Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Transformace BPMN modelů technologických procesů do modelů využívajících barvené Petriho sítě

Bc. David Le

Diplomová práce
2023

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. David Le**
Osobní číslo:	**I21279**
Studijní program:	**N0613A140007 Informační technologie**
Téma práce:	**Transformace BPMN modelů technologických procesů do modelů využívajících barvené Petriho sítě**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování

V úvodní části práce je nutné představit formalismus a zásady evoluce barvených Petriho sítí (Coloured Petri Nets) a dále formalismus standardu BPMN (Business Process Model and Notation).
Primárním cílem diplomové práce je návrh a ověření metody transformace (výchozích) modelů technologických procesů specifikovaných pomocí BPMN do (cílových) modelů využívajících formalismus CPN. Výchozí modely budou specifikovány v rámci vhodného softwarového nástroje (např. Camunda). Transformované cílové modely budou vytvářeny ve vhodném integrovaném vývojovém prostředí, které podporuje výpočty evoluce CPN (např. v nástroji CPNTools).
Navržené řešení bude ověřováno na vhodné případové studii využívající množinu odlišných technologických procesů soupeřících o obslužné zdroje.

Rozsah pracovní zprávy: **cca 60 stran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

JENSEN, Kurt a Lars Michael KRISTENSEN. Coloured Petri Nets: modelling and validation of concurrent systems. Dordrecht: Springer, c2009, xi, 384 s. ISBN 978-3-642-00283-0.
VON ROSING, Mark, Stephen WHITE, Fred CUMMINS a Henk DE MAN. Business Process Model and Notation—BPMN. *The Complete Business Process Handbook*. Elsevier, 2015, 2015, s. 433-457. ISBN 9780127999593. doi:10.1016/B978-0-12-799959-3.00021-5

Vedoucí diplomové práce: **prof. Ing. Antonín Kavička, Ph.D.**
Katedra softwarových technologií

Datum zadání diplomové práce: **8. listopadu 2022**
Termín odevzdání diplomové práce: **19. května 2023**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

prof. Ing. Antonín Kavička, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 30. listopadu 2022

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 18. 5. 2023

Bc. David Le v. r.

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat svému vedoucímu práce prof. Ing. Antonínu Kavičkovi, Ph.D za pomoc a rady při zpracování práce. Dále bych chtěl poděkovat Ing. Jaroslavu Lachovi za konzultace při zpracování praktické části práce. V neposlední řadě bych rád poděkoval rodičům, přátelům a blízkým za trpělivost a podporu během studia.

ANOTACE

Diplomová práce se zabývá problematikou transformace modelů technologických procesů specifikovaných pomocí formalismu BPMN do formalismu barvených Petriho sítí. Úvodní část práce se zaměřuje na obecný popis problematiky analýzy a modelování technologických a business procesů. Další kapitoly detailně popisují oba formalismy, jejich výhody a nevýhody a možné alternativy těchto formalismů. Praktická část se zaměřuje primárně na popis nově navržené a vytvořené metodiky *PetriBPMN* popisující postup automatizované transformace modelů. Dále je zde popsána vytvořená webová aplikace využívající tuto metodiku pro validaci a následnou transformaci jednoho či více vstupních modelů najednou. K implementaci webové aplikace byly převážně použity technologie .NET a JavaScript. V neposlední řadě je zde ověřena správnost metodiky na případové studii.

KLÍČOVÁ SLOVA

automatizovaná transformace, BPMN, barvené Petriho sítě, modelování procesů, technologické procesy, podnikové procesy

TITLE

Transformation of BPMN models of technological processes into models using coloured Petri nets

ANNOTATION

The thesis deals with the transformation of models of technological processes specified using the BPMN formalism into the formalism of coloured Petri nets. The introductory part of the thesis focuses on a general description of the analysis and modelling of technological and business processes. The following chapters describe in detail both formalisms, their advantages and disadvantages, and alternatives to these formalisms. The practical part focuses primarily on the description of the newly designed and created *PetriBPMN* methodology describing the process of automated model transformation. Furthermore, the created web application using this methodology for validation and subsequent transformation of one or more input models at once is described here. Implementation of the web application was done in .NET and JavaScript technologies. Finally, the correctness of the methodology for the case study is verified.

KEYWORDS

automatic transformation, BPMN, coloured Petri nets, process modelling, technological processes, business processes

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

ANSI	American National Standards Institute
ARIS	Architektura integrovaných informačních systémů
BPD	Business Process Diagram
BPM	Business Process Management
BPMN	Business Process Model and Notation
CPN	Coloured Petri net
CSV	Comma-separated values
ČSN	Česká technická norma
DMN	Decision Model and Notation
EPC	Event-driven Process Chain
EPML	EPC Markup Language
ISO	International Organization for Standardization
ITIL	Information Technology Infrastructure Library
SCOR	Supply chain operations reference
UML	Unified Modeling Language
XML	Extensible Markup Language
XSD	XML Schema Definition
XSLT	Extensible Stylesheet Language Transformations

ÚVOD

V dnešní době musí podniky a firmy fungovat v dynamickém a neustále se vyvíjejícím prostředí. Aby podniky zůstaly konkurenceschopné a dosáhly svých cílů musí stále zlepšovat a přizpůsobovat všechny své interní i externí procesy. K tomu existuje mnoho nástrojů a technik v podobě procesního řízení a modelování business procesů, které je velice cenným nástrojem pro úspěšné řízení celého podniku. Modelováním svých podnikových a technologických procesů a následným vytvářením jejich vizuálních reprezentací, mohou podniky lépe porozumět těmto procesům, hledat jejich nedostatky a následně pak najít možnosti pro jejich zlepšení.

Jedním z hlavních důvodů, proč je modelování podnikových procesů v dnešní době tak důležité, je rostoucí složitost podnikových procesů. S rozvojem technologií a globalizací se podniky potýkají se složitějšími procesy a náročnějšími zákazníky, kteří očekávají stále kvalitnější služby. Modelování podnikových procesů může podnikům pomoci rozdělit tyto složité procesy na menší, lépe zvládnutelné celky, což umožňuje efektivnější a účinnější řízení procesů.

Hlavním cílem celého modelování je ovšem snížení finančního zatížení procesů. Upravením procesů a vyřešením různých neefektivit je především ušetřen cenný čas, který jde ruku v ruce s náklady celé společnosti. Kromě toho mohou organizace optimalizací svých procesů snížit zdroje potřebné k dokončení procesu, včetně snížení počtu potřebných zaměstnanců a snížení množství potřebného materiálu, což může v konečném důsledku vést k další úspoře nákladů a zvýšení ziskovosti.

Simulace modelovaných podnikových procesů je dalším důležitým aspektem modelování podnikových procesů. Simulací procesu mohou organizace testovat různé scénáře a analyzovat dopad změn ještě před jejich implementací v reálném světě. To může organizacím pomoci identifikovat potenciální problémy a provést vylepšení procesu předtím, než investují čas a zdroje do jeho implementace. Simulace může také pomoci organizacím činit informovanější rozhodnutí, protože mohou vidět potenciální dopad různých možností na proces a organizaci jako celek.

V této práci je proto vysvětleno procesní řízení od samotné analýzy procesů až po možnosti simulace a vyhodnocení modelovaných procesů.

MOTIVACE A CÍLE PRÁCE

Cílem práce je vytvoření jednoduchého postupu pro modelování technologických procesů a jejich následné analýzy v podobě provádění simulačních experimentů a jejich vyhodnocování.

Pro modelování procesů existuje mnoho formalismů, kde některé jsou méně či více vhodné pro modelování podnikových procesů. Pro účely této práce byl zvolen standard BPMN, který je široce uznávaným standardizovaným formalismem pro modelování podnikových či technologických procesů. Standard BPMN obsahuje širokou paletu prvků, pomocí kterých je možný grafický popis složitých procesů. Proto je v této práci zavedena nová redukce standardu pod názvem BPMN-light, která je dostačující na popis jednoduchých procesů soupeřících o společně sdílené zdroje. Redukcí prvků je navíc zkrácena doba potřebná pro pochopení formalismu, jelikož neobsahuje všechny elementy a charakteristiky v původním standardu.

V praxi je také velice často zapotřebí modelované procesy před jejich implementací a použitím zkušebně simulovat a analyzovat za účelem zjištění jejich nedostatků a možných problémů, které by mohly nastat v průběhu realizace procesu. K tomu se dobře hodí formalismus barvených Petriho sítí. Ty umožňují nejen statickou analýzu procesu například ve formě analýzy stavového prostoru, ale také dynamickou analýzu formou vyhodnocení výsledků simulací.

Proto je v práci popsána nová metodika PetriBPMN pro automatizovanou transformaci výchozích modelů vytvořených ve formalismu BPMN-light do formalismu barvených Petriho sítí. Dále je zde popsána vytvořená webová aplikace, která umožní uživateli automatizovanou transformaci modelovaných procesů bez nutnosti manuálního modelování procesu v Petriho sítích. Uživatel tedy nepotřebuje mít hlubokou znalost tvorby modelu barvené Petriho sítě, ale pouze základní znalost použití a provádění simulačních experimentů a analýz.

1 PROCESNÍ MODELOVÁNÍ

1.1 Procesní řízení

Procesní řízení neboli *Business Process Management* (BPM) je disciplína, která kombinuje business praktiky společně s informačními technologiemi s cílem zlepšit fungování organizace. Zahrnuje identifikaci, modelování, analýzu, návrh, implementaci, monitorování a neustálé zlepšování podnikových (business) procesů za účelem dosažení požadovaných výsledků a cílů. Využívá k tomu systematický přístup popisu procesů včetně jejich vstupů, výstupů, činností, aktivit, zainteresovaných stran (stakeholderů) a přidružených technologií. BPM lze aplikovat na různé typy procesů, jako jsou provozní, manažerské, zákaznické, technické a podpůrné procesy. Je možné jej využít ve všech různých průmyslových odvětvích a sektorech. [1]

1.1.1 Historie a vývoj procesního řízení

Základním konceptem BPM z pohledu organizace a řízení práce je stanovení priorit procesů v organizaci. Ačkoli se tato myšlenka může zdát zřejmá a jasná, trvalo několik etap vývoje, než se stala základní součástí organizací. Pro pochopení důvodů, proč se organizace zabývají procesním řízením, je zapotřebí se podívat na historický vývoj lidí a lidské společnosti jako takové. [2]

V prehistorických dobách byl člověk při různých činnostech univerzální a všestranný. Dokázal si sám zhotovit nástroje a předměty nebo například sehnat potravu. S postupem času a postupným vznikem měst a městských států se objevila částečná specializace, která ve středověku vedla k vytvoření řemesel a dalších oborů. S nástupem industrializace a druhou průmyslovou revolucí se tato specializace prohlubovala. Organizace se pak začaly strukturovat na základě dělby práce, přičemž bylo potřeba, aby na funkční jednotky a skupiny dohlíželi různí manažeři a vedoucí v hierarchické struktuře. Ta je podobná dnešní struktuře v moderních organizacích, kde se vyskytují různá řídící oddělení, jako je oddělení pro nákupy, prodeje, účetnictví, marketing a mnoho dalších. [2]

S rychlým vývojem informačních technologií a zejména informačních systémů hrají tyto systémy velice důležitou roli v managementu a řízení procesů v organizacích. V organizacích existují některé procesy, které jsou prováděny pouze manuálně bez informačních systému. Ovšem velké množství procesů se v dnešní době bez podpory informačních technologií neobejde nebo jsou zcela nahrazeny automatizovanými procesy bez nutnosti zásahu člověka. Proto je velice důležité, aby v procesním řízení správně spolupracovaly informační technologie společně s lidskými a jinými zdroji. [1]

1.2 Proces

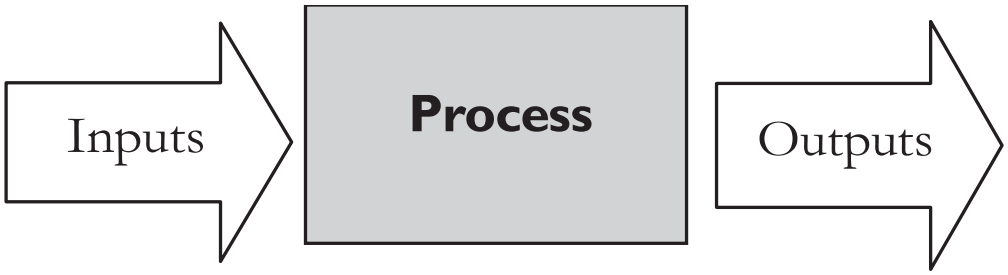
Procesy jsou základním konceptem v různých oblastech a lze je rozdělit do mnoha typů v závislosti na kontextu. Pod pojmem proces je možné si představit řadu různých druhů procesů jako jsou provozní procesy, obchodní procesy, technické procesy, biologické procesy, politické procesy, finanční procesy, počítačové procesy a tak dále. [3]

Pojem proces je možné chápat jako soubor činností či aktivit, které jsou určitým způsobem logicky uspořádány. Pro účely této práce stačí základní definice procesu:

„Proces je přístup, který se skládá z řady činností, které přijímají určité vstupy za účelem dosažení nějakého cíle (výstupu).“ [3]

Definice business procesu je velice podobná:

„Podnikový proces se skládá ze souboru činností, které jsou prováděny koordinovaným způsobem v organizačním a technickém prostředí. Tyto činnosti společně realizují podnikový cíl. Každý podnikový proces je realizován jednou organizací, ale může se vzájemně ovlivňovat s podnikovými procesy prováděnými jinými organizacemi.“ [1]



Obrázek 1: Transformační model procesu[1]

Obchodní cíl ovšem nemusí být ryze fyzický, ale může jím být například zvýšení zisků společnosti či snížení celkových nákladů. Obecně lze procesy tedy považovat za určitý způsob transformace vstupů na nějaké výstupy viz Obrázek 1. Podle druhu takové transformace se procesy dělí na několik základních druhů: [4]

- fyzické – procesy pro transformaci surových materiálů na výsledné produkty
- lokační – transportní procesy
- transakční – například proces transformace peněžní, hotovostní hodnoty na akcie
- informační – procesy pro interakci a nakládání s daty

[1] LAGUNA, Manuel a Johan MARKLUND. *Business Process Modeling, Simulation and Design*. 3rd edition. London: Chapman & Hall, 2018. ISBN 9780429673337.

