Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Diplomová práce

Možnosti využití metodiky DEMO pro tvorbu BPMN modelů

Bc. Štěpán Heller

Vedoucí práce: Ing. Pavel Náplava

12. ledna 2016

Poděkování

Rád bych poděkoval mému vedoucímu Ing. Pavlu Náplavovi za cenné rady v průběhu tvorby práce a také Stevenu Van Kervelovi ze společnosti Formetis, jehož postřeby byly velkým přínosem pro vznik této práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 12. ledna 2016 .....

Odkaz na tuto práci

Heller, Štěpán. *Možnosti využití metodiky DEMO pro tvorbu BPMN modelů.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato diplomová práce se zabývá kombinací silných stránek notace BPMN (Business Process Model and Notation) a metodologie DEMO (Design & Engineering Methodology for Organizations). Za silné stránky notace BPMN v tomto kontextu autor označuje zejména srozumitelnost pro business uživatele a dále širokou paletu nástrojů, které umožňují modelování podnikových procesů v BPMN. V případě metodologie DEMO autor identifikuje silné stránky v hlubokém teoretickém základu, který zajišťuje vytváření modelů, které jsou ontologicky kompletní, konzistentní a jsou oproštěné od implementačních detailů.

V této práci čtenář nalezne srovnání běžných technik pro modelování podnikových procesů a dále hlubokou analýzu BPMN a DEMO. Hlavním přínosem této diplomové práce je však předpis, jak mapovat základní koncepty metodologie DEMO na BPMN primitiva a především návrh metody, jak využít důležité teoretické koncepty metodologie DEMO pro tvorbu BPMN modelů, které jsou kompletní, konzistentní, jednoznačné a esenciální. Navržená metoda je dále demonstrována na příkladu.

Klíčová slova DEMO, Design & Engineering Methodology for Organizations, BPMN, Business Process Model and Notation, Enterprise ontology, Business Process Management, Proces, Procesní model

Abstract

This thesis combines the strong aspects of BPMN (Business Process Model and Notation) such as good understanding of the notation among business users and large amount of software tools available that support the notation with strong aspects of DEMO (Design & Engineering Methodology for Organizations) such as the profound theoretical background which assures the user that the model he gets as a result is ontologically complete, consistent and abstracts away from all unnecessary implementation details.

This thesis contains an analysis of available business process modelling techniques along with a thorough analysis of BPMN and DEMO. The main contribution of the thesis however is a proposal of a method that makes use of important theoretical concepts of DEMO in order to create BPMN models which are complete, consistent, unique and essential. The use of the method is then demonstrated on an example.

Keywords DEMO, Design & Engineering Methodology for Organizations, BPMN, Business Process Model and Notation, Enterprise ontology, Business Process Management, Process, Process Model

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Každá organizace (nezáleží zda firma, úřad nebo spolek) od určité velikosti začne řešit problémy s efektivitou a udržitelností růstu. Jedním z řešení, ať už k němu řídící pracovníci přistoupí vědomě či nevědomě, je nějaká forma *procesního řízení*.

Procesní řízení, jehož základům se podrobně věnuje kapitola 1, ve své podstatě znamená standardizaci opakujících se postupů, jejich zaznamenání ve formě, která umožní pozdější analýzu a optimalizaci za účelem zvyšování efektivity jejich provádění a eliminaci chyb, které mohou vzniknout.

Přístupy, jak procesy zaznamenávat, analyzovat a optimalizovat se vyvíjí stejně jako se vyvíjí organizace, společnost, požadavky zákazníků a v neposlední řadě technologie. Od intuitivního zaznamenávání průběhu procesů pomocí *vývojových diagramů*, které neumožňovaly nic jiného než prosté grafické znázornění posloupnosti aktivit, se vývoj posunul k automatizaci procesů pomocí výpočetních prostředků a analýze velkého množství dat o každém kroku analyzovaného procesu.

Tato práce se snaží přispět k tomu, aby bylo možné procesy zaznamenávat a analyzovat s větší mírou konzistence podle metody, jejíž návrh tato práce představuje v kapitole 5.

Motivace

V této práci jsou analyzovány dvě techniky použitelné k modelování podnikových procesů – DEMO a BPMN. BPMN je v současnosti zřejmě nejpoužívanější notací pro vizuální reprezentaci podnikových procesů. Její předností je zejména velká srozumitelnost pro business uživatele, kteří jsou dobře obeznámeni s vývojovými diagramy a jsou tak schopni číst i vytvářet diagramy v BPMN bez větších problémů. Slabinou BPMN je však absence jasných pravidel *jak* diagramy vytvářet, *které* části procesu v nich zaznamenávat a *z čeho*

se procesy vlastně skládají. Výsledkem jsou BPMN modely, které jsou často *nekompletní*, *nekonzistentní* a *nejednoznačné*.

DEMO je metodologie založená na silném teoretickém základu, který se skládá především z *Enterprise ontology* a Ψ*-theory*. DEMO má jasná pravidla co v modelech zachycovat, jak při vytváření modelů postupovat a jak ověřit, zda jsou vzniklé modely korektní a správné. Metodologie DEMO zajišťuje, že pokud dodržíme veškeré postupy, které tato metodologie stanoví, tak nám při modelování stejného procesu musí na konci vždy vzniknout ten samý model. Tato jistota má zásadní pozitivní důsledky pro analýzu, sdílení i diskusi nad procesy v rámci organizace.

Tato práce si klade za cíl zkombinovat výhody obou technik, tedy dobrou srozumitelnost business uživateli na straně jedné a pevný teoretický základ na straně druhé, vyvinutím metody, která umožní vytvářet BPMN modely, které budou *kompletní*, *konzistentní* a *jednoznačné*. V kombinaci s možnostmi automatizace by se mohlo jednat o krok dopředu v celé oblasti Business Process Managementu.

Struktura práce

Práce je rozdělena do šesti kapitol. V první jsou definovány základní pojmy, se kterými je ve zbytku textu pracováno a také je zde rozebrán vývoj přístupů k práci s podnikovými procesy. Druhá kapitola analyzuje nejpoužívanější techniky pro modelování podnikových procesů a srovnává jejich silné a slabé stránky. Třetí a čtvrtá kapitola rozebírají notaci BPMN, respektive metodologii DEMO do hloubky – jejich základní principy a postupy. V páté kapitole se nachází klíčová část této práce, kterou je jednak jednak návrh toho, jak vyjádřit klíčové prvky metodologie DEMO pomocí primitiv z notace BPMN a také návrh metody, která obsahuje sedm kroků dle kterých je možné vytvořit BPMN model procesu. Výsledný model by měl být *kompletní*, *konzistentní* a *jednoznačný*. Závěrečná kapitola demonstruje navrženou metodu na konkrétním příkladu a diskutuje výsledky její aplikace.

Překlad cizojazyčných termínů

Při psaní textu se autor potýkal s problémem, že zejména v případě metodologie DEMO existuje jen mizivé množství textů v českém jazyce, které by popisovaly tuto metodologii. Základní termíny, kterými jsou označeny jednotlivé prvky metodologie, tak nemají český překlad. Autor se rozhodl toto řešit částečným překladem těch termínů, u kterých je přeložení přímočaré a jinak pracoval s původními anglickými výrazy. Při případné aktualizaci práce by bylo možné přeložit více termínů, pokud by se na českých ekvivalentech našla shoda v rámci české komunity DEMO.

Klíčové zdroje

Pro čerpání informací použil autor této diplomové práce několik desítek zdrojů, ale tři níže uvedené stojí za vypíchnutí a krátký komentář, neboť byly pro vznik této práce zásadní.

Enterprise ontology – Jan L. G. Dietz

Jan Dietz je tvůrcem metodologie DEMO a autorem celé řady publikací na téma fungování organizací, sociálních interakcích v nich, stejně jako na modelování jejich činnosti. Kniha *Enterprise ontology – Theory and Metodology* podrobně popisuje všechny tyto fenomény stejně jako Enterprise ontology, Ψ-theory a celou metodologii DEMO.

Enhancing the Formal Foundations of BPMN by Enterprise Ontology – Van Nuffel, Mulder, Van Kervel

Tato publikace je jednou z mála, které se zabývají nějakým druhem kombinace DEMO a BPMN. V tomto případě se jedná zejména o analýzu již existujících BPMN modelů z hlediska požadavků na ontologickou kompletnost a konzistentnost. Tato práce zároveň obsahuje postup, jak zajistit úpravu těchto BPMN modelů tak, aby tyto požadavky splňovaly.

V této souvislosti by autor rád zmínil, že v rámci tvorby této diplomové práce navštívil jednoho z autorů této publikace Stevena Van Kervela a jeho tým ve společnosti Formetis, za účelem diskutování závěrů a přínosů práce. Tato návštěva byla pro vznik této diplomové práce velkým přínosem.

Business Process Modeling and Simulation: DEMO, BORM and BPMN – Zuzana Vejražková

Třetí publikací, která byla zásadní pro vznik této diplomové práce je diplomová práce Zuzany Vejražkové, která vznikla na Fakultě informačních technologií ČVUT. Tato práce se zabývá analýzou technik DEMO, BORM a BPMN s důrazem na simulaci podnikových procesů, která umožňuje jejich efektivnější analýzu.

Rozdíl mezi touto diplomovou prací a prací Zuzany Vejražkové je jednak ve volbě technik, které jsou zkombinovány (Zuzana Vejražková kombinuje DEMO s Petriho sítěmi, tato práce kombinuje DEMO a BPMN) a jednak v menším důrazu na simulaci a automatizaci, kterou autor této diplomové práce přenechává dalšímu výzkumu.

Kapitola 1

Definice základních pojmů

1.1 Motivace k řízení podnikových procesů

Každá firma, která se snaží efektivně řídit svůj chod a neustále se rozvíjet, stále hledá nové a nové cesty, jak toho docílit. Takovými cestami může být uvádění nových produktů na trh, hledání nových trhů a příležitostí na nich, nabírání nových zaměstnanců, investice do propracovaného marketingu a mnoho dalších. Stále více firem ale v posledních několika dekádách obrací svou pozornost také dovnitř vlastní organizace. Hledají oblasti, kde je možné najít úspory nebo kde by bylo možné práci zefektivnit.

Aby bylo něco takového vůbec možné, musí mít manažeři a odpovědní vedoucí pracovníci především přehled o své organizaci a její hlubokou znalost. Pouze z takové hluboké znalosti pak mohou vzejít příležitosti k efektivnějšímu dosahování podnikových cílů.

Z toho důvodu firmy a organizace hledají cesty, jak lépe pochopit a následně standardizovat *procesy* v rámci vlastního podniku, které ve svém souhrnu nejsou nic jiného než soubor postupů, kterými podnik nebo organizace dosahuje svých cílů. O tom, kterak takové procesy pozorovat, standardizovat a řídit, byla napsána celá řada publikací a je nezbytné mít na zřeteli, že se jednotlivé přístupy od sebe více či méně odlišují.

Ačkoliv akademici i odpovědní lidé z prostředí samotných firem a organizací se často přou, který z přístupů je lepší, zůstává bez nejmenších pochyb, že adoptování jakéhokoliv přístupu vedoucího k lepšímu porozumění chodu vlastní organizace je lepší, než nahodilý přístup k řízení společnosti, kdy jsou změny vykonávány ad hoc a jakékoliv plánování do budoucna je tak velmi obtížné. Pochopení vlastní organizace je jedním (nikoli jediným) ze základních předpokladů pro její dlouhodobě udržitelný rozvoj a právě k tomu účelu je *procesní řízení* široce akceptovaným přístupem.

> Hlavním cílem procesního řízení je nalezení nejefektivnější cesty pro transformaci zákaznických požadavků na zákaznickou spoko-

jenost. [1] [1]

1.2 Vývoj přístupů k řízení podnikových procesů

Obsah této sekce vychází zejména z [2].

Velkou změnou v myšlení společností v moderních dějinách byla bezpochyby průmyslová revoluce v 18. a 19. století. Přechod k centralizované manufakturní výrobě nutil firmy přemýšlet nad efektivitou práce a vytvářet postupy (byť třeba neformální), které měly k požadované efektivitě přispívat.

Dalším velkým impulsem byl konec 2. světové války, kdy docházelo ke změně orientace z hospodářství orientovaného na válečná snažení jednotlivých států zpět k trhu, kde měl hlavní slovo zákazník. Firmy (zejména továrny a průmyslové závody) začaly více měřit, kolik jim konkrétní činnost zabere času a kolik spolyká finančních prostředků.

V 70. letech přicházejí na scénu první počítače, které budou mít zásadní dopad na to, jak organizace vnitřně fungují. Jedním z prvních rozpoznatelných přístupů tak byl *Just In Time (JIT)*, který vyvinula a dále precizovala japonská Toyota. Jak již název napovídá, praktiky JIT se soustředily na to, aby dodávky materiálu proběhly až ve chvíli, kdy byly potřeba při výrobním procesu. Firmy tak šetří finanční prostředky na nutnosti materiál držet na skladě a pečovat o něj.

Přístup označovaný jako *Total Quality Management (TQM)*, který zejména v 80. a 90. letech adoptují japonské továrny, se soustředí na maximální kvalitu ve všech fázích výrobního procesu a minimalizaci chybovosti. Hlavními cíli pro organizace jako Ford Motor Company, Motorola nebo Toyota Motor Company bylo dosáhnout 100% zákaznické spokojenosti a 0% chybovosti.

V druhé půlce 80. let firmy začaly pociťovat, že je potřeba začít lépe využívat statistická data nasbíraná za běhu procesů dle TQM. Tento přístup se nazývá *Business Process Improvement* a je využíván dodnes. Firmy se začaly více ptát *proč* proces vlastně děláme místo toho, *jak* jej vykonat efektivněji. Termín Business Process Improvement byl dnes již pohlcen obecnějším Business Process Management, stále je však možné se s ním někde setkat.

Koncept *Six Sigma* byl velmi populární v první polovině 90. let. Úspěch slavil především ve velkých společnostech jako Motorola nebo Sony. Six Sigma je soubor postupů, kterak zlepšit fungování procesů v organizaci eliminací chybovosti. *Six Sigma Defect* je definován jako cokoliv, co se odchyluje od očekávání zákazníka.

Přístup nazývaný jako *Lean* byl hojně využíván v druhé polovině 90. let. Jeho jádrem je snaha postupně eliminovat ztráty při výrobě za použití nejmenšího možného množství lidské práce, investic do nástrojů a času pro vývoj nového produktu.

[1] The main objective of process management is to find the most efficient and effective way to transform customer requirements into customer satisfaction. [1]

S příchodem 21. století se objevil nový přístup zvaný *Service Science*. Jedná se o přístup, který klade vyšší nároky na flexibilitu procesů ve společnosti. Firma by měla být schopná se plně přizpůsobit požadavkům zákazníka místo toho, aby mu nabízela soubor předem definovaných produktů nebo služeb. Dobrým příkladem využití přístupu Service Science je automobilka, která nevyrobí auto, dokud si ho zákazník neobjedná a může tak při jeho výrobě plně vycházet z požadavků zákazníka na úpravy na míru.

1.3 Definice základních pojmů

V rámci této sekce jsou definovány základní pojmy, jejichž znalost a plné porozumění je nezbytné pro orientaci v obsahu dalších kapitol.

1.3.1 Podnikový proces

V nadsázce řečeno definic pojmu *proces* existuje tolik, kolik existuje publikací, které jsou jim věnovány. Intuitivně si člověk s těmito definicemi neseznámený představí určitou po sobě jdoucí posloupnost aktivit, na jejichž konci je nějaký výsledek.

Norma EN ISO 9000:2000 definuje pojem proces následovně: [3]

> Proces je soubor vzájemně působících nebo vzájemně souvisejících činností, které přeměňují vstupy na výstupy. [2]

O něco podrobnější definici můžeme najít v [4]:

> Podnikový proces se skládá ze souboru činností, které jsou prováděny koordinovaně v organizačním a technickém prostředí. Tyto činnosti společně plní podnikový cíl. Každý podnikový proces je prováděn jednou organizací, ale může vzájemně působit s procesy prováděnými jinými organizacemi. [3]

Jako poslední si můžeme uvést definici podnikového procesu z české literatury: [5]

> Podnikový proces je souhrnem činností, transformujících souhrn vstupů do souhrnu výstupů (zboží nebo služeb) pro jiné lidi nebo procesy, používajíce k tomu lidi a nástroje.

[2] Set of interrelated or interacting activities which transforms inputs into outputs.

[3] A business process consists of a set of activities that are performed in coordination in an organizational and technical environment. These activities jointly realize a business goal. Each business process is enacted by a single organization, but it may interact with business processes performed by other organizations. [4]

I když je možné proces vnímat jako izolovanou jednotku, je na tomto místě dobré si uvědomit, že proces v organizaci je velmi často výstupem jiného procesu. Mezi hlavní atributy procesu patří: [6]

- *Název*, který proces identifikuje.
- *Účel*, pro který je proces prováděn.
- *Vlastník*, který je za proces zodpovědný (osoba nebo složka v organizaci).
- *Specifikace vstupů*. Věci, které jsou potřebné k provádění procesu.
- *Specifikace výstupů*. Věci, které jsou vytvořeny v průběhu provádění procesu.
- *Vstupní a výstupní podmínky*, které musí být splněny při spuštění a ukončení procesu.
- *Činnosti* definují jednotlivé kroky (operace) při provádění procesu.
- *Role a zodpovědnosti* definují, kdo je zodpovědný za provedení konkrétní činnosti.
- *Measurements*. Jaké parametry jsou vyhodnocovány při provádění procesu.

1.3.1.1 Produkt

Dle [3] je *produkt* definován jednoduše jako výsledek procesu. Produkt se dále může skládat z dalších produktů, které jsou výsledkem činnosti jiných procesů.

1.3.1.2 Procedura

Proceduru definujeme dle [3] jako *„určený způsob, jak vykonat činnost nebo proces“*. [4]

Pro správné pochopení rozdílu mezi procesem a procedurou je třeba si uvědomit, že proces nám říká „co je potřeba udělat, a které role jsou zastoupeny“ a procedura „jak to udělat“ a většinou se týká pouze jedné role. [6]

1.3.1.3 Instance

Instance podnikového procesu odpovídá jeho jednomu konkrétnímu provedení dle specifikovaného modelu.

[4] Specified way to carry out an activity or a process.[3]

1.3.2 Řízení podnikových procesů

Ačkoliv slovo „řízení“ v názvu sekce čtenáři již trochu napovídá, význam řízení podnikových procesů (často se používá také anglický výraz Business Process Management, zkráceně BPM) je širší než jen samotný dohled nad prováděním konkrétního procesu. Konkrétně je řízení podnikových procesů definováno jako: [4]

> Řízení podnikových procesů obsahuje principy, metody a techniky, které podporují návrh, konfiguraci, administraci, standardizaci a analýzu podnikových procesů. [5]

Ačkoliv taková definice může být pro čtenáře obtížně uchopitelná, nejedná se o nic jiného než o neustále se opakující proces, který zahrnuje mapování podnikových procesů, jejich modelování, provádění a dohled nad prováděním, sbírání dat, pomocí, kterých lze proces hodnotit, hodnocení a hledání cest, jak provádění nebo návrh procesu vylepšit tak, aby bylo jeho provádění efektivnější.

Hlavním cílem řízení podnikových procesů je nalezení co možná nejúčinnější cesty, kterou organizace může vyplnit požadavky zákazníka k jeho co možná nejvyšší spokojenosti. [7]

Proto, aby mohlo být řízení podnikových procesů v rámci organizace zavedeno, je nejprve nezbytné procesy definovat. Obvykle se definice podnikových procesů provádí manuálně sběrem informací od odpovědných pracovníků, kteří jsou odpovědní za provádění jednotlivých činností ideálně ve spolupráci s jejich nadřízenými pracovníky.

1.3.3 Business Process Management System

Při řízení podnikových procesů je pro firmy většinou výhodné, když použijí nějaký software, který jim umožní lepší dohled nad všemi aspekty životního cyklu procesu. Takový software označovaný jako BPMS (Business Process Management System) definujeme dle [4] jako:

> BPMS je obecný software, který využívá explicitní reprezentaci podnikových procesů k řízení jejich provádění. [6]

Jednou z charakteristik BPMS je, že není používán jen jedním typem uživatelů v rámci organizace, ale díky přizpůsobeným pohledům může sloužit napříč organizací. Mezi uživatele BPMS mohou patřit: [8]

- Procesní analytik

[5] Business process management includes concepts, methods, and techniques to support the design, administration, configuration, enactment, and analysis of business processes. [4]

[6] A business process management system is a generic software system that is driven by explicit process representations to coordinate the enactment of business processes. [4]

- Procesní architekt
- Koncoví uživatelé
- Liniový management
- Zákazníci
- Vývojáři

Přínosy používání některého BPMS systému tkví zejména v získání uceleného přehledu o procesech a jejich provádění v rámci organizace. Díky tomu je možné snáze identifikovat slabá místa, která přináší ztráty času nebo peněz. Pokud dojde při provádění procesu k nějakému problému, je díky BPMS snazší ho identifikovat a zjednat nápravu. BPMS systém rovněž usnadňuje provádění změn v nastavení procesů a také přispívá k lepšímu vyhodnocování výkonnosti firmy a predikci dalšího vývoje díky množství dat, které je možné nasbírat při provádění procesů.

1.3.4 Business Process Model

Pro potřeby standardizace, ale vlastně i obyčejné diskuse v rámci organizace, je nejprve třeba podnikový proces popsat. Popisují se jednotlivé činnosti, tok informací, zastoupené role, interakce s jinými procesy apod. Z toho důvodu je vytvářen tzv. *Business Proces Model*, který můžeme dle [4] definovat následovně:

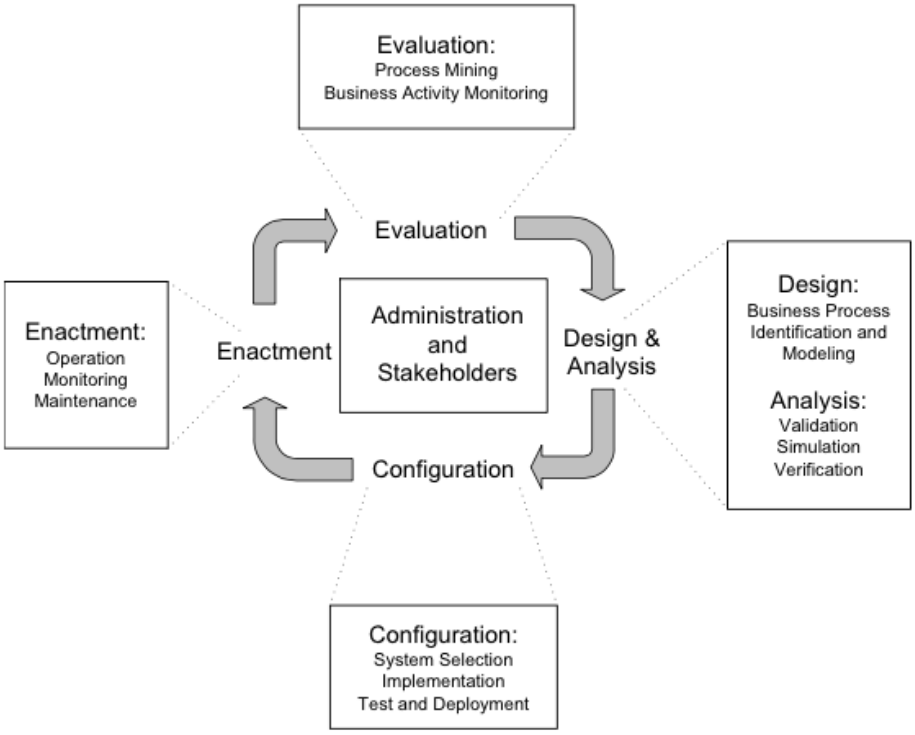
> Business Process Model se skládá ze souboru modelů činností a prováděcích pravidel mezi nimi. Každý Business Process Model představuje plán pro soubor instancí podnikového procesu a každý model činnosti představuje plán pro soubor instancí činnosti. [7]

Podnikové procesy v Business Process Modelu mohou být popsány textově nebo graficky. Grafický způsob v dnešní době jasně převládá zejména díky možnosti rychlejšího zorientování v procesu a jednoduššímu pochopení toho, jak proces probíhá.

Je žádoucí, aby byl Business Process model co nejlépe pochopitelný pro všechny zainteresované osoby. Právě grafické znázornění je pro toto nejvhodnějším řešením. Výsledný model by však neměl umožňovat více než jednu intepretaci situace, kterou znázorňuje.

Mezi známější nástroje pro grafické znázornění podnikových procesorů řadíme: [8]

[7] A business process model consists of a set of activity models and execution constraints between them. Each business process model acts as a blueprint for a set of business process instances, and each activity model acts as a blueprint for a set of activity instances. [4]



Obrázek 1.1: Business Process Lifecycle [4]

- Vývojový diagram (Flowchart)
- UML
- BPMN
- BPEL

Petriho sítě, Flowchart, UML, BPMN, BPEL a DEMO si podrobněji rozebereme v kapitolách 2, 3 a 4.

Business Process Model se soustředí zejména na strukturu a organizaci procesu, nikoliv na technické aspekty jeho implementace. Realizace procesu se tak může upravit, aniž by se musel měnit příslušný model.

V dalším textu budeme používat spíše české označení *procesní model*.

1.3.5 Životní cyklus procesu

Obrázek 1.1 ilustruje všechny fáze, kterými podnikový proces prochází. Obrázek také ilustruje ideu neustálé probíhající optimalizace procesu.

Ve fázi *Návrh a analýza* dochází ke sběru informací o procesech většinou formou pohovorů a dotazování zodpovědných pracovníků. Na základě takto

nasbíraných informací je možné procesy identifikovat, popsat, potvrdit se zodpovědnými pracovníky a vytvořit jejich procesní modely.

Po návrhové fázi většinou následuje fáze *Implementace*. Podnikový proces může být implementován například pouze „slovně“ pomocí souboru pravidel a opatření, které musí zodpovědní zaměstnanci dodržovat. Často je ale k realizaci podnikových procesů využíván specializovaný software. V takovém případě dochází k obohacení procesního modelu o specificky technické informace a také o doplnění interakcí pracovníků se softwarem.

Ve chvíli, kdy je dokončena implementace je možné přikročit k fázi *Provádění*. Provádění konkrétní instance podnikového procesu je většinou spuštěno nějakou událostí (např. jiným procesem). V této chvíli systém BPMS aktivně kontroluje provádění procesu a upozorňuje zodpovědné pracovníky na případné problémy nebo požadavky na součinnost. V průběhu této fáze jsou také shromažďována cenná data, která prováděním procesu vzniknou. Tato data slouží jako vstup poslední vyhodnocovací fáze.

Fáze *Vyhodnocení* využívá nasbíraná data, která jsou vyhodnocována a jsou v nich hledány příležitosti k optimalizaci procesu. Příkladem může být například, že některá část procesu trvá příliš dlouho z důvodu zpožďování dodávek materiálu. Na základě takové informace mohou manažeři přistoupit k úpravě procesu dodávání materiálů a dosáhnout tak časové úspory.

1.3.6 Capability Maturity Model

Capability Maturity Model (CMM) definuje pět stádií vyspělosti, kterými firmy a organizace procházejí při snaze porozumět vlastním podnikovým procesům. Jednotlivá stádia popisuje tabulka 1.1.

1.3.7 Klasifikace podnikových procesů

Podnikové procesy lze klasifikovat a zařazovat do tříd z několika hledisek.

1.3.7.1 Hlavní a podpůrné procesy

Základní a zřejmě nejlépe uchopitelné hledisko je rozdělení podnikových procesů na *hlavní* a *podpůrné*.

Hlavní procesy jsou takové, které přímo vytvářejí hodnotu pro organizaci. Aby mohl být proces označen jako hlavní, je zde třeba přímé interakce se zákazníkem. Hlavních procesů není většinou v organizaci mnoho, spíše tyto procesy kopírují nabídku služeb zákazníkům.

Podpůrné procesy slouží, jak už z názvu vyplývá, jako podpora procesů hlavních. Tyto procesy přímo hodnotu většinou nevytvářejí, ale jsou nezbytné k jejímu vytváření hlavním procesem. Na rozdíl od hlavních procesů je možné (a často výhodné) podpůrné procesy outsourcovat.

Stádium vyspělosti	Stav
1. Initial	Procesy jsou vykonávány ad hoc bez jasně definovaných postupů. Úspěch procesu závisí zejména na výkonu zodpovědných pracovníků.
2. Repeatable	Základní techniky projektového řízení jsou již zavedeny pro sledování nákladů, plánování a definování funkcionalit. Je zaveden základní rámec, který umožňuje opakovat předchozí úspěšné provedení nějakého úkonu.
3. Defined	V této fázi je proces již dokumentován a standardy pro jeho provádění jsou definovány.
4. Managed	Probíhá sběr výkonnostních měřítek pro vyhodnocování kvality provádění procesů. Procesy jsou prováděny pod dohledem.
5. Optimizing	Procesy jsou na základě nasbíraných dat a zpětné vazby neustále vylepšovány za účelem jejich zefektivnění.

Tabulka 1.1: Jednotlivá stádia vyspělosti v práci s podnikovými procesy dle modelu CMM [9]

1.3.7.2 Vnitřní a vnější procesy

Vnitřní proces je takový proces, který probíhá uvnitř jedné organizace bez interakce s procesem jiné organizace.

Vnější procesy naopak interagují s procesy jiné organizace a je tedy třeba u nich dbát na optimální synchronizaci.

1.3.7.3 Další členění podnikových procesů

Podnikové procesy můžeme členit dle mnoha dalších hledisek. Jedním z takových hledisek je například *úroveň automatizace* konkrétního podnikového procesu. Některé procesy mohou probíhat plně automatizovaně bez zapojení člověka, u jiných je menší či větší zapojení lidského elementu nezbytné.

Dalším možným pohledem je *úroveň opakování*. Podnikové procesy, u kterých dochází k častému opakování, často podléhají vyšší úrovni automatizace, ale není to pravidlem. Existují však v organizaci i procesy, které se opakují jen výjimečně. Jako příklad si můžeme uvést návrh nového modelu auta v automobilce.

