. Všeobecne používaným formátom na vyjadrenie štruktúrovaných dát sa v dnešnej dobe považuje XML. XMI vypĺňa medzeru medzi objektami vytvorenými UML nástrojmi a XML. Poskytuje štandardy pre mapovanie objektov, ktoré boli definované UML do štandardizovanej podoby XML. Medzi jeho hlavné výhody patrí:

1. XMI využíva XML technológie
2. Softvér, ktorý podporuje XMI umožňuje vytváranie schém z modelovací
3. Softvér, ktorá podporuje XMI poskytuje vyššiu úroveň abstrakcie ako XMI elementy a atribúty
4. XMI pomáha vytvárať XML dokumenty, ktoré možno ľahko zameniť
5. XMI umožňuje vytvárať jednoduché dokumenty a upravovať ich podľa toho ako aplikácia rastie
6. XMI umožňuje pracovať s dátami a metadátami.

XMI (XMI Metadata Interchange) je produkt OMG (Object Management Group) a bol vyvinutý ako splynutie dvoch štandardov XMI (Extersible Markup Language - W3C štandard), a MOF (Meta object Facility – OMG jazyk pre meta-modely) pre manipuláciu a výmenu modelov a meta dát. Pretože XMI je pod kategória jazyka XML, dáta sú vytvárané a vymieňané použitím štruktúrovaného textu, ktoré dodržujú pravidlá modelu. Samotný model je definovaný meta modelom, pričom štruktúrú tohto modelu popisuje

MOF meta-meta model. Obe úrovne, DTD a XMI sú odvodené z príslušného meta modelu. Obrázok 25 znázorňuje využitie XMI pri kompatibilite vytvorených modelov rôznymi nástrojmi.



Obr. 25: Význam XMI štandardu pri kompatibilite

3.1.2 XPath

Xpath je jazyk, ktorý popisuje spôsob ako nájsť a spracovávať položky a elementy v XML pomocou adresovania a syntaxe založenej cez logickú štruktúru a hierarchiu elementov skrz XML. Vďaka tomu je možné pristupovať k XMI dokumentu výrazne jednoduchšie a nie je potrebné skúmať podrobne XML dokument. Xpath taktiež umožňuje programátorovi vysporiadať sa s dokumentom na vyššej úrovni abstrakcie. Patrí medzi štandardy, ktoré sú súčasťou základných balíkov dnes používaných programovacích jazykov. Práca s XML a Xpath je široko podporovaná taktiež v jazyku Java, ktorý využívam pri implementácií.

3.1.3 Java

Pretože že pri výbere programovacieho jazyka hral významnú úlohu fakt, že program by mal byť multiplatformový, rozhodol som sa využil programovací jazyk Java. Ide o objektovo orientovaný programovací jazyk. Je vyvíjaný spoločnosťou Oracle. Jeho syntax vychádza z jazykov C a C++. Najväčším prínosom Javy je nepochybne plná prenositeľnosť programov na ľubovoľnú platformu bez nutnosti ich rekompilácie. Programy sa totiž neprekladajú do strojového kódu konkrétneho procesora, ale do nezávislého podoby, tzv. bytového kódu (bytecode). Tento kód potom môže byť interpretovaný na akomkoľvek počítači alebo priemyselnom zariadení. Kompatibilita je teda zabezpečená na binárnej úrovni. Hardvérové rozdiely zastrešuje tzv Java Platforma, ktorá obsahuje dve základné časti:

- **JVM** - virtuálny stroj (Java Virtual Machine - JVM), ktorý pozostáva z runtime systému, čo je časť, ktorá realizujúce väzbu na hardvér, a interpretra, ktorý vykonáva bytový kód. Pre urýchlenie môže byť interpret voliteľne nahradený tzv JIT (Just-In-Time) kompilátorom, ktorý pri behu programu vykonáva najprv preklad do strojového kódu príslušného procesora.
- **Java Core API** - čo sú základné knižnice pre písanie programov. Výhodou je, že tieto knižnice nemusia byť s programom distribuované, pretože sú súčasťou Java plat-

formy. V súčasnej dobe prebieha návrh niekoľkých rozširujúcich programových rozhraní. Ide napríklad o podporu 2D a 3D grafiky (Java 2D a 3D API), zvuk (Java Audio API), správu siete (Java Management API), bezpečnosť (Java Security API) atď.

Platforma Java bola navrhnutá pre široké spektrum zariadení. Existuje v niekoľkých edíciách, ktoré sú určené pre rozdielne skupiny zariadení a rozdielne využitie. Rozdeľuje sa na:

- **Java SE** - Java Standard Edition (J2SE) - Základná platforma pre aplikácie napísané v jazyku Java. Jej využitie je pre tvorbu desktopových aplikácií. J2SE obsahuje základné knižnice ako napríklad java.lang, java.io, java.nio, java.math, java.net, java.text, java.util, java.applet, java.beans, java.awt a ďalšie.
- **Java EE** - Java Enterprise Edition je platforma využívaná pre serverové riešenia. Oproti Java SE obsahuje naviac knižnice pre vývoj bezporuchových, distribuovaných, viac-vrstvových aplikácií bežiacich na aplikačnom serveri.
- **Java ME** - Java Micro Edition je platforma navrhnutá pre mobilné zariadenia a vložené (embedded) systémy.

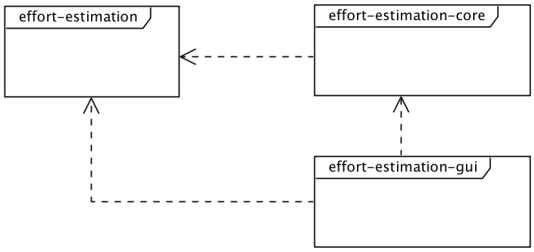
3.1.4 Maven

Od začiatku vzniku programovacieho jazyka java programátori narážali na problém zautomatizovanie činností, ktoré sa rutinne opakujú ako zostavenie projektu, spúšťanie testov a množstvo iných úloh, ktoré úzko súvisia z programovaním modulárnych systémov postavených nad platformou java. Platforma Java so sebou prináša veľké množstvo otvorených nástrojov a knižníc na riešenie rôznych problémov. Bola potreba tieto knižnice pri vývoji udržiavať, zhromažďovať a spravovať. Všetky tieto problémy za nás rieši nástroj Maven. Maven je automatizovaný nástroj primárne učený na zostavovanie Java projektov. Medzi jeho hlavné výhody a úlohy patrí popis a vykonávanie zostavovania programov a druhou významnou úlohou je popis závislostí jednotlivých knižníc, ktoré môžu byť využité pri vývoji. Takto vytvorený projekt spolu s jeho programovými závislosťami je jednoduché spravovať a udržiavať a poskytuje nám flexibilitu pri zmene verzií jednotlivých knižníc. Môj projekt bol vytváraný a spravovaný týmto nástrojom. Maven sa nachádza aktuálne vo verzií 3.3.1.

3.2 Softvérová implementácia

Nasledujúca časť obsahuje implementáciu navrhnutej aplikácie. Aplikáciu som sa rozhodol implementovať v programovacom jazyku Java. Obrázok 26 znázorňuje závislosť jednotlivých modulov. Aplikácia sa skladá z 3 základných modulov. Všetky tieto moduly boli vytvárané ako maven module, čo nám umožňuje lepšie správu vytvorených jar archívov:

1. **effort-estimate** – Ide model, ktorý zastrešuje 2 hlavné moduly a obsahuje definíciu používaných knižníc a modulov.
2. **effort-estimate-core** – Hlavný modul aplikácie pre odhad zložitosti. Obsahuje časti aplikácie pre vytváranie metrík, vytvárania objektového modelu UML diagramu aktivít a taktiež časti aplikácie, ktoré sú zodpovedné za vytváranie a učenie umelej neurónovej siete.
3. **effort-estimate-gui** – Modul, ktorý je závislý na effort-estimate-core. Ide o modul, ktorý obsahuje grafické rozhranie aplikácie vytvorenej v prostredí Swing. Táto knižnica je štandardom pre vytváranie grafických desktopových aplikácií nad patformou Java SE.