Procesy je možno dělit i podle dalších charakteristik jako je úroveň automatizace procesů, opakovanost procesu, úroveň struktury a organizace procesu, zaměření procesu aj. [4] Nejběžnější rozdělení je podle strategické důležitosti procesu pro organizaci: [2]

- primární/klíčové procesy
- podpůrné procesy
- řídící/management procesy

Jako primární procesy je možné si představit ty procesy, které jsou základním pilířem organizace. Jedná se zejména o procesy, které pokrývají ty nejpodstatnější hodnototvorné činnosti či služby, za které zákazníci platí. Podpůrné procesy napomáhají plnění obchodního cíle klíčovým procesům. Může se například jednat o procesy infrastrukturního charakteru nebo o lidské zdroje. Jako poslední kategorii procesů autoři uvádí takzvané řídící procesy. Tyto procesy reprezentují aktivity manažerského typu a to takové, které nepřináší společnosti zisky, ale jsou zapotřebí pro správný chod jiných procesů. Příkladem může být proces průběžného hodnocení konkurenceschopnosti a sledování pozice na trhu oproti konkurenci. [2]

Správnou kategorizací procesů do jednotlivých skupin a jejich plánováním může společnost efektivně splňovat své obchodní cíle a minimalizovat tak své náklady. Proto je zapotřebí procesy analyzovat, dokumentovat a modelovat tak, aby mohly zodpovědné osoby správně rozhodovat o budoucím směru a vývoji společnosti. [2]

1.3 Identifikace procesů

Identifikace procesů je soubor činností, jejichž cílem je systematicky definovat soubor procesů podniku a stanovit jasná kritéria pro jejich prioritizaci. V podniku takových procesů existuje nepřeberné množství, ale jen málo organizací má potřebné zdroje a kapacitu k podrobnému popisu všech svých procesů. Investovaný čas a prostředky na identifikaci a následné modelování procesů je tedy proto potřeba dobře rozmyslet. Je tedy nezbytné zaměřit pozornost na určitou podmnožinu nejdůležitějších procesů. Zejména na ty, které se nacházejí v oblastech přinášejících největší zisky, nebo tam, kde jsou přítomny významné problémy (nebo obojí). Takové procesy je vhodné detailně modelovat a ostatní, nedůležité procesy zcela vynechat. [2]

Pro účely modelování existuje mnoho modelovacích přístupů, jazyků a metod, které jsou podrobněji popsány v následujících kapitolách. Všechny zmíněné modelovací přístupy jsou zaměřeny na modelování obecných procesů. Není zde rozlišováno mezi procesem podnikovým, technickým, technologickým atd. Všechny tyto pojmy jsou tedy v práci zaměnitelně používány.

1.4 Životní cyklus procesu

V procesním řízení prochází procesy svým životním cyklem. Cyklus se skládá ze vzájemně propojených fází, které jsou uspořádány cyklicky viz Obrázek 2. Uspořádání ale neurčuje nutně striktní chronologickou posloupnost provádění jednotlivých fází. Často se stává, že se během každé fáze provádějí různé úkoly návrhu a vývoje současně a běžně se některé činnosti překrývají ve více fázích. [1]

1.4.1 Analýza a návrh

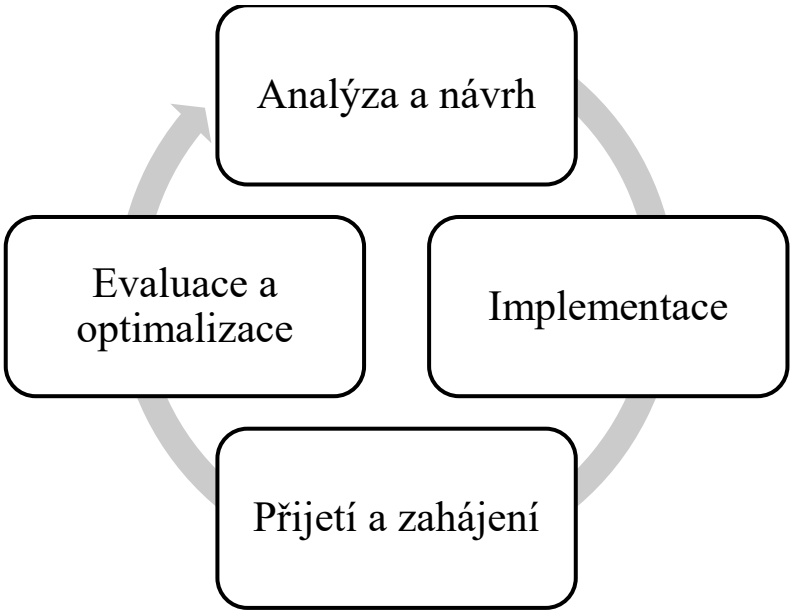
Cyklus začíná fází analýzy a návrhu procesů, kde prvním krokem je již zmíněná identifikace procesů. Ta může probíhat v podobě průzkumů a dotazníků. Poté jsou nalezené procesy analyzovány, přezkoumány a ověřeny. Ověřené procesy je potřeba navrhnout a namodelovat. Modelování podnikových procesů v podniku je hlavní technickou fází při návrhu procesů. Tím je neformální popis procesu převeden pomocí konkrétního modelovacího přístupu do formalizovaného výstupu pomocí standardizované notace. [1]

Jakmile je vytvořen prvotní návrh procesu, je zapotřebí jej ověřit se skutečností. Jednou z možností je uspořádání schůze, kde proběhne diskuse zúčastněných stran v procesu. Diskutující strany mohou vytvořený model zkontrolovat a ověřit jeho správnost, a tím se ujistit, že jsou tyto procesy správně realizovány. Další možností validace procesů je pomocí simulací. Simulační techniky mohou kromě základní validace odhalit potenciální problémy v průběhu procesu. Sledováním výsledků simulace lze tedy zjistit, zda se proces chová tak jak bylo zamýšleno. [1]

1.4.2 Implementace

Implementace navrženého procesu je možná různými způsoby. Procesy bez podpory specializovaného systému pro řízení podnikových procesů mohou být implementovány prostřednictvím souboru postupů a zásad, které budou zaměstnanci dodržovat. V opačném případě, kdy je k realizaci procesu použit systém v podobě informačního systému, je zapotřebí v konfigurační fázi zvolit implementační platformu. Ta je závislá na konkrétní organizaci a prostředí, ve kterém bude proces realizován. Konfigurace zahrnuje interakce zaměstnanců se systémem a také integraci stávajících softwarových systémů. [1]

Po úspěšné konfiguraci procesu a implementaci je opět ověřen a otestován. V závislosti na konkrétním procesu mohou být nakonec vyžadovány další činnosti jako je například školení personálu či migrace aplikačních dat do nového systému. [1]



Obrázek 2: Životní cyklus procesu[2]

1.4.3 Přijetí a zahájení procesu

V této fázi je implementovaný proces přijat a inicializován. Zahájení procesu obvykle následuje po definované události. Spuštěný proces je dále monitorován a kontrolován systémem pro řízení podnikových procesů. Kontroluje se zde správné provádění procesu v porovnání s namodelovaným procesem v předchozích fázích. Celý průběh procesu je systematicky monitorován a zaznamenáván. Tyto informace jsou cenné pro rychlou reakci na případné problémy a vytváření aktuálních stavových informací, které mohou být využity například na tvorbu takzvaných dashboardů. Stavové informace jsou poté logovány pro budoucí analýzu a zpracování. [1]

1.4.4 Evaluace a optimalizace procesů

Poslední fáze celého cyklu se zaměřuje na zhodnocení a budoucí plánování procesu. K tomu využívá dostupné protokolované informace z předchozí fáze, pomocí kterých vyhodnocuje kvalitu procesů. Kvalitu je možné měřit z několika hledisek. Hlavními hledisky pro společnost jsou například náklady a čas. Správnou interpretací výsledných měřítek je možné hledat v procesu místa pro budoucí zlepšení. Těmito místy mohou být úzká hrdla, kde je proces zpomalován z důvodu čekání na jiné externí procesy či zdroje. Optimalizací procesů je možné snížit celkové náklady potřebných na vykonávání procesu a zefektivnit celkový průběh procesu nejen z časového hlediska. [1], [5]

[2] vlastní zdroj

1.5 Modelovací přístupy

Modelování podnikových či jiných procesů je důležitou a nedílnou součástí v různých fázích procesního řízení. Před samotným začátkem modelování je zapotřebí si uvědomit, proč a za jakým účelem je takový proces modelován. Výsledný model se může markantně lišit v závislosti na důvodu takového modelování. Takových důvodů existuje mnoho. Nejdůležitějším z nich je beze sporu detailní pochopení samotného procesu. Procesy mohou ke své realizaci využívat mnoho zdrojů v podobě fyzických objektů (vybavení, materiály, produkty), aktérů (lidé nebo organizace) a nehmotných věcí (elektronické dokumenty a jiné materiály). Dokonce může operovat s dalšími procesy a podprocesy. Takový proces může být velice komplexní a obtížný na přesné pochopení průběhu a rozsahu. [2]

Dalším velice důležitým cílem procesního modelování je zjištění aktuálního stavu procesů, které jsou momentálně využívány. Tím je možné všechny procesy podrobně analyzovat a dokumentovat pro výměnu informací mezi všemi zapojenými účastníky v procesu. Navíc umožňuje taková analýza hledat možné nedostatky a úzká hrdla procesů. [2]

Popis libovolného procesu je možný i v podobě prostého textu. Nicméně takové texty mohou být těžkopádné a lze v nich snadno najít nepřesnosti a nejednoznačnosti, které mohou vést k nepřesnému popisu a následného nepochopení procesu čtenářem. To je hlavní důvod, proč vznikly v praxi vizuální diagramy pro popis procesů, které umožňují jednoduchým způsobem zápis a popis procesů. Použitím jednotné a standardizované notace se také minimalizuje riziko nedorozumění mezi autorem diagramu a cílovou osobou. Toto riziko se ale nedá použitím samotných diagramů zcela eliminovat. Proto bývají vizuální diagramy doplněny o textový popis, který popisuje vlastnosti a charakteristiky procesu, které nelze zachytit pouze grafickou reprezentací. [2]

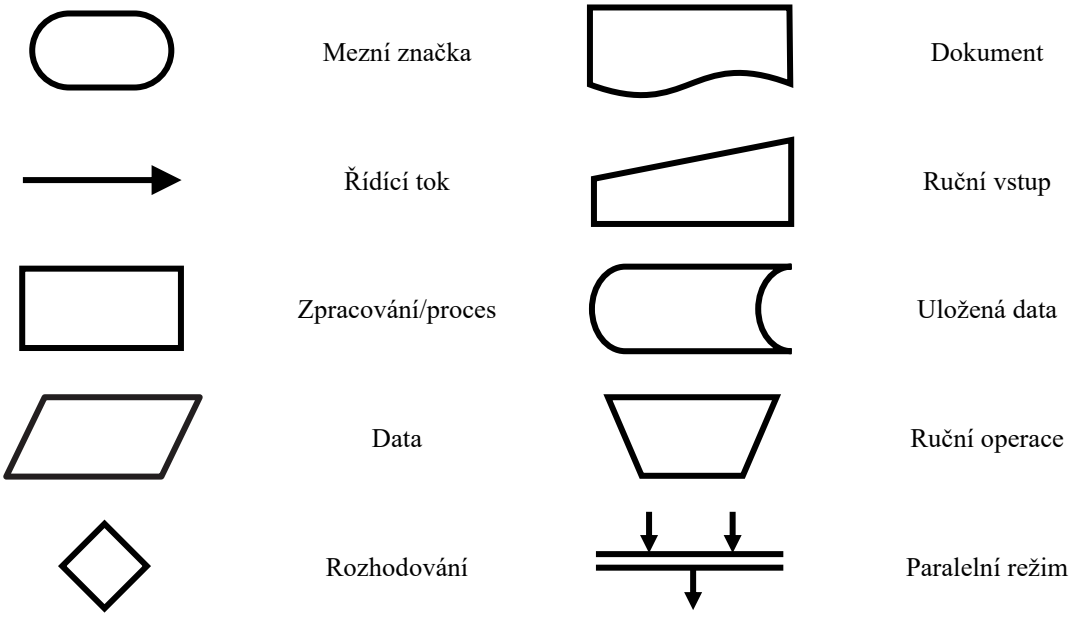
Existuje řada modelovacích technik a přístupů použitelných pro modelování procesů. Mezi ty nejznámější spadají:

- vývojový diagram (Flow chart)
- EPC (Event-driven Process Chain)
- UML (Unified Modeling Language)
- BPMN (Business Process Model and Notation)
- Petriho sítě (Petri net)

1.5.1 Vývojový diagram

Jak již bylo zmíněno, existuje mnoho způsobů, jak lze proces vyjádřit pomocí diagramu. Mezi ty nejstarší se bezpochyby zařazuje vývojový diagram. Historie použití vývojových diagramů sahá již do 20. let 20. století. Vývojové diagramy našly široké uplatnění v různých průmyslových odvětvích a bylo vyvinuto mnoho nástrojů a technologií, které vývojové diagramy využívají nebo z nich přímo vycházejí. [2], [3]

Vývojové diagramy mohou také být jednoduchou grafickou metodou pro znázornění toku programu, sekvencí a algoritmů pomocí standardní sady symbolů. Mohou být proto velice cenné při řešení programátorských úloh, které odráží problematiky z oblastí reálného světa jako je například oblast řízení, dynamiky tekutin, komunikace nebo výroby, které mohou být také vyjádřeny a modelovány pomocí vývojových diagramů. [6], [7]



Obrázek 3: Základní symboly vývojového diagramu[3]

Obrázek 3 zobrazuje pouze podmnožinu symbolů používaných pro modelování vývojových diagramů. Významný problém notace vývojových diagramů je v neexistenci všeobecně přijímaného standardu. Symboly se mohou v různých textech a formalizovaných normách, jako je například norma ISO, ČSN nebo ANSI, lišit, což vede k tomu, že existuje více "definitivních" verzí vývojových diagramů s drobnými odchylkami v jejich symbolech a zápisech. Symboly vývojového diagramu je možné použít ve vývojových diagramech toku dat, programu, systému, v síťových diagramech programu a také v diagramech zdrojů systému. [3], [8], [9]

[3] vlastní zdroj

Každý vývojový diagram by měl začínat a končit mezní značkou reprezentovanou obdélníkem se zakulacenými rohy reprezentující start a konec. Název mezní značka je zde na místě, jelikož začátek a konec procesu představuje přechod mezi procesem a vnějším prostředím (například jiným procesem). Další běžně užívaným symbolem je obdélník reprezentující určitý krok zpracování. Krok procesu může představovat provedení kódu v případě softwaru nebo provedení jakékoli jiné činnosti. Přechod z jednoho kroku na další umožňuje symbol orientované šipky. Spojením procesů a případných dalších symbolů vzniká cesta v diagramu. Tento procesní tok ovšem neindikuje žádný datový tok nýbrž tok řídící. V mnoha případech neexistuje přímočarý tok procesu, kde by existovala pouze jedna varianta. K takovému větvení existuje symbol rozhodování v podobě diamantu a symbol paralelního režimu v podobě dvou rovnoběžných čar. Při větvení programu vzniká více výstupních toků z jednoho vstupního. U větvení s následnou synchronizací je tomu přesně naopak – z více vstupů vychází pouze jeden výstup. [3], [9]

Na tvorbu vývojového diagramu existuje mnoho nástrojů od placených až po open source možnosti. Některé z nich jsou dostupné dokonce jako online aplikace v prohlížeči bez potřeby stažení a místní instalace programu. Těmito nástroji jsou například:

- Canva
- Lucidchart
- Smart Draw
- Draw.io
- Visual Paradigm
- Microsoft Visio

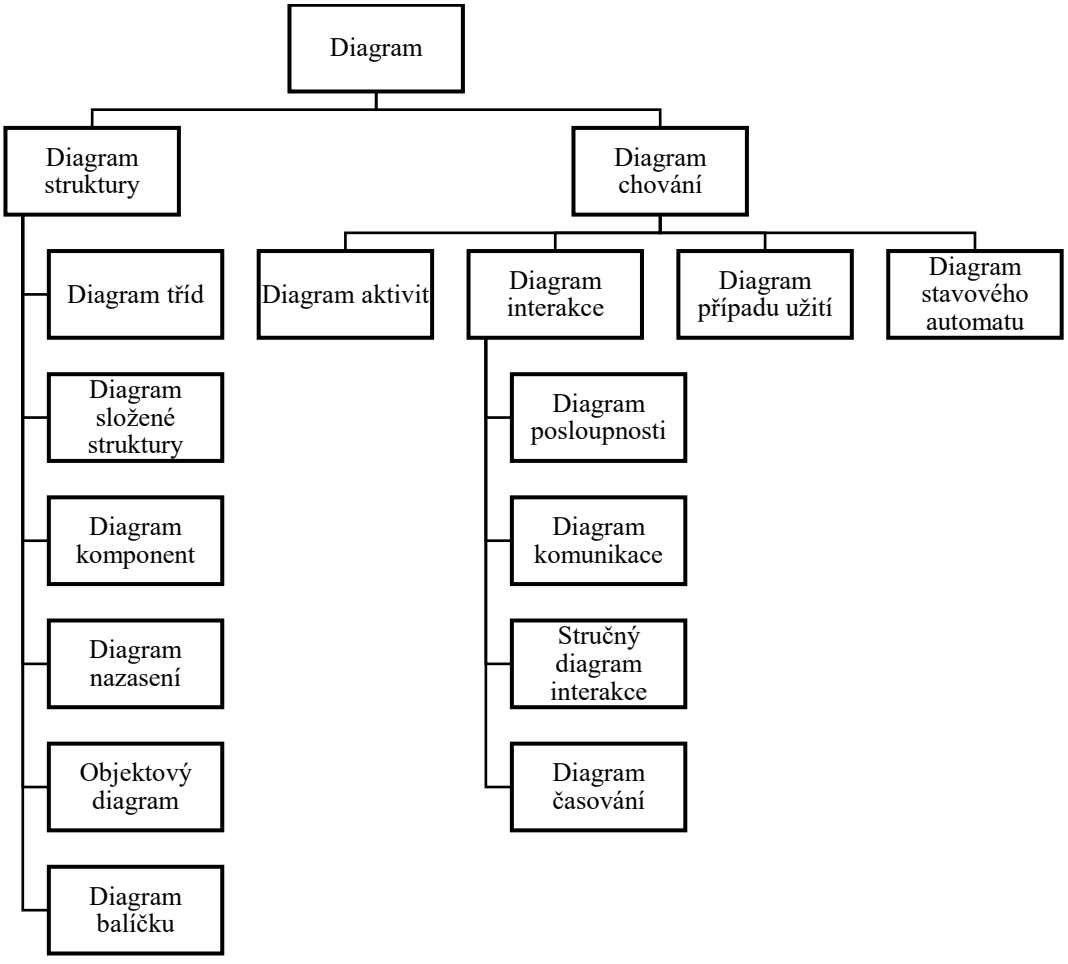
1.5.2 UML

Unified Modeling Language (UML) je univerzální vizuální modelovací jazyk používaný zejména v softwarovém inženýrství k popisu, specifikaci, návrhu a dokumentaci softwarových systémů. Byl vyvinut ve spolupráci tří autorů tehdejších nejpoužívanějších modelovacích jazyků a metodik. Autoři Grady Booch, James Rumbaugh, a Ivar Jacobson měli většinový podíl na trhu modelovacích technik a metod. To ovšem znamenalo přílišnou rozmanitost a nejednotnost, což způsobovalo značné potíže a nedostatky z pohledu kvality a jednolitosti jazyků. Každý jazyk měl své zastánce i odpůrce a nebylo tedy možné využít pouze jediný způsob. Prvním pokusem o sjednocení byla metodika Fusion vytvořená v roce 1994, která ovšem z důvodu absence již zmíněných autorů při tvorbě metodiky nebyla veřejností kladně

přijata. Definitivní konec metodiky Fusion nastal vytvořením prvních specifikací jazyka UML ve firmě Rational Corporation, kde působili Booch i Rumbaugh. [3], [10]

V roce 1997 byl jazyk UML přijat sdružením OMG (*Object Management Group*) jako průmyslový standard objektově orientovaného jazyka, který byl kolektivně přijat. Ostatní dosud existující metody upadly v zapomnění a jazyk UML se stal standardem v mnoha různých odvětvích. [10]

Na první pohled se může zdát, že UML je složitý a moc rozsáhlý na pochopení, zejména pro ty, kteří nemají zkušenosti s podobnými technologiemi nebo modelováním. Nicméně podobně jako u jiných jazyků platí, že ne všechny části UML se využívají současně a pro konkrétní aplikace jsou vhodné specifické podmnožiny UML. Jazyk UML od verze 2.0 obsahuje celkem 13 diagramů, které pokrývají různé pohledy na model viz Obrázek 4. [3], [10]



Obrázek 4: Diagramy v UML[4]

[4] upraveno z: ARLOW, Jim, Ila NEUSTADT a Bogdan KISZKA. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2008. ISBN 978-80-251-1503-9

Pro komplexní analýzu a modelování procesů je možné využít všech 13 pohledů. To by ovšem znamenalo velké časové investice, a proto se v praxi používají zejména diagramy tříd, případů užití, sekvenční diagramy a diagramy aktivit. [3]

Některé koncepty a pojmy procesů ovšem jazyk UML neobsahuje. Pro účely procesního modelování vzniklo rozšíření jazyka UML v podobě Eriksson-Penkerovi notace. Tato rozšíření mohou vyjadřovat podnikový proces při zachování soudržnosti se softwarovým modelem. Rozšíření k tomu používá čtyři pohledy: [11]

- strategický pohled (vize organizace)
- procesní pohled
- strukturní pohled
- pohled na chování organizace

1.5.3 EPC

V 90. letech 20. století si EPC (*Event-driven Process Chain*) získal popularitu jako konceptuální jazyk pro modelování podnikových procesů, které zahrnovalo dokumentování obecných podnikových operací, jako jsou například procesy obchodních a zákaznických služeb. Původně byly EPC diagramy vytvořeny pro návrh referenčního procesního modelu systému SAP R/3. Široké uznání si však získaly, když se staly hlavním modelovacím jazykem v sadě nástrojů ARIS, a následně je začali používat i další dodavatelé pro návrh referenčních modelů, které nejsou závislé na SAP, například ITIL[5] a SCOR[6]. [2], [12]

Za účelem interoperability a usnadnění vzájemné výměny EPC modelů vytvořených v různých softwarových nástrojích, byl vytvořen nástrojově neutrální souborový formát EPML, který je založen na formátu XML. EPML umožňuje kromě již zmíněné interoperability také vytvoření XSLT skriptů, které zajistí jednoduchý převod mezi EPML a jinými formáty založenými na XML. [13]

EPC notace nabízí čtyři hlavní typy elementů pro modelování procesů: [12]

- událostní
- funkční
- spojovací
- rozhraní procesu

[5] Soubor konceptů a postupů pro řízení služeb využívající informační technologie.
[6] Referenční model pro řízení dodavatelského řetězce v rámci společnosti.

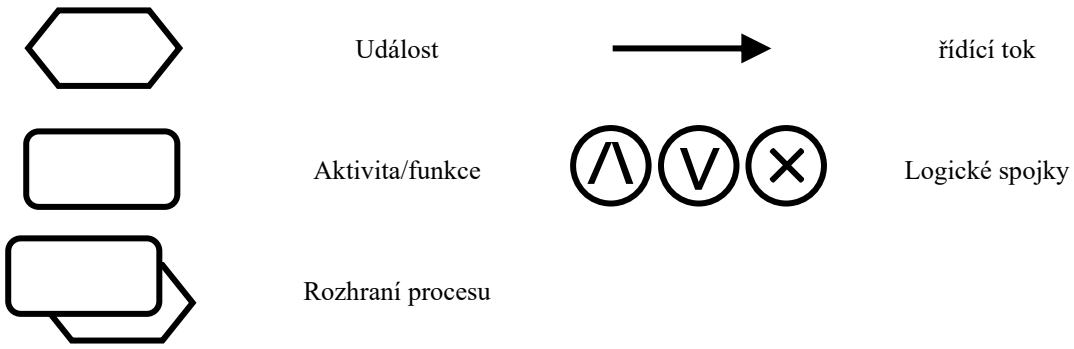
Funkční elementy jsou základním stavebním kamenem diagramu. Představují konkrétní aktivitu, činnost nebo úkol v procesu. Aktivity jsou znázorněny obdélníky se zaoblenými rohy. Stejně jako u vývojového diagramu obsahuje v sobě textový popis uvádějící název aktivity či činnosti, kterou reprezentuje. Aktivita se navíc může podílet na rozhodování v procesu. [12]

Dalším základním elementem diagramu jsou události znázorněné pomocí symbolu roztaženého šestiúhelníku. Událostní elementy popisují situaci a podmínky před a po vykonání aktivity. Jedná se tedy o pasivní prvky diagramu a znamenají určitý aktuální stav procesu. [12]

Funkční a událostní elementy jsou vzájemně spojeny spojovacími prvky mezi které se řadí logické spojky a šipky znázorňující řídící toky mezi prvky. Logickou spojkou může být spojka *„AND“* (a současně) se symbolem „∧“, spojka *„OR“* (logické nebo) se symbolem „∨“ a spojka *„XOR“* (vzájemné vylučování) se symbolem křížku. Spojky jsou zde z důvodu modelování komplexních směrovacích pravidel a synchronizačních úloh. [12]

Posledním typem prvku je prvek rozhraní procesu. Ten představuje hranici mezi jednotlivými procesy – tedy vstup či výstup z jednoho procesu do jiného, a tedy jeho konec či začátek. Konec a začátek procesu je možné modelovat také pomocí událostí, které celý proces spouští nebo ukončují. [12]

Obrázek 5 zobrazuje grafické reprezentace zmíněných prvků.