1.3.8 Klíčové role při řízení podnikových procesů

V každé organizaci, která se zabývá řízením vlastních podnikových procesů, existuje několik zainteresovaných rolí osob, které jsou pro fungující řízení nezbytné. Mezi klíčové role dle [4] patří:

- *Chief Process Officer*: Chief Process Officer je zodpovědný za standardizaci a správné fungování procesů v rámci organizace. Dále má také na starosti přizpůsobování podnikových procesů novým požadavkům trhu.
- *Business Engineer*: Pracovníci v této roli jsou zodpovědní za definování strategických cílů společnosti a organizačních podnikových procesů.
- *Process Designer*: Tito pracovníci mají na starost modelování podnikových procesů za neustálé probíhající komunikace s Business Engineer a dalšími zainteresovanými rolemi.
- *Process Participant*: Pracovníci, kteří provádějí skutečné operativní činnosti při provádění samotného procesu.
- *Knowledge Worker*: Účastníci procesu, kteří používají softwarové nástroje k provádění činností v rámci procesu.
- *Process Responsible*: Osoba zodpovědná za správné provedení všech procesů používajících jeden konkrétní procesní model.
- *System Architect*: Lidé zodpovědní za vývoj a nastavení BPMS v organizaci.
- *Vývojáři*: Vývojáři mají na starost vývoj softwarových prostředků nezbytných pro implementaci podnikových procesů.

Pro optimální fungování řízení podnikových procesů je nezbytná úzká spolupráce všech výše uvedených rolí.

Kapitola 2

Techniky modelování podnikových procesů

2.1 Procesní model a důvody pro jeho tvorbu

Ať už člověk vytváří jakýkoliv model, jeho cílem je zachytit nějaký jev, který je potřeba kvůli své komplexnosti zobrazit zjednodušenou vizuální formou, která bude pochopitelná i pro jiné lidi než je sám tvůrce modelu. Umět jev zachytit ve formě modelu je jedním z prvních kroků na cestě k tomu tento jev upravovat.

Přeneseno do světa podnikových procesů je situace velmi podobná. Jedním z hlavních důvodů, proč organizace přistupují k BPM je potřeba procesy upravovat a zejména optimalizovat. Aby to bylo možné, je potřeba nejdřív stanovit metriky a tyto metriky být pak schopen měřit. Základem pro všechny tyto kroky je ale korektní procesní model, který proces věrně popisuje.

2.1.1 Definice procesního modelu

Základní definice procesního modelu podle [10]:

> Procesní model je konceptualizací podnikového procesu v organizaci. [8]

Čtenářsky přístupnější definici pak nabízí [11]:

> Procesní model popisuje (většinou grafickou formou) aktivity, události, jejich pořadí a propojení, které utváří podnikový proces. [9]

[8] Process model is a conceptualization of the (business) process in an enterprise. [10]

[9] Process model describe, typically in a graphical way, the activities, events and control flow logic that constitues a business process.

2.2 Základní techniky

V této sekci si popíšeme populární techniky pro tvorbu procesních modelů.

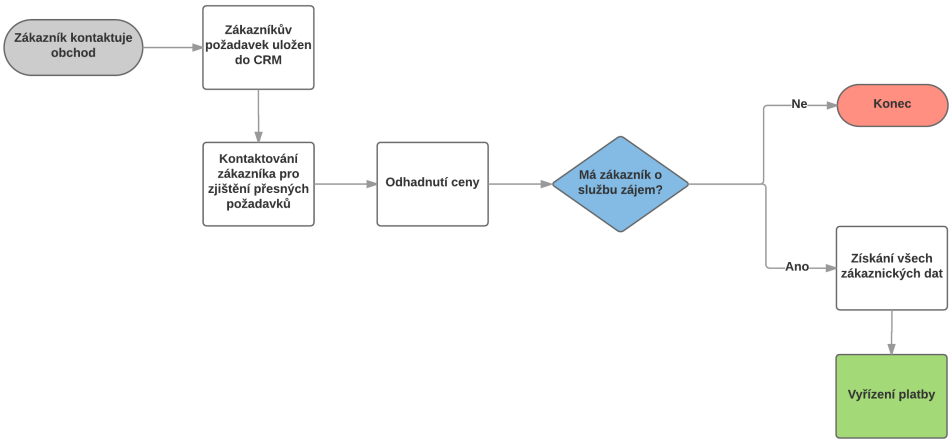
2.2.1 Vývojový diagram (Flowchart)

Vývojový diagram je pravděpodobně nejpopulárnější technikou pro modelování podnikových procesů. Vděčí za to zejména své jednoduchosti, dostupnosti mnoha nástrojů, které tuto techniku podporují a také její velké srozumitelnosti, která jí činí velmi snadno uchopitelnou i pro ty uživatele v organizaci, kteří nejsou příliš obeznámeni s problematikou modelování podnikových procesů.

2.2.1.1 Základní pravidla

Vývojové diagramy se skládají z několika málo základních symbolů. Tyto symboly se nazývají: [12]

- *Startovací a ukončovací symboly* – používají se pro vyznačení začátku a konce procesu.
- *Šipky* – zobrazují tzv. „řídící tok“, tedy přechod v čase mezi jednotlivými symboly.
- *Dílčí kroky procesu* – jsou reprezentovány obdélníkem.
- *Podprogramy* – zobrazeny obdélníkem se svislými čarami po stranách. Používají se pro zobrazení skupiny kroků procesu pomocí jediného symbolu.
- *Vstupy a výstupy* – zobrazují tok informací směrem dovnitř i vně procesu. Pro jejich reprezentaci se používají lichoběžníky respektive rovnoběžníky.
- *Podmíněný cyklus* – zobrazuje událost, která se opakuje dokud je splněna jasně definovaná podmínka. Zobrazuje se pomocí šestiúhelníku.
- *Podmíněný výraz* – kosočtvercem je symbolizováno rozhodnutí a určuje tedy místo, kde dochází k větvení procesu.
- *Spojovací symbol* – inverzním symbolem ke kosočtverci je ve vývojovém diagramu kruh, který se používá ke spojení více řídících toků do jednoho.



Obrázek 2.1: Nákupní proces pomocí vývojového diagramu

2.2.1.2 Výhody a nevýhody

Nespornou výhodou vývojových diagramů je právě jejich přístupnost pro uživatele a velmi strmá křivka učení, což dělá z této techniky první volbu pro případy, kdy je potřeba velmi rychle vymodelovat nějaký proces a organizace nemá zavedeny sofistikovanější metody BPM. Vývojové diagramy umožňují efektivnější komunikaci o problému v rámci týmu.

Největší přednost vývojových diagramů je zároveň jejich největší slabinou. Právě přílišná jednoduchost této techniky dělá z modelování komplexnějších procesů poměrně komplikovanou a nepřehlednou záležitost. Ve vývojových diagramech je také složitější modelovat některé jevy, jako například tzv. „unhappy paths“ a další nestandardní události, která však v životě procesů nastávají poměrně běžně. U vývojových diagramů je také obtížné dělat změny, protože to často vyžaduje kompletní překreslení celého diagramu.

2.2.1.3 Použití

Vývojové diagramy mají mnoho využití. Hodí se například pro komunikaci mezi organizací a jejími externími zákazníky, protože se dá předpokládat, že se s vývojovými diagramy už v minulosti setkali a budou jim tedy rozumět. Vhodné je také použít vývojový diagram v dokumentaci k softwaru nebo jinému systému, kterou budou číst různorodé skupiny uživatelů.

2.2.2 BPMN

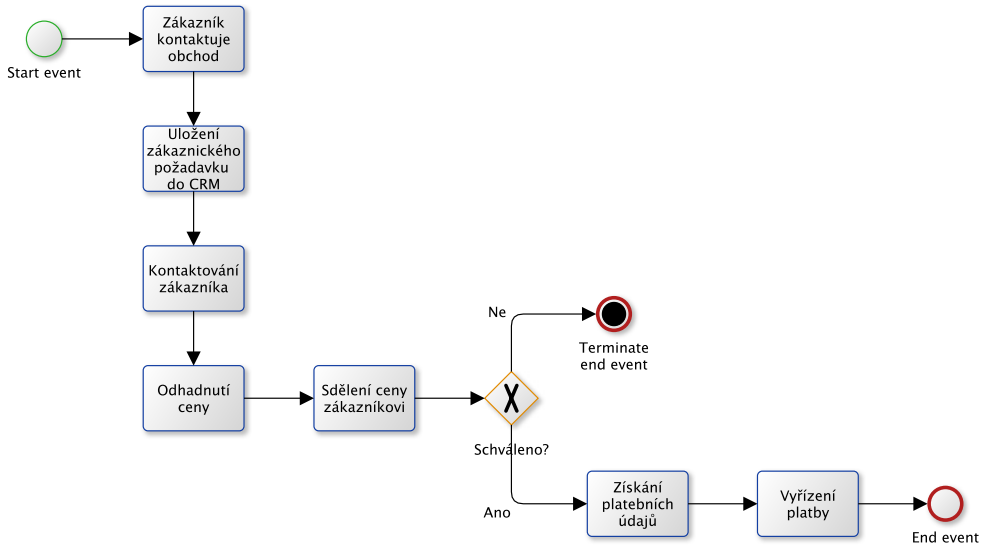
BPMN nebo celým názvem *Business Process Model and Notation* je v současnosti de facto standardem na poli modelování podnikových procesů. Jeho využití je široké od IT přes obchod až například po komplexní dopravní systémy. Za svou popularitu vděčí zejména tomu, že se stále jedná o dobře pocho-

pitelnou grafickou notaci, která je svým vzhledem velmi podobná vývojovým diagramům, ale na rozdíl od nich je BPMN standardem, tedy má oficiální specifikaci s popisem jednotlivých elementů a pravidel, jak je používat.

S trochou nadsázky by se dalo říct, že BPMN je vlastně rozšířením vývojového diagramu. Je určitě pravdou, že se BPMN touto jednoduchou technikou v mnohém inspirovalo a na jejích základech postavilo notaci, která umožňuje poměrně jednoduše modelovat i komplexní podnikové procesy a zároveň si stále uchovává dobrou srozumitelnost pro uživatele.

2.2.2.1 Základní pravidla

Jelikož se BPMN budeme podrobně věnovat v kapitole 3 nemá smysl na tomto místě zacházet do přílišných detailů. Zatím si vystačíme s tím, že v BPMN jsou zásadními objekty *aktivity*, *události*, *brány*, *počáteční a koncové symboly* a *„šipky“* neboli symboly reprezentující *sekvenční tok* procesu nebo zasílání *zpráv*. Vzhledově se tyto symboly příliš neliší od korespondujících symbolů ve vývojovém diagramu.



Obrázek 2.2: Nákupní proces pomocí BPMN

2.2.2.2 Výhody a nevýhody

Mezi hlavní výhody BPMN určitě můžeme zařadit fakt, že BPMN je *standard*, tedy je přesně definované, co který symbol vyjadřuje. O BPMN se stará organizace OMG (Object Management Group) a kontinuálně pracuje na jeho rozvoji. Díky širokému rozšíření BPMN existuje na trhu velké množství placených i neplacených nástrojů, které umožňují modelování podnikových procesů pomocí této notace. Stránka `https://bpmnmatrix.github.io` eviduje 58 dostupných nástrojů pro vytváření modelů podnikových procesů v BPMN.

Další neoddiskutovatelnou předností je srozumitelnost notace, která je vysoká právě díky své podobnosti s vývojovými diagramy. Bez nutnosti zdlouhavého studia dokumentace je BPMN modelu schopen porozumět člověk z managementu společnosti a stejně tak i softwarový inženýr nebo vývojář. Právě pro ty ukrývá BPMN další výhodu a tou je poměrně přímočará převoditelnost BPMN modelů do strojově čitelných formátů, jako je jazyk XML nebo na něm založený jazyk BPEL.

Ačkoliv je používání notace BPMN částečně definované v dokumentaci, reálné procesy v organizacích mohou být obtížně modelovatelné bez hluboké znalosti BPMN a může tedy docházet k vytváření nekorektních modelů nebo více různých modelů toho samého jevu. To je důsledkem absence metodologie, která by předepisovala postup pro modelování podnikového procesu v BPMN, jeho strukturu a další pravidla. Dalším problémem dle [13] je, že někteří výrobci softwaru pro modelování v BPMN si tento standard ohýbají podle sebe nebo ho „obohacují“ o vlastní „vylepšení“, která pak dělají takto vytvořené modely obtížně přenositelnými.

2.2.2.3 Použití

Organizace OMG, která BPMN nyní spravuje, uvádí jako hlavní poslání BPMN *přenositelnost* procesních modelů vytvořených v této notaci bez závislosti na tvůrcích konkrétního modelovacího nástroje. BPMN je vhodné pro modelování podnikových procesů v celé jejich šíři.

2.2.3 BPEL

BPEL neboli Business Process Execution Langauge (a správněji WS-BPEL Web Service Business Process Execution Language) je v našem výčtu jedinou technikou pro modelování podnikových procesů, který nemá grafickou reprezentaci. Je to totiž *jazyk*. Své využití nachází zejména při automatizaci podnikových procesů. BPEL je založen na XML a v podstatě standardizuje definici podnikových procesů právě pomocí XML. [14]

2.2.4 UML

UML neboli *Unified Modeling Language* je velmi populární grafický jazyk, zvláště v oblasti IT a vývoje softwarových systémů. Jak píše [15] právě s tímhle cílem bylo UML původně také vytvořeno. Jenže jeho popularita se rychle rozšířila i do světa byznysu a UML přestalo dostačovat potřebám svých uživatelů. Proto bylo postupně rozšiřováno o další aspekty, které pokrývaly modelování podnikových procesů v celé jejich šíři.

UML obsahuje standardizovaný mechanismus jak jazyk rozšiřovat tak, aby jeho obecné principy mohly být doplněny o další vyhovující specifickému účelu, jako je například právě modelování podnikových procesů. Proto byl již v době vzniku UML vytvořen standardní profil pro modelování podnikového procesu

[5]. Tento profil pracuje zejména s *Diagramem tříd* a s *Diagramem Use-Case*. Diagram Use-Case je v tomto profilu používán pro zobrazení podnikových procesů a jejich interakce s aktéry a zákazníky. Oproti tomu Diagram tříd se používá spíše k zobrazení vnitřní struktury popisované organizace. Faktem je, že se standardní profil pro modelování podnikového procesu v praxi příliš neujal, snad kvůli své přílišné snaze podnikové procesy přiblížit k IT a informačním systémům. [5]

UML se přesto pro modelování podnikových procesů používá a to zejména v neformální podobě, kdy je organizacemi používán především *Diagram aktivit*, který je pro modelování podnikových procesů vhodný. Diagram aktivit totiž umožňuje sekvenční i paralelní zobrazování aktivit a objekty na vstupu i výstupu procesu a také závislosti mezi jednotlivými aktivitami.

Pro větší přenositelnost a potřeby standardizace UML jazyka pro modelování podnikových procesů však vznikla celá řada rozšíření třetích stran. Mezi nejpoužívanější pak patří rozšíření podle H. Erikssona. [5, 16].

> Erikssonův přístup je nejenom rozšířením UML, ale do značné míry plnohodnotnou metodou modelování procesů – určuje sadu modelů a diagramů, postavených vesměs na standardních diagramech UML. [5]

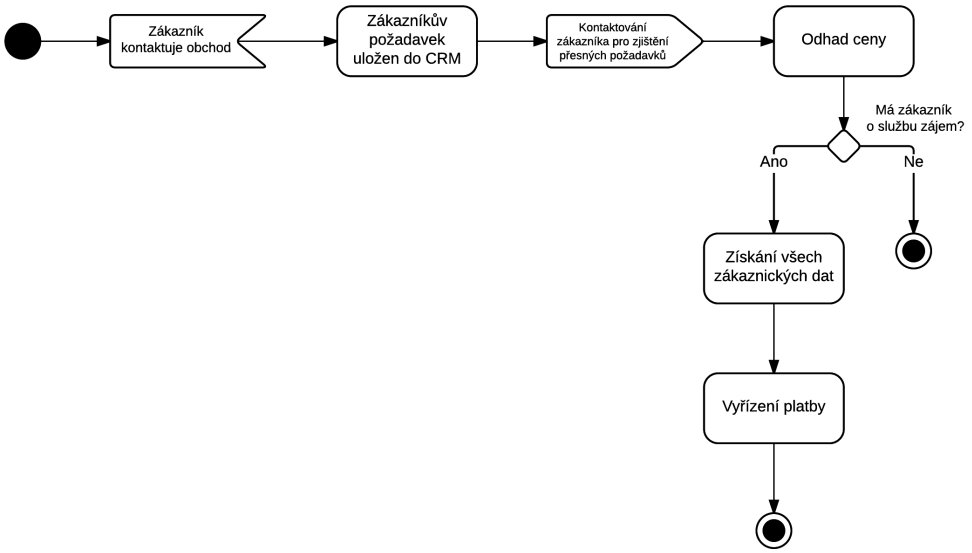
Erikssonovo rozšíření obsahuje více různých diagramů, přičemž pro potřeby samotného modelování podnikového procesu je nejzásadnější *Diagram procesů*, který je rozšířením právě již výše zmíněného Diagramu aktivit. Právě tomu se budeme v této kapitole šířeji věnovat.

2.2.4.1 Základní pravidla

Diagram aktivit obsahuje několik základních prvků: [17]

- *Aktivita* – proces, který se modeluje.
- *Akce* – jedná se o konkrétní činnost v rámci aktivity.
- *Zahájení a ukončení* – počáteční a koncový uzel.
- *Řídící tok* – určuje pořadí vykonávání jednotlivých kroků aktivity, znázorněné pomocí šipek.
- *Rozhodnutí* – rozvětvení procesu na základě určité podmínky.
- *Spojení* – spojuje více toků do jednoho.
- *Rozdělovník a spojovník* – pro provádění více akcí paralelně se používá rozdělovník a spojovník.

- *Příchozí událost* – používá se, pokud je k provedení některých kroků potřeba akce zvenčí.
- *Spuštění události* – používá se, pokud je potřeba znázornit akci, která spouští jiný proces nebo akci.



Obrázek 2.3: Nákupní proces pomocí UML

2.2.4.2 Výhody a nevýhody

Mezi výhody UML můžeme určitě zařadit velkou rozšířenost tohoto jazyka a s tím spojenou i velkou dostupnost nástrojů umožňujících modelování v UML, literatury i komunit, kde je možno hledat pomoc či radu.

Jako nevýhodu lze určitě uvést faktickou neexistenci standardu pro modelování podnikových procesů v UML. Standardní profil pro modelování podnikového procesu, který UML obsahuje, se příliš nepoužívá a od verze UML 2.0 už dokonce není součástí množiny standardních profilů [5]. Existují sice další rozšíření, ale ty nejsou standardem, což snižuje možnost jejich přenositelnosti i udržitelnosti do budoucna. UML je přeci jen jazyk stále velmi blízký vývoji softwaru a modelování podnikových procesů spíše umožňuje, než podporuje.

2.2.4.3 Použití

Jak už bylo řečeno výše, UML nachází uplatnění v mnoha oborech lidské činnosti, ale prosadilo se zejména v IT a tvorbě softwaru. Některé nástroje, které umožňují modelování v UML, dokonce nabízí možnost z UML modelu exportovat přímo zdrojový kód například pro tvorbu databáze.

Jelikož UML obsahuje různé druhy modelů, dostává tím celý jazyk další možnosti využití i mimo IT. Síla UML je v zachycení struktury jevu a již tolik nezáleží na tom, jestli se jedná o databázi nebo organizační strukturu podniku. Například Use-Case diagram najde využití nejen při modelování chování člověka při interakci se softwarem, ale například také obsluhy zákazníka v bance apod.

UML je tak mimo jiné využíváno v IT, bankovnictví, telekomunikacích, zdravotnictví nebo v obranném průmyslu.

2.2.5 DEMO

DEMO neboli *Design & Engineering Methodology for Organizations* je metodologie se silným teoretickým základem určená k modelování, analýze a grafickému zobrazení podnikových procesů. Za jejím vznikem stojí zejména Jan Dietz. DEMO obsahuje 4 modely, každý pro jiný účel a jiný pohled na proces.

> Tímto způsobem jsme schopni rozmotat uzel, který dnešní organizace připomínají, odstranit chyby v jejich návrhu a přitom zajistit, aby vše fungovalo. Stejně jako by inženýr opravil most, letadlo nebo počítač. [18] [10]

Fundamentální vlastnosti, který by měl procesní model vytvořený v metodologii DEMO splňovat dle [10] jsou:

- jednoznačnost,
- konzistentnost,
- kompletnost,
- výstižnost,
- obsahovat jen nezbytné množství informací.

Zjednodušeně řečeno, důvodem pro vytvoření této metodologie byla narůstající nespokojenost se stavem jiných metodologií, notací a jazyků obvykle používaných k modelování podnikových procesů. Modely vytvořené v těchto nástrojích jsou totiž většinou příliš podrobné a příliš technicky zaměřené, takže stěžují pohled na organizaci z vrchu – jen na to důležité, co se v ní odehrává.

DEMO je naopak postaveno na modelování podnikových procesů pomocí *ontologií*, což znamená, že je zachyceno jen jádro problému, které většinou tvoří *komunikace* mezi lidmi. DEMO má silný teoretický základ v teorii PSI, o které si více řekneme v kapitole 4.

[10] This way we can untangle the complex knot that organizations have become, fix the constructional mistakes, and put everything back together. Just like an engineer would repair a bridge, airplane or computer. [18]

2.2.5.1 Základní pravidla

Detailnímu popisu modelování v DEMO se věnuje kapitola 4, takže na tomto místě zmíníme jen základní věci. Kompletní model organizace (tzv. *Essential model*) se skládá ze 4 různých modelů: [19]

1. Construction Model (CM)
2. Process Model (PM)
3. Action Model (AM)
4. State Model (SM)

Jedním z nejdůležitějších aspektů metodologie DEMO je tzv. *transaction pattern*, který dává přesnou strukturu tomu, jak probíhají transakce (například objednávka). V DEMO se takové transakce skládají vždy ze stejných kroků a tudíž mají všechny stejnou strukturu, což zanechává malý prostor pro více interpretací stejné transakce.

Nejdůležitější aspekty v DEMO jsou dva:

1. Ontologická transakce
2. Actor

Podrobně budou tyto elementy rozebrány v kapitole 4.

2.2.5.2 Výhody a nevýhody

Mezi silné stránky metodologie DEMO určitě musíme zařadit absolutní *jednoznačnost* modelů vzniklých dle této metodologie. Zatímco u jiných technik, jako je třeba vývojový diagram, ale i BPMN nebo UML, vzhledem k jejich poměrně vysoké úrovni detailu, může docházet k nejednoznačnostem, tj. že stejný jev je vymodelován různě. V DEMO díky jeho poměrně rigidním pravidlům a modelu postavenému na jasně strukturovaných transakcích dostáváme stejné modely pro stejné jevy, což je velmi pozitivní pro celkovou konzistenci BPM v organizaci a zároveň to usnadňuje analýzu a vylepšování procesů.

Jako hlavní přednost DEMO uvádí [20] schopnost modelů vytvořených pomocí této metodologie zachytit pouze základní podstatu každé organizace a schopnost modely abstrahovat od technických detailů, které jsou pro účely modelování organizací podružné.

Jako nevýhodu celé metodologie je určitě nutné označit poměrně dlouhou a pozvolnou křivku učení, která je v tomto případě dána dvěma věcmi:

1. Široký teoretický základ, který sahá až do oblasti ontologie, filozofie a dalších oborů. I samotná metodologie má za sebou velmi robustní teorii, která není na první pohled zřejmá.

2. Na rozdíl od vývojových diagramů nebo BPMN jsou modely vytvořené v DEMO pro člověka nezasvěceného do toho, jak DEMO funguje, prakticky nečitelné.

Když by chtěla organizace začít používat DEMO pro modelování svých procesů, je nutné investovat čas i finanční prostředky do zasvěcení odpovědných lidí do metodologie a jejího používání, přičemž vycvičení lidí na potřebnou úroveň může být poměrně zdlouhavé. Právě toto by mohlo být jednou z hlavních překážek, které brání většímu rozšíření metodologie DEMO mimo akademickou půdu. Další komplikací je pak velmi malé množství dostupných nástrojů pro vytváření DEMO modelů. Oficiální stránka věnovaná metodologii DEMO uvádí na `http://www.ee-institute.org/en/demo/tools` pouze 4 nástroje, z nichž žádný nepodporuje všechny modely, které DEMO obsahuje.

2.2.5.3 Použití

Tato část je založena zejména na [19], kde jsou uvedeny nejméně 3 možné způsoby, jak využít DEMO:

1. **Návrh a optimalizace organizací** – Díky přednostem metodologie DEMO, které jsou popsány výše, je možné se na organizaci podívat z ontologického hlediska, což umožňuje snáze pochopit, jak organizace skutečně funguje a je možné fungování jednotlivých procesů vylepšit a zefektivnit.

2. **Softwarová podpora organizace** – Velká část podnikových procesů je dnes podporována IT systémy. DEMO rozděluje druhy softwaru do tří úrovní, aby korespondovaly se strukturou organizace tak, jak jí vidí DEMO (ontologická, infologická, datalogická). Pomáhá tak jasně určit strukturu používaného softwaru uvnitř organizace.

3. **Vývoj softwaru** – Ačkoliv DEMO abstrahuje při modelování procesů od implementačních detailů, přesto může být užitečné při jeho vývoji. DEMO modely totiž mohou sloužit jako odrazový můstek na začátku vývoje. DEMO model totiž obsahuje základní informace o každém procesu: kdo ho iniciuje, kdo ho provádí a jaké aktivity obsahuje a v jakém pořadí. Dle [21] jsou DEMO modely velmi snadno převoditelné do Use Case modelů.

2.3 Srovnání technik

Tato sekce se soustředí na porovnání objektivních měřítek, podle kterých můžeme jednotlivé techniky porovnávat. Její ambicí rozhodně není rozhodnout,

která z nich je „lepší“ nebo „horší“, jelikož to vždy závisí především na konkrétním účelu, ke kterým chceme danou techniku použít. Z porovnání je vyjmut BPEL, jelikož nepodporuje grafické znázornění procesu.

Tabulka 2.1: Srovnání základních technik pro modelování podnikových procesů

	Vývojový diagram	**UML**	**BPMN**	**DEMO**
Použití	Používá se pro modelování podnikových procesů, pracovních postupů, kroků algoritmu atd.	Používá se pro zobrazení a návrh softwaru, kroků algoritmu, interakce uživatele s aplikací, organizační struktury, datových toků, rozhodovacích postupů atd.	BPMN je používáno exkluzivně pro modelování podnikových procesů v celé řadě odvětví. Verze 2.0 navíc nabízí větší podporu i pro implementaci procesů pomocí IT systémů.	DEMO se používá k zobrazení základních interakcí uvnitř organizace, může být tak využito pro její návrh nebo optimalizaci. Modely vytvořené v DEMO jsou rovněž vhodným základem pro tvorbu softwaru.
Dostupnost nástrojů	Na trhu existuje celá řada nástrojů pro modelování vývojových diagramů a to jak volně dostupných, tak komerčních. Existují i on-line nástroje.	Pro UML je dostupné velké množství volného i komerčního software. Existují i on-line nástroje.	Modelování v BPMN umožňuje celá řada aplikací, velká část z nich je komerčních.	Aplikací, které podporují modelování v DEMO jsou na trhu jen jednotky. Jejich výčet je dostupný na webu `http://www.ee-institute.org`. Jsou mezi nimi komerční i nekomerční aplikace.

Tabulka pokračuje na další straně

Tabulka 2.1 – *Pokračování tabulky z předchozí strany*

	Vývojový diagram	**UML**	**BPMN**	**DEMO**
Dostupnost literatury	O vývojových diagramech existuje mnoho literatury, nicméně z důvodu neexistence standardizujících pravidel se mohou ve výkladu lišit.	Díky své celosvětové popularitě existuje o UML opravdu velké množství literatury. Přímo na stránkách organizace OMG `http://www.omg.org` je k dispozici celá řada materiálů.	O BPMN se (stejně jako o UML) stará organizace OMG. Vzhledem k tomu, že BPMN je velmi rozšířené, existuje velké množství dostupné literatury.	Literatury zabývající se metodologií DEMO je na trhu poměrně málo. Významná část je napsaná přímo autorem celé metodologie Janem Dietzem. Základní literatura se dá dohledat na webu `http://www.ee-institute.org` v anglickém nebo v holandském jazyce.
Velikost komunity	Vzhledem k faktu, že na rozdíl od ostatních technik neexistuje organizace, která by se starala o vývojové diagramy, těžko lze v tomto případě hovořit o existenci jasně identifikovatelné komunity akademiků, organizací nebo autorů literatury.	Komunita kolem UML je široká. UML je vyučováno na univerzitách, v podnicích i dalších organizacích. Je k dispozici velké množství literatury i nástrojů.	I BPMN má kolem sebe širokou komunitu autorů, lektorů i organizací, které se zabývají tvorbou BPMN nástrojů.	Komunita kolem DEMO se skládá hlavně z akademických obcí v Evropě. Po světě existuje i několik „center excelence“, dá se však říct, že v porovnání s ostatními, je komunita kolem DEMO malá.

Tabulka pokračuje na další straně

Tabulka 2.1 – *Pokračování tabulky z předchozí strany*

	Vývojový diagram	**UML**	**BPMN**	**DEMO**
Výhody	• Srozumitelné různým typům uživatelů • Přímočaré učení • Dobrá dostupnost nástrojů podporujících modelování	• Mnoho různých typů modelů • Standard, který je dále vyvíjen • Dobrá dostupnost nástrojů podporujících modelování • Možnost modely jednoduše implementovat pomocí IT systémů	• Notace zaměřená výhradně na modelování podnikových procesů • Standard, který je dále vyvíjen • Dobrá dostupnost nástrojů podporujících modelování • Možnost modely jednoduše implementovat pomocí IT systémů	• Silný teoretický základ • Jednoznačnost a konzistence modelů • Abstrahuje od implementačních detailů

Kapitola 3

Notace BPMN

3.1 O BPMN

BPMN neboli *Business Process Modelling and Notation* je soubor pravidel a grafických prvků, pomocí kterých mohou organizace modelovat svoje obchodní procesy. Jedná se pravděpodobně o světově nejpoužívanější *standard* pro modelování podnikových procesů. Jeho nespornou výhodou je, že na rozdíl od hojně rozšířených vývojových diagramů, je BPMN standardizované, tudíž je možné modely vytvořené v BPMN automatizovat, je možné volně měnit nástroje, kterými jsou modely vytvářené a uživatelé tak nejsou závislí na jejich výrobcích.

Za vznikem BPMN stála iniciativa BPMI (Business Process Management Initiative), jejíž primární motivací bylo vytvořit grafickou notaci, která bude srozumitelná všem účastníkům životního cyklu procesu (management, vývojáři, analytici). [22] Faktem je, že modely v BPMN v dnešní době slouží pro popis procesů na vysoké úrovni abstrakce, ale i pro popis těch nízkoúrovňových, které slouží jako podklad pro implementaci procesu v nějakém softwarovém nástroji.

Dalším cílem, se kterým bylo BPMN vytvořeno, bylo ustanovit notaci, která umožní zobrazovat jednoduché i komplexní obchodní procesy [22], protože v té době obvyklé modelovací metody byly při vytváření rozsáhlých modelů velmi obtížně použitelné a vzniklé modely bylo složité udržovat.