Obr. 26: Aplikačné moduly a ich závislosti

3.2.1 Načítanie a vytvorenie modelu

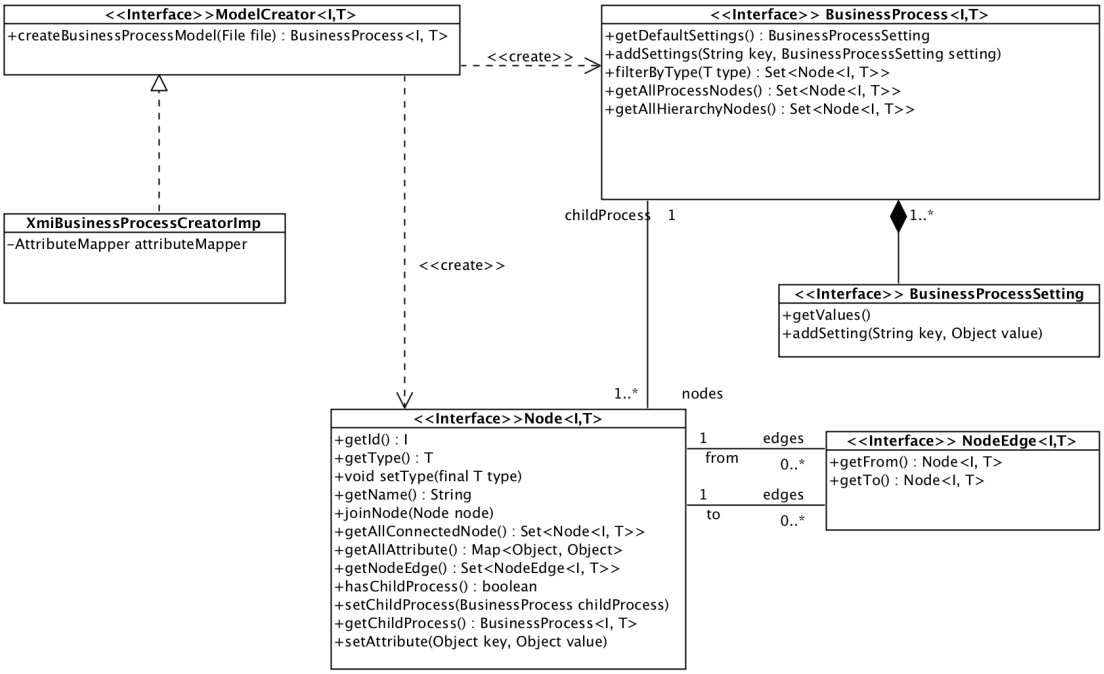
Aby bolo možné aplikačne pracovať s UML diagramom aktivít, je potrebné ho načítať do aplikačného modelu. Pri načítavaní využívam už spomenutý XPath na prechádzanie xmi dokumentu a hľadanie vhodných elementov, ktoré následne tranformujem do objektového modelu. Objektový model podnikovhé procesu spolu s časťou aplikácie, ktorá je zodpovedná za vytvorenie modulu je zobrazená na obrázku 27. Model sa skladá z nasledovných častí:

1. **Node** - Predstavuje element podnikového procesu a môže obsahovať niekoľko na seba nadvezujúcich hrán spolu. Taktiež návrh počíta s aplykovaním pod procesov a preto element Node taktiež môže obsahovať 0..1 podnikových procesov.

2. **BusinessProcess** - Ide o kontajner, ktorý obsahuje všetky elementy a nastavenie podnikového procesu. Obsahuje 0..n nastavení a a 1..n elementov podnikovhé procesu.

3. **NodeEdge** - Ide o model, ktorý predstavuje hranu, ktorá spája elementy BP. Spája vždy Node from a Node to.

4. **ModelCreator** - Komponenta je zdpovedná za vytvorenie objektovej reprezentácie BP. Implementácia tejto komponenty pracuje s XMI modelom.



Obr. 27: Triedny diagram objektového modelu podnikového procesu

3.2.2 Analýza modelu

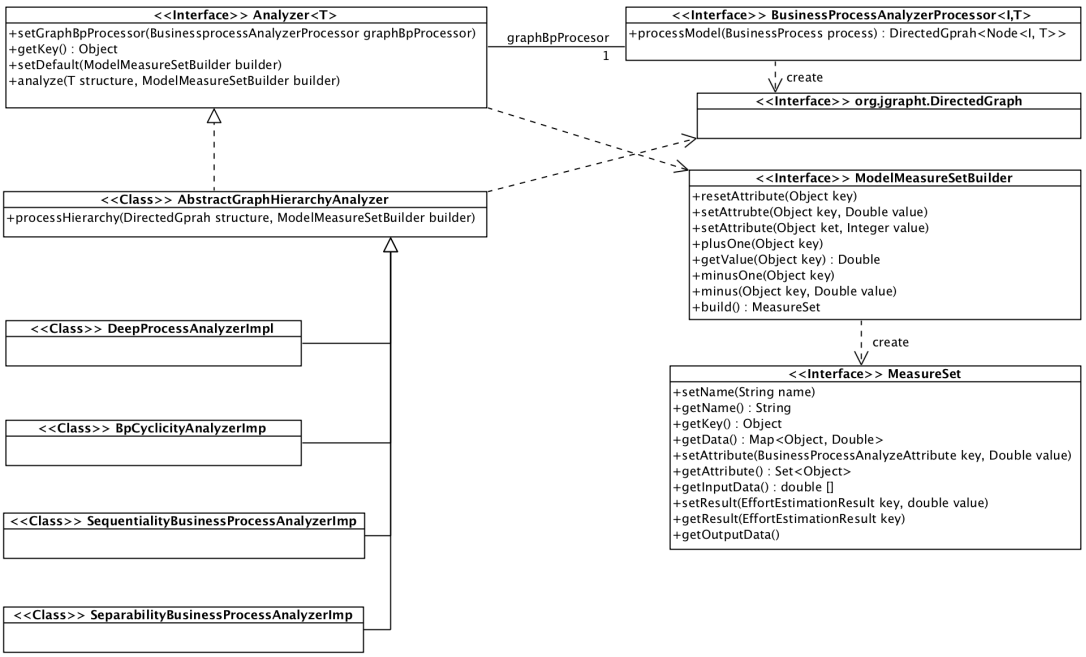
Po vytvorení objektového modelu BP nasleduje druhá fáza projektu a tou je analýza vytvoreného modelu a vytvorenie metrík, ktoré budu následne skúmané. Pre potreby analýzy modelu je aplikácia navrhnutá do 3 základných komponent:

1. **Analyzer** - Ide o jednu z hlavných častí aplikácie. Každá z implementácií je zodpovedná za vytvorenie konkrétnej meratelnej metriky. Pretože aplikácia môže spracovať podnikové procesy, ktoré sú vnorené, je potrrebné prejsť každý element hierarchicky. Medzi implementácie rozhrania Analyzer patrí **DeepProcessAnalyzerImpl, BpCyclicityAnalyzerImp, SequentialityBusinessProcessAnalyzerImp, SeparabilityBusinessProcessAnalyzerImp**

2. **ModelMeasureSetBuilder** - Hlavnou úlohou komponenty je držanie nameraných hodnôt počas procesu ohodnocovania a vytvorenie výslednej množiny nameraných hodnôt.