Obrázek 5: Základní symboly EPC diagramu[7]

Důležitou vlastností EPC diagramů jsou podmínky návazností jednotlivých prvků. Je zde pravidlem, že spojení událost – aktivita se musí střídat, a to buď přímo, nebo nepřímo. To například znamená, že aktivita nemůže vykonáním spustit vykonání jiné aktivity a vyvolaná událost nemůže vyvolat událost následující. Obecně je v EPC diagramu je zakázáno spojení mezi uzly stejného typu (například aktivita – aktivita nebo událost – událost). [12]

[7] vlastní zdroj

Další velice důležitou syntaktickou podmínkou je nemožnost použití spojek „OR“ a „XOR“ pro rozpojení toku procesu po události. Toto pravidlo vychází z logiky událostí, které se nemohou na rozdíl od aktivit podílet na rozhodování. Nebylo by tedy jasné, kterou z následujících větví se má průběh procesu vydat. [12]

Někteří autoři uvádí také další dodatečná pravidla pro tvorbu EPC diagramů. Některé formalismy například definují kardinalitu předchůdců a následovníků jednotlivých elementů, povolení smyček v diagramu, hierarchický pohled na sub procesy aj. Tyto definice se ovšem odlišují, a je potřeba je tedy brát s určitou rezervou. [12]

1.5.4 BPMN

Business Process Model and Notation (BPMN) je široce přijatý standard pro modelování, specifikaci a vizualizaci podnikových procesů. Poskytuje grafický zápis, který umožňuje podnikovým analytikům modelovat, analyzovat a optimalizovat podnikové procesy jasným a srozumitelným způsobem. Tento grafický zápis nazývaný *Business Process Diagram* (BPD) vychází z tradičních technik vývojových diagramů obdobně jako jazyk EPC zmíněný v předchozí kapitole. [14]

Oproti vývojovým diagramům či EPC diagramům, je standard BPMN záležitostí spíše 21. století. Vznikl spojením dvou tehdejších hlavních myšlenkových proudů popisu procesů. První z těchto proudů se zaměřoval na plánování pracovních postupů, zatímco druhý se věnoval modelováním a architekturou procesů. První verzí dostupnou pro veřejnost byla verze 1.0 uvolněná v roce 2004, kdy členové institutu BPMI[8] položili základy na vytvoření jednotného standardu. Dnes již dobře známá verze 2.0 byla vydána v roce 2011 pod záštitou sdružení OMG, které si převzalo správu dalšího vývoje standardu. [3], [14], [15]

BPMN si bere za cíl zjednodušit modelování podnikových procesů zejména pro méně technicky zaměřené uživatele v podniku tím, že poskytuje notaci, která je pro běžného uživatele intuitivní, ale zároveň dokáže reprezentovat složitou sémantiku procesů. To umožňuje použití jednotného způsobu pro výměnu informací mezi všemi stranami v podniku jako jsou například business analytici, kteří tyto procesy analyzují a modelují, techničtí vývojáři implementující takto modelované procesy a obchodní manažeři, kteří tyto procesy spravují a monitorují. [14]

Detailní popis jednotlivých elementů diagramu a jejich význam je popsán v kapitole Standard BPMN.

[8] Business Process Management Institute

1.5.5 Petriho sítě

Petriho sítě jsou typem matematického modelu, který zavedl Carl Adam Petri v 60. letech 20. století. Jsou velice podobné konečným stavovým automatům pro popis sekvenčních systémů. Petriho sítě se používají zejména k reprezentaci a analýze rozsáhlých systémů obsahující velký počet souběžných aktivit. Souběžnost je v reálném fyzikálním světě velice častým jevem v mnoha oblastech a odvětvích. V podniku mohou různé operace a činnosti probíhat současně, což může způsobovat značné potíže při plánování a organizaci těchto úkonů. Návrh takového systému není vůbec jednoduchou záležitostí, protože na jeho chodu se podílí řada interaktivních, a dokonce v reálném čase současně probíhajících akcí, takže jeho logické struktury a dynamické chování jsou velmi složité. Proto potřebujeme vhodný formalismus pro popis těchto proměnlivých systémů jako tomu jsou již zmíněné Petriho sítě, které díky své schopnosti přesně modelovat souběžnosti a interakce mezi jednotlivými procesy. [16], [17]

Petriho sítě mají, i přes svůj přesný matematický základ, věrnou grafickou reprezentaci, která umožňuje grafické modelování dynamiky systému. Pro celkový pohled na proces, podnikový proces a podnik je ale jednodušší využít uživatelsky přívětivější notace jako je například zmíněný standard BPMN. [7], [16]

Jednotlivé prvky a principy grafu, druhy grafu a typy Petriho sítí jsou detailně popsány v samostatné kapitole.

2 STANDARD BPMN

2.1 Klasifikace elementů BPMN diagramu

Jak již bylo řečeno v minulé kapitole, primárním cílem standardu BPMN bylo vytvoření jednoduchého a srozumitelného formalismu pro popis a vytváření modelů podnikových procesů. Na druhou stranu bylo ale zapotřebí vytvořit formalismus, který by byl schopný popsat komplexní problémy, které jsou nedílnou součástí většiny podnikových procesů. K tomu by bylo zapotřebí velkého množství různých elementů, což by bylo protichůdné s prvním cílem. Proto vzniklo menší množství základních elementů, které jsou rozděleny do specifických kategorií. Tím, že nabízí omezený počet kategorií prvků, BPMN zajišťuje, že jednotlivci, kteří si prohlížejí vytvořený diagram, mohou rychle identifikovat základní typy prvků a snadno pochopit význam diagramu. Těmito kategoriemi jsou[9]: [14], [18]

- tokové objekty
- data
- spojovací objekty
- plavecké dráhy
- artefakty

Specifikace BPMN od verze 2.0 definuje celkem 116 elementů, což je více jak dvojnásobný počet oproti původním 55 elementům. Z toho důvodu jsou dále v této práci zmíněny pouze nejdůležitější a běžně používané elementy. Ostatní elementy a upřesňující elementy, které jsou zde opominuty je možné najít v oficiální specifikaci zmíněné v předchozí kapitole. [14]

2.2 Tokové objekty

První a zároveň nejdůležitější kategorií prvků jsou tokové objekty. Tyto objekty jsou hlavním stavebním kamenem diagramu a definují chování procesu. Tokové objekty dále členíme na tyto tři podkategorie: [18]

- události
- aktivity
- brány

[9] Některé prameny (například [1] a [3]) uvádí pouze čtyři kategorie, kde spojují data a artefakty v jedinou společnou kategorii. Oficiální specifikace od OMG ovšem uvádí již zmíněných pět kategorií.

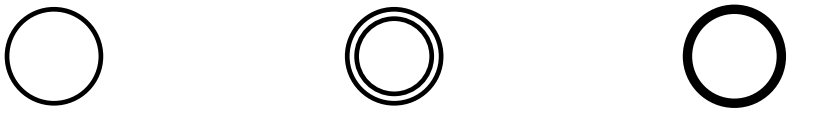
2.2.1 Události

Události hrají důležitou roli v podnikových procesech, jelikož reprezentují spojení mezi situacemi a procesy, které na tyto situace reagují. Existují události dvojího charakteru. Prvním z nich jsou události s určitou kauzalitou – *trigger events*, která způsobí například start procesu. Nazývají se také jako *catching events*, jelikož ve svém smyslu „chytají“, čekají na příchod spouštěče, který tuto událost spustí. Druhým typem jsou události s určitým dopadem/výsledkem – *result events*, které představují výsledek nějaké činnosti. Tento typ událostí je naopak nazýván jako *throwing events*, protože „vyhazují“ určitý výsledek. [1], [3]

Podle těchto charakteristik se události člení na tři typy:

- počáteční
- koncové
- průběžné

Všechny druhy událostí jsou reprezentovány symbolem kruhu viz Obrázek 6. Pro rozlišení jednotlivých typů událostí, mají tyto kruhy různý obrys a případnou upřesňující ikonu/značku. V této práci jsou použity a zohledněny pouze jednoduché a prázdné události – *none events*. Jedná se o události bez specifikované ikony, a tím pádem bez upřesňujícího spouštěče a/nebo výsledku. [18]



Obrázek 6: Symboly událostí v BPMN[10]

Prvním typem události je počáteční událost reprezentovaná obyčejným prázdným kruhem. Účelem počátečních událostí je reprezentace začátku (startu) procesu či choreografie (je vysvětlena dále v textu). Jelikož charakterizuje pouze počáteční impuls, nemohou počáteční události obsahovat vstupní toky ale pouze výstupní. [18]

Dalším typem události jsou události koncové. Ty jsou přesným opakem počátečních událostí. Reprezentují konec/výsledek procesu či choreografie a mohou mít pouze vstupní toky. Označují se symbolem prázdného kruhu obdobně jako počáteční události ovšem s tlustou obrysovou čarou. [18]

[10] upraveno z: BPMN coverage. *Camunda Platform 8 Docs* [online]. Berlin: Camunda Services, c2023 [cit. 2023-04-29]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/bpmn-coverage/

Posledním typem jsou události průběžné, které obsahují charakteristiky obou předchozích typů. Mohou mít vstupní i výstupní toky a nacházejí se v diagramu mezi počáteční a koncovou událostí (startem a koncem procesu). Nepodílejí se tedy na iniciaci nebo ukončování procesu samotného, ale mají vliv na průběh procesu. Mohou se podílet například na posílání zpráv v procesu či mezi procesy, zobrazení časové složky v procesu či narušení běžného toku procesu zpracováváním výjimek. Průběžná událost používá symbol dvou vnořených kruhů s tenkou obrysovou čarou jako je tomu u počáteční události. [18]

2.2.2 Aktivity

Aktivity hrají klíčovou roli při reprezentaci úkolů a činností, které je třeba provést v rámci procesu. Pomocí různých typů aktivit mohou BPMN diagramy poskytnout podrobný a komplexní přehled o procesu. [1]

Aktivita může být buď atomická nebo složená z více aktivit a procesů. Atomická aktivita je nazývána jako *Task*. Označují se symbolem obdélníku se zakulacenými rohy viz Obrázek 7. *Sub-Process* je naopak aktivita složená z více aktivit. Může tedy obsahovat nejen elementy typu *Task*, ale i další podprocesy. Podproces se označuje stejným symbolem jako *Task*, ale obsahuje navíc ikonu čtverce se znaménkem plus na spodní straně. Procesy sami o sobě nemají grafickou reprezentaci. Jsou to pouze sety grafických objektů (nemusí se jednat pouze o aktivity). [18]

Obrázek 7: Symboly aktivit v BPMN[11]

Aktivity mohou obsahovat značky a ikony – *Markers* dvojího typu. Prvním typem jsou značky určující chování aktivity. Tyto značky/symboly se zapisují na spodní straně aktivity. Může se například jednat o symbol točící se šipky určující cyklické chování aktivity v procesu. Druhým typem značek jsou značky určující typ aktivity jako je například servisní aktivita nebo manuální aktivita. Tato ikona se nachází v levém horním rohu uvnitř symbolu atomické aktivity. [18]

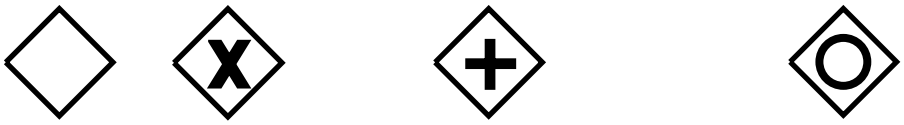
Aktivita může mít více vstupních a výstupních toků. Pokud aktivita neobsahuje vstupní tok, musí být její instance vytvořena s instancí procesu. Pokud nemá výstupní hrany, je tato aktivita zároveň implicitním koncem procesu bez nutnosti explicitní definice koncové události. [18]

[11] upraveno z: BPMN coverage. *Camunda Platform 8 Docs* [online]. Berlin: Camunda Services, c2023 [cit. 2023-04-29]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/bpmn-coverage/

2.2.3 Brány

Brány zajišťují řízení sekvenčních toků v procesu. Pokud proces neobsahuje větvení, nejsou brány potřeba. Ovšem taková situace není častá a v mnoha procesech je zapotřebí mít místa pro větvení a rozhodování. Termín brána (*gateway*) implikuje její samotný význam v diagramu. Brány umožňují nebo zakazují průchod tokenů branou pomocí definovaných pravidel a jejich vyhodnocování. [18]

Pro řízení různých situací a variant, existují různé brány, které kontrolují různé typy rozhodování. Všechny brány jsou reprezentovány stejným symbolem diamantu podobně jako tomu je pro symbol rozhodování u vývojových diagramů. Pro rozlišení jednotlivých druhů bran jsou doplněny značkou uvnitř diamantu viz Obrázek 8. [15]