O BPMN se dnes stará organizace Object Management Group (OMG). Za svojí popularitu vděčí BPMN, kromě výše zmíněných důvodů, zejména přístupnosti notace pro business uživatele, kteří jsou obeznámení s tradičními vývojovými diagramy, kterým se struktura diagramů i některé elementy v mnohém podobají [23]. Rozdílem oproti vývojovým diagramům je ale již výše zmíněná standardizace použití jednotlivých elementů a tedy v tomto případě účelné omezení svobody uživatelů. Tento fakt umožňuje *validovat* výsledné BPMN modely oproti specifikaci. Dalším rozdílem je možnost modelovat chování na základě výskytu nějaké definované události, což je situace, která se v

reálném životě stává velmi často, ale ve vývojovém diagramu ji není možné vyjádřit. V neposlední řadě je v BPMN také možné modelovat komunikaci s entitami mimo organizaci nebo proces.

Jak tvrdí například [23] v praxi vzniká velké množství „špatného BPMN“, tedy modelů, které nejsou validní, kompletní nebo jednoznačné. Důvod je zřejmý – absence pevného teoretického základu, který by říkal více než k čemu který element z notace slouží a s kterým elementem je možné ho propojit. BPMN chybí *metodologie*, která by přesně popisovala jak modely vytvářet a jak zaručit jejich konzistenci, jednoznačnost a kompletnost. V praxi tedy vidíme vznik „metodologií“, které nejsou součástí standardu BPMN, ale jsou adoptovány ve firmách právě kvůli požadavkům na výše zmíněné vlastnosti a rovněž pro zajištění kontinuity. Jednou z velmi rozšířených je metoda popsaná v publikaci *BPMN Method and Style* vyvinutá Brucem Silverem, který se rovněž podílí na vývoji standardu BPMN. Cílem metody je umožnit vyvářet modely, které jsou:

- korektní,
- jednoznačné,
- kompletní,
- konzistentní.

> „Špatné BPMN“ je dnes normou spíše než výjimkou. [23] [11]

3.1.1 Verze 1.2 vs 2.0

BPMN je v současnosti ve verzi 2.0, nicméně v praxi je stále hojně využívána i verze 1.2. Klíčovým rozdílem je standardizace převodu BPMN konstruktů do jazyka XML, což by umožnilo z grafického vyjádření modelu v notaci BPMN vygenerovat *metamodel* v jazyce XML, který by bylo možné automatizovat pomocí softwarových nástrojů. Taková řešení sice již existovala i u starších verzí BPMN, ale byla vždy závislá na interpretaci výrobce konkrétního BPMS řešení a tudíž jen obtížně přenositelná. Standard BPMN 2.0 by měl toto změnit. Co se týče grafických elementů, tak do verze 2.0 jich oproti 1.2 přibylo jen velmi málo a většina business uživatelů tak pravděpodobně ani rozdíl nepostřehne.

3.2 Základní koncepty notace BPMN

Dříve než přikročíme k popisu základních elementů notace BPMN, popíšeme si základní koncepty, ze kterých BPMN vychází.

[11] Bad BPMN is the norm rather than the exception. [23]

3.2.1 BPMN diagram

Diagram v BPMN není pouze grafickým vyjádřením podnikového procesu, ale zároveň vstupním bodem pro *sémantický model* v jazyce XML, který je (pokud to modelovací software umožňuje) vytvářen zároveň s grafickým diagramem, aniž by do toho uživatel musel jakkoli zasahovat. Jak píše [23] BPMN dovoluje existenci sémantického modelu bez jeho grafického vyjádření, ale ne naopak – sémantický model tedy musí vždy existovat.

Procesní model v BPMN neříká nic o tom, jak jsou jeho jednotlivé aktivity prováděny nebo proč jsou prováděny. Definuje pouze následující:

- *pořadí* aktivit,
- *kdy* se aktivity provádějí,
- za jakých *podmínek* se aktivity provádějí.

3.2.2 Aktivita v BPMN

[23] definuje *aktivitu* v BPMN jako *akci*, jednotku provedené práce. Aktivita je akce prováděná v rámci organizace opakovaně a je neměnná, neboli je vždy prováděna stejným způsobem a má jasně vymezený začátek a konec. Aktivita je v rámci modelu dále nedělitelná, tj. není možné ji rozložit na subaktivity.

3.2.3 Proces v BPMN

Samotný pojem *proces* lze v BPMN jednoduše popsat jako *posloupnost aktivit* z počátečního stavu do konečného stavu [23]. *Procesní model* je pak mapou všech možných cest – posloupností aktivit – z počátečního stavu do některého z konečných stavů. Podobně jako aktivita i proces je prováděn opakovaně a každá jeho instance musí být prováděna dle některé z cest definovaných v procesním modelu.

Dle [24] má proces v BPMN 4 důležité aspekty:

- **Orchestrace** – jako orchestraci označujeme fakt, že procesy v BPMN se skládají z aktivit, které jsou vždy prováděny v určitém pořadí tak, jak je definováno v procesním modelu. Tyto aktivity jsou prováděny opakovaně a průběh jejich vykonávání má jasně definovaný začátek a konec. Procesní model navíc obsahuje všechny signifikantní možnosti, jak proces může proběhnout, ne jen jednu nejčastější nebo ad-hoc případy provádění.
- **Participant** – samotný proces v BPMN je vnímán jako samostatná entita a je účastníkem *spolupráce (collaboration)* s jinými entitami. Každý participant je unikátně propojen s jedním procesem.

- **Množina vykonavatelů** – každá aktivita v BPMN má vykonavatele ač není v diagramu znázorněn. Pokud je aktivita součástí orchestrace popsané výše, je pak součástí procesu a to samé platí i pro jejího vykonavatele.
- **Nezávislý actor** – BPMN používá z hlediska reálného života poměrně neobvyklou sémantiku, kdy u aktivity je tím, kdo požaduje její vykonání *samotný proces* (jeho instance) a aktivitu provádí její vykonavatel (například zaměstnanec).

3.2.4 Procesní logika

Procesní logika definuje všechny signifikantní možnosti, jak může proces probíhat (sekvence aktivit) od začátku do konce. Každý procesní model by měl obsahovat kompletní procesní logiku, pokud pro vypuštění některých možných cest neexistuje vážný důvod. Častým problémem modelů, které [23] označuje jako „špatné BPMN“ je popis pouze jedné cesty (obvykle té „šťastné“, tzv. *happy flow*) a ignorování případů, které z nějakého důvodu probíhají odlišně.

3.3 Základní elementy notace BPMN

Standard BPMN ve verzi 2.0 obsahuje již více než 100 symbolů. Popis všech je mimo rozsah této práce. Nicméně v této sekci budou popsány všechny elementy, které populární publikace [23] označuje jako *Level 1* a několik vybraných elementů z množiny, které stejný autor označuje jako *Level 2*, které se nám budou hodit později v dalších částech práce.

Symbolika použitá v BPMN je odvozená od klasických a hojně rozšířených vývojových diagramů, která jsou intuitivní na pochopení i pro pozorovatele, který není obeznámen s problematikou modelování podnikových procesů.

Jak uvádí [23] a [22] pro základní práci obvykle stačí pouze základní druhy elementů, které jsou:

- *Počáteční událost (Start event)*
- *Konečná událost (End event)*
- *Aktivita (Activity)*
- *Sekvenční tok (Sequence flow)*
- *Tok zpráv (Message flow)*
- *Brána (Gateway)*
- *Bazén (Pool)*
- *Plavecká dráha (Swimming lane)*

- *Datový objekt (Data object)*
- *Datové úložiště (Data store)*
- *Datová asociace (Data association)*

Pro potřeby této práce si ještě popíšeme následující element:

- *Průběžná událost typu Signál a Zpráva (Intermediate event)*

3.3.1 Aktivita

Aktivita je v procesu graficky znázorněna obdélníkem se zaoblenými rohy. Jak již bylo popsáno výše, reprezentuje jednotku provedené práce. Jako jediný BPMN element má *vykonavatele* [23].

Obrázek 3.1: Element *aktivita*

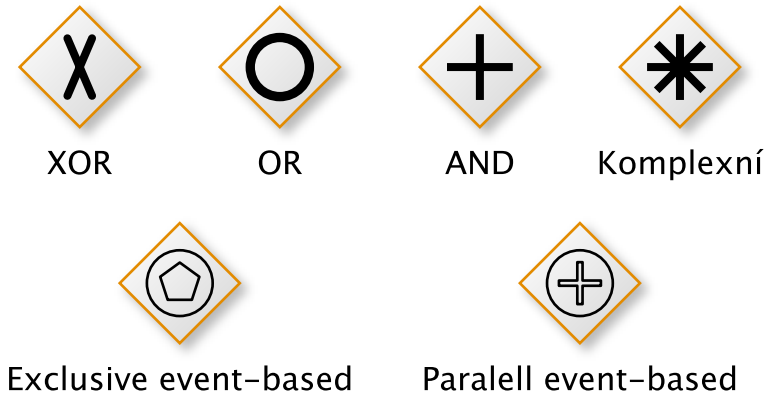
Každá aktivita je buď *task (Task)* nebo *subproces (Subprocess)*. Task je atomický typ aktivity, tedy v modelu jej není možné rozdělit na další aktivity, ze kterých se skládá na rozdíl od subprocesu, který je definované má a v modelu jsou znázorněné jako samostatný proces s počátečním i koncovým stavem. V závislosti na preferenci uživatele může být subproces zobrazen v rodičovském procesu jako jedna „sbalená“ aktivita nebo jsou přímo v rodičovském procesu vidět všechny aktivity, které subproces obsahuje. Subproces musí vždy začínat *počáteční událostí typu None*. Task se dělí na 8 podtypů, jejichž definice je k dohledání ve specifikaci [25]. V rámci této práce budeme používat pouze *abstraktní task*.

3.3.2 Brána

V případě, že je potřeba rozštěpit sekvenční tok, obvykle na základě nějaké podmínky, je tu element *brána*. Bez použití brány není možné sekvenční tok rozdělit, na rozdíl například od vývojových diagramů. Brána je v BPMN diagramu zobrazena použitím kosočtverce nepovinně se symbolem uvnitř. Existuje několik typů bran:

- Exkluzivní (XOR)

- Paralelní (AND)
- Inkluzivní (OR)
- Komplexní
- Event-based



Obrázek 3.2: Různé druhy elementu *brána*

3.3.2.1 Exkluzivní brána (XOR)

Fungování exkluzivní brány je poměrně přímočaré. Jak už jméno naznačuje, tato brána umožní pokračovat pouze jednomu sekvenčnímu toku na svém výstupu. Rozhodnutí v bráně by měla předcházet aktivita, která provede samotné rozhodnutí a brána by pak již jen „ověřila“ výsledek tohoto rozhodnutí a příslušně němu vybrala na výstupu sekvenční tok.

3.3.2.2 Paralelní brána (AND)

I v tomto případě je chování brány předvídatelné. Paralelní brána rozdělí vstupní sekvenční tok na více výstupních toků, po kterých proces pokračuje bez jakékoli podmínky. Paralelní bránu lze využít i v případě, kdy potřebujeme různé sekvenční toky opět spojit a je nutné zajistit jejich synchronizaci, tj. vyčkat na všechny příchozí sekvenční toky a teprve v momentě, kdy dorazí všechny, pokračovat v procesu.

3.3.2.3 Inkluzivní brána (OR)

Inkluzivní brána na rozdíl od exkluzivní varianty umožňuje na základě splněné podmínky na výstupu $1...n$ aktivních sekvenčních toků.

3.3.2.4 Komplexní brána

Komplexní brána se využívá pouze v případě, kdy není možné rozdělení sekvenčního toku modelovat použitím jiného typu brány. Lze ji vidět použitou v momentech, kdy rozdělení sekvenčního toku předchází nějaké velmi komplexní rozhodnutí. Podmínky, za kterých je použit určitý výstupní sekvenční tok jsou specifikovány textovým popisem na jednotlivých větvích sekvenčního toku.

3.3.2.5 Event-based brána

Event-based brána se používá v případě, kdy o výběru výstupního sekvenčního toku „rozhodne“ situace, kdy nastala nějaká událost (například obdržení zprávy).

3.3.3 Počáteční, průběžná a konečná událost

Každý proces v BPMN by měl začínat nějakým typem *počáteční události* a končit v některém typu *konečné události*. Obě tyto události jsou zobrazeny pomocí kruhu, který může obsahovat symbol určující, o jaký typ události se jedná. Proces může mít více počátečních událostí (speciální typy procesů naopak žádnou) a víc konečných událostí.

Navíc existují také *průběžné události*, které se vyskytují v průběhu procesu. Průběžné události mají dva základní podtypy – *throwing* a *catching*. Například throwing průběžná událost typu Zpráva znamená, že jakmile tato událost v procesu nastane odešle se zpráva definovanému příjemci. Oproti tomu když nastane catching průběžná událost typu Zpráva znamená to, že průběh procesu se zastaví dokud nepřijde zpráva, na kterou událost čeká. V této práci pracujeme pouze s průběžnými událostmi typu Zpráva nebo Signál.

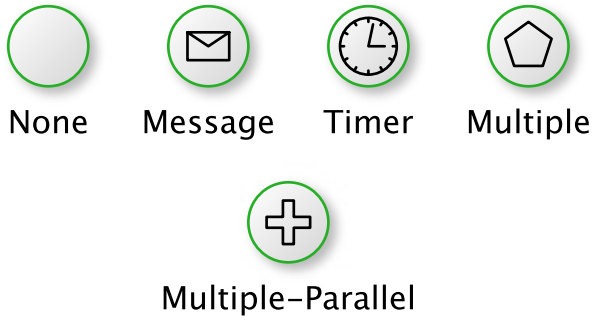
Symbol uvnitř kruhu označuje, o jaký typ počáteční nebo konečné události se jedná. V případě té počáteční symbol označuje jev, který proces spouští (například zpráva nebo signál). Typy počátečních událostí, které rozeznáváme, jsou:

- Prázdný (None)
- Zpráva (Message)
- Časovač (Timer)
- Multiple a Multiple-Parallel

BPMN 2.0 definuje 9 typů konečných událostí, ale v praxi se používají zejména tyto 4: [23]

- Prázdný (None)
- Zpráva (Message)

- Terminate
- Multiple



Obrázek 3.3: Různé druhy *počátečních událostí*

Obrázek 3.4: Různé druhy *konečných událostí*

3.3.3.1 Prázdná počáteční událost

Prázdná počáteční událost se používá v případech, kdy proces není spouštěn žádným jevem nebo tento jev není specifikován. Často se tento typ počáteční události využívá v případě, kdy je proces spouštěn manuálně vykonavatelem určitého tasku.

3.3.3.2 Počáteční událost typu Zpráva

Proces jehož počáteční událost je typu *Zpráva* se spouští okamžikem přijetí nějaké specifické zprávy, kterou zašle jiný proces.

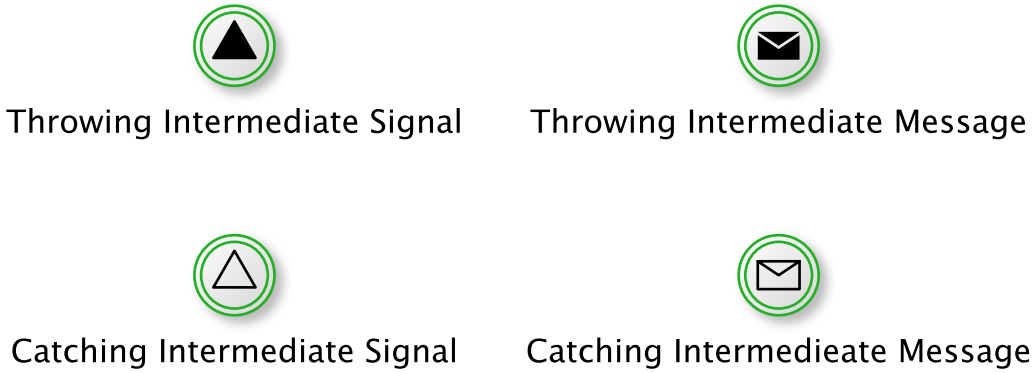
3.3.3.3 Počáteční událost typu Časovač

Pokud je projekt spouštěn pravidelně v určitý čas dle nějakého plánu, používá se počáteční událost typu *Časovač*. Štítek (Label) počáteční události tohoto

typu by měl vyjadřovat frekvenci s jakou je proces spouštěn (např. „denně“, „kvartálně“).

3.3.3.4 Průběžná událost typu Signál

Průběžná událost typu *Signál* vysílá nebo přijímá signál, což je neadresná zpráva, kterou může přijmout každý, kdo je připraven naslouchat. Je například vhodná pro neadresnou komunikaci uvnitř procesu (na rozdíl od zprávy). Na obrázku 3.5 vidíme průběžnou throwing i catching událost typu Signál.



Obrázek 3.5: Různé druhy průběžných událostí používaných v této práci

3.3.3.5 Průběžná událost typu Zpráva

Analogicky s předchozí sekcí funguje i průběžná událost typu *Zpráva*. na rozdíl od signálu však zprávy podléhají některým omezením a není možné je využít pro komunikaci uvnitř procesu. Na obrázku 3.5 vidíme průběžnou událost typu Zpráva ve verzi throwing i catching.

3.3.3.6 Prázdná konečná událost

Pokud proces končí v prázdné konečné události znamená to zjednodušeně řečeno, že o svém konci proces nedá veřejně vědět, jelikož není vyslán žádný signál nebo zpráva.

3.3.3.7 Konečná událost typu Zpráva

Konečná událost typu *Zpráva* je reprezentována kruhem s černou obálkou uvnitř. Tento typ konečné události vyjadřuje, že proces při svém ukončení vyšle zprávu nějakému jinému procesu.

3.3.3.8 Konečná událost typu Terminate

Konečná událost typu *Terminate* se používá v případě speciálních událostí, kdy je potřeba v případě dosažení tohoto konečného stavu ukončit i všechny aktivní paralelní sekvenční toky, podprocesy atd.

3.3.4 Sekvenční tok

Sekvenční tok je v diagramu zobrazen pomocí nepřerušované šipky, která je vždy připojena z obou stran k jiným BPMN elementům v diagramu. Jeho úkolem je určovat pořadí provádění aktivit v procesu. Sekvenční tok dle specifikace může propojovat pouze aktivity, brány a události. Jinými slovy sekvenční tok reprezentuje *orchestraci*. [23]

U sekvenčních toků je velmi důležité mít na paměti pravidlo, že žádný sekvenční tok nesmí nikdy překročit hranice subprocesu nebo bazénu, neboť toto chování specifikace BPMN zapovídá.

3.3.5 Tok zpráv

na rozdíl od sekvenčního toku je *tok zpráv* v diagramu zobrazen šipkou s přerušovanou čárou. Vyjadřuje zaslání zprávy od odesílatele k příjemci, kterým je nějaká externí entita (black-box bazén nebo aktivita, zpráva nebo událost uvnitř jiného procesu). Je důležité si uvědomit, že tok zpráv neindikuje vždy jistotu, že komunikace opravdu proběhne. Někdy se jedná jen o vyjádření, že je v tomto momentě možné odeslat nebo přijmout zprávu.

3.3.6 Bazén a plavecké dráhy

Jasně ohraničený obdélník orientovaný vertikálně či horizontálně ohraničuje *bazén*. Bazén slouží k vymezení hranic mezi procesem a externími entitami. Bazén obsahuje BPMN elementy procesu nebo se jedná o tzv. *black-box bazén*. Ten se používá v případě, že chceme modelovat komunikaci s externí entitou, ale nechceme zobrazovat jak vnitřní entita funguje.

Plaveckou dráhu je možné použít v bazénu i mimo něj. Používá se většinou k zachycení rozdělení odpovědností za danou aktivitu v rámci organizace, nicméně specifikace BPMN 2.0 dovoluje prakticky neomezené použití plaveckých drah k jakémukoliv typu kategorizace. [23]

Grafická reprezentace se podobá bazénu, nicméně rozdíl je ve štítku vlevo, který u plavecké dráhy není ohraničen a oddělen od zbytku elementu. V jednom bazénu bývá typicky více plaveckých drah, nicméně plaveckou dráhu je možné zobrazit i vně bazénu.

Obrázek 3.6: Element *bazén* s *plaveckými drahami*

3.3.7 Datový objekt, úložiště, asociace

Reprezentace dat a práce s nimi došla v BPMN 2.0 výrazné změny, kdy *datový objekt* je v nové verzi plnohodnotným BPMN elementem a zároveň byl přidán nový objekt *datové úložiště*.

Datový objekt je graficky reprezentován elementem, který připomíná list papíru s ohnutým rohem a jedná se spíše o programátorský konstrukt, jak uvádí [23]. Reprezentuje lokální proměnnou, data, která existují dočasně v rámci instance procesu.

Datové úložiště oproti tomu reprezentuje data, která jsou dostupná trvale, například v nějaké databázi nebo na jiném místě. Data je možné z procesu číst či měnit. Datové úložiště je reprezentováno symbolem válce se třemi čárkami ve vrchní části.

S ostatními elementy v procesu jsou výše popsané datové objekty spojené prostřednictvím datových asociací, které jsou znázorněny tečkovanými šipkami s nevyplněným „V“ na vrcholu nebo bez něj.

3.4 Základní pravidla modelování v BPMN

Specifikace BPMN předepisuje poměrně málo pravidel, jak procesní modely vytvářet ve smyslu jak elementy používat a jak ne. Je tedy běžné, že pravidla

Obrázek 3.7: Element *datový objekt* a *datové úložiště*

vznikají přirozeně uvnitř organizací, kde chtějí sjednotit styl, podle kterého modely vznikají, aby byly modely vytvořené uvnitř organizace navzájem kompatibilní. [23] uvádí, že BPMN má 3 zdroje, ze kterých lze pravidla vyčíst a těmi jsou:

- oficiální specifikace,
- BPMN metamodel,
- XML schéma příslušné metamodelu.

V této sekci najdeme přehled nejdůležitějších oficiálních pravidel dle specifikace BPMN, které by měl dodržovat každý model vzniklý v BPMN a měl by být proti nim validován.

1. Sekvenční tok nesmí překročit hranice bazénu.
2. Sekvenční tok nesmí překročit hranice podprocesu.
3. Tok zpráv nesmí spojovat elementy uvnitř stejného bazénu.
4. Sekvenční tok může spojovat mezi sebou pouze aktivity, brány, události a oba konce sekvenčního toku musí být k některému z těchto elementů připojené.
5. Tok zpráv může spojovat pouze aktivity, události (typu Zpráva nebo Multiple) nebo hranice black-box bazénu a oba konce musí být k některému z těchto elementů připojené.

3.5 Možnosti automatizace BPMN modelů

Čím dál častěji jsou v dnešní době BPMN modely vytvářeny ne pouze za účelem jejich zdokumentování, ale stále více také za účelem jejich budoucí automatizace [26]. Myšlenka na automatizaci procesních modelů však existovala již na samotném začátku, kdy BPMN vzniklo jako grafická vrstva v rámci systému vyvíjeného konsorciem BPMI.org [23]. Vznikl tak jazyk BPML, který měl být nezávislým standardem pro automatizaci procesů a popis procesu v

jazyce BPML by nebyl „programován“, ale generován automaticky z diagramu v notaci BPMN. BPML se ale nikdy neuchytil, společnosti IBM a Microsoft vyvinuly vlastní jazyk BPEL, který staví na standardu pro webové služby WSDL, a tento jazyk rychle na trhu BPML jasně předčil.

Kromě některých proprietárních řešení, které zde popsány nebudou, stojí za popsání dvě řešení automatizace BPMN, kterými jsou BPMN 2.0 a BPEL.

3.5.1 BPMN 2.0

BPMN 2.0 se od předchozí verze s označením BPMN 1.2 ve skutečnosti, co se nových elementů a jejich vzhledu týká, liší jen velmi málo. Hlavní změny se odehrály tak říkajíc pod povrchem na *metamodelu* a na jeho reprezentaci v jazyce XML. Hlavním cílem těchto změn bylo vytvořit standardizovaný XML formát popisující model procesu, který bude sloužit vzájemné kompatibilitě modelů mezi různými BPMN nástroji a zároveň umožní modely automatizovat. [23]

BPMN 2.0 tak standardizuje reprezentaci dat v procesu, zpráv, služeb, přidělení tasků apod. v XML, které je vytvářeno spolu s každým diagramem. Jak uvádí [23], pro každý BPMN element je nutno specifikovat parametry jako:

- Proměnné procesu
- Data na vstupu i výstupu tasku a jejich vazby na proměnné
- Zprávy
- Definice událostí
- Podmíněné výrazy

Jak poznamenává [23], BPMN 2.0 na rozdíl od BPEL popsaného níže, není jazykem pro popis automatizace procesů a tedy není možné v BPMN 2.0 procesy přímo spouštět. Každý softwarový nástroj bude muset implementovat vlastní exekuční prostředí pro BPMN 2.0 XML metamodel. V současnosti však není adopce této části BPMN 2.0 u výrobců softwarových nástrojů příliš markantní. [27]

3.5.2 BPEL

BPEL neboli *Business Process Execution Language* (přesněji WS-BPEL Web Service Business Process Execution Language) je jazyk pro popis automatizace procesů. Je založený na XML, ve kterém popisuje veškeré detaily o procesu a jeho elementech. Proces popsaný v BPEL je možné přímo spouštět v některém z dostupných BPEL exekučních prostředí.

Jazyk BPEL nemá žádnou oficiální grafickou reprezentaci, ale jelikož většina business uživatelů chce procesní modely vytvářet v uživatelsky příjemném grafickém editoru a ne pomocí programovacího jazyka, používá každý z BPEL nástrojů nějakou grafickou notaci, kterou následně převádí do BPEL. Tato notace může být proprietární (vyvinutá výrobcem nástroje), častěji se však využívá notace BPMN, se kterou jsou uživatelé dobře obeznámeni.

Jak již bylo uvedeno výše, oficiální název pro BPEL je WS-BPEL, kde první dvě písmena znamenají *Web Service* neboli *webová služba*. Celý jazyk BPEL je založen na *WSDL* neboli *Web Service Description Language*. Jednou ze základních funkcí BPEL tak je orchestrace webových služeb. Úkolem BPEL je integrace funkcionalit, které poskytují webové služby pro implementaci v konkrétním podnikovém procesu.

Proces v BPEL je specifikován pomocí XML dokumentu, který popisuje veškeré elementy a umožňuje popsat i vztahy mezi nimi a webovými službami. Tento XML dokument je pak spustitelný pomocí exekučního prostředí, které je součástí BPEL nástroje.

Jako již bylo zmíněno výše, BPMN je velmi často využíváno pro modelování procesu na grafické úrovni a takový model je pak převeden do BPEL. Že je to velmi běžná praxe napovídá i samotná specifikace jazyka BPMN, která přímo obsahuje informace o tom, jak BPMN do BPEL převádět. Tento postup však může narážet na problémy, neboť BPMN je omezenější a méně strukturovaný než BPEL. Řešením v takovém případě je buď smířit se se ztrátou informace vlivem nedokonalého převodu a nebo upravit vstupní BPMN model tak, aby korespondoval s konstrukty, které najdeme v BPEL. [28]

Kapitola 4

Metodologie DEMO

4.1 O DEMO

4.1.1 Rozdíl mezi notací a metodologií

V celém textu této diplomové práce se vyskytují dva jevy. O BPMN mluvíme jako o *notaci* a o DEMO jako o *metodologii*. Takto tyto techniky označujeme záměrně a myslím, že je na tomto místě účelné si vysvětlit rozdíl mezi pojmy *notace* a *metodologie*. Pochopení této odlišnosti totiž vnáší trochu světla k lepšímu pochopení rozdílu mezi BPMN a DEMO.

4.1.1.1 Notace

Notace označuje *formální prostředky* pro popis reality. Například právě v oblasti analýzy a modelování podnikových procesů je notací sada grafických objektů, které pak používáme pro popsání samotného procesu. Na notaci je obvykle navázána související metodika. [29]

Metodikou nazýváme *popis pracovního postupu* nějaké činnosti, který je více či méně formalizovaný [30]. V případě modelovací techniky by taková metodika tedy popisovala jak při modelování postupovat a jak a kdy jednotlivé elementy přesně používat. To však v případě BPMN neplatí, jelikož BPMN není svázané s žádnou metodikou [22] a tedy pokud se někde vyskytuje označení BPMN jako metodiky, je takové označení chybné.

4.1.1.2 Metodologie

Metodologie je oproti tomu vědní disciplína, která se zabývá tvorbou metod a jejich aplikací. Metodologie vědy je tedy naukou o metodách. Jak píše [31] je teorií k výběru výzkumných metod a návodem, jak vybrané metody (metodu) používat ve vědeckém zkoumání.

Pod vlivem angličtiny se však i v češtině často setkáváme s tím, že pojmy metodika a metodologie splývají a jsou časfo zaměňovány. Můžeme nicméně

vidět, že rozdíl BPMN a DEMO je na první pohled zřejmý v tom, že notace BPMN nemá za sebou zdaleka tak robustní základ jako metodologie DEMO. Tento fakt má několik důsledků týkajících se přístupnosti a přímočarosti použití obou technik. Tyto důsledky budou ještě v této práci dále rozebrány.

4.1.2 Motivace k vytvoření DEMO

Jak shrnuje [32], u základní úvahy tvůrců DEMO byl současný stav podniků a organizací, které jsou velmi komplexní a z toho důvodu je velmi obtížné mít pomocí současných nástrojů jasnou představu o tom, jak přesně fungují a co se v nich děje.

Moderní organizace jsou totiž založeny na propojení sociálních a technických komponent, které spolu vzájemně komunikují. Komunikace je tedy nejdůležitějším aspektem celé metodologie DEMO. Dle [10] je ontologie podniků (*Enterprise ontology*) nejvhodnějším prostředkem k pochopení konstrukce a operací v podniku.

> DEMO bylo vytvořeno jako metodologie pro vytváření ontologického modelu podniku. [19] [12]

4.2 Ontologie

V úplně základním pojetí je ontologie definována jako *nauka o bytí.* Taková definice je samozřejmě pro čtenáře velmi abstraktní. Lepší by bylo ontologii definovat jako nauku o Bytí, tedy s velkým B, neboť ontologie se právě zabývá „pouze“ tím, co to znamená, že něco „je“, jak to bytí vypadá a jak to funguje.

> Ontologie (nebo ontologický model) organizace je definován jako porozumění chodu organizace, které je kompletně oproštěné od realizace a implementace vlastních činností. [13]

Pro lepší pochopení toho, co ontologický model představuje je užitečné uvést rozdíl mezi *teleologickým* pojetím systému a *ontologickým* modelem systému.