3. **BusinessProcessAnalyzerProcessor** - Pretože väčšina použitých metrík sú grafovo orientované. Pri vytváraní metrik som využíval knižnicu jgraphT, ktorá slúži na prácu s orientovaným grafom a taktiež obsahuje radu užitočných funkcií na výpočet vlastností grafu. Komponenta je zodpovedná za vytvorenie jgraphT grafu z objektovej reprezentácie podnikovhé procesu.

4. **MeasureSet** - Ide o výsledky nameraných metrík BP, ktoré sú následne spracované ANN.



Obr. 28: Triedny diagram analýzy podnikového procesu

3.2.3 Návrh aplikácie odhadu zložitosti

Ide o hlavné časti aplikácie, ktoré fungujú ako fasáda a zabezpečujú funkcionalitu všetkých hlavných častí aplikácie.

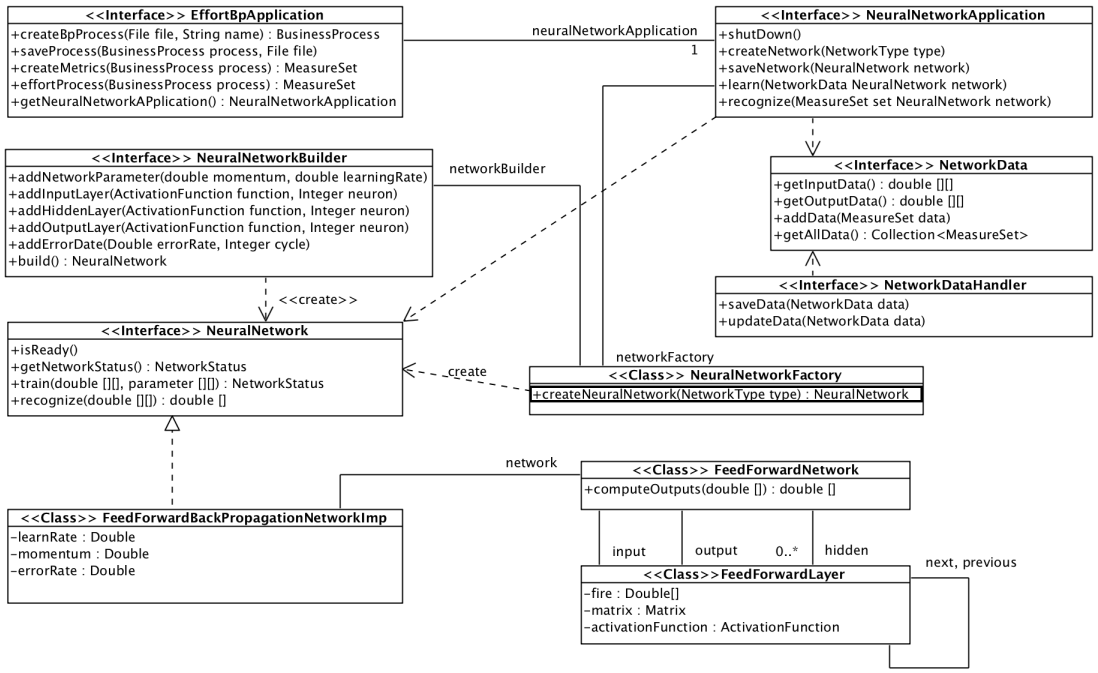
1. **NeuralNetworkApplication** - Hlavnou úlohou tohto rozhrania je práca s ANN. Umožňuje vytvorenie siete, načátanie naučenie siete, zastavenie učenia a taktiež ponúka funcionalitu pre spustenie rozhodovania na základe naučenej ANN.

2. **NeuralNetworkBuilder** - Ide o návrhový vzor Builder a umožňuje nám vytvoriť a nakonfigurovať ANN.

3. **NetworkData** - Ide o adaptré spolu s **NetworkDataHandler**, ktorý je zodpovedný za udržiavanie a správu naučených dát aplikácie.

4. **NeuralNetwork** - Ide o jednu z hlavných častí aplikácie, ktorá predstavuje prístup k ANN a zaobaľuje základnú funkcionalitu potrebné pre prácu s ANN. Aplikácia implementuje toto rozhranie triedou **FeedForwardBackPropagationNetworkImp**, ktorá predstavuje simuláciu ANN.

5. **EffortBpApplication** - Predstavuje hlavnú časť aplikácie a spája hlavné časti, ktoré boli spomenuté v predchádzajúcich častiach. Je zodpovedná za vytvorenie procesu, uloženie, spracovanie a vyhodnotenie na základe vytvorenej a naučenej ANN.



Obr. 29: Triedny diagram aplikácie odhadu úsilia s ANN

3.2.4 Návrh GUI

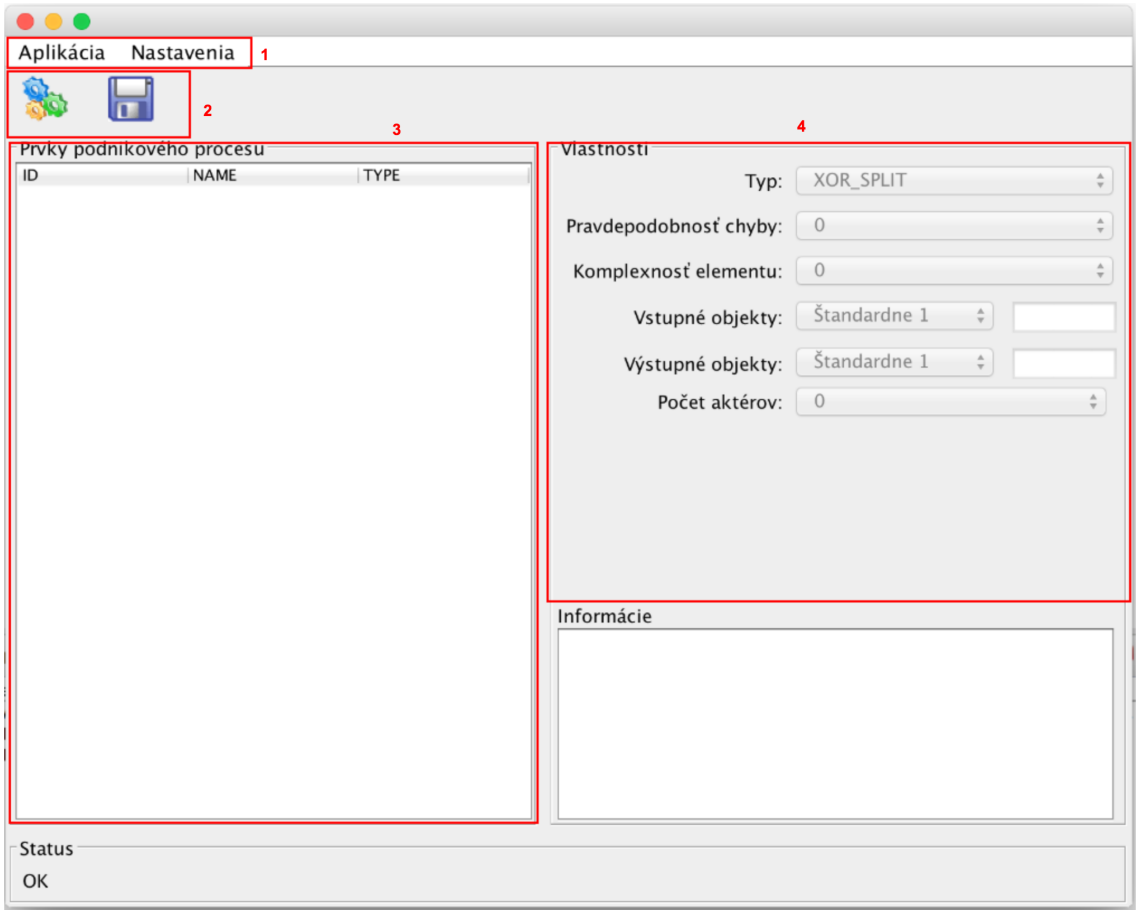
Aplikácia by mala taktiež obsahovať grafickú časť a mala by umožniť užívateľovi:

- Musí poskytnúť užívateľovi možnosť načítať XMI model a zobraziť zoznam načítaných elementov.
- Takto načítaný podnikový proces by malo byť možné ďalej parametrizovať a nastaviť tie vlastnosti, ktoré je možné ovplyvniť. Metriky, ktoré je možné pred samotným vytvorením ohodnotenia ešte parametrizovať som spomenul v predchádzajúcej kapitole.
- Zobrazenie výsledkov a po vytvorení odhadu tento odhad zobraziť alebo ďalej spracovať a uraviť.
- Spracovávať dáta, ktoré boli využité pri učení ANN.
- V neposledom rade by malo byť užívateľovi umožnené spustiť nové učenie siete, pokiaľ bola trénovacia množina upravená k lepším výsledkom.

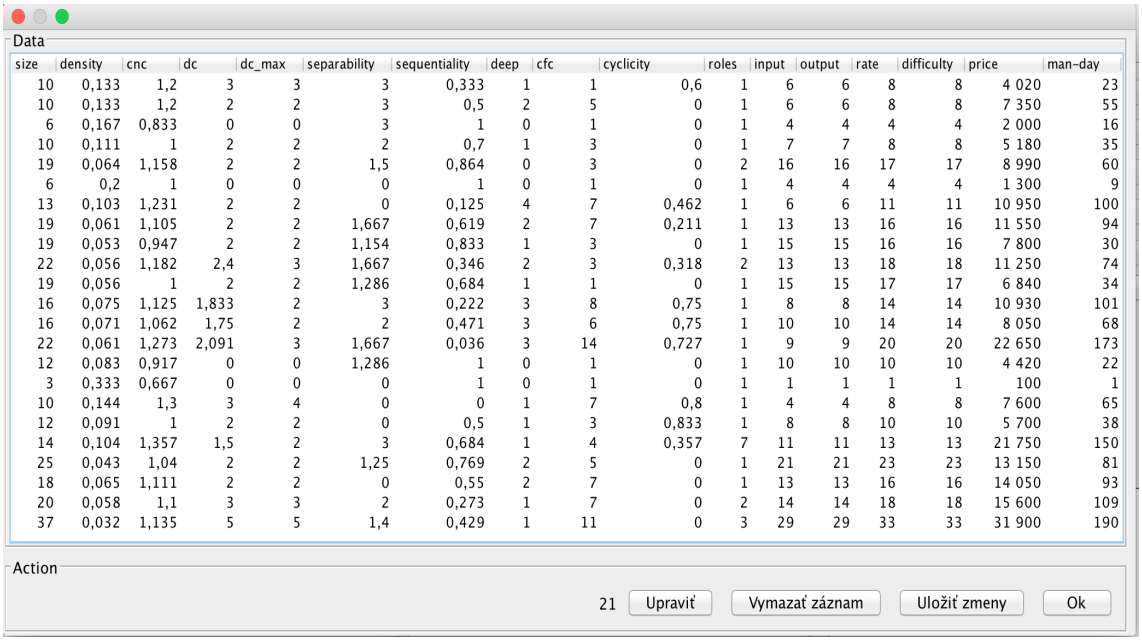
Pre vytvorenie grafického prostredia aplikácie som využil štandardnú knižnicu Swing java platformy. Po rozbore spomenutých požiadaviek, ktoré boli spomenut som navrhol nasledovné grafické prostredie. Grafické prostredie je rozdelené na dve základné obrazovky. Orázok 30 znázorňuje zobrazenie hlavnej obrazovky aplikácie odhadu úsilia. Hlavnú obrazovku môžeme rozdeliť do 2 základných častí:

- Ľavá strana slúži na zobrazenie načítaných elementov podnikového procesu. Zobrazuje elementy ako interne XMI id, názov aktivity alebo elementu a typ elementu.
- Pravá strana slúži na parametrizáciu jednotlivých elementov. Ide o parametre, ktoré je možné ovplyvniť a tak zmeniť výsledok ohodnotenia.

Poskytuje rozhranie pre zmenu trénovacej množiny a spúštanie učenia a parametrizáciu ANN. Obrazovka nastavení zobrazuje použitú trénovaciu množinu s možnosťou zmeny parametrov a spustenia nového učenia ANN.Obrázok 31 čislo znázorňuje obrazovku pre nastavenie ANN a trénovacej množiny. Výsledky odhadu zložitosti je možné vidieť na obrázku 32.



Obr. 30: Hlavná obrazovka aplikácie



Obr. 31: Nastavenie ANN a trénovacej množiny.

Obr. 32: Obrazovka výsledku odhadu úsilia ANN

4 Výsledky

Po stanovení metrík, ktoré budú použité pri vyjadrení odhadu úsilia podnikové procesu je potrebné stanoviť vlastnosti umelej neurónovej siete. Ide o vlastnosti, ktoré majú vplyn na učenie neurónovej siete ako aj na vyjadrenie celkovej prestnosti neurónovej siete. Ide o vlastnosti ako:

- **chyba siete** - Ide o parameter, ktorý slúži na vyjadrenie celkovej chyby siete. Podstata učenia umelej neurónovej siete spočíva práve nájdení čo najmenšej chyby.
- **learning rate** - Určuje veľkosť zmien vnútorných synaptických váh.
- **momentum** - Ide o parameter, ktorým je možné upravovať rýchlosť učenia neurónovej siete. Výsoká hodnota tohto parametru môže urýchliť učenie siete avšak nastavenie hodnoty príliš vysoko môže viesť k nestabilite systému. Hodnota príliš nízko zasa môže spôsobiť nevyhnutiu sa lokálnym minimám výslednej trénovacej funkcie.

4.1 Testy ANN

Pre potreby najoptimálnejšieho naučenia siete som vytvoril niekoľko testov ANN, kedy som sa snažil nastaviť sieť aby ideálne konvergovala. Pre potreby parametrizácie som nastavoval niekoľko parametrov ako:

Error rate - Obecne môžeme povedať, čím menšia chyba siete tým nám sieť poskytne lepšie výsledke. Pri veľmi nízkej chybe však môže nastať situácia, že sieť nebude možné naučiť danému problému v reálnom čase. Pri chybe siete okolo 0.001 môže trvať naučenie siete aj niekoľko hodín.

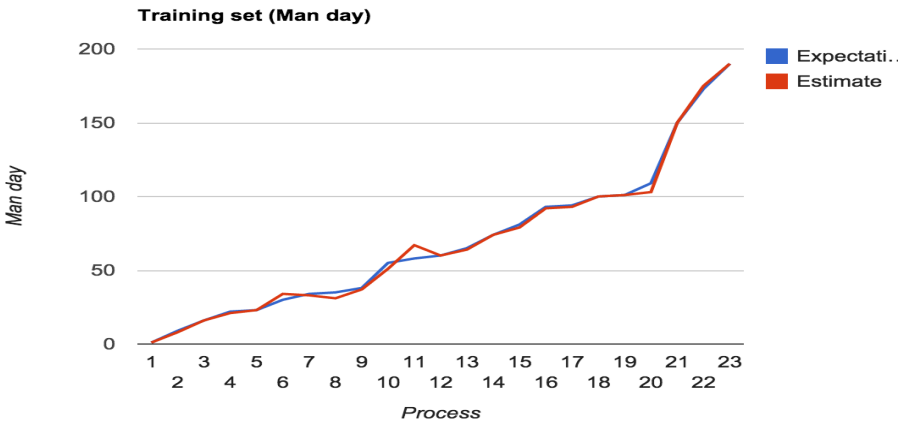
Moment - V testoch som využíval prevažne konštantnu 0.6

Rate - V testoch som prevažne používal konštantu 0.09

Počet neurónov - V testoch optimálneho nastavenia ANN som sa snažil otestovať niekoľko nastavení počtu neurónov vnútornej vrstvy ANN.