Exkluzivní Paralelní Inkluzivní

Obrázek 8: Symboly bran v BPMN[12]

Každá z bran může být dvojího typu – konvergující a divergující. Brána má spojovací (*joining)* charakteristiku, pokud má dva a více vstupních toků. Naopak rozdělovací (*forking*), pokud má dva či více toků výstupních. Jedna brána může mít obě tyto charakteristiky najednou. [18]

Prvním běžně užívaným typem brány je exkluzivní brána – *XOR Gateway*. Ta umožňuje vytvoření několika alternativních cest toku procesu. U rozdělovací exkluzivní brány může pro danou instanci procesu nastat pouze jedna z vycházejících cest. To umožňuje exkluzivitu při výběru následné cesty. Spojovací exkluzivní brána pak umožňuje opětovné sjednocení bez synchronizace. Exkluzivní brána může být symbolizována prázdným diamantem nebo nepovinně doplněna o značku **X**. [18]

Druhým typem brány je paralelní brána – *AND Gateway* určená pro modelování paralelismů. Ta se naopak od exkluzivní brány používá pro spuštění všech toků vycházejících z brány bez nutnosti kontroly podmínek. Spojovací paralelní brána, na rozdíl od exkluzivní brány, zaručuje synchronizaci příchozích toků. Pouze po aktivaci všech vstupních hran je možné aktivovat hrany vycházející. Symbolizuje se opět diamantem ale navíc se znakem plus – +. [18]

[12] upraveno z: BPMN coverage. *Camunda Platform 8 Docs* [online]. Berlin: Camunda Services, c2023 [cit. 2023-04-29]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/bpmn-coverage/

V neposlední řadě je zde v praxi velice často používaná inkluzivní brána – *OR Gateway*. Ta se odlišuje od ostatních bran symbolem kruhu uvnitř diamantu. Její význam je na pomezí mezi bránou exkluzivní a paralelní. Oproti exkluzivní bráně může být aktivováno více výstupních toků. Brána tedy ověřuje všechny podmínky výstupních toků na rozdíl od exkluzivní brány, která ukončuje evaluaci ostatních hran po nalezení prvního validního toku. Vycházející toky na spojovací inkluzivní bráně jsou navíc synchronizovány. Brána tedy čeká na příchod všech aktivovaných toků z předchozí inkluzivní brány a pouze poté generuje token u vycházejících toků. [18]

2.3 Datové objekty

Velice často je v procesech zapotřebí mít k dispozici způsob pro modelování a reprezentaci dat, které jsou v procesu vytvářeny, používány či jinak zpracovávány. Může se jednat nejen o informační data, ale také o fyzické objekty jako jsou například tištěné dokumenty nebo produkty. [18]

Obrázek 9: Symboly pro reprezentaci dat v BPMN[13]

Datové objekty – *Data Objects* jsou reprezentovány symbolem obdélníku s přehnutým pravým horním rohem viz Obrázek 9. Tento symbol ale pouze reprezentuje referenci na samotný datový objekt – *Data Object Reference*. Objekty totiž mohou během procesu měnit svůj stav. Například jeden objekt objednávky v procesu vyřízení objednávky může během celého procesu mít několik různých stavů (akceptováno, zpracováno, vyřízeno…). [1], [18]

Životní cyklus datových objektů existuje pouze s existencí samotného procesu. Pro ukládání, aktualizování a načítání persistentních dat i po ukončení procesu je zapotřebí uložit data do datového úložiště – *Data Store*. Datové úložiště se nepoužívá pouze jako místo pro ukládání dat, umožňuje také načítání a aktualizování již existujících dat. Může se tedy jednat například o informační databázi, anebo v případě fyzických dat o skladiště produktů nebo kabinetu pro tištěné dokumenty. V diagramu je zobrazen jako reference – *Data Store Reference* pomocí symbolu datového úložiště viz Obrázek 9. [1], [18]

[13] upraveno z: BPMN coverage. *Camunda Platform 8 Docs* [online]. Berlin: Camunda Services, c2023 [cit. 2023-04-29]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/bpmn-coverage/

2.4 Koncept tokenu a toku procesu

V předchozích kapitolách byly mnohokrát použity pojmy jako je token a sekvenční tok. Tokeny a sekvenční toky jsou nepostradatelnými elementy při popisu průběhu a vztahů mezi aktivitami uvnitř procesu.

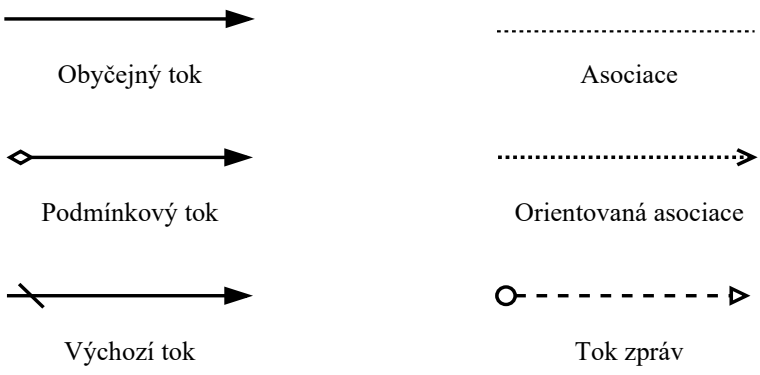
Koncept tokenu představuje jakýsi teoretický objekt používaný pro popis průběhu a postupu v diagramu. Token prochází jednotlivými tokovými objekty jako jsou aktivity, brány a události, které již byly zmíněny dříve. Tím, jak diagramem postupují, lze popsat chování procesu. Token je vytvořen na začátku procesu například počáteční událostí, a poté putuje do některé z koncových událostí (ať už explicitně či implicitně graficky definovaných). [18]

2.5 Spojovací objekty

Aby mohl token putovat mezi jednotlivými tokovými objekty, musí tyto objekty být nějakým způsobem spojeny. K tomuto účelu existují sekvenční toky – *Sequence Flows*. Ty mají tvar orientované šipky reprezentující hrany orientovaného grafu a spojují tokové objekty v podobě vrcholů grafu. Token těmito toky putuje s nulovým časem, tedy okamžitě. S průchodem tedy není asociována žádná časová režie. Existují ovšem také tokové objekty, kterými token putuje s nenulovým časem. Pomocí těchto objektů je poté možné modelovat složku času procesu. [15]

Existují tři druhy sekvenčních toků (graficky viz Obrázek 10):

- normální – obyčejný tok mezi dvěma tokovými objekty
- podmínkový – obsahuje podmínku určující propustnost pro token
- výchozí – tok použitý v případě, kdy všechny ostatní vycházející toky brány jsou vyhodnoceny jako nepravda



Obrázek 10: Symboly spojovacích objektů v BPMN[14]

[14] vlastní zdroj

Kromě sekvenčních toků existují i toky zpráv, kde tokem neputuje token ale zpráva. Pomocí zprávy lze předávat informace mezi jednotlivými plaveckými drahami. [3]

Třetím typem spojovacích objektů jsou asociace. Ty mohou být obyčejné či orientované jako je tomu u datových asociací. Asociace se používají pro spojení artefaktů s jinými elementy a datové asociace pro modelování toku dat. [18]

2.6 Artefakty a plavecké dráhy

Artefakty se nepoužívají pro popis průběhu procesu. Mají účel doplnit diagram o informace, které není možné vyjádřit ostatními elementy. Řadí se mezi ně textové anotace a skupiny. Pomocí textové anotace lze doplnit diagram o textový popisky. Skupiny mají stejný účel – zlepšení přehlednosti diagramu pomocí seskupení více elementů do jedné skupiny. [18]

Poslední kategorií elementů jsou plavecké dráhy skládajících se z bazénů – *Pools* a jednotlivých drah – *Swim Lane*. Jeden bazén představuje jednoho účastníka v procesu, a může být pro kategorizaci a přehlednost rozdělen do jedné či více drah viz Obrázek 11. [3]

Textová anotace | Skupina | Plavecké dráhy

Obrázek 11: Symboly artefaktů a plaveckých drah v BPMN[15]

2.7 Další prvky standardu BPMN

Standard BPMN pokrývá problematiku modelování procesů nejedním druhem diagramu. Kromě diagramu pro popis jednoho procesu používá BPMN další rozšířené pohledy na proces jako jsou diagramy choreografie, kolaborace a konverzace. Ty pomáhají popsat podnikové procesy, kde se nachází více účastníků a je zapotřebí modelovat jejich interakce.

V neposlední řadě je vhodné zmínit modelovací jazyk DMN (*Decision Model and Notation*), který se často používá právě v kombinaci s jazykem BPMN. Jazyk DMN poskytuje přehledný grafický popis složitých rozhodovacích procesů pomocí rozhodovacích tabulek a dalších elementů. Díky tomu je možné v BPMN diagramech přehledněji popsat rozsáhlé a komplexní rozhodování v procesu. [1]

[15] upraveno z: BPMN coverage. *Camunda Platform 8 Docs* [online]. Berlin: Camunda Services, c2023 [cit. 2023-04-29]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/bpmn-coverage/

3 PETRIHO SÍTĚ

V Petriho sítích existuje mnoho rozšiřujících tříd a podtříd, z nichž každá má své jedinečné vlastnosti a použití. [20]

V následujících kapitolách jsou proto probrány pouze základní principy fungování barvených Petriho sítí a základní prvky sítě jako jsou místa, přechody a hrany. Dále zde jsou podrobněji rozebrány principy barvených Petriho sítí, které rozšiřují klasické Petriho sítě zavedením barev, z důvodu jejich užití v této práci.

3.1 Barvené Petriho sítě

Barvené Petriho sítě rozšiřují základní strukturu Petriho sítí o další vlastnosti, což umožňuje modelovat a analyzovat složitější systémy. Hlavním rozdílem oproti klasickým sítím je účel použití. Klasické sítě se řadí do kategorie takzvaných sítí nízké úrovně. Jsou proto spíše vhodné pro modelování teoretických modelů souběžnosti. Oproti tomu barvené sítě jsou řazeny do sítí vysoké úrovně, které jsou charakteristické kombinací Petriho sítí s vyšším programovacím jazykem. Jsou proto více vhodné pro modelování situací v praxi, především proto, že umožňují vytvářet komplexní a parametrizované modely. [21]

Barvené Petriho sítě (*Coloured Petri nets* – CPN) využívají pro své účely programovací jazyk CPN ML vycházejícího z funkcionálního programovacího jazyka Standard ML. Jazyk CPN ML rozšiřuje jazyk Standard ML o konstrukce potřebné pro popis barvených Petriho sítí. Modely vytvořené ve formalismu barvených sítí se tedy skládají ze dvou složek: grafické síťové struktury a formální inskripce v jazyce CPN ML pro definování množin barev a datových typů, deklarování proměnných anebo také manipulaci s multimnožinami. [21]

Další podkapitoly popisující definice a koncepty se zabývají pouze barvenými Petriho sítěmi.

3.2 Definice nehierarchické barvené Petriho sítě

V této kapitole je stručně uveden aparát nehierarchické barvené Petriho sítě. Úplnou formální definici barvené Petriho sítě lze najít ve zdroji [21], kde jsou podrobně popsány všechny vlastnosti barvených Petriho sítích, které jsou zde opomenuty. Zde jsou formálně uvedeny pouze základní charakteristiky, které jsou zde pouze neformálně popsány.

Barvené Petriho sítě rozšiřují trojici míst, přechodů a hran o další prvky a vlastnosti. Celkově tak vzniká devítice, pro kterou je charakteristika následující:

N-tice $CPN = (P, T, A, \Sigma, V, C, G, E, I)$ se nazývá nehierarchická barvená Petriho síť právě tehdy když: [21]

- P je konečná množina míst
- T je konečná množina přechodů
- $A \subseteq P \times T \cup T \times P$ je množina orientovaných hran
- Σ je konečná množina neprázdných množin barev
- V je konečná množina proměnných, kde pro všechny proměnné $v \in V$ platí:

$$Type[v] \in \Sigma$$

- $C: P \rightarrow \Sigma$ je funkce barev, která každému místu přiřazuje množinu barev.
- $G: T \rightarrow EXPR_V$ je funkce strážních podmínek, která každému přechodu $t \in T$ přiřazuje strážní podmínku, pro kterou platí:

$$Type[G(t)] = Bool$$

- $E: A \rightarrow EXPR_V$ je funkce hranových výrazů, která každé hraně $a \in A$ přiřazuje hranový výraz, pro který platí:

$$Type[E(a)] = C(p)_{MS}$$

  kde p je místo spojené s hranou a
- $I: P \rightarrow EXPR_\emptyset$ je inicializační funkce, která každému místu $p \in P$ přiřazuje inicializační výraz, pro který platí:

$$Type[I(p)] = C(p)_{MS}$$

Pozn: V definici se vyskytují další výrazy jako:

- $EXPR$ – označení množiny výrazů popisného jazyka
- $Type[e]$ – označení typu hodnoty získané vyhodnocením výrazu $e \in EXPR$
- MS – označení multimnožiny

Úplná definice všech použitých výrazu je dostupná ve zdroji [21].

3.2.1 Koncept tokenu v Petriho sítích

Pro pochopení jednotlivých částí definice barvené Petriho sítě, je zapotřebí zavedení konceptu tokenu, pomocí kterého jsou popsány další koncepty jako jsou množiny barev, hranové výrazy či evoluce sítě.