Teleologický pohled na systém se zabývá funkcemi a službami, které systém poskytuje navenek. Teleologický model pak vypadá jako tzv. „black-box model“ neboli vidíte, že se vstupy změní na nějaké výstupy, ale už není vidět, jak k tomu došlo. Tento pohled (model) je vhodný pro užívání a řízení (věcí, systémů, organizací).

[12] DEMO was developed to be a methodology for creating an ontological model of an enterprise. [19]

[13] The ontology (or ontological model) of an enterprise is defined as an understanding of its operation, that is completely independent of the realization and the implementation of the enterprise. [33]

Ontologický pohled se naopak zabývá tím, jak systém funguje uvnitř, tedy tím, *jak* dojde k proměně vstupů na výstupy. Zabývá se tedy konstrukcí a chodem systému. Ontologický model je tedy typem tzv. „white-box modelu“. Ontologický model najde uplatnění při budování a úpravách (věcí, systémů, organizací).

4.2.1 Motivace pro zabývání se ontologiemi v organizaci

Jak už bylo popsáno výše, ontologie v organizaci slouží zejména k porozumění jejímu chodu bez nutnosti zabývat se, jak jsou jednotlivé činnosti implementovány. Takový přístup je užitečný pro následující skupiny uživatelů: [33]

- **Manažeři** – Pro řízení větších celků je užitečné mít možnost oprostit se od detailů a dokázat se na chod takového celku podívat z vyšší perspektivy, tzv. „big picture pohled“.
- **Návrháři, inženýři, architekti** – Pro účely návrhu a úprav fungování chodu organizace je důležité mít podnikové procesy definované metodicky a nezávisle na jejich implementaci.
- **Uživatelé** – Existují skupiny uživatelů (uvnitř i vně organizace), pro které je užitečné mít vhled i do fungování organizace nebo jejího celku.

4.3 Teorie PSI (Ψ-theory)

Teorie PSI neboli Ψ-theory je teorie o fungování organizací. [33] [14]

Zkratka PSI znamená *Performance in Social Interaction*. Paradigma, na kterém je tato teorie založena říká, že subjekty, kterými jsou lidé v organizaci, vstupují do závazků a dodržují je. Tímto způsobem pak vzniká spolupráce mezi lidmi.

Cílem Ψ-theory je umožnit porozumění funkcím organizace bez vlivu toho, jak jsou tyto funkce ve skutečnosti operativně vykonávány. Jak uvádí [32], stejné cíle si klade i metodologie DEMO, takže je jen logické, že je právě na Ψ-theory postavena. Porozumění této teorii je tedy nezbytné pro správné pochopení a používání DEMO.

Teorie PSI se skládá ze čtyř axiomů:například

1. operační,
2. transakční,
3. kompoziční,
4. distinkční.

[14] The Ψ-theory is a theory about the operation of organizations. [33]

4.3.1 Operační axiom – The operation axiom

První axiom Ψ-theory se nazývá *operační*. Jeho základem jsou dvě tvrzení: [10]:

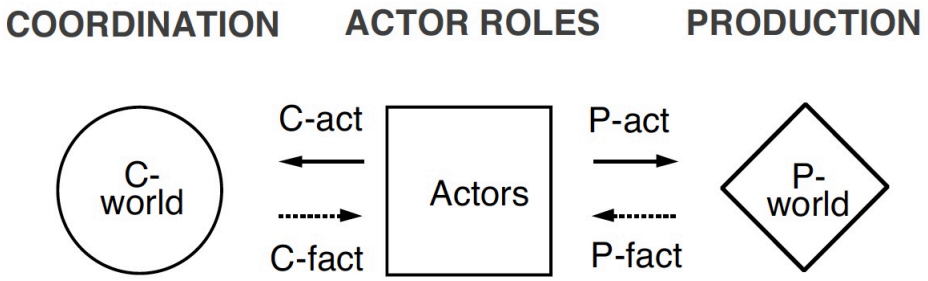
1. Chod organizace se skládá z aktivit, které vykonávají actoři. Actoři jsou kombinací *zodpovědnosti* a *autority* k provádění dané aktivity.

2. Při tom provádějí dva druhy činností: *coordination a production acts (C-acts a P-acts)*. Výsledkem těchto činností jsou *coordination a production facts (C-facts a P-facts)*.

Provádět P-acty a P-facty znamená přivádět na svět něco nového a přispívat tak k podnikovým funkcím nebo službám. P-facty mohou být hmotné i nehmotné. Příkladem těch hmotných může být například vyrobení pizzy, příkladem nehmotných zase například vynesení rozsudku soudem.

Provádět C-acty a C-facty znamená, že actoři jednají v souladu se závazky, které se týkají tvorby P-factů. Zjednodušeně řečeno se jedná o *komunikaci* ohledně vytváření P-factů.

Kromě C-factů a P-factů rozlišujeme ještě C-world a P-world. C-world je množina C-factů a stejně tak P-world je množina P-factů. Oba světy jsou tedy množinou factů, které byly vytvořeny do konkrétního momentu v čase. Vztah všech základních stavebních kamenů operačního axiomu je graficky znázorněn na obrázku 4.1.



Obrázek 4.1: Grafické znázornění operačního axiomu [10]

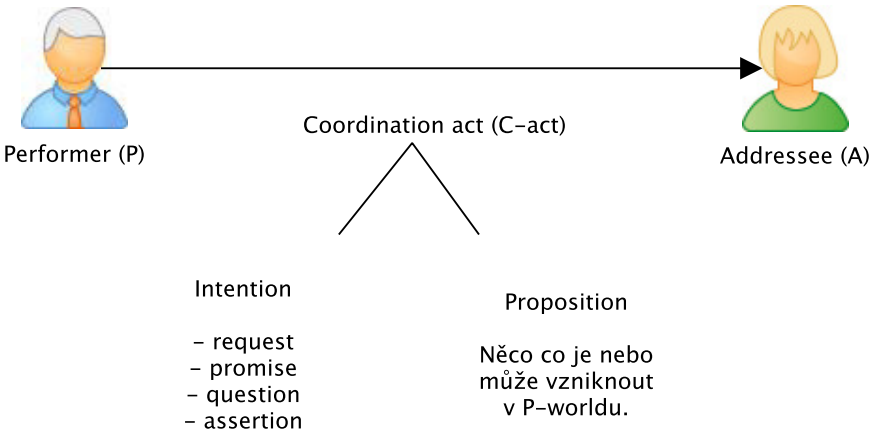
4.3.1.1 C-acty

C-acty probíhají mezi dvěma subjekty z nichž jeden se nazývá *performer* a druhý *adressee*. C-actů je několik typů, které můžeme rozdělit na intenční (*intention*) a propoziční (*proposition*).

Mezi příklady intenčních koordinačních činností řadíme:

- *request*
- *promise*
- *question*
- *assertion*

V případě propozičních C-actů executor „ohlašuje“ P-fact a příslušný čas, kdy má být proveden.



Obrázek 4.2: Grafické znázornění C-actu [10]

4.3.1.2 P-facty

Jak již bylo naznačeno, P-facty jsou buď hmotné nebo nehmotné. Zde je nezbytné poukázat na to, kdy tyto skutky začnou v P-worldu skutečně existovat. Před tím, než se to stane, je totiž nutné provést ještě dva C-facty a to *state* a *accept*. Teprve ve chvíli, kdy jsou tyto C-facty provedeny začíná P-fact skutečně existovat v P-worldu.

4.3.1.3 Actoři

Actoři jsou aktivní subjekty uvnitř organizace. Jednají autonomně, tedy jejich činnost není vyvolána nějakou událostí [10]. U actorů existují tři důležité vlastnosti, kterými jsou *kompetence*, *autorita* a *zodpovědnost*.

Kompetencí je myšlena schopnost subjektu provádět P-acty a související C-acty. [10] uvádí příklad instalatéra, který má znalosti a zkušenosti, které jsou nezbytné pro to být profesionálním instalatérem.

Aby mohl být subjekt schopen vykonávat určitou profesi, musí pro to mít nějaký autoritativní základ, jako například být zaměstnancem určité organizace a podobně.

Subjekt je vázán normami, které se vztahují k řečené autoritě nebo k obecným normám platným ve společnosti, které očekávají, že bude svojí autoritu vykonávat odpovědným způsobem. V příkladu instalatéra to znamená, že se očekává, že bude jednat zodpovědně se svými zákazníky.

4.3.2 Transakční axiom – The transaction axiom

Transakční axiom dále rozebírá P-acty a C-acty a zejména to, jak spolu tyto činnosti souvisí. Základní myšlenkou transakčního axiomu, kterou formuluje [10], je, že C-acty probíhají postupně za sebou ve stejných *vzorech*. Tyto vzory se nazývají transakce a vždy zahrnují dva actory (initiator a executor) a jejich cílem je dosáhnout určitého výsledku, kterým je P-fact.

Každá transakce má tři fáze:

1. *order phase*,
2. *execution phase*,
3. *result phase*.

V rámci order phase se initiator a executor snaží dojít k *dohodě* ohledně výsledku, kterého má být dosaženo (co, kdy). V execution phase je tento výsledek vytvořen a v result phase opět dochází k jednání mezi initiatorem a executorem, jestli vytvořený výsledek odpovídá požadavku initiatora.

Výsledek transakce (P-fact) začne existovat až ve chvíli, kdy je dokončena result phase, tedy když je P-fact schválen a přijat iniciátorem transakce. Do toho okamžiku P-fact v našem výkladu neexistuje.

Transakční vzory rozlišujeme dva: základní a standardní.

4.3.2.1 Základní transakční vzor

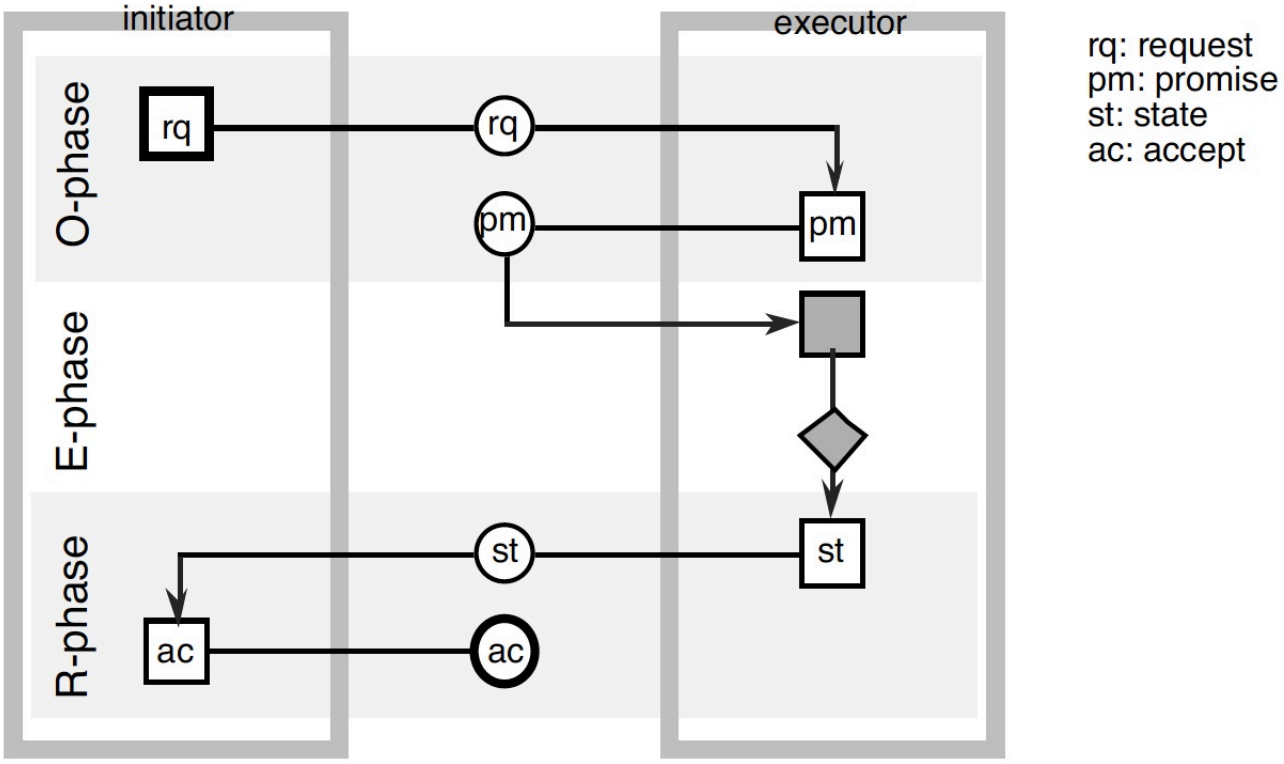
Základní transakční vzor je zjednodušený průběh transakce oproštěný od tzv. *unhappy paths* neboli „nešťastných scénářů“. Popisuje postup transakce v případě, kdy nenastanou žádné problémy, tedy executor vytvoří P-fact, který odpovídá požadavkům initiatora a tento P-fact je tedy bez komplikací akceptován.

Průběh zjednodušeného transakčního vzoru:

1. Initiator formuluje požadavek (*request*)
2. Executor učiní *promise*
3. Executor provede požadavek (vytvoří P-fact) (*execution*)

4. Executor prohlásí výsledek za hotový (*state*)
5. Initiator akceptuje výsledek (*accept*)

Základní transakční vzor je graficky znázorněn na obrázku 4.3.



Obrázek 4.3: Grafické znázornění základního transakčního vzoru [10]

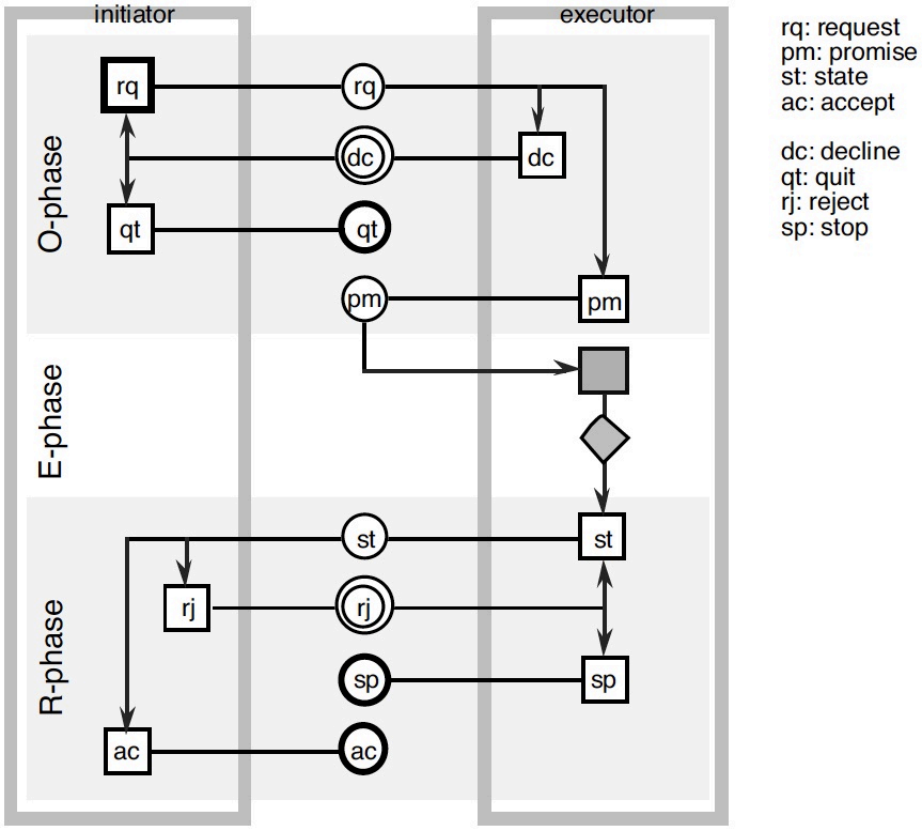
4.3.2.2 Standardní transakční vzor

Standardní transakční vzor počítá i se situacemi, které v běžném životě neustále nastávají. Například se situací, kdy executor nemůže učinit *promise* na požadavek iniciátora nebo initiator odmítne akceptovat výsledek transakce. Pokud toto nastane, dostává se celá transakce do tzv. *diskusních stavů* (*discussion states*), které mohou být buď *nepřijmuto* (*declined*) nebo *odmítnuto* (*rejected*).

Důvody pro odmítnutí *požadavku* executorem vycházejí z těchto tří typů tvrzení (*validity claims*):

- tvrzení pravdivosti (*claim to truth*),

- tvrzení oprávněnosti (*claim to justice*),
- tvrzení upřímnosti (*claim to sincerity*).



Obrázek 4.4: Grafické znázornění standardního transakčního vzoru [10]

4.3.2.3 Odvolávací vzory (Revoke patterns)

V rámci standardního transakčního vzoru je možné odvolat kterýkoliv koordinační čin pomocí *odvolávacího vzoru* pro daný C-act.

4.3.3 Kompoziční axiom – The composition axiom

Umět popsat strukturu konkrétní transakce je rozhodně přínosné, ale na ontologický popis organizace umět popsat jednotlivé transakce nestačí. Zde pak nastupuje *kompoziční axiom*, který se zabývá tím, jak jsou jednotlivé transakce, respektive P-facty, propojené.

Kompoziční axiom tvrdí, že každá transakce je buď součástí jiné, je transakcí zákazníka organizace nebo je *self-activated*. Logickou implikací tedy je, že transakce mohou obsahovat další transakce. *Takto propojené transakce pak dohromady tvoří podnikový proces.*

Jak píše [10] kompoziční axiom je tak základem pro definici podnikového procesu, která říká, že podnikový proces je množina volně propojených transakcí.

4.3.4 Distinkční axiom – The distinction axiom

Distinkční axiom tvrdí, že lidé mají tři různé typy schopností, které hrají roli v jejich chování.

- **forma** – jak už název napovídá, tato schopnost se týká formy v jaké jsou informace uchovávány, předávány, přijímány atd.
- **informa** – v tomto případě jde o obsah informace a její komunikaci mezi lidmi a plně abstrahujeme od formy v jaké je informace komunikována.
- **performa** – jedná se o nejvyšší formu lidských schopností. Jde zde o vytváření nových originálních věcí přímo nebo nepřímo pomocí komunikace. To se týká závazků, rozhodnutí, posuzování apod.

Jak píše [32], pro jeden infologický čin (performa) musíme provést více infologických činů (informa) a pro jeden infologický čin musíme provést více datalogických činů (forma). Toto rozlišení umožňuje výrazně zjednodušit procesní modely, protože se při jejich tvorbě zabýváme pouze ontologickými činy.

4.4 Teorém organizace – The organization theorem

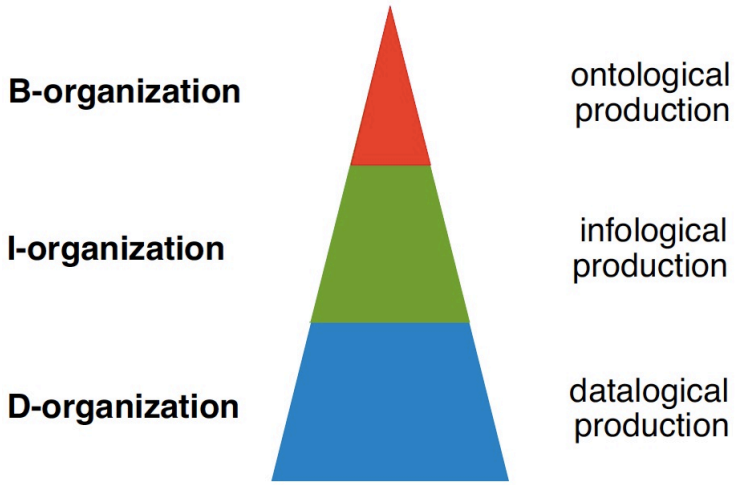
V předchozí sekci jsou popsány čtyří axiomy Ψ-theory, které přináší z různých úhlů pohled na chod organizace, její operace a uspořádání. Tyto pojmy však samy o sobě nestačí k vytvoření ontologického modelu, který bude výstižný, úplný, ucelený a konzistentní. Teorém organizace se zabývá právě tím.

Dle teorému organizace je každá organizace strukturovaná jako heterogenní systém, který je tvořen třemi vrstvami, z nichž každá představuje jeden homogenní systém: [10]

- B-organizace (B=Business)
- I-organizace (I=Intellect)
- D-organizace (D=Document)

Mezi těmito vrstvami (systémy) existuje velká provázanost. D-organizace podporuje I-organizaci a I-organizace podporuje B-organizaci. Provázanost mezi těmito systémy zajišťuje člověk. U tohoto konstatování je třeba se zastavit. Rozdělení organizace do tří systému si není možné představovat tak, že existují v organizaci nějaká B-oddělení, I-oddělení nebo D-oddělení nebo dokonce B-lidé, I-lidé či D-lidé. Naopak v realitě nic takového neexistuje. Lidé i skupiny lidí zastávají role ve všech systémech najednou a volně mezi nimi „přechází".

Rozdíl mezi jednotlivými systémy tvoří jejich výstupy. Jak uvádí [10] výstup B-organizace je ontologický, výstup I-organizace je infologický a výstup D-organizace je datalogický.



Obrázek 4.5: Grafické znázornění organizačního teorému [10]

Na vrcholu této pyramidy je B-organizace neboli ontologický level. Tím je naznačeno, že porozumění chodu organizace na této úrovni je kompletní, tedy není z něj nic vynecháno.

Pro lepší pochopení provázanosti jednotlivých systémů v organizaci bude nyní jejich provázanost hlouběji rozebrána. I-organizace poskytuje „informační zdroje" pro fungování B-organizace. [10] uvádí příklad s výpočtem denního obratu, který B-actor v B-organizaci chce vytvořit. Za tímto účelem je ale nejprve v I-organizaci I-actorem třeba provést několik jasně definovaných výpočtů (I-transakce) a následně doručit B-actorovi výsledek, kterým je právě denní obrat.

Dále je nutné rozebrat propojení mezi I-organizací a I-actorem s D-organizaci a D-actorem. V příkladu počítání denního obratu musí nejdříve I-actor sčí-

tající jednotlivé položky, které denní obrat tvoří, někde tyto položky (čísla) opatřit. Tyto údaje získá samozřejmě v D-organizaci pomocí D-transakcí.

Jak už bylo naznačeno výše, není důležité, jestli v tomto konkrétním případě je B-actor, I-actor a D-actor několik osob nebo jeden člověk a stejně tak B-organizace, I-organizace či C-organizace mohou být na několika kontinentech nebo uvnitř jedné kanceláře.

4.5 Základní modely DEMO

V této sekci bude rozebrána vlastní metodologie DEMO. Rozebrány budou zejména dvě věci: typy modelů, ze kterých se DEMO skládá a doporučený postup, jak v DEMO modelovat. Přesný popis jednotlivých elementů je možné dohledat například v [10].

Jak píše [19], základními elementy DEMO jsou *ontologické transakce* a *actoři*.

4.5.1 Ontologická transakce

Ontologické transakce jsou transakce, jejichž výsledkem je vytvoření něčeho nového, ať už se jedná o věc hmotnou či nehmotnou. Jak už uvádí distinkční axiom Ψ-theory, který byl popsán výše, ontologické transakce probíhají na ontologické úrovni (performa).

4.5.2 Actor

Každá transakce má vždy jednoho initiatora a jednoho executora.

4.5.3 Modely DEMO (Aspect models)

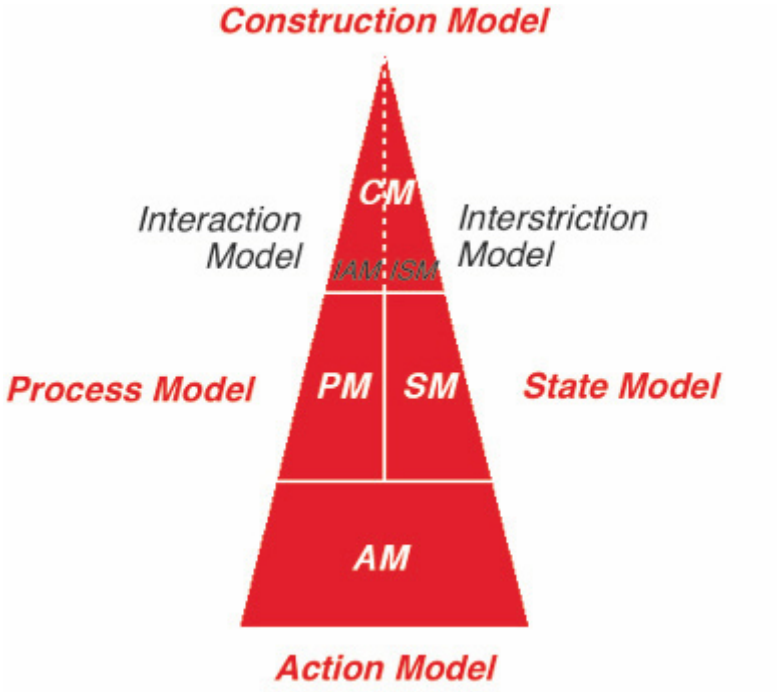
Struktura a popis jednotlivých modelů vychází zejména z [19] a [33].

Jak už bylo zevrubně popsáno v kapitole 2, DEMO se skládá ze 4 hlavních modelů a dalších podmodelů. Jedná se o:

- Construction Model (CM)
- Process Model (PM)
- State Model (SM)
- Action Model (AM)

Každý z těchto modelů se pohybuje na jiné úrovni abstrakce, což se dá dobře popsat na principu pyramidy, který je naznačen na obrázku 4.6. Construction Model, který je usazen na vrcholu této pyramidy pracuje s nejvyšší úrovní abstrakce a naopak Action Model, který je na obrázku zobrazen při

základu pyramidy je už velmi podrobný a detailní. Všechny tyto modely dohromady tvoří kompletní ontologickou znalost organizace.



Obrázek 4.6: DEMO Aspect models [10]

4.5.3.1 Construction Model

Jak už jeho název napovídá, Construction Model se zabývá konstrukcí organizace na té nejvyšší úrovni abstrakce. Construction Model tvoří dva podmodely: *Interaction Model (IAM)* a *Interstriction Model (ISM)*. Cílem Construction Modelu je v podstatě jen identifikovat jednotlivé ontologické transakce a jejich výsledky.

Interaction Model (IAM) zobrazuje actory (initiatory a executory) a transakce, které tito actoři vykonávají. Jedná se o vysoce abstraktní pohled, takže v něm není vidět jednotlivé kroky transakcí, ale čistě jen interakci mezi actory, která je prováděna právě prostřednictvím transakce.

IAM se skládá z *Transaction Result Table (TRT)* a *Actor-Transaction Diagramu (ATD)*. TRT je jednoduchá tabulka identifikovaných ontologických transakcí a jejich výsledků. ATD je diagram, který ukazuje transakce mezi actory.

ISM je založen na IAM a obsahuje dva diagramy a tabulku:

- **Actor Bank Diagram (ABD)** – zobrazuje vztah mezi actory a informačními bankami

- **Organization Construction Diagram (OCD)** – jedná se o kombinaci ABD s ATD. Tedy actoři, transakce mezi nimi a navíc ještě propojení s informačními bankami.

- **Bank Contents Table (BCT)** – tabulka popisující obsah bank faktů.

4.5.3.2 Process Model (PM)

Pokud se Construction Model nacházel na vysoké úrovni abstrakce, tak *Process Model (PM)* jde o krok více do detailu. Kde CM pouze pojmenovával transakce, PM rozvádí jejich jednotlivé kroky tak, jak jsou popsány v transakčním axiomu Ψ-theory. Zároveň ukazuje také vztahy mezi jednotlivými transakcemi. Stále je však potřeba mít na paměti, že ačkoliv PM detailně popisuje strukturu podnikového procesu, tak je stále abstrahován od implementace a vlastní realizace daného procesu včetně takových věcí jako je výměna dat apod.

V případě přípravy na vývoj informačního systému je PM dobré místo, odkud začít se soupisem požadavků a případů užití.

PM se skládá z:

- **Process Structure Diagram (PSD)** – jak už název napovídá, zobrazuje strukturu procesu. Jak už bylo popsáno výše, podnikový proces se skládá z jedné či více vzájemně provázaných transakcí. Kroky, které nejsou popsány v PSD nejsou povolené a PSD by měl obsahovat i odvolávací vzory.

- **Information Use Table (IUT)** – přímo se váže na State Model. Pro každou objektovou třídu, typ skutku a výsledku ze State Modelu specifikuje, v jakých krocích PM se používají jejich instance.

4.5.3.3 State Model (SM)

State Model (SM) se zabývá především P-worldem – jeho objektovými třídami, typy skutků, typy výsledků a existenčními ontologickými pravidly. SM je přímo navázaný na Action Model, zobrazuje ale pouze informace, které jsou relevantní pro chod organizace. SM obsahuje:

- **Object Fact Diagram (OFD)** – tento diagram zobrazuje vztah mezi objektovými třídami a typy výsledků.

- **Object Property List (OPL)** – popisuje objektové třídy a jejich vlastnosti.

4.5.3.4 Action Model (AM)

Action Model (AM) existuje v metodologii DEMO na té nejnižší úrovni abstrakce a tudíž je velmi detailní a ostatní modely na něm stojí, AM popisuje především pravidla, kterými se operace v organizaci řídí. Tato pravidla jsou v AM popsána slovně pomocí pseudo-algoritmického jazyka, ve kterém specifikují co má být uděláno při request, promise, state a accept.

Už z popisu výše je zřejmé, že AM bude velmi užitečný nástroj při implementaci softwaru. Action Model je ontologicky atomický, což znamená, že již nemůže být dále rozdělen na podmodely.

4.6 Postup při tvorbě DEMO modelů

[10] přímo nespecifikuje, jak a v jakém pořadí modely vytvářet. Pro zápis postupu v této sekci je proto využit a doplněn postup uvedený v [19].

1. Prvním krokem při tvorbě DEMO modelů je získání textového popisu organizace nebo procesu. Textový popis lze získat od doménových a procesních vlastníků.

2. Na textový popis je aplikována *Performa-Informa-Forma analýza* a jednotlivé aktivity v popisu jsou klasifikovány jako ontologické, infologické nebo datalogické.

3. V označených aktivitách jsou rozlišeny C-acty a P-acty a dále actoři, kteří jsou za jejich provedení zodpovědní.

4. Vytvoření *Transaction Result Table (TRT)* obsahující všechny identifikované transakce, jejich výsledky a transakční kroky identifikované v předchozím kroku.

5. Aplikace kompozičního axiomu a zápis struktury výsledků transakcí dle závislostí, které je možné odhalit v textovém popisu.