Testy vytvorej umelej neurónovej siete možeme rozdeliť do niekoľko krokov. V prvom kroku som porovnal procesy, ktoré boli súčasťou trénovacej množiny. Ide o ujistenie sa či veľkosť chyby siete vyhovuje naším požiadavkám. Pre všetky nastavenia ANN boli tieto výsledky veľmi podobné a preto uvádzam výsledky iba použitej neurónovej siete. Graf z obrázka 33 a 34 znázorňuje očakávané hodnoty a ohodnotené hodnoty price a man day procesov, ktoré boli súčasťou trénovacej množiny.

V ďalšom kroku testovania ANN som vytvoril množinu 22 procesov, ktoré neboli súčasťou trénovacej množiny. Tieto som ohodnotil z nadobudnutých skúseností, ktoré som aplikoval taktiež na ohodnotenie trénovacej množiny a tieto procesy som podrobil



Obr. 33: Graf validácia trénovacej množiny pre parameter man day

Obr. 34: Graf validácia trénovacej množiny pre parameter overall price

testom ohodnotenia ANN. Pri vytváraní testovacej množiny som daktiež využíval odvodzovanie a spájanie podnikových procesov, ktoré boli súčasťou trénovacej množiny. Procesy testovacej množiny môžeme rozdeliť do niekoľkych kategórií:

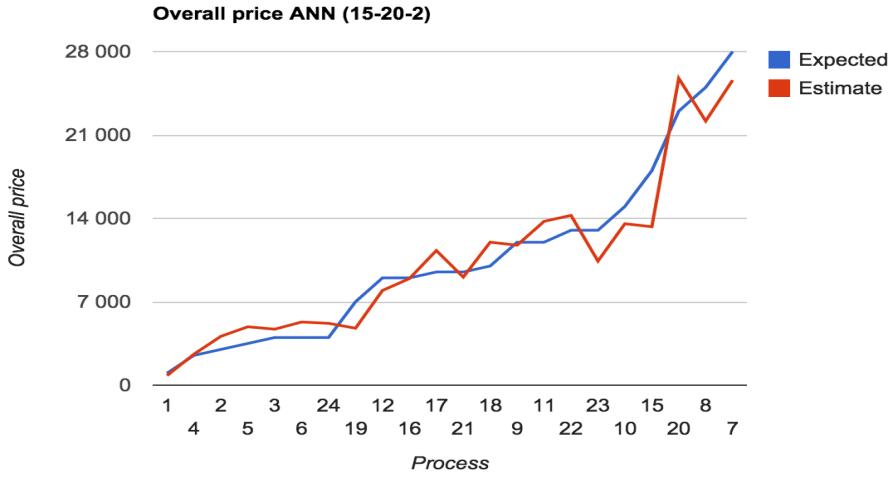
1. Procesy, ktoré boli ohodnotené bez odvodzovania.
2. Procesy vytvorené ako spojenie dvoch procesov, ktoré boli súčasťou trénovacej množiny. Predpoklad takéhoto odhadu je že výsledný odhad sa bude v ideálnom prípade s časti približovať k súčtu parciálnych odhadov. Taktiež sa odhad môže približovať k procesu, ktorý sa svojou veľkosťou a komplexnosťou podobá na iný proces trénovacej množiny.
3. Pridaním alebo odobraním niekoľkých elementom z procesu, ktorý bol súčasťou trénovacej množiny. V takto prípade môžeme predpokladať drobné navýšenie alebo zníženie celkového odhadu.

4.1.1 Test číslo 1

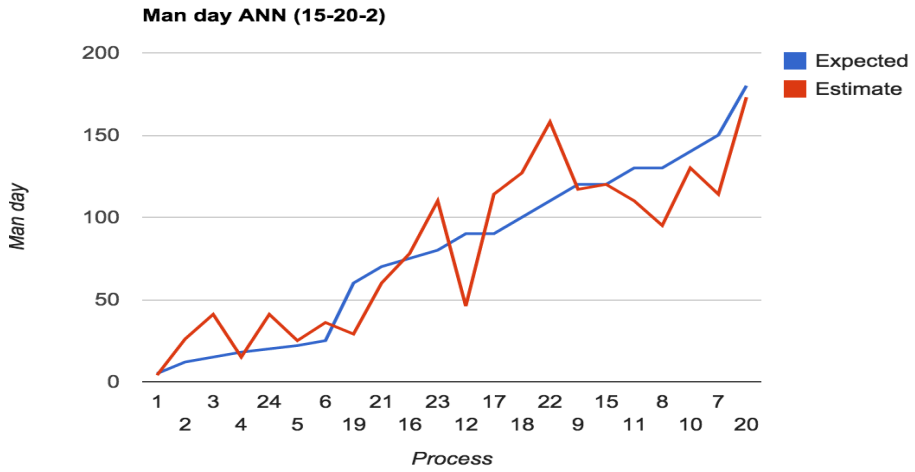
V prvom teste bola ANN nakonfigurovaná nasledovným spôsobom

Neurons	Overal error	Momentum	Learning rate
15-20-2	0.003	0.6	0.09

Nasledovný graf znázorňuje ohodnotenie 22 procesov, ktoré neboli súčasťou trénovacej množiny. Ide o testovaciu množinu procesov.



Obr. 35: Test ANN číslo 1 overall price testovacej množiny



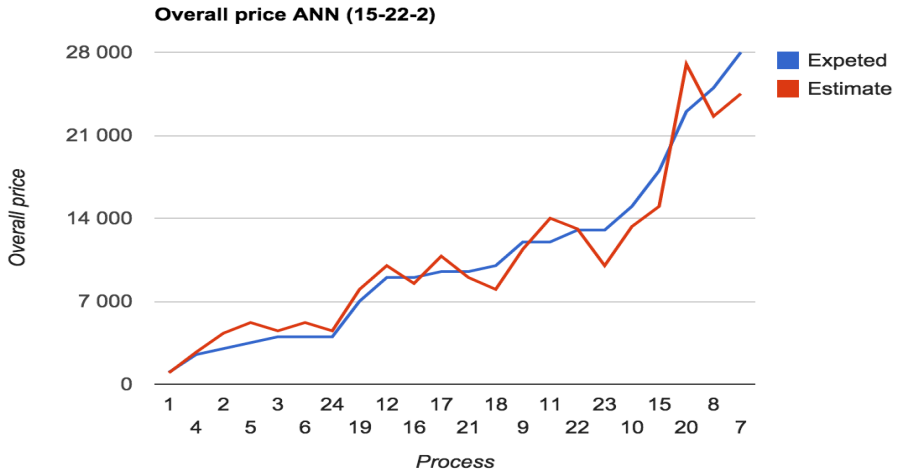
Obr. 36: Test ANN číslo 1 man day testovacej množiny