Petriho sítě, obdobně jako notace BPMN, používají pro popis stavu modelu koncept tokenu. Význam tokenu je ovšem odlišný. V BPMN se tokeny využívají ke sledování instancí procesu. Tokeny putující přes různé aktivity brány a jiné elementy představují jednu instanci procesu a její aktuální pozici v rámci toku.

V Petriho sítích představují tokeny diskrétní entity (odrážející např. předměty, výrobky apod.), které se pohybují systémem. Používají se k reprezentaci stavu systému a interakcí mezi jeho součástmi. Tokeny v Petriho sítích nejsou vázány na konkrétní instanci procesu a mohou reprezentovat různé objekty v závislosti na konkrétní aplikaci. V klasických Petriho sítích nejsou jednotlivé instance tokenů rozlišovány a pracuje se pouze s jejich počty. V grafické podobě jsou reprezentovány jako černé body, a proto jsou nazývány jako *black tokens.* [20]

Barvené Petriho sítě rozšiřují základní token tím, že umožňují, aby tokeny měly další atributy, například prioritu, požadavky na zdroje nebo stav. To umožňuje podrobnější reprezentaci modelovaného systému. Barvené Petriho sítě také umožňují modelovat souběžné procesy, kdy se na stejném místě může současně vyskytovat více tokenů s různými atributy, které představují různé entity nebo aspekty systému. Tokeny jsou od sebe odlišeny barvou z čehož vychází název barvené sítě. [20]

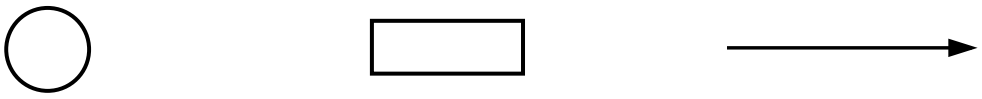
3.2.2 Základní elementy

Petriho sítě využívají pro popis své „infrastruktury“ grafovou strukturu sestávající ze dvou hlavních elementů – konečnou množinou míst *P* (*places*) a množinou přechodů *T* (*transitions*). Místa se používají pro modelování pasivních komponent v modelu jako jsou například množiny zdrojů, fronty apod. Tyto elementy uchovávají tokeny. Graficky se znázorňují symbolem oválu či kruhu. [20]

Druhým typem elementů jsou přechody. Ty naopak reprezentují aktivní komponenty jako jsou akce, události nebo vysílání zprávy. Mohou tedy produkovat, konzumovat, přesouvat či jinak ovlivňovat tokeny. Přechody jsou graficky znázorněny obdélníky nebo čtverci. [20]

Tyto elementy jsou společně propojeny množinou orientovaných hran *A* v podobě šipek – *arcs*. Ty nereprezentují komponenty v modelu ale pouze jakýsi logický vztah mezi komponentami. Jak je z předchozí formální definice patrné, hrany mohou existovat pouze mezi elementy různého typu. Znamená to tedy, že nelze mít hranu mezi elementy místo – místo nebo přechod – přechod ale pouze mezi přechodem a místem. [20], [21]

Společně tato trojice prvků představuje síťovou strukturu, která je doplněna o další informace a inskripce, které jsou specifické pro barvené Petriho sítě.



Místo Přechod Hrana

Obrázek 12: Symboly elementů Petriho sítí[16]

3.2.3 Doplňující inskripce

Již zmíněné formální inskripce, které určují typ, hodnotu a barvu tokenů a také podmínky, za kterých se tokeny mohou pohybovat mezi místy a přechody, jsou důležitou a neoddělitelnou součástí sítě. Inskripce doplňují orientovaný graf o informace definující chování celé sítě. Tyto inskripce zahrnují: [21]

- množiny barev – *colour sets / colsets*
- počáteční značení – *initial marking*
- hranové výrazy – *arc expressions*
- strážní podmínky – *guards*

Nejdůležitější složkou jsou množiny barev Σ, které specifikují typy značek v modelu. Lze je chápat jako datové typy ve vyšších programovacích jazycích. Jazyk CPN ML poskytuje jednoduché datové typy, ze kterých lze poté vytvářen strukturované datové objekty (záznamy, seznamy apod.). Ke každému místu v síti je přiřazena inskripce pomocí funkce C definující množinu barev, které se na tomto místě mohou vyskytovat. Hodnoty tokenu představují barvy a množiny barev určují množinu hodnot, kterých mohou barvy nabývat. [21]

Počáteční značení umožňuje určit počáteční multimnožinu značek v jednotlivých místech. To je možné pomocí inicializačního výrazu v jazyce CPN ML, který je k místům přiřazen pomocí funkce I. [21]

Hranové výrazy a strážní podmínky jsou vysvětleny v následující kapitole popisující evoluci sítě a její pravidla.

3.2.4 Evoluční pravidla

Jak již bylo uvedeno dříve, Petriho sítě se používají zejména pro popis a modelování dynamických systémů. Příslušná dynamika je reprezentována pomocí evoluce sítě, která je dána evolučními pravidly a principy. Prvním z principů je princip duality, který uvádí existenci dvou druhů elementů, které tu již byly zmíněny – místa a přechody. [23]

[16] vlastní zdroj

Druhým principem je princip lokality, který říká, že chování přechodu závisí pouze na něm a stavu jeho okolí. Toto okolí je definováno místy, které vstupují a vystupují z přechodu. Takové vstupní okolí ${}^\bullet t$ a výstupní okolí $t^\bullet$ přechodu lze tedy definovat jako: [22], [23]

$$\begin{aligned} {}^\bullet t &:= \{p \mid p \in P \wedge (p,t) \in A\} \\ t^\bullet &:= \{p \mid p \in P \wedge (t,p) \in A\} \end{aligned} \tag{3.1}$$

Princip souběžnosti úzce souvisí s principem lokality. Ten říká, že pokud existují dva přechody, které mají disjunktní množinu lokalit, mohou události přechodů nastat nezávisle. [23]

Události jsou v Petriho sítích reprezentování provedením (odpálením) přechodu. Jako každá reálná událost má i přechod své podmínky před provedením a následky po provedení události. Předpoklady a podmínky jsou reprezentovány stavem vstupního okolí přechodu a následky jsou reprezentovány stavem výstupního okolí přechodu. [22]

Provedením přechodu se změní značení (stav) $M(p)$ na následné značení $M'(p)$. Pokud jsou splněny předpoklady proveditelnosti přechodu, je tento přechod označován jako proveditelný. Pro zjištění proveditelnosti přechodu je zapotřebí vyhodnocení hranových výrazů vstupních a výstupních hran přechodu. Hranové výrazy se vyskytují na hranách grafu a specifikují multimnožinu značek, které jsou odebrány ze vstupních míst, resp. přidávány do míst výstupních. Výrazem nemusí být pouze jednoduchý výraz v podobě volné proměnné, ale i velice komplexní výraz v rozsahu jazyka Standard ML jako jsou řídící struktury (if – else) či dokonce funkce. Podmínkou takového výrazu je dodržení návratového typu (množiny barev), který musí být stejný jako je typ přilehlého místa. To je patrné z formální definice funkce E, která přiřazuje hranový výraz hranám. [21]

Podmínku proveditelnosti přechodu lze navíc doplnit také strážními podmínkami na samotném přechodu. Ty představují omezující výrazy ve formě predikátů, které určují podmínky pro dosazení tokenů do volných proměnných. Návratovým typem takového výrazu je tedy booleovská hodnota, která musí být vyhodnocena jako pravdivá, jinak není dosazení vyhodnocovaného tokenu do hranových výrazů dovoleno. Přechod je tedy proveditelný pouze pokud je pravdivě vyhodnocena strážní podmínka a zároveň existuje navázání proměnných u hranových výrazů. Strážní podmínky obdobně jako hranové výrazy mohou obsahovat proměnné, do kterých jsou dosazovány konkrétní tokeny pro vyhodnocení podmínek a výrazů. [21]

3.3 Koncept hierarchie

Obdobně jako programy lze modely barvených Petriho sítí organizovat do menších částí v podobě modulů. Tento koncept je vhodný hned z několika důvodů. Při modelování rozsáhlého modelu sítě, se model stává velice rychle nepřehledným a používání takové sítě je velice nepraktické. Rozdělením modelu do menších částí se může uživatel modelující síť lépe soustředit na menší celek a lépe se soustředit na detaily daného modulu. Dalším důvodem je využití znuvupoužitelnosti modulů. Takto modelované systémové komponenty lze poté použít v modelu sítě opakovaně a předejít případným duplicitám, které by bylo nutné explicitně modelovat. Navíc je možné moduly využít vnořením jednoho modulu do jiného a vytvořit tak hierarchickou strukturu. To opět vede k větší přehlednosti celého modelu a také k úspoře času potřebného pro modelování. Při změně vlastností či jiné úpravě takového modulu je tato změna promítnuta do všech částí, kde je modul využit a není proto nutné opravovat tuto změnu na více místech. Graficky lze tyto moduly znázornit v podobě hierarchicky organizovaných stránek. Může tedy existovat i stránka, která je podstránkou jiné stránky. [21]

3.4 Koncept fúzních míst

S konceptem hierarchie souvisí i koncept takzvaných fúzních setů. Ty umožňují fúzi několika míst do jednoho společného setu. Místa z jednoho setu se poté v celém diagramu sítě chovají jako jeden jediný společný prvek. Takto vzniklý prvek je globálně dostupný mezi všemi moduly a může tedy být použit jako společné úložiště značek mezi moduly. [21]

Fúzní sety jsou užitečné zejména při modelování souběžných dějů, kde může být nutné synchronizovat činnosti více aktivit. Jsou také užitečná při modelování systémů se složitými řídicími toky nebo datovými závislostmi, kde je synchronizace událostí kritická pro správné fungování systému. [21]

3.5 Koncept reprezentace času

Barvené Petriho sítě je možné obohatit také o časovou složku. Časované barvené Petriho sítě pak umožňují analyzovat a simulovat systémy zahrnující plynutí času, v nichž čas hraje důležitou roli. Časová složka je v CPN implementována pomocí časových razítek – *timestamps*, které reprezentují hodnotu simulačního času. Při konstrukci časovaných modelů je definice barev opatřena o parametr *timed*, která definuje barvu jako časovanou barvu. Jednotlivé tokeny takové barvy poté obsahují časové razítko, které udává podmiňující čas, od kterého je tento token možné použít při čerpání z místa při odpálení přechodu. Hodnota časového razítka tokenu je ovlivněna přechody, které obsahují informaci o časové režii

vykonání přechodu. V praxi se může jednat o dobu trvání činnosti, kterou tento přechod reprezentuje. Časová režie nemusí být určena pouze na přechodu samotném, ale také na vycházejících hranách. Součet časové režie je poté po odpálení přechodu připočten k tokenu vycházející z onoho přechodu danou hranou. [21]

4 VLASTNÍ ŘEŠENÍ

Hlavním cílem praktické části práce bylo vytvoření metodiky pro transformaci modelů procesů vytvořených v notaci BPMN do formalismu barvených Petriho sítí. V úvodu jsou zmíněny nástroje vybrané pro modelování BPMN modelů a modelů barvených Petriho sítí. Dále je pak definována redukovaná sada elementů BPMN-light a s ní související validační pravidla pro tvorbu výchozího modelu. Poté následuje vysvětlení principu nově vytvořené metodiky transformace modelů a také je zde popsána implementovaná webová aplikace, která tuto metodiku využívá. Poslední podkapitola zmiňuje možnost analýzy a simulace výstupního modelu barvené Petriho sítě.

4.1 Vymezení problematiky

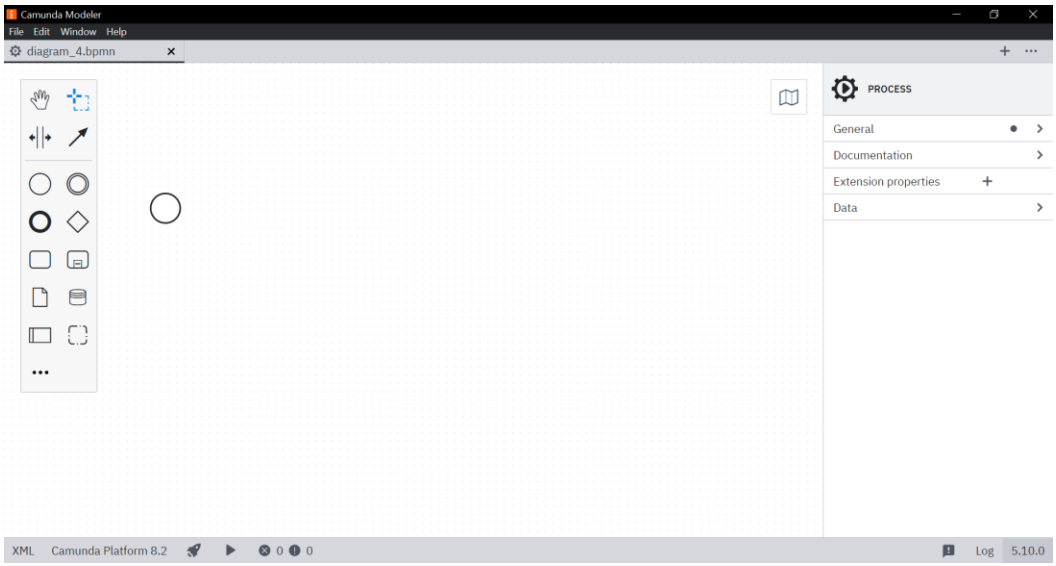
V této kapitole jsou úvodem popsány vybrané modelovací nástroje pro modelování vstupních a výstupních modelů v metodice PetriBPMN. Nová metodika PetriBPMN je popsána konkrétně pro použití s těmito nástroji. Myšlenka metodiky je ovšem univerzální a jednotlivé principy metodiky jsou přenositelné i pro jiné modelovací nástroje.