6. Identifikace rolí actorů pro každou transakci (Actor-Initiator, Actor-Executor).

7. Rozhodnutí, které transakce patří dovnitř organizace, a které probíhají vně. Vytvoření *Actor-Transaction Diagramu (ATD)*. Nyní máme k dispozici kompletní *Interaction Model (IAM)*.

8. Vytvoření *Process Structure Diagramu (PSD)*, čímž jsme vytvořili *Process Model (PM)*.

9. Na základě textového popisu je možné identifikovat všechny *action rules* a vytvořit na jejich základně *Action Model (AM)*.

10. Díky AM a provedení analýzy tříd objektů a typů vlastností je možné vytvořit *Object Fact Diagram (OFD)* a *Object Property List (OPL)* ze *State Modelu (SM)*, který tímto vytvoříme.

11. Nyní je možné vytvořit poslední část PM, kterou je *Information Use Table (IUT)*.

12. Na závěr je nutné identifikovat toky informací a na jejich základě a IAM je možné vytvořit *Actor Bank Diagram (ABD)*, *Organization Construction Diagram (OCD)* a *Bank Contents Table (BCT)*. Po dokončení tohoto kroku máme k dispozici i *Interstriction Model (ISM)*..

Kapitola 5

Využití metodologie DEMO pro vytváření BPMN modelů

5.1 Motivace

V předchozích kapitolách jsme se zabývali notací BPMN a metodologií BPMN odděleně a popsali jsme jejich silné i slabší stránky. Motivací pro napsání této práce však je silné stránky obou přístupů zkombinovat a na takovém základě vytvořit metodu, které umožní vytvářet BPMN modely podnikových procesů, které budou *jednoznačné*, *kompletní* a *konzistentní*. Že je spojení těchto přístupů vhodné potvrzuje i [34], když píše, že BPMN začíná tam, kde DEMO končí.

5.1.1 Slabé stránky DEMO

Jak již bylo v průběhu této práce několikrát zmíněno, na metodologii DEMO je nejcennější to, že říká tvůrcům procesních modelů „jak“ by měli při vytváření modelů podnikových procesů postupovat a nedává jim tedy pouze notaci, pomocí které je možné modely „kreslit“. Nutno však zároveň říct, že teoretický základ metodologie DEMO (tedy kombinace Enterprise ontology a Ψ-theory) je pro běžného uživatele velmi nepřístupný a obtížný na pochopení, protože jeho základy sahají až k fundamentálním filozofickým konceptům, jakými právě ontologie jsou.

Křivka učení je tedy v tomto případě velmi pozvolná a k plnému pochopení metodologie DEMO je potřeba hodně času a učení metodou pokus-omyl. Autor této práce má zkušenosti z účasti na kurzu MI-MEP (Modelování ekonomických procesů) na Fakultě informačních technologií ČVUT, kde po dobu jednoho semestru bylo certifikovanými DEMO profesionály vyučováno DEMO po teoretické i praktické stránce. Z autorových zkušeností i po absolvování celého tříměsíčního kurzu měli jeho účastníci problém se základními technikami

metodologie DEMO, jakou je například korektní aplikace distinkčního axiomu Performa-Informa-Forma analýzou, ale i s dalšími fundamentálními věcmi.

Nebudeme tedy daleko od pravdy, když konstatujeme, že v případě běžných uživatelů (návrhářů, analytiků, vývojářů) v rámci podniků bude problém podobný. Dalším nedostatkem DEMO je malé množství nástrojů, které DEMO podporují a žádný z existujících dosud nepokrývá všechny aspect modely, které DEMO obsahuje. Nutno však říci, že komunita kolem DEMO je velmi aktivní a na vývoji těchto nástrojů se intenzivně pracuje.

Jedním z cílů této práce je umožnit uživatelům nadále využívat nástroje, které znají a umí používat (BPMN), ale v rámci metodologie DEMO identifikovat „nezbytné minimum“ z teoretického základu, které zvýší kvalitu výsledných procesních BPMN modelů a tím umožní i jejich lepší analýzu a optimalizaci.

5.1.2 Slabé stránky BPMN

U BPMN nacházíme slabiny tam, kde jsme u DEMO identifikovali silné stránky a naopak. Jak již bylo naznačeno v kapitole 2, velkým problémem BPMN je absence teoretického základu, který dává návrhářům rámec, podle kterého se mohou řídit a vytvářet modely podle pevně daných pravidel a postupů. Absence takového pevného rámce pak ústí v to, že modely jsou neúplné a chybí v nich podstatné informace nebo v to, že v momentě, kdy různí uživatelé modelují jeden a ten samý proces, tak skončí s různými modely i když jsou všechny dle specifikace BPMN korektní.

Problematická je i absence seznamu pravidel, co který element v BPMN přesně vyjadřuje, kdy ho používat a kdy ne. Popis jednotlivých elementů je rozprostřen na 500 stranách v celé specifikaci BPMN [23] a uživatelé jsou tak odkázáni na interpretaci této specifikace, která pochází většinou od tvůrců modelovacího nástroje, který tito uživatelé aktuálně používají. To pak dále tříští způsob, jakým jsou elementy v praxi používány.

V praxi v organizacích často vidíme, že mezi procesními analytiky a vývojáři přirozeně vzniká jakási pseudo-metodologie, právě z důvodu toho, aby bylo možné uvnitř organizace konzistentně vytvářet modely podnikových procesů stejným způsobem i v týmu více lidí a aby byla zajištěna kontinuita i v případě fluktuace členů tohoto týmu. Jedním z přínosů této práce, by mělo být vytvoření pevného rámce založeného na Enterprise ontology a Ψ-theory, který by mohl nahradit tyto pseudo-metodologie.

Zajímavá zjištění uvádí [35], kde autoři při svém výzkumu aplikovali zásady, na kterých stojí metodologie DEMO, na dva klíčové procesy ve velké organizaci (více než 2000 zaměstnanců). Tyto procesy obsahovaly dohromady přes 500 aktivit a více než 60 actorů. Analytici prověřovali, zda je procesní model *konzistentní*, tedy zda pořadí kroků v tomto procesu odpovídá transakčnímu axiomu. Dále se zajímali, zda je model *kompletní*, neboli jestli všechny kroky transakce dle transakčního axiomu DEMO se dají namapovat na akti-

vity uvedené v BPMN modelu. Po důkladné analýze těchto procesů došli k následujícím zjištěním:

- V původních procesech chybělo 25 % P-factů neboli tyto procesy obsahovaly aktivity, které nebylo možné vztáhnout k vytváření nějakého výsledku (hmotného či nehmotného).
- Chybělo rovněž 25 % *request* C-actů. Tedy výsledky byly produkovány bez explicitního požadavku na jejich vytvoření a nebylo tak možné jasně identifikovat Initiatora celé transakce.
- Chybělo 50 % *promise* C-actů. Requesty byly implicitně potvrzovány bez jakékoliv jasné dohody mezi actory.
- Chybělo 25 % *state* C-actů. Výsledky transakce tak nebyly jasně komunikovány s jejím iniciátorem a tedy není jasné, zda dohlédnutí na výsledek transakce leží na Initiatorovi nebo Executorovi.
- Chybělo 40 % *accept* C-actů. Nebylo tedy možné identifikovat, jak jsou výsledky transakce akceptovány jejím Initiatorem a jestli takové výsledky splňují požadavky, které na ně byly kladeny při requestu.

Je tedy zřejmé, že absence pevného teoretického základu je reálným problémem (ač ho uživatelé sami nepociťují [34]) a jeho vytvoření by zvýšilo kvalitu vytvářených modelů podnikových procesů.

5.1.3 Benefity kombinace DEMO a BPMN

Hlavním přínosem této práce je vytvoření základů metody, která vychází z Enterprise ontology, Ψ-theory a metodologie DEMO. Tato metoda by měla v budoucnu umožní vytváření kvalitnějších BPMN modelů. Slovo „kvalitnější“ si na tomto místě zaslouží hlubší rozebrání.

Cílem této práce není vytvoření metodologie, která bude dokonale kopírovat DEMO a v podstatě jen umožní vytvoření DEMO modelů za použití notace BPMN (abstrahujme zde od toho, zda je to vůbec možné, to musí být případně potvrzeno dalším výzkumem). Pokud bychom se k takovému přístupu rozhodli, nedosáhli bychom vyřešení problémů uvedených v sekci 5.1. Cílem této práce je nalézt takový *teoretický rámec*, který bude stále pochopitelný pro uživatele a umožní jim vytvářet kvalitnější BPMN modely, ale nebude mít tak dlouhou křivku učení, jakou pozorujeme u DEMO. Co je zde tedy míněno „kvalitnějšími“ BPMN modely?

- Modely takto vytvořené budou *kompletní*, tedy nebude chybět žádný krok dle transakčního axiomu a bude jasně specifikováno, který actor je zodpovědný za provedení konkrétního transakčního kroku.

- Modely takto vytvořené budou *konzistentní*, neboli pořadí provádění transakčních kroků bude odpovídat transakčnímu axiomu.
- Modely takto vytvořené budou *jednoznačné*, neboli při modelování toho samého procesu by vždy měl vzniknout stejný výsledný model.
- Modely takto vytvořené budou *esenciální*, neboli oproštěné od implementačních detailů a budou zachycovat jen skutečné jádro činnosti organizace.

5.2 Existující možnosti společného využití DEMO a BPMN

Vhodnost spojení obou přístupů neušla pozornosti jiných autorů, například [34], [36] nebo [35]. Ve stručnosti budou na tomto místě závěry těchto prací popsány, protože i na jejich zjištěních autor této práce svůj návrh metody staví.

5.2.1 Analýza procesních modelů pomocí Ψ-theory

V [35] a [36] se autor zabývá využitím Ψ-theory pro analýzu modelů podnikových procesů s cílem zvýšit jejich *kompletnost* a *konzistentnost*. Za tímto účelem představuje autor metodu, jejímž vstupem je procesní model v notaci BPMN. Metoda se skládá z následujících čtyř kroků:

1. Provedení analýzy vstupního BPMN modelu s cílem klasifikovat jednotlivé elementy jako C-act nebo P-act a také rozlišit, zda jsou ontologické, infologické nebo datalogické. Dále je nutné se pokusit v modelu identifikovat actory odpovědné za vykonání C-actů a P-actů.

   Ontologické C-acty jsou následně klasifikovány dle jednotlivých transakčních kroků *standardního transakčního vzoru*.

2. V dalším kroku je vygenerován DEMO Process Structure Diagram (PSD) na základě seznamu actorů, transakcí, C-actů a P-actů získaných v předchozím kroku.

3. Ve třetím kroku je PSD diagram z předchozího kroku doplněn o C-acty a P-acty, které v něm chybí oproti transakčnímu vzoru. Dále je vytvořen DEMO Actor-Transaction Diagram (ATD), který bude sloužit zejména pro verifikaci upraveného BPMN modelu.

4. V následujícím kroku je provedena revize vstupního BPMN modelu a ten je na odpovídajících místech doplněn o elementy představující chybějící C-acty nebo P-acty identifikované v předchozím kroku. Takto revidovaný vstupní model je nyní *kompletní* a *konzistentní* dle Ψ-Theory.

Opakovanou aplikací této metody za účasti doménových i procesních vlastníků je možné dosáhnout procesních modelů, které uspokojí business uživatele a zároveň budou *kompletní* a *konzistentní* dle Ψ-Theory.

5.2.2 Rozšíření formálních základů BPMN pomocí Enterprise Ontology

Práce [34] nazvaná *Enhancing the Formal Foundations of BPMN by Enterprise Ontology* se na teoretické úrovni zabývá možností, jak využít Enterprise ontology jako ontologický základ pro vytváření BPMN modelů a argumentuje, že fundamentální koncepty BPMN jsou „podobné“ konceptům DEMO:

- Koncept *aktivity* je „podobný“ konceptu výkonu (performance) neboli DEMO P-factu nebo určité komunikaci o tomto výkonu neboli DEMO C-factu.
- Koncept *plavecké dráhy* je „podobný“ konceptu actora (Actor-Initiator nebo Actor-Executor).

Na tomto základě a následnou důkladnou analýzou axiomů Ψ-theory autoři došli k 14 pozorováním. Jelikož jsou tato pozorování jedním ze základních stavebních kamenů metody navrhované v této práci, budou zde uvedeny:

- Existují identifikovatelní *actoři* plnící určité role, kde se actor vztahuje k roli a nikoli nutně k jedinci, který tuto roli fyzicky vykonává.
- Existují identifikovatelné P-facty reprezentující nějaký *úkon*.
- Existují C-facty reprezentující *komunikaci* o konkrétním úkonu, který má být vykonán.
- Každý P-fact je vztažen pouze k jednomu unikátnímu C-factu a naopak.
- Každý actor vyjma kořenového Actora-Initiatora má jeden a pouze jeden vztah Actora-Executora k jednomu a pouze jednomu P-factu, ale může mít 0...n Actor-Initiator vztahů k jiným P-factům (potomkům).
- Každý P-fact může být definován jako množina 0...n jeho potomků.
- U každého rodičovského P-factu musí nejdřív dojít k dokončení (*state* a *accept*) P-factů, které jsou jeho potomky, dříve než může začít exekuce rodičovského P-factu.
- Actor, který je Executorem nějakého P-factu, je zároveň Initiatorem každého P-factu, který je jeho potomkem.
- Pouze Initiator kořenového P-factu není Initiatorem žádného jiného P-factu.

- U každého modelu existuje alespoň jeden P-fact, který je ve vztahu k actorovi, který je jen a pouze Executorem tohoto P-factu.
- Existuje množina osmi atributů (*request*, *promise*, etc.), které se unikátně vztahují ke konkrétnímu C-factu, který popisuje aktuální stav komunikace ohledně úkonu (vytvoření P-factu).

Autoři ve své práci představují nástin dvou přístupů, jak využít DEMO a Enterprise Ontology pro vytváření či vylepšení BPMN modelů. Jedním z přístupů je analýza a doplnění existujících BPMN modelů na základě čtyř základních axiomů Ψ-theory v základu podobně jak bylo popsáno v sekci 5.2.1.

Pro tuto práci přínosnější je nástin druhé možnosti – vytváření nových BPMN modelů na formálním základě, který z DEMO a Enterprise Ontology vychází. Vytváření takového BPMN modelu začíná sběrem informací o organizaci, jejím chodu a funkci. Dalším krokem je aplikace *distinkčního axiomu* a vyřazení všech informací, které nejsou ontologické. Třetím krokem je pak ve zbylých informacích nalézt ontologické transakce a actory, kteří vystupují v rolích Actor-Initiator a Actor-Executor ve vztahu k těmto transakcím. Závěrečným krokem je pak vytvoření DEMO Actor-Transaction Diagramu, který zobrazuje vztah mezi transakcemi a actory.

Výsledkem tohoto postupu tedy není BPMN model, ale kroky v něm nastíněné je nezbytné provést, aby bylo možné vytvořit BPMN model, který je *konzistentní* a *kompletní*. Metoda nastíněná v této práci právě na těchto krocích staví a posouvá je směrem ke skutečnému BPMN modelu.

5.3 Mapování základních DEMO konceptů na BPMN primitiva

5.3.1 Výchozí předpoklady

Důkladným studiem notace BPMN i metodologie DEMO bylo vypozorováno několik výchozích předpokladů, které navazují na pozorování dle [34], která jsme uvedli v sekci 5.2.

- V notaci BPMN nenajdeme žádné transakce tak, jak je definuje DEMO. Ve verzi BPMN 2.0 sice konstrukt nazvaný transakce existuje, ale v tomto případě se jedná pouze o speciální druh elementů, který vyžaduje definici kompenzačních aktivit v případě, že je nutné transakci odvolat.
- Neexistují tedy ani žádné pevně dané *transakční kroky* a výsledný procesní model může mít takřka jakoukoliv strukturu.
- Nejsou pevně specifikovány role účastníků (actorů) podnikového procesu.
- Není pevně dáno, že každý proces musí obsahovat popis „šťastných“ i „nešťastných“ scénářů a ty „nešťastné“ se v praxi často nemodelují.

- Není řečeno, které druhy aktivit se mají v modelu zachytit, a které jsou zbytné nebo příliš detailní.

5.3.2 Základní pravidla používání notace BPMN

V této sekci jsou vypsány základní pravidla, jak používat některé elementy z notace BPMN tak, aby jejich použití korespondovalo s vybranými koncepty z metodologie DEMO dle základu metody představené v této práci. Z metodologie DEMO byly vybrány zejména ty koncepty, které jsou součástí transakčního axiomu, který je pro vlastní tvorbu modelů podnikových procesů nejzásadnější a vyjádření tohoto axiomu pomocí korespondujících elementů BPMN by výrazně přispělo k možnosti vytvářet kvalitnější modely v této notaci tak, jak bylo popsáno v sekci 5.1.3

5.3.2.1 Vyjádření actorů

Actor je v DEMO definován jako kombinace *odpovědnosti* a *kompetence* a není nezbytně svázán s konkrétním jedincem, který ve výsledku aktivitu či transakci fyzicky provádí. Pokud chceme v BPMN transakce zobrazovat, musí se podařit v paletě elementů, které BPMN nabízí, nalézt takové, pomocí kterých je možné actory korektně vyjádřit. Vybraný element musí splňovat tyto předpoklady:

- Musí jasně určovat, které kroky transakčního axiomu (C-acty a P-acty) patří do pole *zodpovědnosti* daného actora.
- Musí umožňovat *komunikaci* s druhým actorem v transakci.
- Musí umožňovat navazování dalších actorů a transakcí v *transakčním stromu*.

Pro znázornění actora se nabízí využití BPMN elementu *plavecká dráha (swimlane)*. BPMN nijak konkrétně nespecifikuje, jak plavecké dráhy využívat a nechává to tedy v rukou osoby, která model vytváří [23]. Pro znázornění dvou actorů v transakci by pak sloužily dvě sousedící plavecké dráhy. Komunikace mezi nimi by pak probíhala především pomocí *sekvenčních toků*.

Další možností, jak reprezentovat actora, je teoreticky použitím elementu *bazén*, který by tak fungoval samostatně pro každého actora a jelikož standard notace BPMN 2.0 zakazuje, aby sekvenční toky překročily hranice bazénu, museli by actoři mezi sebou koordinovat kroky (C-acty, P-acty) pomocí zasílání zpráv. A právě zde tkví hlavní úskalí tohoto přístupu. V odborné literatuře (například [23]) je obecně uváděno, že není dobrým přístupem modelovat aktivity v rámci jednoho procesu uvnitř více bazénů. Vznikají pak problémy v navazování posloupnosti aktivit na sebe, ale i s korektností výsledného BPMN modelu dle zásad DEMO neboť využití bazénů nám rozdělí transakci fakticky do dvou procesů, což přináší problémy, které budou podrobně popsány v sekci 5.4.3.

5.3.2.2 Vyjádření C-actů a P-actů

C-acty a *P-acty* jsou jednotlivé kroky v transakčním vzoru, které jsou prováděny v definovaném pořadí a jsou v transakci vždy přítomny, i když jsou někdy prováděny *mlčky*. Pro vyjádření C-actů a P-actů navrhovaná metoda využívá BPMN element *úloha (Task)*, který je dále nedělitelným typem *aktivity*. Úlohy mohou být dále rozčleněny na ty, které provádí člověk, které systém nebo nějaký stroj atp. Pro účely modelování popsané v této práci se nejlépe hodí abstraktní úlohy, které jsou bez označení.

V předchozím odstavci bylo zmíněno, že některé transakční kroky mohou být prováděny *mlčky*. To znamená, že ve skutečnosti neexistuje žádná aktivita, která by daný C-act prováděla, nicméně i provedení C-actu *mlčky* se dle metodologie DEMO počítá jako faktické provedení C-actu. [10]

5.3.2.3 Vyjádření C-factů

C-fact neboli *coordination fact* je výsledkem provedení C-actu. Je tedy přímo navázán na C-act, který jej „vytváří“. V BPMN lze C-fact vyjádřit třemi způsoby:

1. C-fact v modelu není explicitně vyjádřen. Jeho existence je zajištěna implicitně pomocí *sekvenčních toků* vycházejících z *úloh*, které reprezentují C-act.

2. C-fact je vyjádřen pomocí *signálu*. Důvod pro použití elementu signál je ten, že druhý actor by měl být o vzniku C-factu (respektive změně *C-worldu*) informován, což způsob představený v bodě 1 nezaručuje.

3. C-fact je vyjádřen pomocí *zprávy*, kterou zašle actor, který příslušný C-act vykonal, druhému actorovi.

V [10] se uvádí, že v DEMO modelech je C-fact zakreslen mezi elementy, které reprezentují role actorů, což znamená, že C-fact, existuje v jejich společném *mezisvětě*, neboli oba actoři mohou o vzniku C-factu vědět. Z toho vyplývá, že pro reprezentaci C-factu je nejvhodnější některý ze způsobů uvedených v bodech 2 nebo 3.

5.3.2.4 Vyjádření P-factů

Podobně jako C-fact je i *P-fact* výsledkem provedení konkrétního P-actu. V transakci se objevuje pouze jednou a zjednodušeně lze říci, že je „výsledkem“ celé transakce. P-fact může být hmotné povahy (např. upečení pizzy), ale i nehmotné (např. rozhodnutí poroty). Metoda, která je nastíněna v této práci, P-fact v modelu BPMN explicitně neuvádí. Důvody pro toto rozhodnutí jsou následující:

- Korespondující P-act (vyjádřený *úlohou*) nazvaný *Execute* a *sekvenční tok* z něj vycházející už sami o sobě vyjadřují vznik P-factu.
- Actor-Initiator je o vzniku P-factu informován ve chvíli, kdy je mu předložen pomocí C-actu *state*, který dle transakčního axiomu nastane vždy.
- Ve standardním i základním transakčním vzoru v metodologii DEMO je P-act i P-fact uváděn pouze v rámci agendy patřící Actorovi-Executorovi.

5.3.2.5 Vyjádření Agendy

> Slovo „agenda“ vyjadřuje „co musí být uděláno“, neboli agenda je „úkol k vyřešení“. [10] [15]

V metodologii DEMO nejednají Actoři náhodně, ale každý jejich krok je striktně definován pomocí *action rules* v DEMO Action Modelu. Například je jasně řečeno za jakých okolností se vytvoření P-factu přislíbí (*promise*) a kdy je nutné *request* odmítnout. *Agenda* dle DEMO je tvořena čtveřicí parametrů:

- *C-fact*, který má být vykonán
- *P-fact*, kterého se C-fact týká
- *Čas* vztahující se k provedení P-factu
- *Čas*, kdy by měl být C-fact vykonán

Jelikož návrh metody, který tato práce popisuje, pracuje pouze s existujícími BPMN elementy, není v tuto chvíli možné kompletní agendu pomocí nich vyjádřit. Řešením by bylo buď vytvořit vlastní doplňující model podobný DEMO Action Modelu nebo použít přímo DEMO Action Model. Dalším výzkumem je nutné ověřit, zda je vytvoření takové vrstvy nutné a případně takovou vrstvu vytvořit tak, aby byla připravena pro implementaci v softwarovém nástroji, který by rovněž umožnil simulaci modelovaného procesu.

Metodologie navrhovaná v této práci si vystačí s částečným zobrazením agendy přímo v BPMN modelu, kterého je docíleno pomocí *plavecké dráhy* (případně *bazénu*), *sekvenčního toku* a *časového hlediska*. Jinými slovy díky plavecké dráze nebo bazénu, který obsahuje elementy, jež spadají do kompetence a odpovědnosti actora, kterému je plavecká dráha nebo bazén příslušná a díky sekvenčnímu toku, který určuje pořadí provádění jednotlivých úloh daný actor v každém momentě v čase ví, kterou úlohu má provést. Pravidla pro vyhodnocení, zda přijmout *request* nebo *state* pro zjednodušení opomíjíme a předpokládáme, že jsou actorům známy.

[15] The word “agendum” is the singular form of the (plural) Latin word “agenda”. It means: what has to be done. In other words, an agendum is a to-do item. [10]

5.4 Vyjádření transakčního vzoru v BPMN

Pro metodologii, která je nastíněna v této práci, je *transakční vzor* dle transakčního axiomu DEMO tím nejdůležitějším stavebním kamenem. Dává totiž řád a jasnou strukturu tomu, jak popsat podnikový proces, který je dle metodologie DEMO jen uspořádanou strukturou na sebe navázaných transakcí – *transakčním stromem*.

Vyjádření transakčního vzoru pomocí elementů z notace BPMN je tak zcela zásadní pro to, aby mohl být naplněn účel nově vznikající metody, kterým je přispět k vytváření BPMN modelů, které budou *kompletní*, *konzistentní*, *jednoznačné* a *esenciální*. Jak bylo popsáno v sekci 5.3.2, vyjádření jednotlivých komponent transakčního vzoru je možné provést pomocí různých elementů a v této sekci jsou tyto různé přístupy rozebrány do většího detailu.

5.4.1 Vyjádření pomocí *úloh*

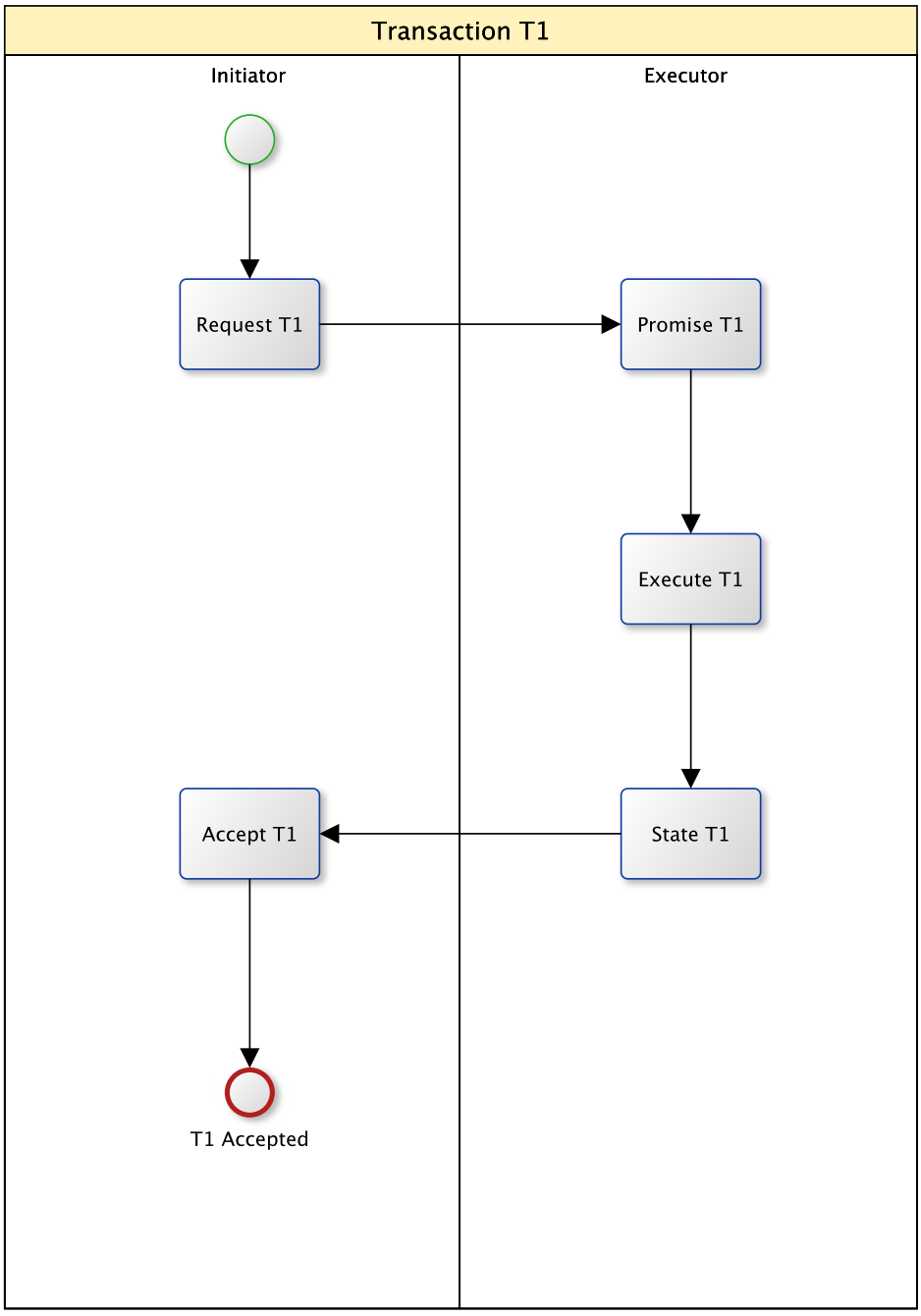
Prvním a nejjednodušším přístupem je vyjádřit transakční vzor čistě pomocí *úloh* a *sekvenčních toků*. Úlohy jsou použity pro vyjádření *C-actů* a *P-actů*. Sekvenční tok je použit ve více významech:

- jednak pro určení *pořadí* provádění jednotlivých C-actů,
- jednak pro zadávání C-factů do *agendy* druhému actorovi,
- jednak sekvenční tok vycházející z každé úlohy reprezentující C-act reprezentuje úspěšný *vznik korespondujícího C-factu*.