4.1.2 Test číslo 2

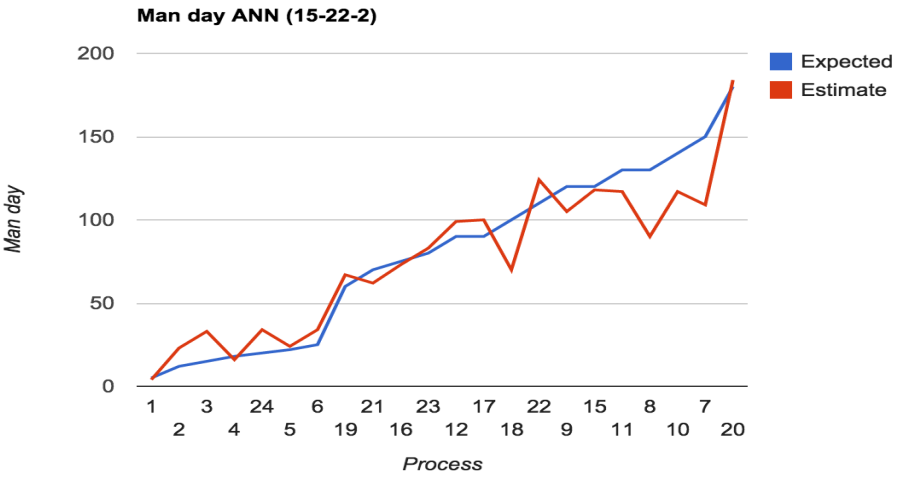
V druhom teste bola ANN nakonfigurovaná nasledovným spôsobom. Ide o nastavenie, ktoré som taktiež využil v aplikácií pretože takto nakonfigurovaná sieť poskytovala lepšie výsledky oproti iným konfiguráciam.

Neurons	Overal error	Momentum	Learning rate
15-22-2	0.003	0.6	0.09

Nasledovný graf znázorňuje ohodnotenie 22 procesov, ktoré neboli súčasťou trénovacej množiny. Ide o testovaciu množinu procesov.



Obr. 37: Test ANN číslo 2 overall price testovacej množiny



Obr. 38: Test ANN číslo 2 man day testovacej množiny

4.2 Vyhodnotenie testovacej množiny

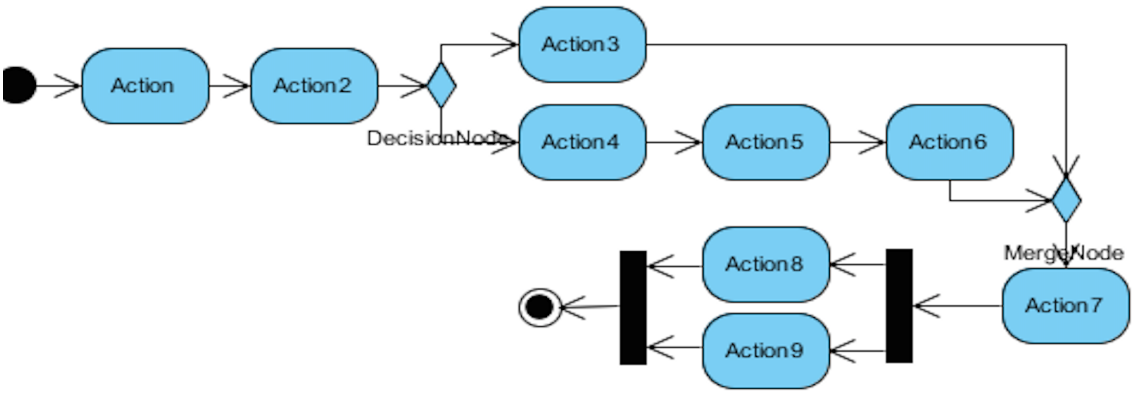
Zo zobrazených grafov vyplýva, že nastavenie neurónovo ANN v rozložení 15-22-2 poskytuje lepšie nastavenie oproti iným nstaveniam. V práci som uviedok iba dva a to rozloženie neurónov 15-22-2 a 15-20-2. Ďalej som testoval taktiež rozdelenie s 18 a 16 neurónmi skytej vrstvy. Tieto nastavenia však neposkytovali také výsledky ako spomenuté rozdelenie s 22 neurónmi. Z grafov môžeme usúdiť, že naučená sieť dokázala reagovať na predloženú vzorku testovacích dát s prijateľnou chybou. V niektorých miestach grafu môžeme vidieť výrazné odskoky od očakávaných hodnôt. Takýto stav môže mať viac príčin a to:

1. Trénovacia množina neobsahovala dostatok prvkov a teda nedokázala pokryť všetky možné vektory a kombinácie metrík, ktoré su spojené s modelovaním podnikových procesov.
2. Trénovacia množina obsahovala anomálie, ktoré sa vymykali z bežného dátového rozloženia trénovacej množiny.
3. Je potrebné pridať d'alšie neuróny do skytej vrstvy a otestovať ako sa bude chovať výsledné ohodnotenie pri nastavení väčšom ako je 22 neurónov skytej vrstvy.

4.3 Testovacie dáta

Nasledujúci test funkčnosti spočíva v porovnaní procesov, ktoré sú si podobné. Ako hlavný proces som si vybral model, ktrorý je zobrazený na obrázku 39. Tento proces bol programom ohodnotený nasledovne:

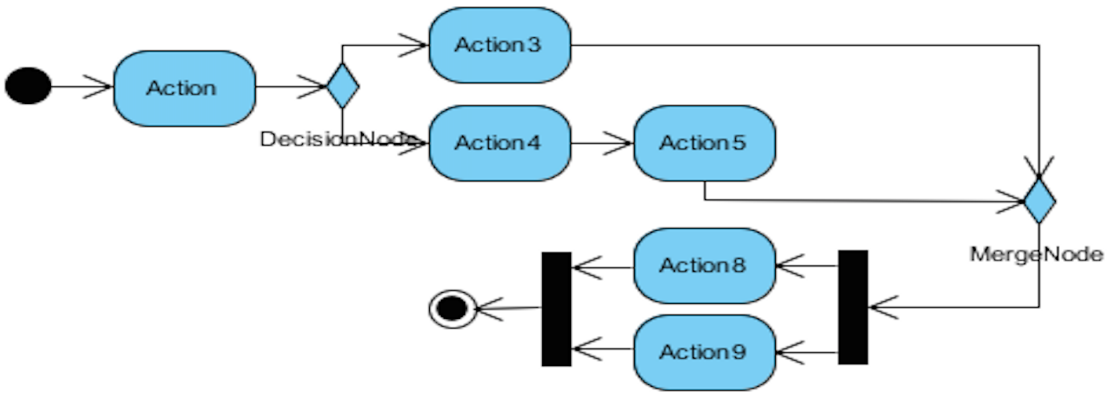
Test	Man-day	Price
Reálne	57	8745



Obr. 39: Model hlavného testovacieho procesu

Nasleduje proces, ktorý by mal mal byt svojou komplexnosťou a veľkosťou menší ako hlavný testovací proces. Obsahuje menej akcií a teda by mal byť finančne menej náročnejší. Tento proces je na obrázku 40 a bol ohodnotený nasledovne:

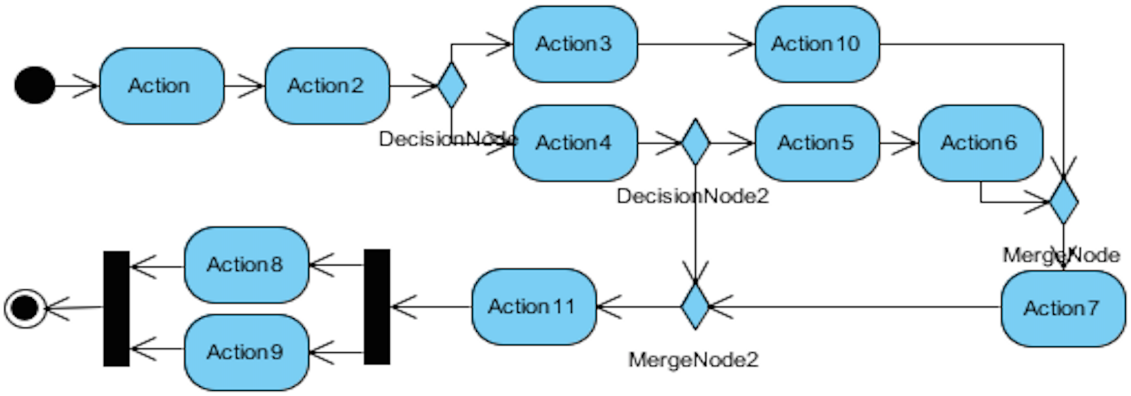
Test	Man-day	Price
Reálne	54	7900



Obr. 40: Model testovacieho procesu, ktorý je veľkosťne menší ako hlavný testovací proces.