Dále je zde uvedena nová redukovaná sada BPMN elementů BPMN-light a důvody této redukce pro použití v metodice PetriBPMN.

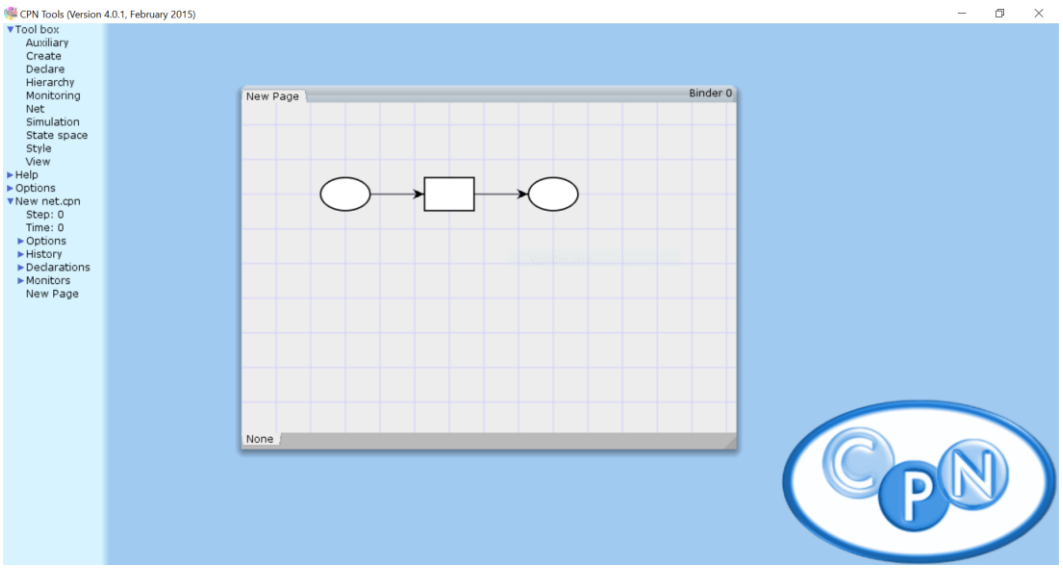
4.1.1 Volba modelovacích nástrojů

Pro modelování BPMN a CPN modelů existuje řada nástrojů. Pro modelování výchozích modelů pomocí notace BPMN byl zvolen nástroj Camunda Modeler viz Obrázek 13, resp. jeho implementace ve formě webového nástroje bpmn-js. Nástroj od společnosti Camunda Services GmbH byl vybrán zejména z důvodu jeho rozšířenosti, jednoduchosti a otevřenosti kódu. Nástroj je navíc velice intuitivní a podporuje implementaci validačních pravidel pro kontrolu správnosti modelu, která jsou popsána v následujících kapitolách.

Pro modelování, analýzu a simulaci cílových CPN modelů byl vybrán nástroj CPN Tools viz Obrázek 14. CPN Tools podporuje práci s barvenými a časovanými Petriho sítěmi. Navíc poskytuje pokročilé analytické funkce, jako je analýza dosažitelnosti, průzkum stavového prostoru a simulace modelu, které mohou uživatelům pomoci ověřit správnost a výkonnost jejich modelů CPN. Nástroj byl spravován a vyvíjen skupinou AIS Group, která je součástí Technické univerzity Eindhoven v Nizozemí. Nyní je nástroj oficiálně nahrazen nástrojem CPN IDE, který je stále ve vývoji. V této práci byl použit starší nástroj CPN Tools z důvodu kompletnosti využitých funkcionalit a lepší stability.



Obrázek 13: Desktopová verze nástroje Camunda Modeler[17]



Obrázek 14: Nástroj CPN Tools[18]

4.1.2 BPMN-light – redukce standardu BPMN

Standard BPMN, který byl popsán v předchozích kapitolách je velice rozsáhlý a komplexní standard zastřešující rozsáhlou problematiku modelování procesů. Obsahuje proto mnoho elementů a vlastností, které nejsou pro účely této práce potřeba. Proto byla sada symbolů a elementů, které pokrývá standard BPMN, redukována a poupravena. Tento krok byl učiněn zejména za účelem vytvoření elementárního grafického jazyka pro modelování podnikových a technologických procesů i pro uživatele, kteří BPMN běžně nepoužívají a nemají s ním velké

[17] vlastní zdroj
[18] vlastní zdroj

zkušenosti. Zavedením minimalistické sady elementů je navíc zjednodušena tvorba vstupních modelů procesů, které jsou velice jednoduché na pochopení.

Nová sada elementů pojmenovaná jako BPMN-light obsahuje redukovanou sadu elementů podporovaných editorem Camunda Modeler, který sám o sobě také nepodporuje všechny modelovací elementy z oficiálního standardu. Seznam elementů podporovaných editorem Camunda Modeler je dostupný na stránkách nástroje ve zdroji [19]. Redukovaná sada BPMN-light obsahuje pouze nejzákladnější elementy pro popis průběhu procesu a reprezentaci datových a fyzických zdrojů pro modelování konkurenčních modelů procesu soupeřící o sdílené zdroje.

Jednotlivé elementy v BPMN-light eviduje Tabulka 1.

Tabulka 1: Elementy sady BPMN-light[19]

Kategorie	**Element**
Událost	Prázdná počáteční událost
	Prázdná koncová událost
Aktivita	Nespecifikovaná událost
Data	Datové úložiště
Brána	Exkluzivní brána
	Paralelní brána
	Inkluzivní brána
Spojovací objekt	Obyčejný tok
	Datový tok
Artefakt	Textová anotace
	Skupina

Elementy textové anotace a skupiny jsou dovoleny, ale pro účel transformace jsou ignorovány. Všechny ostatní elementy, které se v tabulce nenachází jsou v BPMN-light zakázány.

Co se týče sémantiky elementů, ta zůstává téměř totožná. Prázdné počáteční události modelují začátek procesu a prázdné koncové události pak konec celého jednoho procesu. Nespecifikované aktivity reprezentují jakoukoli aktivitu, činnost či práci uvnitř procesu, kterou může být jak manuální, tak i automatizovaná činnost prováděná nějakým strojem. Není zde ovšem specifikován přesný typ činnosti jako tomu je v plném rozsahu standardu BPMN. Význam bran, spojovacích objektů a datových úložišť zůstává beze změny.

4.1.3 Validační pravidla

Standard sám o sobě nespecifikuje zcela konkrétní pravidla pro vytváření modelů procesu. Jednu skutečnost je tedy možné modelovat několika způsoby což může způsobovat značné

[19] vlastní zdroj

problémy v konzistenci vytvářených modelů. Proto jsou zde zavedena validační pravidla, která jsou jakousi nadstavbou pravidel validujících správnost modelu. Důvodů zavedení těchto pravidel je hned několik.

Jedním z hlavních důvodů je již zmíněná konzistence. Dodržováním jednotné a konzistentní sady pravidel se diagramy stávají snáze čitelnými a udržovatelnými v průběhu času, zejména ve větších projektech, kde na stejných diagramech pracuje více lidí.

Dalším důležitým důvodem je kontrola syntaktické správnosti modelu. Velice často se stává, že jsou vytvářeny BPMN diagramy, které svým způsobem neporušují pravidla daná standardem, ale jsou syntakticky nesprávné, neúplné a pro čtenáře diagramu zcela nepochopitelné. Obecně lze tedy říct, že zavedením validačních pravidel vznikají kvalitnější a významově správné diagramy, což vede k úspoře času potřebného ke kontrole chyb a nesrovnalostí diagramů.

Použitá validační pravidla v této práci jsou podrobně vysvětlena v kapitole 4.6, kde je popsán vytvořený modul obsahující všechna validační pravidla potřebná pro vytvoření validního modelu procesu využívajícího redukovanou sadu BPMN-light.

4.2 Modelový případ

Pro účely vysvětlení a následné demonstrace použití nově vytvořené metodiky jsou zde zavedeny dva modely: servis automobilů a čerpací stanice. Servis úzce spolupracuje s čerpací stanicí, která poskytuje mycí linky nejen pro zákazníky čerpací stanice, ale i pro místní servis automobilů. Mycí linky jsou tedy sdílené mezi servisem a čerpací stanicí.

4.2.1 Model servisu automobilů

Prvním modelem je servis firemních automobilů viz Obrázek 16, který disponuje dílnami pro opravy různých vozidel a dodatečnými dílnami pro poskytování nadstandardních služeb jako je aplikace reklamních polepů na karoserii či čištění interiéru vozidla.

Celý proces začíná příjezdem vozidla do servisu. Zde je rozhodnuto, do jaké dílny je vozidlo přesunuto podle typu vozidla. Servis nabízí opravy a prohlídky osobních vozidel a nákladních vozidel. Nákladní vozidla jdou k servisnímu technikovi vždy, osobní automobily pouze pokud mají zjištěnou závadu a autobusy zde nejsou vůbec servisovány. Po prohlídce vozidla a případné opravě je vozidlo přesunuto do mycí linky, kde probíhá mytí karoserie. Zároveň během mytí karoserie probíhá zápis protokolu o příjezdu vozidla. Po umytí vozidla může vozidlo opustit servis anebo využít některou z dalších nabízených služeb. Těmi jsou již zmíněné

polepy reklam a čištění interiéru. Zákazník nemusí požadovat pouze jednu z těchto služeb, ale může požadovat klidně obě služby najednou.

4.2.2 Model čerpací stanice

Druhým modelem je model čerpací stanice viz Obrázek 17, která poskytuje tankování klasických fosilních paliv jako je benzín, nafta či LPG. Dále nabízí služby mytí karoserie poskytováním automatizovaných mycích linek.

Proces je v tomto případě velice triviální. Při příjezdu vozidla je rozhodnuto, zda vozidlo tankuje LPG či nikoli. Při tankování LPG je zapotřebí příjezd vozidla ke stojanu s LPG. Pro tankování benzínu a nafty jsou využívány obyčejné tankovací stojany. Po natankování je možné využít mytí karoserie, které probíhá v mycí lince. Po natankování a případném umytí karoserie opouští vozidlo areál čerpací stanice.

4.3 Metodika PetriBPMN

Nově vytvořená metodika PetriBPMN popisuje proces automatizované transformace modelů business procesů modelovaných pomocí redukované sady BPMN-light do formalismu barvených Petriho sítí. V následujícím textu jsou využívány tyto čtyři typy modelů používaných v metodice PetriBPMN:

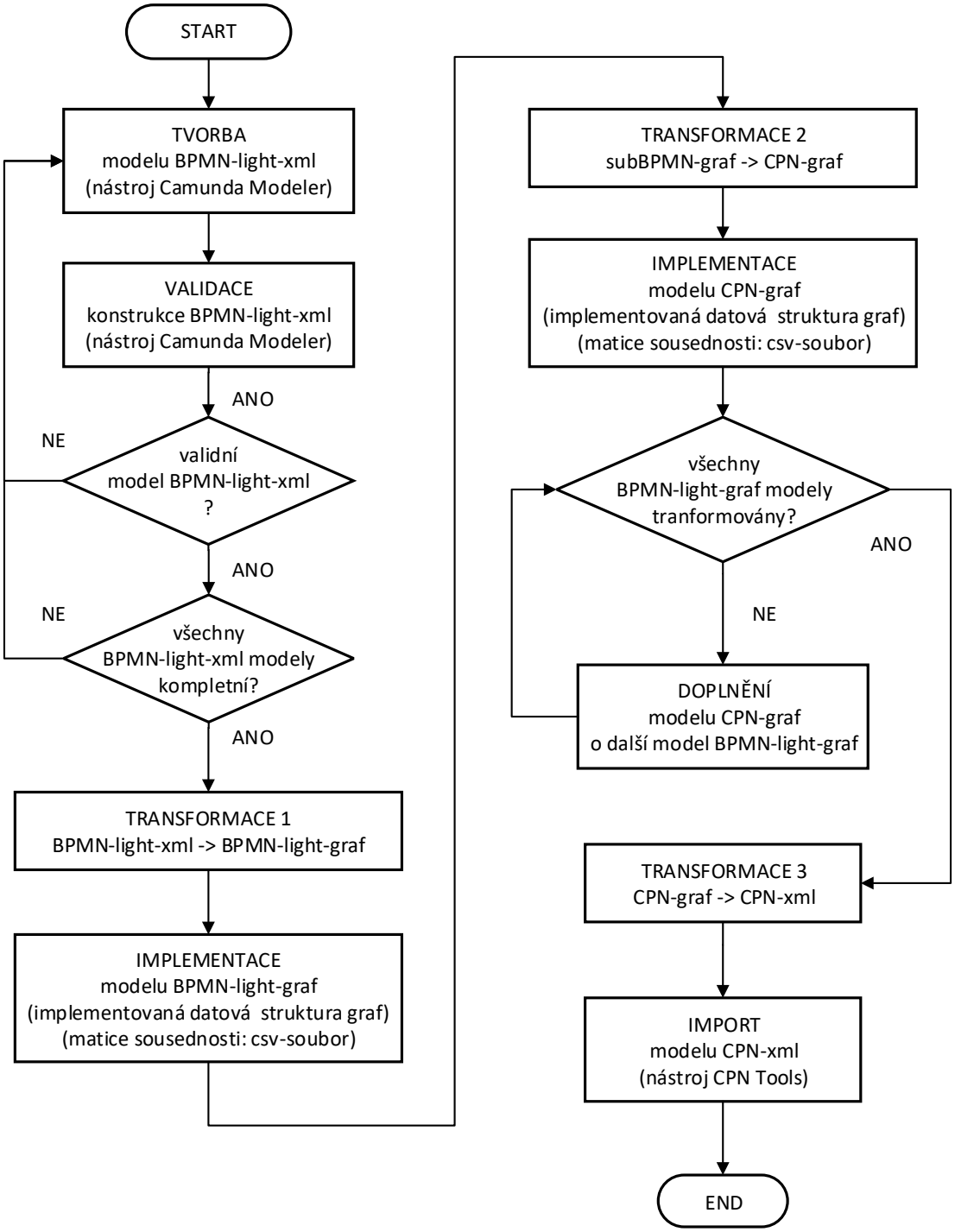
- BPMN-light-xml
- BPMN-light-graf
- CPN-graf
- CPN-xml