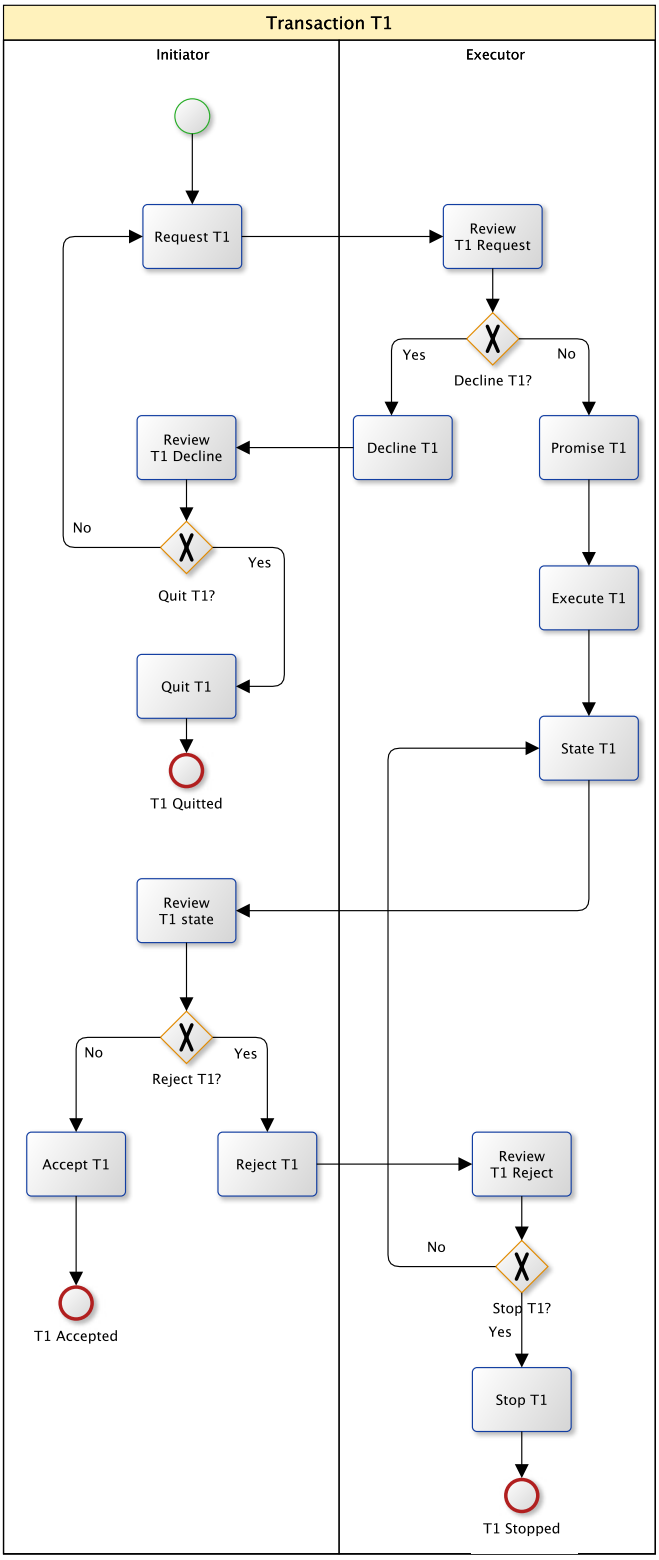
Na obrázku 5.1 je zobrazen *základní transakční vzor* pomocí úloh. *Bazén* slouží pro zastřešení celé transakce a *plavecké dráhy* oddělují agendy obou zúčastněných actorů (Initiatora a Executora). Základní transakční vzor bere v potaz pouze „šťastné scénáře“, takže transakce vždy skončí ve stavu *T1 Accepted*.

Obrázek 5.2 zobrazuje *standardní transakční vzor*, tedy kompletní transakci včetně všech „nešťastných scénářů (unhappy paths)“. Oproti základnímu transakčnímu vzoru zde přibyly čtyři *brány* označující místo, kde actor musí rozhodnout, jak s agendou naloží. Rozhodnutí je exkluzivní, tudíž je použita *exkluzivní brána XOR*. Všechny C-facty, ze kterých transakce již nikam dál nepokračuje, jsou vyjádřeny BPMN elementem *konečná událost (End event)*.

V tomto zobrazení nejsou zobrazeny žádné C-facty, což je z hlediska DEMO problematické. Ačkoliv jejich vznik vyjadřuje sekvenční tok vycházející z C-actu, C-fact není explicitně vyobrazen a druhý actor o něm nemusí vědět, protože v modelu není žádná komunikace zobrazena. Dle [10] však oba Actoři mají mít umožněno vědět o vzniku C-factu.



Obrázek 5.1: Základní transakční vzor pomocí *úloh*



Obrázek 5.2: Standardní transakční vzor pomocí *úloh*

5.4.2 Vyjádření pomocí *úloh* a *signálů*

Další možností, jak transakční vzor zobrazit, je využití *úloh* a *signálů*. Narozdíl od způsobu popsaného v sekci 5.4.1 slouží *sekvenční toky* v tomto případě pouze k vyjádření pořadí, ve kterém jsou C-acty prováděny a pro předávání agendy mezi actory. O reprezentaci vzniklého C-factu se zde však stará BPMN element *signál*, což poskytuje dvě velké výhody. Jednak je díky tomu C-fact v modelu vizuálně zastoupen, takže se výsledný model více blíží transakčnímu vzoru tak, jak ho definuje DEMO a jednak BPMN element signál je dle specifikace [25] i [23] možné použít ke komunikaci uvnitř procesu, respektive k odeslání neadresné zprávy komukoliv, kdo je připraven jí naslouchat. Toto mnohem přesněji vyhovuje popisu společného *mezisvěta*, který jsme uvedli v sekci 5.3.2.3.

Pro reprezentaci actorů jsou v tomto případě opět použity *plavecké dráhy* uvnitř *bazénu*, který zastřešuje celou transakci. C-facty, ve kterých transakce končí jsou i v tomto případě vyjádřeny pomocí BPMN elementu *konečná událost*.

Na obrázku 5.3 vidíme *základní transakční vzor* pomocí úloh a signálů. Lze vidět, že po provedení C-actu *Request T1* se sekvenční tok přesouvá do signálu *T1 Requested*, který vyjadřuje vzniklý C-fact. Předpokládáme, že Actor-Executor tomuto Signálu naslouchá a může tak provést C-act *Promise T1*.

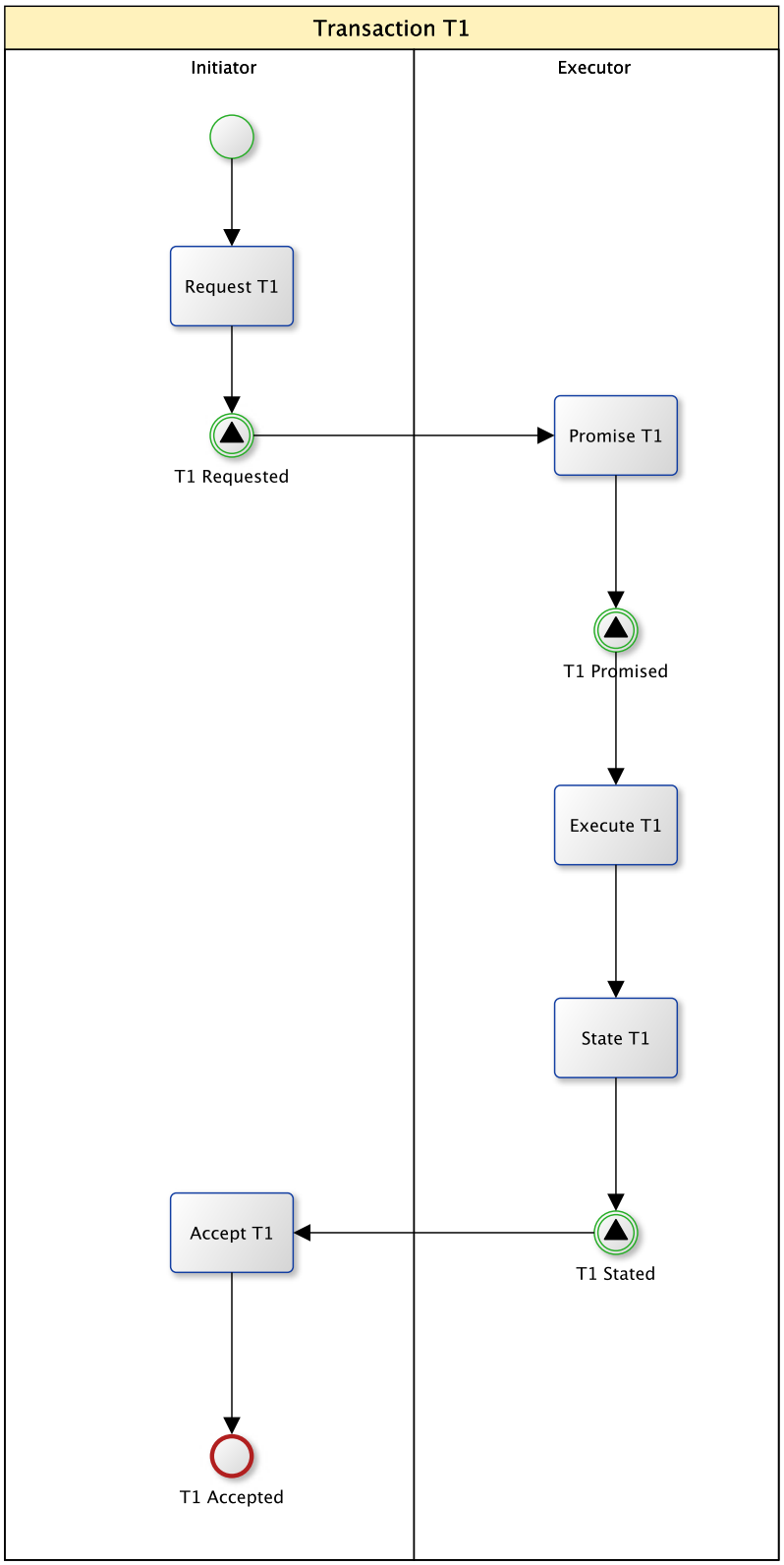
Obrázek 5.4 zobrazuje *standardní transakční vzor* vyjádřený úlohami a signály.

5.4.3 Vyjádření pomocí Úloh a Zpráv

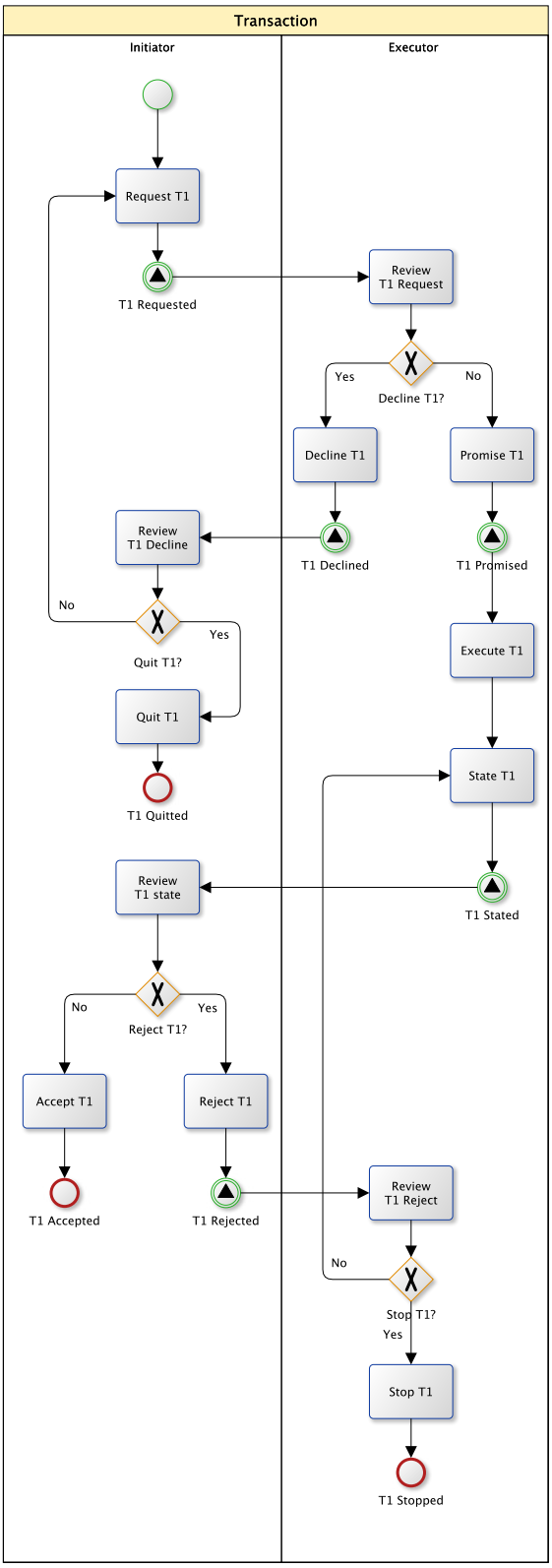
Jiným přístupem k vyjádření transakčního vzoru je využít ke komunikaci mezi actory BPMN element *zpráva*. Jelikož specifikace BPMN předepisuje, že zpráva může být použita pouze ke komunikaci mezi *bazény*, musí být v tomto případě pro každého actora vyhrazen zvláštní bazén. Na obrázku 5.5 lze vidět *základní transakční vzor* vyjádřený právě tímto způsobem.

Předávání agendy v tomto případě probíhá pomocí zaslání zprávy a *průběžné události (Intermediate event)*. Jakmile *Actor-Initiator* provede C-act *Request T1*, vyšle zprávu reprezentující C-fact, kterou přijme *Actor-Executor* na druhé straně pomocí *počáteční události* typu *catch (Message)*, čímž vznikne nová instance procesu na straně Executora. O každém provedeném C-actu je pak druhý actor informován pomocí zprávy.

V případě základního transakčního vzoru se většina problematických míst tohoto přístupu neprojevuje. K diskusi je však rozdělení actorů do dvou bazénů, které v podstatě znamená rozdělení transakce na dva podprocesy, jeden na straně Initiatora a jeden na straně Executora. Toto je poměrně výrazné odchýlení od transakčního axiomu tak, jak ho definuje DEMO a přináší komplikace, které se projeví v případě *standardního transakčního vzoru*.



Obrázek 5.3: Základní transakční vzor pomocí *úloh* a *signálů*



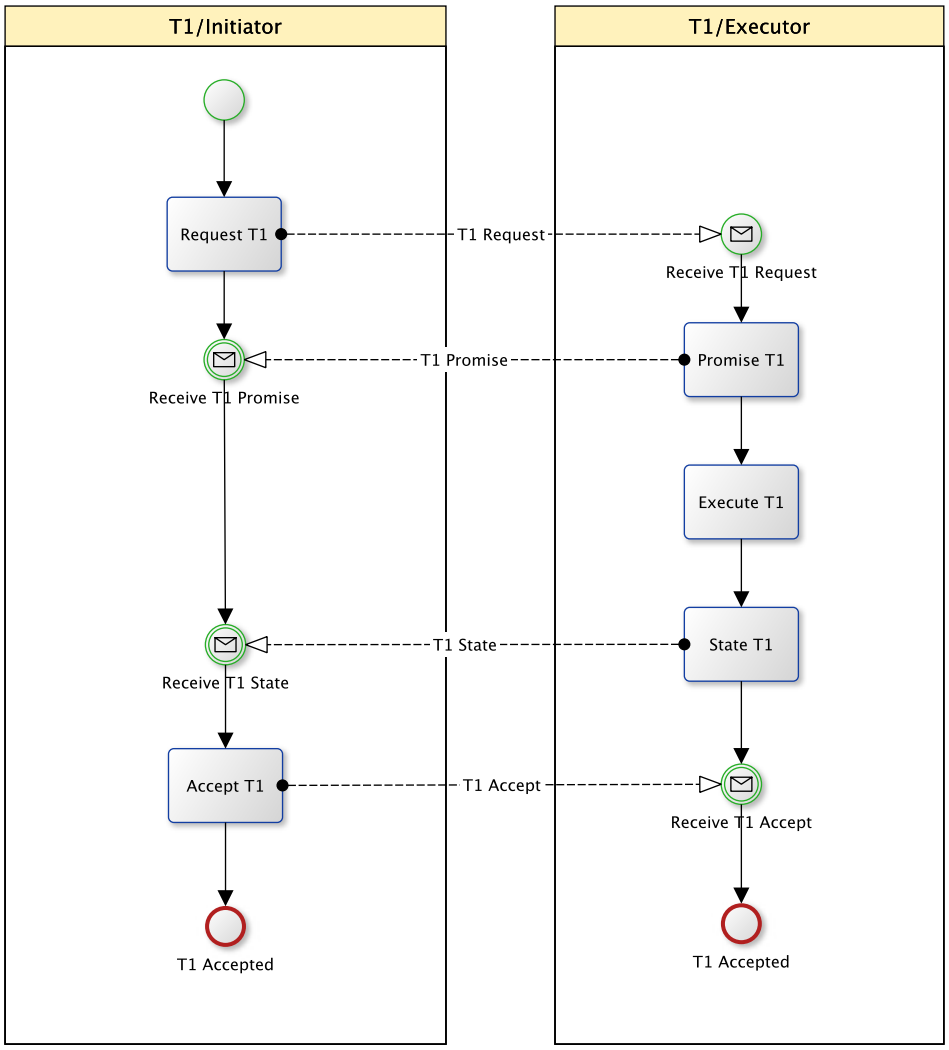
Obrázek 5.4: Standardní transakční vzor pomocí *úloh* a *signálů*

Ve standardním transakčním vzoru totiž existují místa, která při maximální snaze o dodržení standardního transakčního vzoru z DEMO mohou vyústit v *deadlock*. Jedná se o případy, kdy se actor rozhodne zamítnout výsledek C-actu *request* nebo *state*. V takovém případě totiž musí actor odpovědný za C-act rozhodnout, zda transakci zastavit nebo se pokusit o C-act znovu. Transakční vzor dle DEMO, ale nepočítá s tím, že by transakce byla rozdělena mezi dva procesy a že by měl existovat mechanismus, jak o tomto rozhodnutí informovat druhého actora. Řešením je v tomto případě přidat do transakčního vzoru mezistavové zprávy.

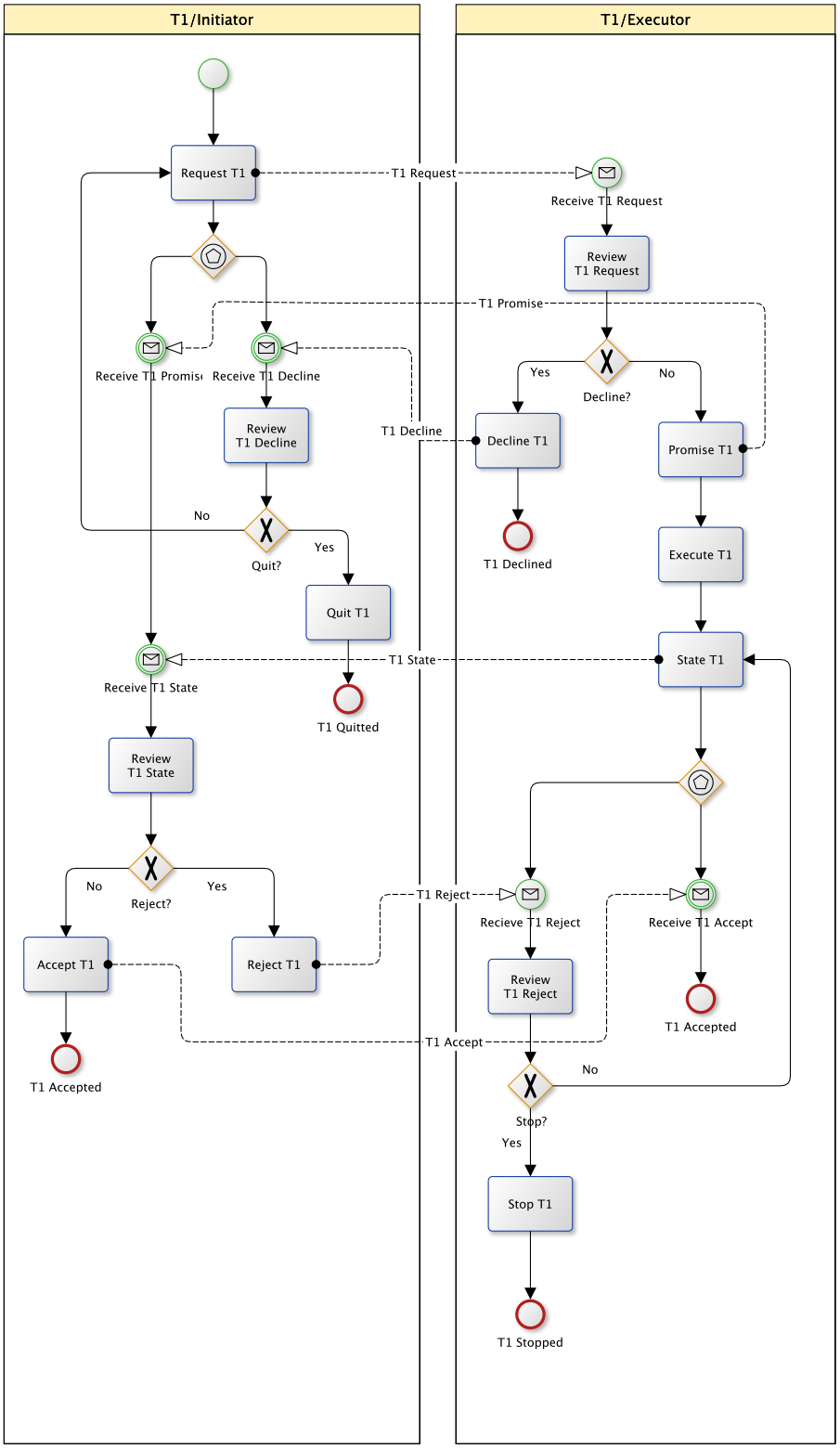
V případě, že bychom nepoužili mezistavové zprávy, nastal by problém hned v případě, že by se Actor-Executor rozhodl pro *Decline T1*. V takovém případě Executor vyšle zprávu Actoru-Initiatorovi a ten může buď transakci zastavit použitím *Quit T1* nebo poslat *Request T1* znovu. Jenže sekvenční tok v procesu na straně Executora je stále v situaci *Decline T1* a bez mezistavové zprávy (neboli zprávy o rozhodnutí jak se Initiator rozhodl naložit s *T1 Declined*) nemůže dále pokračovat. Částečným řešením, které můžeme vidět na obrázku 5.6, by v tomto případě mohlo být použít na tomto místě koncový stav *T1 Declined* a instanci procesu u Executora ukončit. V případě, že by se Initiator rozhodl transakci neukončit a poslal *request* znovu, vznikla by nová instance procesu na straně Executora.

Větší problém však nastane v případě, kdy by se Initiator po obdržení *T1 State* rozhodl tento *state* odmítnout a provést *Reject T1*. Executor by následně obdržel zprávu o *rejectu* a musel by rozhodnout, zda transakci zastavit provedením *Stop T1* nebo provést *State T1* znovu. Jenže bez mezistavových zpráv se Initiator nemá šanci dozvědět výsledek tohoto rozhodnutí Executora. Řešení popsané v předchozím odstavci zde nebude fungovat, protože pokud by se Executor rozhodl provést znovu *State T1*, proces na straně Initiatora by již byl ukončen. Jediným řešením jsou tedy mezistavové zprávy, které informují druhého actora o rozhodnutích v případě, kdy by mohlo dojít k deadlocku. Toto řešení lze vidět na obrázku 5.7. Jakmile nastane *Decline T1*, proces na straně Executora se přesune do *průběžné události (Multiple)* a čeká na příchod mezistavové zprávy od Initiatora, jestli se rozhodl transakci ukončit nebo pošle *request* znovu. Na základě toho buď proces ukončí nebo ho přesune na začátek a bude čekat na *request* od Initiatora. Analogicky je pak mezistavových zpráv využito v případě *state*.

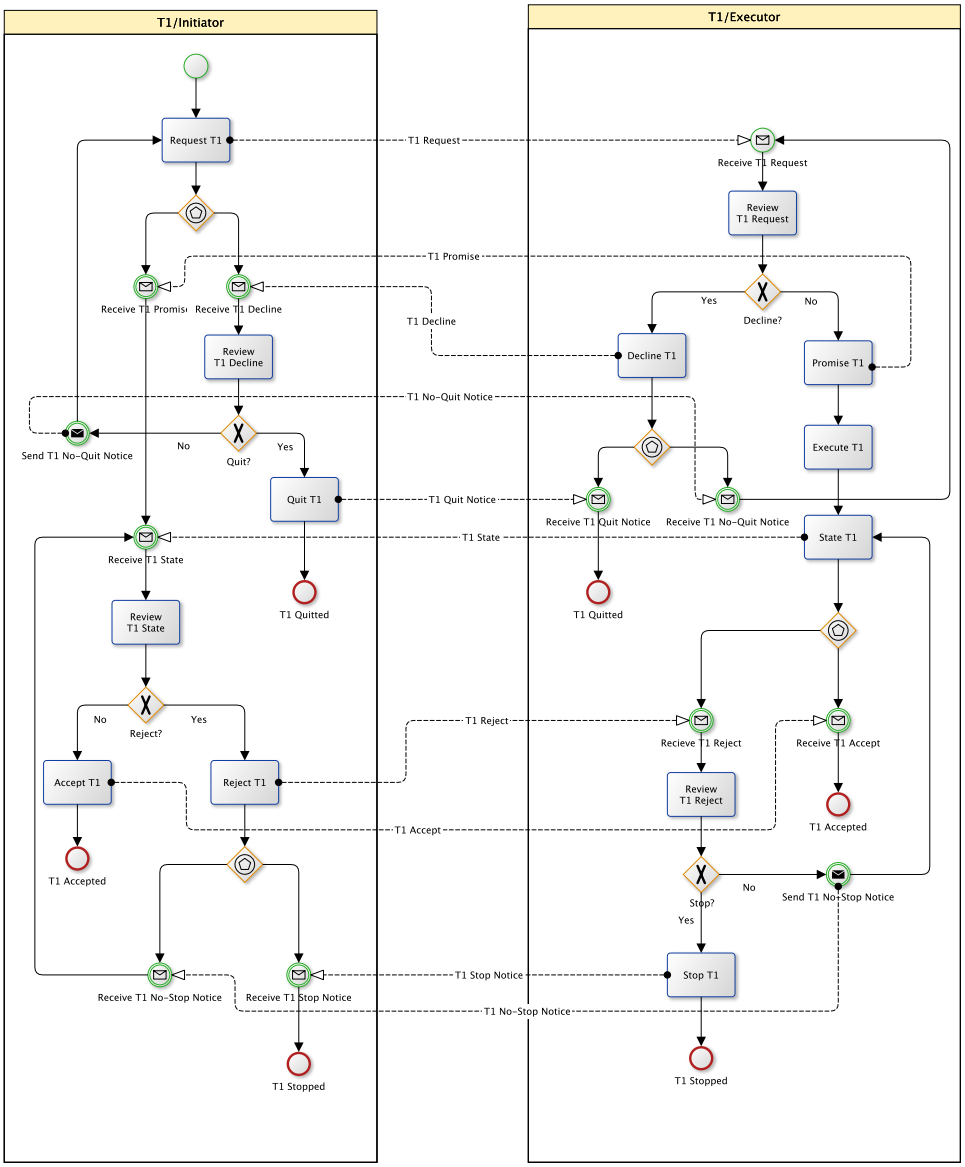
Využití mezistavových zpráv je však výrazným odchýlením od transakčního axiomu, protože tyto zprávy fakticky přidávají další komunikaci, se kterou transakční axiom nepočítá. Z tohoto důvodu je tento způsob pro vyjádření transakčního axiomu nevhodný.



Obrázek 5.5: Základní transakční vzor pomocí *úloh* a *zpráv*



Obrázek 5.6: Standardní transakční vzor pomocí *úloh* a *zpráv*



Obrázek 5.7: Standardní transakční vzor pomocí *úloh* a *zpráv* (včetně mezistavových zpráv)

5.5 Návrh metody pro vytváření BPMN modelů dle zásad DEMO

V této sekci je popsána metoda pro vytváření BPMN modelů dle zásad DEMO. Představená metoda je navržena pro vytváření zcela nových modelů, nikoli k analýze a doplnění již existujících BPMN modelů. Cílem této metody je umožnit popisovat průběh podnikových procesů v oblíbené notaci BPMN a zároveň zajistit, aby výsledný model byl *kompletní*, *konzistentní*, *jednoznačný* a *esenciální*.

5.5.1 Předpoklady

Aby bylo možné implementovat tuto metodu v praxi, je nezbytné, aby odpovědní pracovníci byli vybaveni následujícími teoretickými znalostmi, které vycházejí z Ψ-theory, Enterprise ontology a metodologie DEMO.

- Znalost *operačního axiomu* a schopnost rozlišit komunikační acty od produkčních.
- Znalost *transakčního axiomu*, *základního i standardního transakčního vzoru* včetně všech jeho kroků (C-actů, C-factů, P-actů, P-factů).
- Znalost a schopnost vytvořit DEMO Actor-Transaction Diagram (ATD) a DEMO Process Structure Diagram (PSD).
- Znalost notace BPMN včetně elementů, které oblíbená publikace [23] označuje jako Level 2.
- Znalost a schopnost aplikovat *kompoziční axiom*, tedy umět identifikovat jak jsou na sebe transakce navázané.

5.5.2 Krok 1: Získání textového popisu procesu

Při vytváření modelu určitého podnikového procesu je obvyklým prvním krokem získání informací od osob, které se tohoto procesu účastní nebo mu dobře rozumí. Může se jednat o výkonné i vedoucí pracovníky a obvykle je užitečné posbírat více pohledů na jeden konkrétní proces a následně diskutovat nesrovnalosti, které mohou vzniknout.

Díky tomu, že sběr informací provádějí pracovníci se znalostí transakčního axiomu i transakčních vzorů, mohou se cíleně zeptat na jednotlivé jeho kroky. Tedy zjistit, jak probíhá *request*, jak *state*, jak *accept* a podobně. Na konci by tedy měl vzniknout textový popis procesu s maximem relevantních informací pro další postup.

5.5.3 Krok 2: Aplikace distinkčního axiomu

Nyní jsme schopni na vzniklý textový popis procesu, chodu organizace či její složky aplikovat tzv. *Performa-Informa-Forma analýzu*, což není nic jiného než praktická aplikace *distinkčního axiomu*, který je popsán v sekci 4.3.4. Použijeme stejný postup jako v [10] a v textovém popisu označíme Performa aktivity červeně, Informa zeleně a Forma modře. Je naprosto přirozené, že u některých aktivit v textu bude problematické určit o jaký druh se jedná. S tím, jak budeme provádět další kroky dojdeme i k většímu porozumění celému procesu a je možné rozhodnutí přehodnotit. Výsledkem této části tedy je textový popis procesu s identifikovanými a vizuálně označenými Performa, Informa a Forma aktivitami.

5.5.4 Krok 3: Aplikace operačního axiomu

V tomto kroku budeme pracovat pouze s aktivitami, které jsme v Kroku 2 označili jako Performa (červeně). Úkolem je rozlišit, zda se jedná o C-act, C-fact, P-act nebo P-fact a dále k nim identifikovat příslušné actory tak jak předepisuje *operační axiom* popsaný v sekci 4.3.1. Opět není důvod nedodržet postup doporučený v [10], tedy části textu, které indikují actora označit hranatými závorkami „[“ a „]“. Části textu, které indikují C-act nebo C-fact označíme kulatými závorkami „(“ a „)“. Nakonec P-acty a P-facty nalezené v textu označíme špičatými závorkami „<“ a „>“. Texty uzavřené v závorkách navíc podtrhneme.

5.5.5 Krok 4: Zápis nalezených transakcí a jejich parametrů

Ve čtvrtém kroku označené C-acty, C-facty, P-acty a P-facty roztřídíme dle jednotlivých *transakčních kroků* (*request, promise, state, accept atd.*) dle *transakčního axiomu*, jehož popis najdeme v sekci 4.3.2. Zároveň specifikujeme výsledek transakce a všechny poznatky učiněné v předchozích krocích zapíšeme společně do tabulky, která má strukturu dle tabulky 5.1. V tabulce 5.1 můžeme vidět vyplněné parametry ukázkové transakce. Navržená struktura vychází ze struktury představené v [8].