Posledným porovnaním je proces, ktorý je komplexnejší a zložitejší ako hlavný testovací proces. Obsahuje viac aktivít a pridané cesty podnikovým procesom. Ide o proces na obrázku 41 a jeho odhodnotenie je nasledovné:

Test	Man-day	Price
Reálne	97	13500



Obr. 41: Model procesu, ktorý je komplexnejší ako hlavný testovací model

Test ukázal, že pridanie novej cesty a nových rozhodovacích blokov v spojení s niekoľko aktivitami, môže mať za následok výrazné navýšenie celkového času spojeného s vytvraním takéhoto podnikovhé procesu. Ako som už spomenul, takýto výrazný posun by bolo možné odstrániť využítím väčšieho počtu podnikových procesov trénovacej množiny, ktoré by podrobnejšie opisovali vzťahy jednotlivých premených vstupného datového súboru. Podľa môjho názoru by to výrazne mohlo pomôcť pri spresení odhadov man-day a overall price.

5 Záver

V práci som sa zaoberal vytvorením nástroja pre odhad zložitosti a úsilia podnikového procesu s využitím umelých neurónových sietí. V práci som nadobudol znalosti metrík, ktoré slúžia na vyjadrenie vlastností podnikového procesu a ktoré slúžia taktiež ako parametre iných metód odhadov úsilia. Pri obecných odhadoch úsilia a zložitosti je ťažké určiť presne reálne úsilie pretože do procesu vstupuje množstvo faktorov od začiatku analýzy procesu až po jeho implementáciu. Dôležitú vlastnosť taktiež zohráva znalosť problematiky analýzi podnikového procesu a znalosť domény, ktorej sa skúmaný a modelovaný proces týka. Druhá časť práce sa zaoberala vytvorením nástroja s využitím umelých neurónových sietí. ANN vnímam ako zaujímavý nástroj, ktorý rozhodne vyskúšam aplikovať aj na iné teoretické problémy, v ktorých ANN prinášajú pozitívne výsledky. ANN našli uplatnenie pre odhady zložitosti či už softvérových alebo podnikových procesov. Ide o techniku, ktorá môže pomôcť a priniesť iný pohľad na obecné riešenie problému predikčných odhadoch. Testy, ktorými prešla sieť ukázali dobré výsledky s priateľnou chybou a to ma doviedlo k záveru, že použitie ANN bolo správnou voľbou. Vytvorený nástroj môže nájsť uplatnenie v spoločnosti, ktorá sa zaoberá modelovaním a vytváraním procesov, a ktorá má záznamy o už vytvorených procesoch a ich celkovom úsili, ktoré bolo vynaložené spoločnosťou na ich realizáciu.

Zo vzorky dát, ktorá odpovedá validačnej množiny a testovacieho vzorku je možno povedať že ohodnotenie bolo úspešné a ANN poskytuje pozitívne výsledky.

Peter Rafaj

6 Literatúra

[1] *prof. Ing. Ivo Vondrák, CSc., METODY BYZNYS MODELOVÁNÍ: pro kombinované a distanční studium. [online]. [cit. 2015-3-12]. Dostupné z:* `http://vondrak.cs.vsb.cz/download/Metody_byznys_modelovani.pdf`

[2] *Petr Kovář, Úvod do Teorie grafů. [online]. [cit. 2015-3-12]. Dostupné z:* `http://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/uvod_do_teorie_grafu.pdf`

[3] *Zakladné grafové algoritmy. [online]. Dostupné z:* `http://algoritmy.eu/zga/grafy-a-stromy/`

[4] *Teória grafov. [online]. Dostupné z:* `http://books.fs.vsb.cz/SystAnal/texty/21.htm`

[5] *Jan Mendling, Metrics for Process Models, ISBN-10: 3-540-89223-0 Springer Berlin Heidelberg New York.*

[6] *Banu Aysolmaz, Deniz İren, and Onur Demirörs 1Enterprise, An Effort Prediction Model Based on BPM Measures for Process Automation [154-168], Business-Process and Information Systems Modeling, ISBN 978-3-642-38483-7, e-ISBN 978-3-642-38484-4, Springer Heidelberg Dordrecht London New York.*

[7] *Beniamino Murgante, Osvaldo Gervasi, Sanjay Misra, Nadia Nedjah Ana Maria A.C. Rocha, David Taniar, Bernady O. Apduhan (Eds.), Computational Science and Its Applications – ICCSA 2012, 12th International Conference Salvador de Bahia, Brazil, June 18-21, 2012 Proceedings, Part IV, ISSN 0302-9743, e-ISSN 1611-3349, Springer Heidelberg Dordrecht London New York*

[8] *Beniamino Murgante Osvaldo Gervasi Sanjay Misra Nadia Nedjah Ana Maria A.C. Rocha David Taniar Bernady O. Apduhan (Eds.), Computational Science and Its Applications – ICCSA 2012, ISBN 978-3-642-31127-7, e-ISBN 978-3-642-31128-4, Springer Heidelberg Dordrecht London New York*

[9] *Geoffrey M. Muketha, A.A.A Ghani, M.H Selamat, R. Atan, A survey of Business Process Complexity Metrics, ISSN 1812-5638, Asian Network of Scientific Information 2010, Faculty of Computer Science and Information Technology, University Putra Malaysian.*

[10] *Volker Gruhn and Ralf Laue, Complexity Metrics for Business Process Models„Chair of Applied Telematics / e-Business Computer Science Faculty, University of Leipzig, Germany, [Online], Dostupné z:* `http://subs.emis.de/LNI/Proceedings/Proceedings85/GI-Proceedings-85-1.pdf`

[11] *prof. Ing. Ivo Vondrák, CSc., Neuronové sítě, [Online], Dostupné z:* `http://vondrak.cs.vsb.cz/download/Neuronove_site.pdf`

[12] *Lubica Benuskova, Umelé neurónové siete, [Online], Dostupné z:* `http://dai.fmph.uniba.sk/~benus/books/UNS_revised.pdf`

[13] *Saeed Ayat, Hojjat A. Farahani, Mehdi Aghamohamadi, Mahmood Alian, Somayeh Aghamohamadi, Zeynab Kazemi, A comparison of artificial neural networks learning algorithms in predicting tendency for suicide, Neural Comput Applic (2013) 23:1381–1386*

[14] *Ciobanu Dumitru, Advantages and Disadvantages of Using Neural Networks for Predictions, "Ovidius" University Annals, Economic Sciences Series Volume XIII, Issue 1/ 2013*

[15] *Mamoona Humayun and Cui Gang, Estimating Effort in Global Software Development Projects Using Machine Learning Techniques, International Journal of Information and Education Technology, Vol. 2, No. 3, June 2012*

[16] *Ali Idri, Taghi M. Khoshgoftaar, Alain Abran, Can Neural Networks be easily Interpreted in Software Cost Estimation?*

[17] *I.F. Barcelos Tronto, J.D. Simões da Silva, N. Sant'Anna, Comparison of Artificial Neural Network and Regression Models in Software Effort Estimation, Laboratory for Computing and Applied Mathematics - LAC Brazilian National Institute for Space Research - INPE*