Silnou stránkou DEMO, respektive transakčního axiomu je, že dává každé transakci (a tím i podnikovému procesu) pevně danou strukturu, kterou nelze obejít nebo upravit. I když v textovém popisu procesu nenalezneme popsané všechny transakční kroky (což se stane téměř vždy), neznamená to, že neexistují. V takovém případě je třeba jít zpět do kroku 1 a zjistit chybějící informace od doménových a procesních vlastníků a dle nových zjištění upravit i výsledky následujících kroků metody.

ID transakce	T01
Název transakce	Vyřízení objednávky *O*
Výsledek transakce	Objednávka *O* byla vyřízena
Initiator	Zákazník
Executor	Mia
Request	Zavolání s objednávkou
Promise	Objednávka potvrzena s cenou a časem vyhotovení
State	Zboží předáno zákazníkovi
Accept	Zákazník odchází s balíčkem z prodejny
Decline	Zamítnutí objednávky pokud zboží není na skladě
Reject	*chybí v popisu*
Revoke request	*chybí v popisu*
Revoke promise	*chybí v popisu*
Revoke state	*chybí v popisu*
Revoke accept	*chybí v popisu*

Tabulka 5.1: Tabulka s ukázkovými parametry transakce

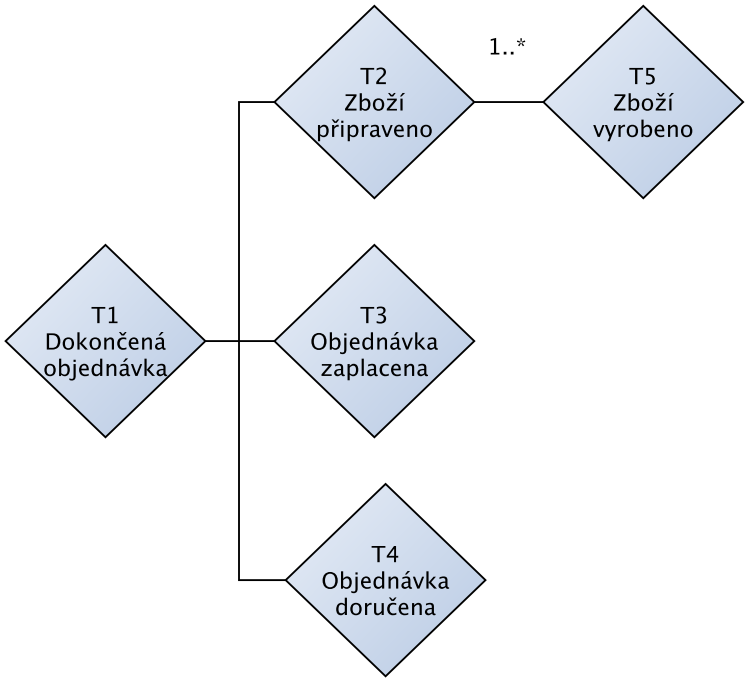
5.5.6 Krok 5: Aplikace kompozičního axiomu

Jak již bylo popsáno v 4.3.3 podnikový proces můžeme definovat jako množinu volně propojených transakcí. Úkolem pátého kroku je tedy identifikovat exekuce transakcí, které jsou závislé na předchozím provedení exekucí jiných transakcí. Jedinou možností jak toto prakticky provést je vrátit se k textovému popisu procesu a pokusit se identifikovat fráze, které značí závislost provádění P-actů nebo P-factů na sobě. Pokud se nám podaří objevit, že vznik P-factu B byl iniciován v průběhu vytváření P-factu A a zároveň je vznik P-factu A na B závislý, poté B je součástí A. Všechny takto objevené závislosti zakreslíme do diagramu, který můžeme vidět na obrázku 6.1, který demonstruje vizualizaci transakčního stromu na příkladě jednoduchého objednávkového procesu.

5.5.7 Krok 6: Vytvoření DEMO modelů

Nyní již máme dostatek informací pro vytvoření dvou DEMO modelů, které budou sloužit jako pevný bod pro verifikaci výsledného BPMN modelu i pro diskusi s business uživateli v případě požadavků na úpravy. Důvodem proč k této diskusi využívat DEMO modely (zejména ATD) a ne přímo výsledný BPMN model je vysoká úroveň komplexnosti BPMN modelu, který znesnadňuje jeho porozumění business uživateli.

Tabulky s popisem transakcí, které vznikly v kroku 4, jsou dostatečným podkladem pro vytvoření DEMO Actor-Transaction Diagramu (ATD) i DEMO Process Structure Diagramu (PSD). ATD model zachycuje jevy v organizaci na nejvyšší úrovni abstrakce, tedy pouze actory a mezi nimi probíhající trans-



Obrázek 5.8: Struktura závislosti transakcí v ukázkovém podnikovém procesu

akce. PSD jde o úroveň níže a zobrazuje jednotlivé transakční kroky stejně jako vzájemnou provázanost transakcí, kterou jsme identifikovali v předchozím kroku. Oba modely jsou podrobně popsány v rámci kapitoly 4.

5.5.8 Krok 7: Vytvoření BPMN modelu

Nyní již máme vytvořený dostatečně silný základ a můžeme přikročit k vytvoření samotného BPMN modelu, který je výsledkem celé metody. Při jeho tvorbě využijeme zejména PSD diagram vytvořený v předchozím kroku a způsob vyjádření standardního transakčního vzoru pomocí *úloh* a *signálů* popsaný v sekci 5.4.2.

Při vytváření BPMN modelu postupujeme tak, že následujeme PSD diagram od místa, kde dojde k provedení C-actu *request* u kořenové transakce celého podnikového procesu a postupně převádíme jednotlivé transakční kroky do BPMN primitiv (*plaveckých drah, úloh, signálů, sekvenčních toků, bran*) popsaných v sekci 5.4.2. V případě, že je podnikový proces složen z více vzájemně provázaných transakcí, musíme při vytváření BPMN modelu v momentě, kdy v PSD diagramu dojdeme na místo, kde je při provedení C-actu *promise* vytvořen *request* na další transakci, odklonit sekvenční tok a pokračovat v provádění transakčního vzoru transakce-potomka. Stejně tak pokračujeme i v dalších případech dokud se opět „nevynoříme“ (v případě,

že všechny transakce-potomci dopadnou úspěšně) v kořenové transakci a jsme schopni provést exekuci P-actu.

5.6 Další výzkum

Obsah této práce je prvním pokusem o vytvoření metody pro vytváření BPMN modelů podnikových procesů za použití teoretických konstruktů z Ψ-theory, Enterprise ontology a metodologie DEMO. Jako každá první verze i tato musí být doplněna dalším výzkumem a všechna představená východiska musí být prověřena a potvrzena. Pro zdůraznění je zde vypsáno několik bodů, kde autor považuje další výzkum za nezbytný:

1. Posouzení důležitosti a případně vytvoření určité formy textového modelu na způsob DEMO Action Modelu, který umožní jednak naprosto přesně zachytit informace o agendě actorů v daném momentě, jednak bude sloužit jako podklad pro implementaci simulace modelu pomocí nějakého softwarového nástroje a jednak umožní formální verifikaci pomocí gramatiky, která bude pro takový model vytvořena.

2. Posouzení, zda je BPMN element *úloha* dostatečný pro vyjádření DEMO C-actů a P-actů.

3. Posouzení použití konkrétních názvů aktivit pro jednotlivé transakční kroky identifikované v kroku 4 metody.

4. V DEMO verze 3 byla přidána do transakčního vzoru další vrstva umožňující prakticky kdykoliv „vzít zpět“ již vyjádřený C-act. Dalším výzkumem musí být posuzeno zda je toto možné znázornit v DEMO, jestli je to žádoucí a jaký to bude mít vliv na komplexnost výsledných BPMN modelů.

5. Návrh implementace automatického generování BPMN modelů z DEMO modelů pomocí vyjádření transakčního vzoru v BPMN dle sekce 5.4. Výsledkem by mělo být propojení se softwarovou aplikací, kterou připravuje firma Formetis.

Kapitola 6

Aplikace metody

6.1 Úvod

V předchozí kapitole byl představen návrh metody pro vytváření BPMN modelů za použití Enterprise ontology, Ψ-theory a metodologie DEMO. Cílem metody má být umožnit vytvářet BPMN modely, které budou vždy *konzistentní*, *kompletní* a *jednoznačné*.

V této sekci je navržená metoda aplikována po jednotlivých krocích na konkrétní příklad a na závěr jsou diskutovány výsledky. Aplikace kroků 1, 2, 3 a 6 vycházejí z analogických postupů popsaných v [10].

6.2 Aplikace metody

6.2.1 Krok 1: Získání textového popisu procesu

Pro demonstraci metody použijeme výňatek (první fázi) z příkladu Pizzeria Mama Mia z [10], kterou používá rovněž [34]. Popis situace je následující:

> Zákazníci si objednávají přímo v pizzerii, nebo si s objednávkou zavolají. V obou případech Mia zapíše jméno zákazníka, objednávku a celkovou cenou na objednávkový formulář. Na pultu leží seznam nabízených pizz a jejich cen. Mia obvykle nové menu vytváří během své každoroční dovolené. V případě telefonické objednávky také zaznamenává telefonní číslo. Navíc zopakuje objednávku a informuje zákazníka o ceně a předpokládaném čase, než bude pizza připravena. Pokud je to nutné, sdělí také zákazníkovi aktuální nabídku pizz. Objednávkové formuláře mají sériové číslo a jsou vyhotoveny ve dvou kopiích – v růžové a bílé kopii. Mia posune růžový formulář přes okno ve zdi do kuchyně, kde se Mario stará o pečení pizzy. Bílou kopii si Mia nechá za pultem. Jakmile Mario dokončil objednávku, podá pizzy v krabicích přes okno Mie,

> včetně růžové kopie objednávky. Mia pak hledá odpovídající bílou kopii, kterou podá spolu s krabicemi zákazníkovi a čeká na platbu. Může se stát, že Mario není schopen objednávce vyhovět kvůli chybějícím ingrediencím. V takovém případě prostrčí hlavu oknem ve zdi a upozorní Miu na problém. Vrátí také růžovou kopii. Pokud je zákazník přítomen v obchodě, Mia se s ním poradí, jak objednávku upravit. V případě, že zákazník není přítomen, což je častější případ, Mia změní objednávku podle vlastního uvážení. To někdy vede k vášnivým debatám v pizzerii, když si zákazník přijde pro svojí objednávku. Díky Miině temperamentu vždycky nakonec dojde k dohodě, která není nevýhodná pro ni. [16]

6.2.2 Krok 2: Aplikace distinkčního axiomu

V rámci druhého kroku navržené metody je třeba na prostý textový proces aplikovat distinkční axiom, neboli provést *Performa-Informa-Forma analýzu*. Tu provádíme tak, že pročítáme text a označujeme barevně (Performa červeně, Informa modře, Forma zeleně) aktivity v popisu. Vznikne nám tak text, ze kterého lze jednoduše barevně odlišit červené Performa aktivity, které budeme dále analyzovat.

Znovu se na tomto místě sluší zopakovat, že je naprosto přirozené, že některé aktivity bude zprvu obtížné barevně klasifikovat. V takovém případě je vhodné se pokusit odhadnout zařazení aktivity a po aplikaci dalších kroků a získání přesnějšího náhledu na problém své rozhodnutí případně přehodnotit. Výsledek po aplikaci druhého kroku vidíme níže:

> Zákazníci si objednávají přímo v pizzerii, nebo si s objednávkou zavolají. V obou případech Mia zapíše jméno zákazníka, objednávku

[16] Customers address themselves to the counter of the pizzeria or make a telephone call. In both cases Mia writes down the name of the customer, the ordered items, and the total price on an order form. On the counter lies a plasticized list of the available pizza's and their prices. Usually she pro- duces this list every year during their holiday. In case of an order by telephone she also records the telephone number. Moreover, she repeats the ordered items and informs the customer about the price and the expected time that the order will be ready. If necessary, she also tells the customer the assortment of pizzas. The order forms have a serial number and are produced in duplicate: a white and a pink copy. Mia shifts the pink one through a hatch in the wall to the kitchen, where Mario takes care of baking the pizzas. She keeps the white copy behind the counter. As soon as Mario has finished an order, he shifts the pizzas in boxes through the same hatch to Mia, including the pink order copy. Mia then seeks the matching white copy, hands it together with the boxes over to the customer, and waits for payment. It may happen that Mario is not able to fulfill an order completely because of missing ingredients. In such a case he puts his head through the hatch and notifies Mia of the problem. He then also returns the pink copy. If the customer is present in the shop, she confers with him or her what to do about it, and modifies the order. If the customer is not present, which is mostly the case for tele- phonic orders, she modifies the order to her own discretion. This leads sometimes to vigorous debates in the pizzeria when the customer comes for taking away the order. Thanks to Mia's temperament she always comes to an agreement that is not disadvantageous for her.

a celkovou cenou na objednávkový formulář. Na pultu leží seznam nabízených pizz a jejich cen. Mia obvykle nové menu vytváří během své každoroční dovolené. V případě telefonické objednávky také zaznamenává telefonní číslo. Navíc zopakuje objednávku a informuje zákazníka o ceně a předpokládaném čase, než bude pizza připravena. Pokud je to nutné, sdělí také zákazníkovi aktuální nabídku pizz.

Objednávkové formuláře mají sériové číslo a jsou vyhotoveny ve dvou kopiích – v růžové a bílé kopii. Mia posune růžový formulář přes okno ve zdi do kuchyně, kde se Mario stará o pečení pizzy. Bílou kopii si Mia nechá za pultem. Jakmile Mario dokončil objednávku, podá pizzy v krabicích přes okno Mie, včetně růžové kopie objednávky. Mia pak hledá odpovídající bílou kopii, kterou podá spolu s krabicemi zákazníkovi a čeká na platbu.

Může se stát, že Mario není schopen objednávce vyhovět kvůli chybějícím ingrediencím. V takovém případě prostrčí hlavu oknem ve zdi a upozorní Miu na problém. Vrátí také růžovou kopii. Pokud je zákazník přítomen v obchodě, Mia se s ním poradí, jak objednávku upravit. V případě, že zákazník není přítomen, což je častější případ, Mia změní objednávku podle vlastního uvážení. To někdy vede k vášnivým debatám v pizzerii, když si zákazník přijde pro svojí objednávku. Díky Miině temperamentu vždycky nakonec dojde k dohodě, která není nevýhodná pro ni.

6.2.3 Krok 3: Aplikace operačního axiomu

Jak již bylo řečeno v předchozí sekci, v rámci aplikace třetího kroku pracujeme pouze s Perfroma aktivitami, které jsme označili červeně. Úkolem je nyní klasifikovat červeně označené aktivity jako C-acty, C-facty, P-acty a P-facty za použití různých druhů závorek popsaných v sekci 5.5.4. Výsledek můžeme vidět na textu níže:

[Zákazníci] si (objednávají) přímo v pizzerii, nebo si s objednávkou zavolají. V obou případech [Mia] zapíše jméno zákazníka, objednávku a celkovou cenou na objednávkový formulář. Na pultu leží seznam nabízených pizz a jejich cen. Mia obvykle nové menu <vytváří> během své každoroční dovolené. V případě telefonické objednávky také zaznamenává telefonní číslo. Navíc zopakuje objednávku a informuje zákazníka o ceně a předpokládaném čase, než bude pizza připravena. Pokud je to nutné, sdělí také zákazníkovi aktuální nabídku pizz.

Objednávkové formuláře mají sériové číslo a jsou vyhotoveny ve dvou kopiích – v růžové a bílé kopii. [Mia] (posune) růžový formulář

> přes okno ve zdi do kuchyně, kde se [Mario] stará o <pečení> pizzy. Bílou kopii si Mia nechá za pultem. Jakmile Mario <dokončil> objednávku, (podá pizzy v krabicích přes okno Mie, včetně růžové kopie objednávky). Mia pak hledá odpovídající bílou kopii, kterou (podá spolu s krabicemi zákazníkovi) a čeká na <platbu>.
>
> Může se stát, že Mario není schopen objednávce vyhovět kvůli chybějícím ingrediencím. V takovém případě prostrčí hlavu oknem ve zdi a upozorní Miu na problém. Vrátí také růžovou kopii. Pokud je zákazník přítomen v obchodě, [Mia] se s [ním] (poradí), jak objednávku upravit. V případě, že zákazník není přítomen, což je častější případ, Mia (změní objednávku) podle vlastního uvážení. To někdy vede k vášnivým debatám v pizzerii, když si zákazník přijde pro svojí objednávku. Díky Miině temperamentu vždycky nakonec dojde k (dohodě), která není nevýhodná pro ni.

Můžeme vidět, že jsme závorkami a podtržením označili i aktivity, které jsme v kroku 2 označili modře nebo zeleně. Není na tom nic špatného a při analýze procesu je zcela přirozené neustále zpřesňovat své závěry. Můžeme se díky tomu vrátit ke krokům 2 a 3 a upravit je tak, aby text, který jsme označili závorkami byl vždy červený. Závěr třetího kroku by tedy mohl vypadat nakonec takto:

> [Zákazníci] si (objednávají) přímo v pizzerii, nebo si s objednávkou zavolají. V obou případech [Mia] zapíše jméno zákazníka, objednávku a celkovou cenou na objednávkový formulář. Na pultu leží seznam nabízených pizz a jejich cen. Mia obvykle nové menu <vytváří> během své každoroční dovolené. V případě telefonické objednávky také zaznamenává telefonní číslo. Navíc zopakuje objednávku a informuje zákazníka o ceně a předpokládaném čase, než bude pizza připravena. Pokud je to nutné, sdělí také zákazníkovi aktuální nabídku pizz.
>
> Objednávkové formuláře mají sériové číslo a jsou vyhotoveny ve dvou kopiích – v růžové a bílé kopii. [Mia] (posune) růžový formulář přes okno ve zdi do kuchyně, kde se [Mario] stará o <pečení> pizzy. Bílou kopii si Mia nechá za pultem. Jakmile Mario <dokončil> objednávku, (podá pizzy v krabicích přes okno Mie, včetně růžové kopie objednávky). Mia pak hledá odpovídající bílou kopii, kterou (podá spolu s krabicemi zákazníkovi) a čeká na <platbu>.
>
> Může se stát, že Mario není schopen objednávce vyhovět kvůli chybějícím ingrediencím. V takovém případě prostrčí hlavu oknem ve zdi a upozorní Miu na problém. Vrátí také růžovou kopii. Pokud je zákazník přítomen v obchodě, [Mia] se s [ním] (poradí), jak objednávku upravit. V případě, že zákazník není přítomen, což je

častější případ, Mia (změní objednávku) podle vlastního uvážení. To někdy vede k vášnivým debatám v pizzerii, když si zákazník přijde pro svojí objednávku. Díky Miině temperamentu vždycky nakonec dojde k (dohodě), která není nevýhodná pro ni.

6.2.4 Krok 4: Zápis nalezených transakcí a jejich parametrů

Pro snazší práci v dalších krocích i případnou budoucí automatizaci si identifikované transakce (korespondující s P-acty a P-facty, které jsme označili „špičatými závorkami“) zapíšeme do tabulky spolu s parametry každé transakce jako například její ID, výsledek transakce nebo jednotlivé transakční kroky. V případu Pizzerie Mama Mia jsme identifikovali následující transakce:

- Vyřízení objednávky *O* (tato transakce nemá v textovém popisu žádný korespondující P-act nebo P-fact, ale její existenci odvodíme z identifikovaného C-actu v podobě objednávání pizzy zákazníky)
- Příprava objednávky *O*
- Platba objednávky *O*

Identifikované transakce a jejich parametry zapíšeme do tabulek 6.1, 6.2 a 6.3.

ID transakce	T01
Název transakce	Vyřízení objednávky *O*
Výsledek transakce	Objednávka *O* byla vyřízena
Initiator	Zákazník
Executor	Mia
Request	Zavolání s objednávkou nebo objednání osobně
Promise	Potvrzení s cenou a časem vyhotovení (telefonicky), *chybí v popisu (osobní odběr)*
State	Pizza předána zákazníkovi
Accept	*chybí v popisu*
Decline	*chybí v popisu*
Reject	*chybí v popisu*
Revoke request	*chybí v popisu*
Revoke promise	*chybí v popisu*
Revoke state	*chybí v popisu*
Revoke accept	*chybí v popisu*

Tabulka 6.1: Parametry transakce T01

ID transakce	T02
Název transakce	Příprava objednávky *O*
Výsledek transakce	Objednávka *O* byla připravena
Initiator	Mia
Executor	Mario
Request	Posunutí objednávkového formuláře do kuchyně
Promise	*chybí v popisu*
State	Podání pizz přes okno Mie
Accept	*chybí v popisu*
Decline	Upozornění Mii na chybějící ingredience
Reject	*chybí v popisu*
Revoke request	*chybí v popisu*
Revoke promise	*chybí v popisu*
Revoke state	*chybí v popisu*
Revoke accept	*chybí v popisu*

Tabulka 6.2: Parametry transakce T02

ID transakce	T03
Název transakce	Platba
Výsledek transakce	Objednávka *O* byla zaplacena
Initiator	Mia
Executor	Zákazník
Request	Podání krabic s objednávkou a objednávkovým formulářem zákazníkovi
Promise	*chybí v popisu*
State	*chybí v popisu*
Accept	*chybí v popisu*
Decline	*chybí v popisu*
Reject	*chybí v popisu*
Revoke request	*chybí v popisu*
Revoke promise	*chybí v popisu*
Revoke state	*chybí v popisu*
Revoke accept	*chybí v popisu*

Tabulka 6.3: Parametry transakce T03

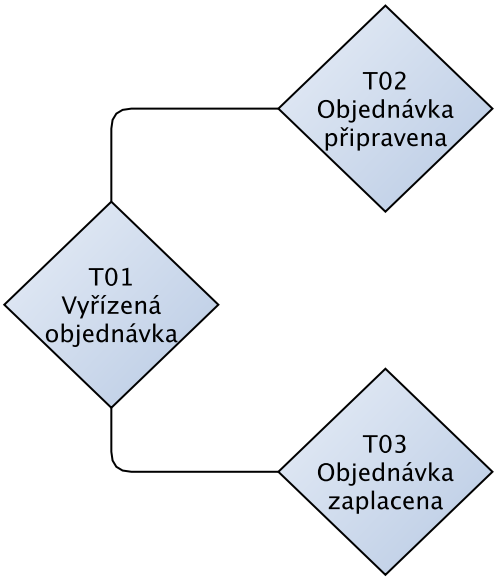
Ve vyplněných tabulkách můžeme vidět, že v nich valná část transakčních kroků chybí, protože nejsou uvedeny v textovém popisu případu Pizzerie Mama Mia. Takový případ bude nastávat v reálném světě velmi často a jeho ideálním řešení je návrat do kroku 1 a získání chybějících informací od vlastníků procesu a dalších zasvěcených osob.

Jelikož případ Pizzerie Mama Mia popsal v [10] Jan Dietz, zeptat se ho by bylo náročné. Pro aplikaci dalších kroků nám však stačí i takovýto neúplný popis. Ve výsledném BPMN modelu však pojmenujeme všechny transakční kroky dle názvů, které tyto kroky nesou v základním transakčním vzoru a ne jmény reálných aktivit, které jsme identifikovali.

6.2.5 Krok 5: Aplikace kompozičního axiomu

Dalším krokem je aplikování kompozičního axiomu a odhalení struktury, v jaké jsou jednotlivé transakce navzájem propojeny. Jinými slovy musíme zjistit, která transakce je kořenová a z jakých transakcí se tato transakce skládá, což nám pomůže určit, v jakém pořadí jsou transakce prováděny.

Tento krok není možné udělat jinak, než že se vrátíme k textovému popisu, který je výsledkem kroku 3 a hledáme náznaky závislostí mezi transakcemi a pořadí v jakém jsou tyto transakce vykonávány. Zjištění pak vyjádříme v jednoduchém grafu. Výsledek pro případ Pizzerie Mama Mia můžeme vidět níže na obrázku 6.1:

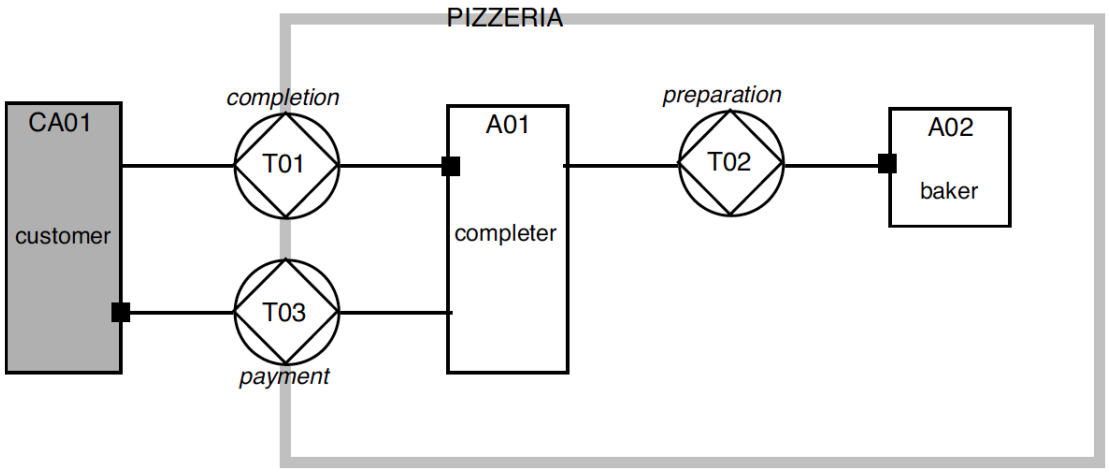


Obrázek 6.1: Struktura závislosti transakcí v případu Pizzerie Mama Mia

6.2.6 Krok 6: Vytvoření DEMO modelů

V předposledním kroku navržené metody je nutné vytvořit dva DEMO modely. Tyto modely slouží zejména pro zpětnou verifikaci vzniklého BPMN modelu.

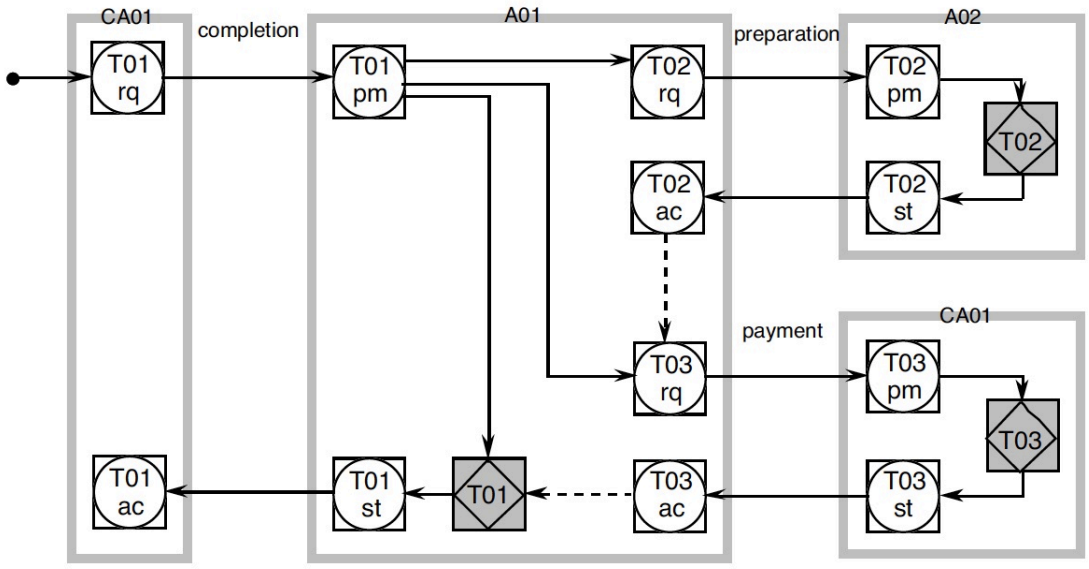
Prvním vytvořeným modelem je Actor-Transaction Diagram (ATD), který zachycuje pouze transakce a actory, kteří se transakce účastní. Na obrázku 6.2 můžeme vidět ATD pro případ Pizzerie Mama Mia. Zákazníka zde reprezentuje actor CA01 a jako jediný je vně organizace Pizzeria. Dalším actorem je Mia, která v tomto ATD vystupuje pod označením A01 a Mario, který je označen jako A02.



Obrázek 6.2: ATD Pizzerie Mama Mia [10]

Pro vytváření výsledného BPMN modelu nám však bude více nápomocný Process Structure Diagram (PSD), který zachycuje všechny transakční kroky dle transakčního vzoru. V kroku 7 bude naším úkolem tyto transakční kroky vyjádřené v PSD zobrazit pomocí BPMN primitiv, které jsou popsány v sekci 5.4.2.

Na obrázku 6.3 můžeme vidět PSD pro případ Pizzerie Mama Mia. Za povšimnutí stojí zejména šipky s přerušovaným tělem, které vyjadřují závislost vykonání C-actu na vykonání C-actu v jiné transakci. Díky tomu lze vidět, že vykonání Request T03 je závislé na Accept T02 a Execution T01 je závislá na Accept T03. Převedeno do lidské řeči to znamená, že než můžeme požádat zákazníka o zaplacení objednávky, musíme jí nejdřív připravit a zákazník ji musí akceptovat, a že než je objednávka vyřízena, musí dojít k jejímu zaplacení. Toto zjištění nám pomůže při vytváření BPMN modelu určit pořadí aktivit, které budeme propojovat pomocí sekvenčních toků.