[18] *Anupama Kaushik, COCOMO Estimates Using Neural Networks, Assistant Professor, Dept. of IT, Maharaja Surajmal Institute of Technology, GGSIP University, Delhi, India, I.J. Intelligent Systems and Applications, 2012, Published Online August 2012 in MECS (http://www.mecs-press.org/)*

[19] *Prasad Reddy P.V.G.D, Sudha K.R, Rama Sree P and Ramesh S.N.S.V.S.C, Software Effort Estimation using Radial Basis and Generalized Regression Neural Networks, ISSN 2151-9617, JOURNAL OF COMPUTING, VOLUME 2, ISSUE 5, MAY 2010, WWW.JOURNALOFCOMPUTING.ORG*

[20] *Prabhakar,Maitreyee Dutta, Prediction of Software Effort Using Artificial Neural Network and Support Vector Machine,Volume 3, Issue 3, March 2013, ISSN: 2277 128X, International Journal of Advanced Research in Computer Science and Software Engineering*

[21] *WIEM KHLIF, NAHLA ZAABOUB, HANENE BEN-ABDALLAH, Coupling metrics for business process modeling, Faculty of Economics and Management Sciences, NTERNATIONAL JOURNAL OF COMPUTERS Issue 4, Volume 4, 2010*

[22] *Krzysztof Kluza, Grzegorz J. Nalepa, Square Metrics for Measuring Business Process Model Complexity, ISBN 978-83-60810-51-4, AGH University of Science and Technology*

[23] *Jorge Cardoso, Business Process Control-Flow Complexity: Metric, Evaluation, and Validation, University of Madeira, Portugal*

[24] *Elvira Rolón, Francisco Ruiz, Félix García, Mario Piattini, APPLYING SOFTWARE PROCESS METRICS IN BUSSINESS PROCESS MODEL, ISSN: 1698-2029, RPM-AEMES, VOL. 3, No 2 Septiembre 2006*

[25] *Beatriz Marín, José Quinteros, A COSMIC Measurement Procedure for BPMN Diagrams, Universidad Diego Portales, Chile*

[26] *Warren Rost Bennett, Jr., PREDICTING SOFTWARE SYSTEM DEVELOPMENT EFFORT VERY EARLY IN THE LIFE-CYCLE USING IDEF0 AND IDEF1X MODELS, Mississippi State, Mississippi, December 1996, [Online], Dostupné z:* `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.50.9357&rep=rep1&type=pdf`

[27] *UML Superstructure Specification, v2.0, [Online], Dostupný z:* `http://www.omg.org/cgi-bin/doc?formal/05-07-04.pdf`

7 Prílohy

Príloha diplomovej práca obsahuje prvky trénovacej množiny, ktoré som využil pri učení umelej neurónovej siete. Ide o 29 prvkovú množinu, ktorá obsahuje hodnoty použitých metrík spolu s ohodnoteniami jednotlivých procesov. Trénovaciu množinu predstavuje tabuľka 1,2.

name	size	density	cnc	dc	dc max	sep	seq	deep	cfc	cyclicity	roles	in	out	rate	diff
10	10.0	0.13	1.2	3.0	3.0	3.0	0.33	1.0	1.0	0.6	1.0	6.0	6.0	8.0	8.0
11	10.0	0.13	1.2	2.0	2.0	3.0	0.5	2.0	5.0	0.0	1.0	6.0	6.0	8.0	8.0
12	6.0	0.17	0.83	0.0	0.0	3.0	1.0	0.0	1.0	0.0	1.0	4.0	4.0	4.0	4.0
13	10.0	0.11	1.0	2.0	2.0	2.0	0.7	1.0	3.0	0.0	1.0	7.0	7.0	8.0	8.0
14	19.0	0.06	1.16	2.0	2.0	1.5	0.86	0.0	3.0	0.0	2.0	16.0	16.0	17.0	17.0
15	6.0	0.2	1.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	4.0	4.0	4.0	4.0
16	13.0	0.1	1.23	2.0	2.0	0.0	0.13	4.0	7.0	0.46	1.0	6.0	6.0	11.0	11.0
17	19.0	0.06	1.11	2.0	2.0	1.67	0.62	2.0	7.0	0.21	1.0	13.0	13.0	16.0	16.0
19	19.0	0.05	0.95	2.0	2.0	1.15	0.83	1.0	3.0	0.0	1.0	15.0	15.0	16.0	16.0
1	22.0	0.06	1.18	2.4	3.0	1.67	0.35	2.0	3.0	0.32	2.0	13.0	13.0	18.0	18.0
20	19.0	0.06	1.0	2.0	2.0	1.29	0.68	1.0	1.0	0.0	1.0	15.0	15.0	17.0	17.0
21	16.0	0.08	1.13	1.83	2.0	3.0	0.22	3.0	8.0	0.75	1.0	8.0	8.0	14.0	14.0
22	16.0	0.07	1.06	1.75	2.0	2.0	0.47	3.0	6.0	0.75	1.0	10.0	10.0	14.0	14.0
24	22.0	0.06	1.27	2.09	3.0	1.67	0.04	3.0	14.0	0.73	1.0	9.0	9.0	20.0	20.0
25	12.0	0.08	0.92	0.0	0.0	1.29	1.0	0.0	1.0	0.0	1.0	10.0	10.0	10.0	10.0
26	3.0	0.33	0.67	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0
28	10.0	0.14	1.3	3.0	4.0	0.0	0.0	1.0	7.0	0.8	1.0	4.0	4.0	8.0	8.0
29	12.0	0.09	1.0	2.0	2.0	0.0	0.5	1.0	3.0	0.83	1.0	8.0	8.0	10.0	10.0
3	14.0	0.1	1.36	1.5	2.0	3.0	0.68	1.0	4.0	0.36	7.0	11.0	11.0	13.0	13.0
6	25.0	0.04	1.04	2.0	2.0	1.25	0.77	2.0	5.0	0.0	1.0	21.0	21.0	23.0	23.0
7	18.0	0.07	1.11	2.0	2.0	0.0	0.55	2.0	7.0	0.0	1.0	13.0	13.0	16.0	16.0
8	20.0	0.06	1.1	3.0	3.0	2.0	0.27	1.0	7.0	0.0	2.0	14.0	14.0	18.0	18.0
9	37.0	0.03	1.14	5.0	5.0	1.4	0.43	1.0	11.0	0.0	3.0	29.0	29.0	33.0	33.0

Tabuľka 1: Trénovacia množina

name	price	m.day
10	4020.0	23.0
11	7350.0	55.0
12	2000.0	16.0
13	5180.0	35.0
14	8990.0	60.0
15	1300.0	9.0
16	10950.0	100.0
17	11550.0	94.0
19	7800.0	30.0
1	11250.0	74.0
20	6840.0	34.0
21	10930.0	101.0
22	8050.0	68.0
24	22650.0	173.0
25	4420.0	22.0
26	100.0	1.0
28	7600.0	65.0
29	5700.0	38.0
3	21750.0	150.0
6	13150.0	81.0
7	14050.0	93.0
8	15600.0	109.0
9	31900.0	190.0

Tabuľka 2: Výsledky odhadu úsilia

8 Obsah CD

Na priloženom CD sa nachádzajú nasledovné dokumenty:

- **application** - Adresár s aplikáciou pripravenou na spustenie.
- **effort-estimation** - Zdrojové súbory aplikácie.
- **test_model.zip** - Archív obsahujúci niekoľko testovacích modelov.
- **rafa014_program_documentation.pdf** - Programová dokumentácia k programu.
- **rafa014_user_documentation.pdf** - Užívateľská dokumentácia k programu k spusteniu a obsluhe programu.