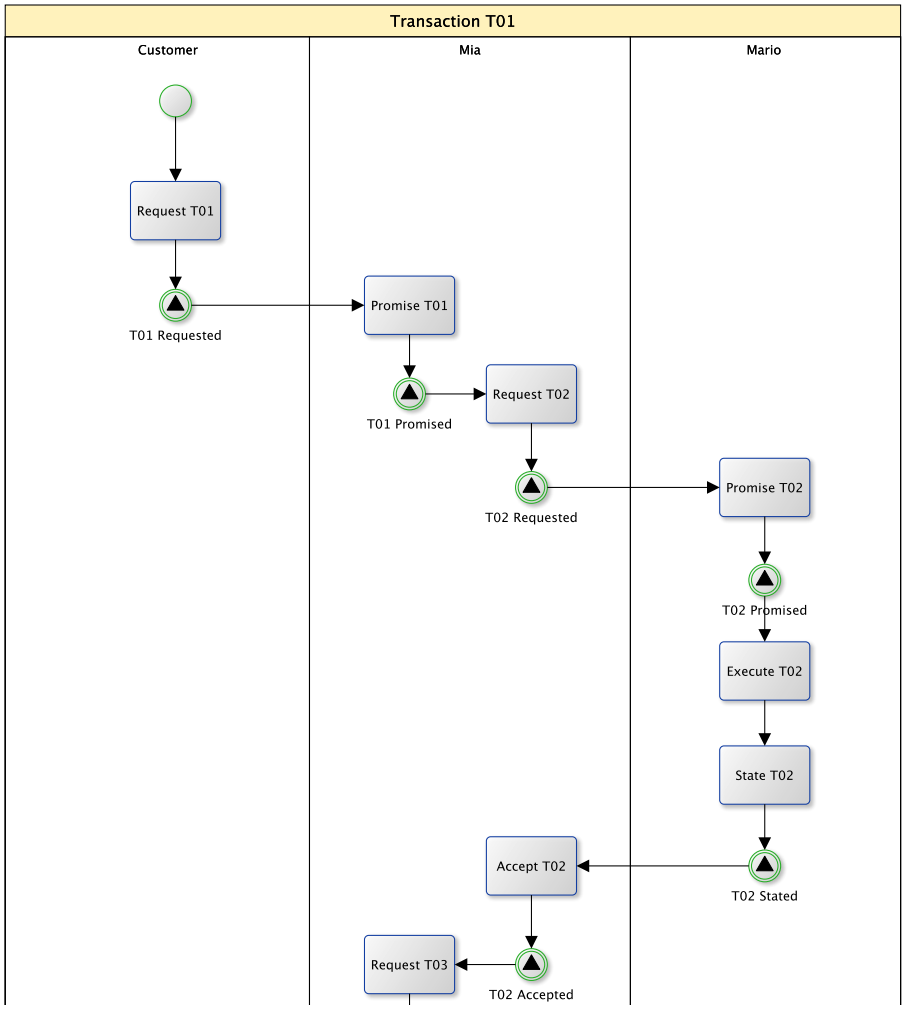
Obrázek 6.3: PSD Pizzerie Mama Mia [10]

6.2.7 Krok 7: Vytvoření BPMN modelu

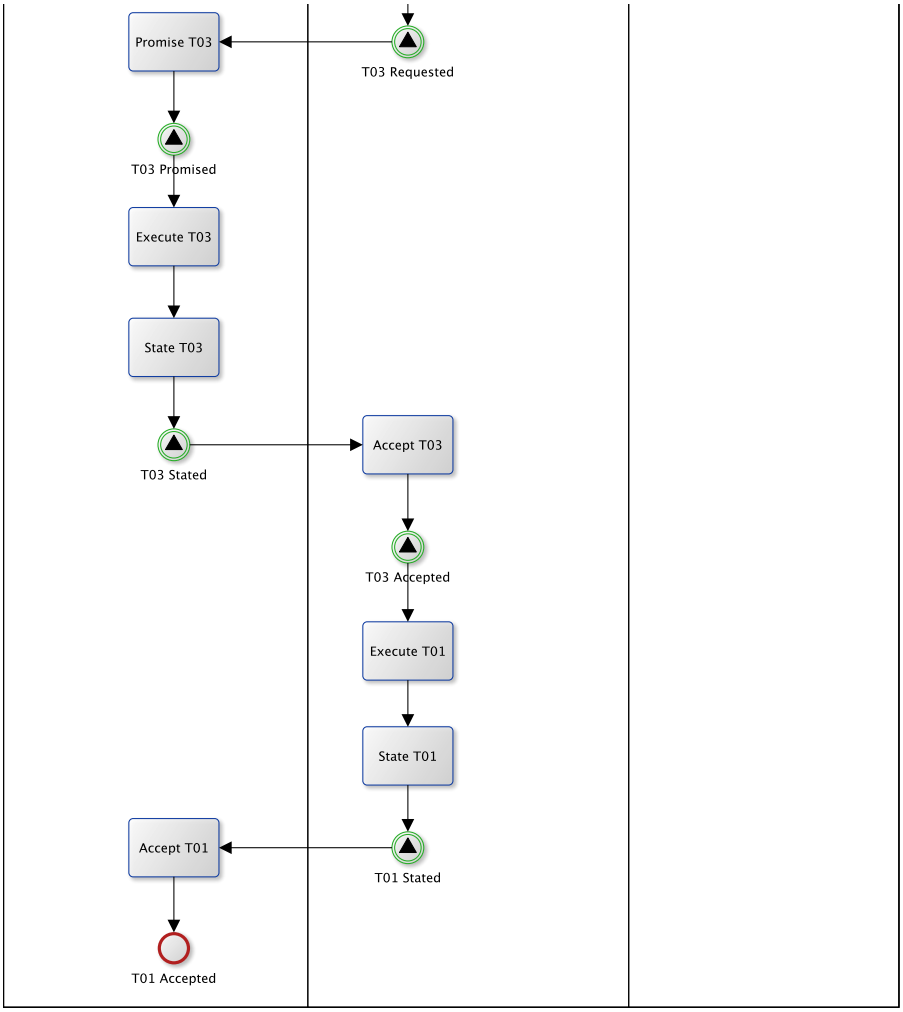
Nyní je vše připraveno pro vytvoření konečného BPMN modelu. Při jeho tvorbě budeme vycházet především z PSD modelu vytvořeného v předešlém kroku a z předpisu pro převedení transakčního axiomu do BPMN primitiv, který je uveden v sekci 5.4.2. Postupujeme tedy po jednotlivých transakčních krocích zobrazených v PSD v obrázku 6.3 a převádíme je dle tohoto předpisu do BPMN. Je důležité správně poskládat aktivity dle závislostí vyjádřených v kroku 5 i v PSD pomocí přerušovaných čar a dle diskuse v předchozím kroku.

Při vytváření PSD v předchozím kroku je použit pouze základní transakční vzor. Důvodem je větší čitelnost výsledného BPMN modelu a také absence popisu nešťastných scénářů v textovém popisu případu Pizzerie Mama Mia. Dle předpisu v sekci 5.4.2 by však nebyl problém vytvořit i BPMN model ze standardního transakčního vzoru.

Zobrazení případu Pizzerie Mama Mia v BPMN za použití metody navržené v této práci můžeme vidět na obrázku:



Obrázek 6.4: Případ Pizzerie Mama Mia v BPMN 1/2



Obrázek 6.5: Případ Pizzerie Mama Mia v BPMN 2/2

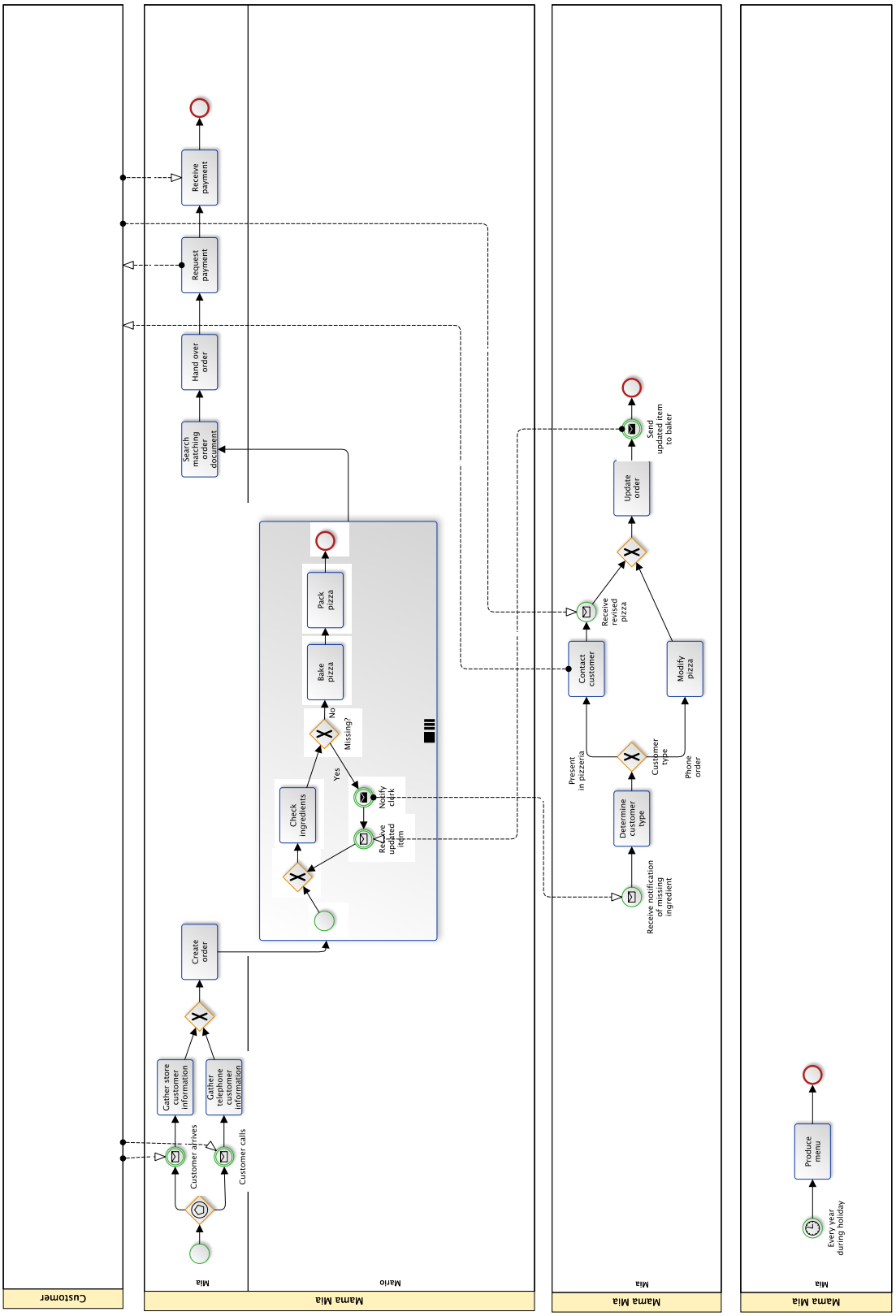
6.3 Diskuse

Na obrázcích 6.4 a 6.5 vidíme, že se nám podařilo vytvořit BPMN model pro případ Pizzerie Mama Mia dle metody navržené v kapitole 5. Vytvořený BPMN model je:

- *kompletní* dle transakčního axiomu – žádný transakční krok dle základního transakčního vzoru v modelu nechybí a díky použití plaveckých drah je jasné, který actor je zodpovědný za vykonání konkrétního transakčního kroku

- *konzistentní* dle transakčního axiomu – pořadí provádění všech transakčních kroků je konzistentní se základním transakčním vzorem
- *jednoznačný* – navržená metoda zajišťuje, že při správném aplikování všech kroků metody vznikne vždy ten samý model
- *esenciální* – výsledný model neobsahuje žádné implementační detaily. Při jeho tvorbě bylo použito pouze aktivit, které jsme vyhodnotili jako Performa.

Na obrázku 6.6 lze vidět případ Pizzerie Mama Mia v notaci BPMN, kterou můžeme najít v [34]. Je vidět, že od diagramu vzniklého aplikací metody se velmi liší, ačkoliv oba tyto diagramy modelují stejný případ a jsou dle specifikace BPMN korektní. V diagramu na obrázku 6.6 evidentně chybí velké množství kroků dle transakčního axiomu, například chybí *promise* objednávky klientovi nebo většina transakčních kroků u platby. Pokud by však neexistovaly DEMO modely, které jsme vytvořily v rámci této kapitoly, neměli bychom jak ověřit, zda je vzniklý model kompletní.



Obrázek 6.6: Případ Pizzerie Mama Mia dle [34]

6.3.1 Slabiny metody

6.3.1.1 Modelování komplexních procesů

Na obrázcích 6.4 a 6.5 jsou zachyceny pouze 3 transakce za použití základního transakčního vzoru a stejně jsme museli diagram rozdělit na 2 obrázky, protože se celý nevešel na jednu stránku. V případě modelování komplexnějších procesů, které obsahují desítky transakcí bude problém jen narůstat a model se bude obtížně vytvářet i číst.

Řešením je v tomto případě automatické generování BPMN diagramu z ATD diagramu (případně PSD dle DEMO 3) , který je mnohem méně obsáhlý a dobře čitelný i když obsahuje desítky transakcí. Pokud by bylo možné automatizovat vytváření BPMN a jejich verifikaci dle DEMO modelů, jednalo by se o velký krok kupředu.

6.3.1.2 Korektní provedení kroku 2

Z mé zkušenosti je pro velké množství lidí velmi problematické správně aplikovat krok 2 navržené metody, neboli správně provést na textovém popisu procesu Performa-Informa-Forma analýzu. Lidská řeč je totiž velmi vágní a rozdíly mezi Performa, Informa i Forma aktivitami jsou často nezřetelné a ani odborníkům se zkušenostmi s DEMO se často nedaří provést tento krok správně.

Na tomto místě je třeba uvést, že i situace, kdy se nepodaří všechny aktivity správně zařadit a jako Performa je například zařazena aktivita, která Performa není, stále pravděpodobně vznikne kvalitnější BPMN model než by vzniknul neaplikováním této metody. Jak popisuje [23] „špatné BPMN“ je dnes spíše pravidlem než výjimkou a aplikace navržené metody by tedy za každých okolností přispěla obecně k lepším výsledkům.

Závěr

Na začátku této práce jsme uvedli důvody, proč je vůbec vhodné se modelování podnikových procesů věnovat – jedná se o cestu k větší efektivitě a udržitelnému růstu každé organizace. K řízení podnikových procesů však existuje celá řada přístupů, které se mění v čase podle toho, jak se mění podniky i technologie. Výkon dnešních výpočetních prostředků je na takové úrovni, že umožňuje prakticky v reálném čase vyhodnocovat efektivitu provádění podnikového procesu a rychle zavádět úpravy v případě, že se podaří odhalit problém.

Kdo z nás, zákazníků velkých podniků jako jsou banky, telefonní operátoři nebo státní úřady, má pocit, že procesy v těchto organizacích jsou skutečně efektivní, a že při jejich provádění nedochází ke zbytečným prodlevám či plýtvání? Běžně používaným tvrzením v médiích či mezi politiky je to, že jen malé organizace dokáží být efektivní a ty velké jsou naopak „těžkopádné molochy“. Čím je to způsobeno? Ve všech takto velkých organizacích se jistě procesnímu řízení věnují, je tedy možné, že je něco špatně s metodami, podle kterých jsou podnikové procesy v dnešní době řízeny?

Ve druhé kapitole této práce najdeme srovnání nejběžnějších technik pro modelování podnikových procesů, jako jsou vývojové diagramy, UML nebo BPMN a také metodologie DEMO, která je sice v praxi využívána zatím jen málo, její potenciál je ale v mnohém revoluční. Tato metodologie totiž dokáže uvnitř operací různorodých organizací rozpoznat vzory a pomocí nich pak popsat chod celé organizace bez zahlcení detaily, které nejsou podstatné. Tento přístup umožňuje řídícím pracovníkům soustředit se jen na to důležité a přesto získat kompletní informace o vlastním podniku.

Faktem však je, že první verze DEMO vznikly již v 80. letech minulého století a do dnešních dnů se DEMO šíří jen pomalu. O důvodech můžeme jen spekulovat, přesto je dokážeme s určitou mírou jistoty odhadnout. DEMO je totiž poměrně složité na naučení, což stojí podniky čas a úsilí, které zatím nejsou ochotny vynaložit.

Oproti tomu notace BPMN těmito „nedostatky“ netrpí. O popularitě této notace, která vychází z vývojových diagramů, svědčí i fakt, že v dnešní době

existuje na trhu několik desítek nástrojů, které ji podporují a nové stále přibývají. BPMN je jednoduchá a přímočará notace, ve které může uživatel začít modelovat „během několika málo hodin“. Důsledkem však je, že vzniká obrovské množství modelů, které svojí kvalitou efektivní řízení procesů v organizaci často spíše komplikují než usnadňují.

Tato práce vznikla na popud Ing. Pavla Náplavy a Stevena Van Kervela z nizozemské firmy Formetis, kteří měli za to, že by kombinací těchto dvou technik mohl vzniknout nadějný nástroj pro zvýšení kvality řízení podnikových procesů. Tato práce se snaží udělat první malý krok tímto směrem v podobě průzkumu možností přenesení základních teoretických konceptů metodologie DEMO do notace BPMN. Zároveň přináší první verzi metody, která na teoretických základech z DEMO umožňuje vytvářet BPMN modely, které nebudou trpět nekompletností či nekonzistentností.

Přínosy práce

Přínosy této diplomové práce vidí autor ve čtyřech oblastech:

1. Srovnání nejběžnějších technik pro modelování podnikových procesů z pohledu jejich silných a slabých stránek zejména pro business uživatele
2. Průzkum přínosů spojení DEMO a BPMN
3. Předpis pro vyjádření základních konceptů DEMO pomocí primitiv notace BPMN
4. Návrh první verze metody pro vytváření BPMN modelů podnikových procesů za použití teoretických konceptů z metodologie DEMO

Zjištění

Z argumentace v kapitolách 2, 3, 4 a 5 je zřejmé, že pro spojení obou technik existuje velký prostor, neboť silné stránky jedné techniky doplňují slabé stránky druhé techniky a naopak. Jednoduchost, přímočarost, velké množství dostupných nástrojů a uživatelská základna BPMN na straně jedné a teoretické koncepty, které zajišťují ontologickou kompletnost, konzistentnost, jednoznačnost a esencialitu, na straně DEMO si přímo říkají o nalezení cesty, jak tyto vlastnosti skloubit.

V kapitole 5 je provedena analýza možností, jak skloubení dosáhnout, pomocí diskuse nad možnými přístupy, kterými je možné základní teoretické koncepty DEMO v BPMN vyjádřit. U každého navrženého přístupu je diskutována jednak míra korektnosti s jakou je pomocí takového přístupu možné dosáhnout požadovaných ontologických vlastností a jednak srozumitelnost a přívětivost použití takového přístupu pro business uživatele.

Podařilo se zjistit, jak dokládá text kapitoly 5, že je skutečně možné v BPMN vyjádřit požadované koncepty z DEMO s minimální ztrátou korektnosti. BPMN modely, které vznikly aplikací metody navržené v kapitole 5, jsou skutečně kompletní, konzistentní, jednoznačné a esenciální. Situací, které můžeme modelovat v BPMN, je však nepřeberné množství a vhodnost navržené metody tak musí být v takových situacích ověřena dalším výzkumem a nadále precizována.

Slabiny navržené metody se nacházejí jednak v modelování komplexních procesů, které obsahují větší množství transakcí. Výsledný model se v takovou chvíli stává poměrně nepřehledným, což v praxi komplikuje jeho následnou analýzu. S tímto problémem se však potýká celé BPMN a navržená metoda ho jen zmírňuje. Nezřídka můžeme vidět v organizacích BPMN modely, které jsou rozprostřené přes několik stran formátu A4. Navržená metoda však ve svém šestém kroku počítá s vytvořením DEMO Actor-Transaction Diagramu, který poskytuje pohled na organizaci na nejvyšší úrovni abstrakce. Tento diagram totiž potřebuje pro zobrazení jedné transakce pouze jeden symbol. Částečným řešením tohoto problému by tak mohlo být generování výsledného BPMN modelu přímo z ATD a PSD diagramů.

Další slabinou navržené metody je hned druhý krok, ve kterém je nutné rozlišit, které aktivity vyjádřené v textovém procesu jsou ontologické, infologické nebo datalogické. Jedná se o poměrně obtížný úkol, se kterým mají čas od času problémy i zkušení profesionálové. Řešení v tomto případě neexistuje, tuto dovednost lze získat pouze praxí.

Další výzkum

Závěry této práce poskytují obrovský prostor pro další výzkum. Vzhledem k tomu, že se jedná o první návrh metody, která kombinuje BPMN a DEMO, je třeba dalším výzkumem prověřit prakticky veškerá východiska, na kterých staví. Zejména pak vhodnost použití konkrétních BPMN primitiv pro vyjádření konceptů metodologie DEMO jako jsou agenda, C-act, C-fact, P-act, P-fact a další.

Velká příležitost pro další výzkum se rovněž skýtá v oblasti automatizace některých částí této metody a v oblasti simulace výsledného BPMN modelu. V tomto případě by bylo vhodné analyzovat možnosti DEMO softwarového enginu, který vyvíjí firma Formetis, a implementovat rozšíření, které by umožnilo jednak z DEMO modelů automaticky generovat BPMN modely s vlastnostmi, které DEMO zajišťuje a dále simulovat běh a agendu BPMN modelu. Veškeré možnosti pro další výzkum jsou popsány v sekci 5.6.

Literatura

[1] Peris-Ortiz, M.; Álvarez-García, J.: *Action-based quality management: Strategy and tools for continuous improvement.* 2014, ISBN 9783319064529, 1–18 s., doi:DOI10.1007/978-3-319-06453-6.

[2] Panagacos, T.: *The Ultimate Guide to Business Process Management: Everything you need to know and how to apply it to your organization.* CreateSpace Independent Publishing Platform, 2012, 186 s.

[3] ISO 9000:2005. Dostupné z: `https://www.iso.org/obp/ui/#iso:std:iso:9000:ed-3:v1:en`

[4] Weske, M.: *Business Process Management.* Springer, 2007, ISBN 9783540735212, 372 s.

[5] Řepa, V.: *Podnikové Procesy. Procesní řízení a modelování.* Grada Publishing, a.s., 2007, ISBN 978-80-247-2252-8, 288 s.

[6] Bandor, M.: Process and Procedure. 2007. Dostupné z: `http://www.sei.cmu.edu/library/assets/process-pro.pdf`

[7] Jedlitschka, A.; Salo, O.; Bomarius, F.: Process Management. *Journal of Software Maintenance and Evolution: Research and Practice*, , č. May, 2010: s. 143–149, ISSN 1532060X, doi:10.4135/9781446251720.n28.

[8] Náplava, P.: A7B16ISP Informační systémy a procesní řízení. 2015.

[9] Harmon, P.: *Business Process Change: A Business Process Management Guide for Managers and Process Professionals.* Morgan Kaufmann, 2014, 505 s.

[10] Dietz, J. L. G.: *Enterprise ontology: Theory and methodology.* 2006, ISBN 3540291695, 1–243 s., doi:10.1007/3-540-33149-2.

[11] Recker, J. C.; Rosemann, M.; Indulska, M.; aj.: Business process modeling: a comparative analysis. *Journal of the ...*, ročník 10, č. 4, 2009: s. 333–363, ISSN 15369323. Dostupné z: http://eprints.qut.edu.au/20105$\delimiter"026E30F$nhttp://eprints.qut.edu.au/20105/

[12] Chytil, J.; Lehocký, Z.: Vývojové diagramy - 1. díl. 2005. Dostupné z: http://programujte.com/clanek/2005080105-vyvojove-diagramy-1-dil/

[13] Polancic, G.: Managing business processes with BPMN – SWOT Analysis. 2014. Dostupné z: http://blog.goodelearning.com/bpmn/managing-business-processes-bpmn-swot/

[14] BPEL – jazyk pro automatizaci procesů. Dostupné z: http://www.trask.cz/publikace/zn-51-bpel-jazyk-pro-automatizaci-procesu/

[15] Eriksson, H.-E.; Penker, M. O. T.: Business Modeling with UML.

[16] Eriksson, H.-E.; Penker, M.: *Business Modeling With UML: Business Patterns at Work*. 2000, ISBN 0471295515, 12 s., doi:978-0471295518.

[17] UML 2 Activity Diagramming Guidelines.

[18] Enterprise Engineering and DEMO. Dostupné z: http://www.ee-institute.org/en/demo

[19] Vejražková, Z.: Design and Engineering Methodology for Organizations, 2012.

[20] Barjis, J.: Enterprise Modeling and Simulation Within Enterprise Engineering. *Journal of Enterprise Transformation*, ročník 1, č. 3, 2011: s. 185–207, ISSN 1948-8289, doi:10.1080/19488289.2011.601399. Dostupné z: http://www.tandfonline.com/doi/abs/10.1080/19488289.2011.601399

[21] Shishkov, B.; Dietz, J.: Deriving Use Cases from Business Processes - The advantages of DEMO. 2005: s. 249–257. Dostupné z: http://dx.doi.org/10.1007/1-4020-2673-0{_}29

[22] Vašíček, P.: Úvod do BPMN. 2008. Dostupné z: http://bpm-sme.blogspot.cz/2008/03/3-uvod-do-bpmn.html

[23] Silver, B.: *BPMN Method and Style, 2nd Edition, with BPMN Implementer's Guide: A structured approach for business process modeling and implementation using BPMN 2.0*. 2011, ISBN 978-0982368114, 286 s. Dostupné z: http://www.amazon.com/Method-Style-Edition-Implementers-Guide/dp/0982368119/ref=sr{_}1{_}2?ie=UTF8{&}qid=1393180856{&}sr=8-2{&}keywords=bpmn

[24] Silver, B.: BPMN: The Four Aspects of Process. 2013. Dostupné z: http://brsilver.com/bpmn-four-aspects-process/

[25] Omg, O. M. G.: Business Process Model and Notation (BPMN) Version 2.0. 2011. Dostupné z: http://www.omg.org/spec/BPMN/2.0/PDF

[26] Leymann, F.: BPEL vs BPMN 2.0: Should you care? 2009. Dostupné z: http://leymann.blogspot.cz/2009/12/bpel-vs-bpmn-20-should-you-care.html

[27] Silver, B.: Executable BPMN 2.0. 2011. Dostupné z: http://brsilver.com/executable-bpmn-2-0/

[28] Černý, O.: *BPEL*. Bakalářská práce, Vysoká škola ekonomická, 2010.

[29] Notace. Dostupné z: https://managementmania.com/cs/notace

[30] Rámce a metodiky. 2013. Dostupné z: https://managementmania.com/cs/ramce-a-metodiky

[31] Ochrana, F.: *Metodologie vědy: úvod do problému*. Praha: Karolinum, 2009, ISBN 978-80-246-1609-4.

[32] Vejražková, Z.: *Business Process Modelling and Simulation: DEMO, BORM and BPMN*. Diplomová práce, České vysoké učení technické, 2013.

[33] Dietz, J. L. G.: Enterprise Ontology. 2005. Dostupné z: http://www.iceis.org/Documents/Previous{_}Invited{_}Speakers/2005/ICEIS2005{_}Dietz.pdf

[34] Van Nuffel, D.; Mulder, H.; Van Kervel, S.: Enhancing the formal foundations of BPMN by enterprise ontology. *Lecture Notes in Business Information Processing*, ročník 34 LNBIP, 2009: s. 115–129, ISSN 18651348, doi:10.1007/978-3-642-01915-9{_}9.

[35] Caetano, A.; Assis, A.; Borbinha, J.; aj.: An Application of the ψ-Theory to the Analysis of Business Process Models. *Enterprise Information Systems of the Future*, 2012. Dostupné z: http://lhrgateway.nu.edu.pk/articles/MetricsforProcessModels.pdf

[36] Caetano, A.; Assis, A.; Tribolet, J.: Using business transactions to analyse the consistency of business process models. *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2011: s. 4277–4285, ISSN 15301605, doi:10.1109/HICSS.2012.626.

[37] Řepa, V.: Řízení procesů versus procesní řízení. 2008. Dostupné z: http://bpm-tema.blogspot.cz/2008/04/procesy.html

[38] Dumas, M.; Rosa, M. L.; Mendling, J.; aj.: *Fundamentals of Business Process Management*. Springer Berlin Heidelberg, 2013, ISBN 9783642331428. Dostupné z: http://fundamentals-of-bpm.org/wp-content/uploads/2013/03/errata{_}Dumas.pdf

[39] Rowley, M.: Why use BPMN for BPEL? 2009. Dostupné z: http://www.activevos.com/blog/soa/why-use-bpmn-for-bpel/2009/11/05/

[40] Hesse, M.: BPMN Tool Matrix. Dostupné z: https://bpmnmatrix.github.io

[41] Dietz, J. L. G.; Hoogervorst, J. a. P.: The principles of enterprise engineering. *Lecture Notes in Business Information Processing*, ročník 110 LNBIP, 2012: s. 15–30, ISSN 18651348, doi:10.1007/978-3-642-29903-2{_}2.

[42] Guerson, J.; Almeida, J. P. A.; Guizzardi, G.: Support for domain constraints in the validation of ontologically well-founded conceptual models. 2014, doi:10.1007/978-3-662-43745-2. Dostupné z: http://www.scopus.com/inward/record.url?eid=2-s2.0-84904541612{&}partnerID=40{&}md5=a86c99fbd92cf15a46172f722a66be06

[43] Dietz, J. L.: The atoms, molecules and fibers of organizations. *Data & Knowledge Engineering*, ročník 47, č. 3, 2003: s. 301–325, ISSN 0169023X, doi:10.1016/S0169-023X(03)00062-4. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S0169023X03000624

[44] zur Muehlen, M.; Recker, J.: How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. *Advanced Information Systems Engineering 20th International Conference, CAiSE 2008 Montpellier, France, June 16-20, 2008 Proceedings*, 2008: s. 465–479, doi:10.1007/978-3-540-69534-9{_}35. Dostupné z: http://link.springer.com/chapter/10.1007/978-3-540-69534-9{_}35

Příloha **A**

Seznam použitých zkratek

ABD Actor Bank Diagram

AM Action Model

ATD Actor-Transaction Diagram

BCT Bank Contents Table

BPMN Business Process Model and Notation

BPEL Business Process Execution Language

BPM Business Process Management

BPMI Business Process Management Initiative

BPML Business Process Management Language

BPMS Business Process Management Software

CMM Capability Maturity Model

DEMO Design & Engineering Methodology for Organizations

IAM Interaction Model

ISM Interstriction Model

IUT Information Use Table

JIT Just In Time

OCD Organization Construction Diagram

OFD Object Fact Diagram

OPL Object Property List

OMG Object Management Group

PSD Process Structure Diagram

SM State Model

TQM Total Quality Management

TRT Transaction Result Table

UML Unified Modeling Language

WSDL Web Services Description Language

Příloha B

Glosář

Black Box Model Jako „Black Box Model“ označujeme model systému, u kterého víme, že určitým způsobem přeměňuje vstupy na výstupy, ale nevíme, jak funguje uvnitř.

White Box Model Jako „White Box Model“ označujeme model systému, u kterého víme, že určitým způsobem přeměňuje vstupy na výstupy, ale také to, jak k tomu dochází. Víme tedy, jak takový systém funguje uvnitř.

WSDL – Web Service Description Language Jazyk pro popis webové služby z pohledu poskytovaných funkcionalit a možností, jak je používat z vnějšku.

XML – Extensible Markup Language Značkovací jazyk používaný většinou ke specifikaci struktury uvnitř různých typů dat pro účely dalšího zpracování.

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
  - `thesis.ps` ..... text práce ve formátu PS