Prvním z nich je model BPMN-light-xml. Ten představuje soubor XML obsahující výchozí model procesu, který používá editor Camunda Modeler. Druhým modelem je BPMN-light-graf, který rovněž představuje model výchozího procesu ale ve formátu datové struktury graf. Třetí model – CPN-graf obsahuje reprezentaci cílového modelu barvené Petriho sítě opět ve formátu datové struktury graf. Posledním modelem je konečný model barvené Petriho sítě CPN-xml, který představuje výstupní soubor XML kompatibilní pro použití v nástroji CPN Tools.

Metodika se skládá ze čtyř hlavních fází:

- modelování vstupních BPMN-light modelů procesů
- deserializace BPMN modelu do modelu BPMN-light-graf
- transformace modelu BPMN-light-graf do modelu CPN-graf
- serializace modelu CPN-graf do výstupního modelu CPN-xml

Obrázek 15 obsahuje podrobný vývojový diagram metodiky obsahující všechny fáze metodiky.



Obrázek 15: Vývojový diagram metodiky PetriBPMN[20]

[20] vlastní zdroj

4.3.1 Modelování vstupních procesů

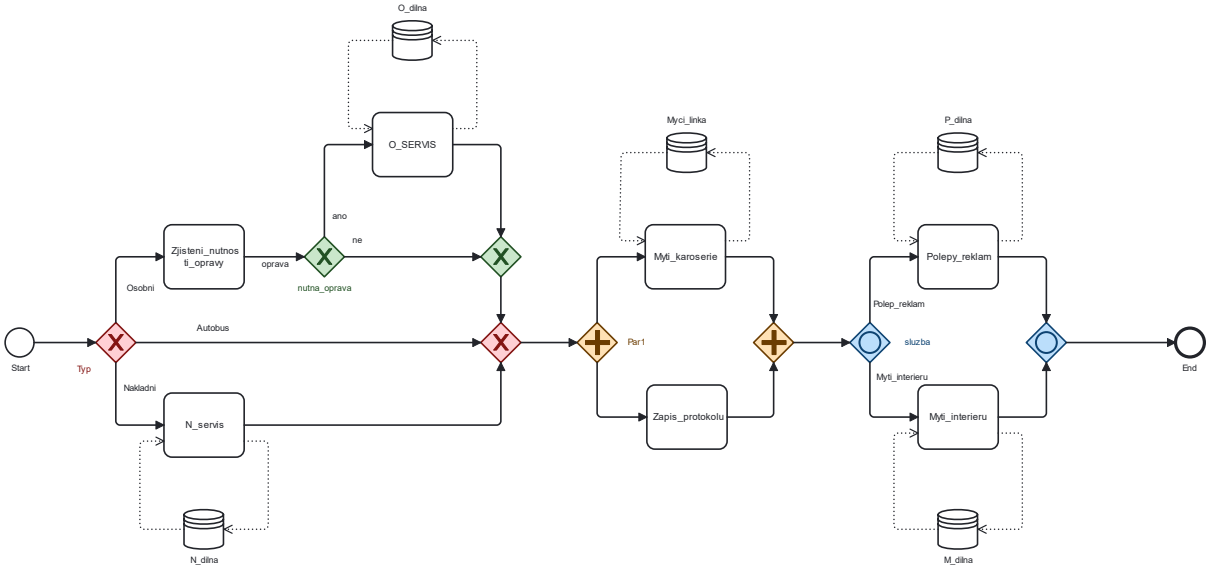
Celý proces metodiky začíná tvorbou BPMN modelu procesu pomocí editoru Camunda Modeler. K tomu je možné použít desktopovou aplikaci nebo webovou variantu nástroje. Editor disponuje možnostmi modelování, které ale nejsou podporovány v nové redukované sadě BPMN-light. Pro ověření správnosti modelu splňující všechna potřebná kritéria, jsou zde již zmíněná validační pravidla. Díky tomu je možné ihned při modelování procesu zjistit případné nedostatky či chyby v modelu před samotnou transformací do Petriho sítě. Výstupem této fáze je validovaný XML soubor modelu procesu ve formátu BPMN-light-xml, který je kompatibilní pro účely transformace.

Při modelování procesu je tedy zapotřebí dodržet zmíněná validační pravidla. Celý proces musí začínat počáteční událostí a končit koncovou událostí. Činnosti, práce a jiné aktivní prvky jsou reprezentovány pomocí aktivit. Pro modelování obslužných zdrojů využívaných procesy je zde element datového úložiště. Při modelování více datových úložišť se stejným názvem jsou tato úložiště modelována jako společné fúzní místo v Petriho sítích. Datová úložiště mohou být propojena s aktivitami pomocí datových asociací, které reprezentují využití/vrácení zdrojů z/do úložiště. Datové asociace navíc obsahují proměnnou určující počet zdrojů, se kterými je touto asociací manipulováno.

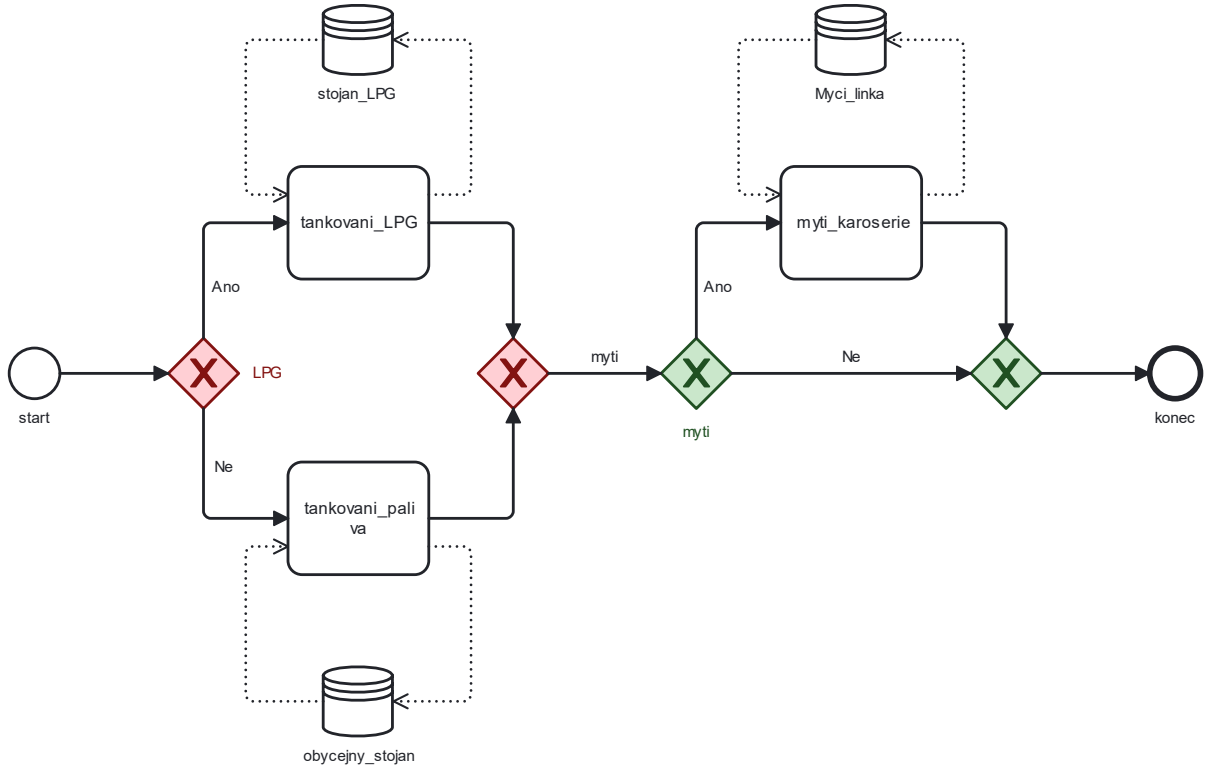
Při modelování podmínkových větvení je zapotřebí dodržení párování bran. Každá rozdělovací brána musí mít párovou spojovací bránu. Tato vlastnost je zapotřebí zejména pro transformaci do Petriho sítě, která je popsána v kapitole 4.3.3.

Formát zápisu počtu u datových asociací a způsob reprezentace podmínek u podmínkových větvení je popsán u popisu implementovaných validačních pravidel v kapitole 4.6. V této kapitole jsou také uvedeny jmenné konvence pro pojmenovávání elementů v BPMN.

Obrázek 16 a Obrázek 17 obsahují BPMN modely modelového případu z kapitoly 4.2.



Obrázek 16: BPMN model servisu automobilů[21]



Obrázek 17: BPMN model čerpací stanice[22]

[21] vlastní zdroj
[22] vlastní zdroj

4.3.2 Deserializace vstupního modelu

Druhým krokem je deserializace exportovaného BPMN-light-xml modelu ve formátu XML souboru z předchozího kroku pro jeho následné zpracování. Tento soubor dodržuje všechny náležitosti definované standardem BPMN, který definuje přesný formát XML souboru pro ukládání BPMN modelu. Díky tomu je deserializace dokumentu částečně automatizovaná bez nutnosti implementace vlastní metody deserializace. Deserializováním vzniká soubor tříd, které ale nejsou reprezentací datové struktury graf. Proto je zapotřebí průchodem elementů tuto datovou strukturu sestavit. Výsledkem je datová struktura graf v podobě BPMN-light-graf modelu reprezentující vstupní model procesu.

Grafová struktura modelu procesu lze pro znázornění zapsat také jako matici sousednosti. Ta popisuje celý diagram pomocí matice obsahující vztahy mezi elementy. Elementy mají navíc přidány prefix, který určuje jejich typ. Co se týče hodnot matice, ty určují typ vtahu mezi elementy. Pokud se jedná o datovou asociaci mezi datovým úložištěm a jiným elementem, číselná hodnota reprezentuje počet zdrojů spojených s touto asociací. Ostatní alfanumerické hodnoty pak reprezentují podmínkové výrazy sekvenčních toků. Pokud je hodnotou symbol orientované šipky, jedná se o obyčejný sekvenční tok.

Vytvořené matice sousednosti modelového případu jsou přiloženy k práci jako příloha z důvodu jejich rozsáhlosti.

Konkrétní implementace fáze deserializace je popsána v kapitole 4.7.2.

4.3.3 Transformace BPMN-light modelu do Petriho sítě

Transformace mezi formalismy je nejrozsáhlejší a implementačně nejsložitější fází. Je zapotřebí zajistit správné mapování BPMN elementů na elementy barvené Petriho sítě. Toto mapování bylo inspirováno článkem ze zdroje [24], který popisuje postup transformace jednotlivých prvků diagramu BPMN. Před touto transformací je ale zapotřebí zjistit a nalézt párování bran. Tento proces byl inspirován článkem ze zdroje [25], který se zabývá segmentací diagramu BPMN na menší části pomocí párování bran.

Operace párování bran je zcela automatizovanou operací bez nutnosti implementace doplňujících informací v diagramu BPMN ze strany uživatele. Obrázek 16 a Obrázek 17 mají tyto páry barevně odlišeny z demonstračních účelů a takové barvení není při modelování potřeba. Z pohledu této práce je ale párování bran klíčové především pro mapování elementu inkluzivní brány, která je implementačně a konstrukčně nejvíce komplexním prvkem. Párování je druhotně významné pro pojmenovávání koncových (spojovacích) bran, které jsou

pojmenovány s prefixem názvu brány počáteční (rozdělovací) pro lepší orientaci v koncovém CPN modelu. Implementace metody hledání párové brány je popsána v kapitole 4.7.3.

Po dokončení operace párování následuje část mapování BPMN elementů do elementů CPN. Toto mapování je prováděno pomocí pravidel, která jsou zmíněna v samostatné kapitole 4.4. Implementace mapování je pak popsána v kapitole 4.7.4.

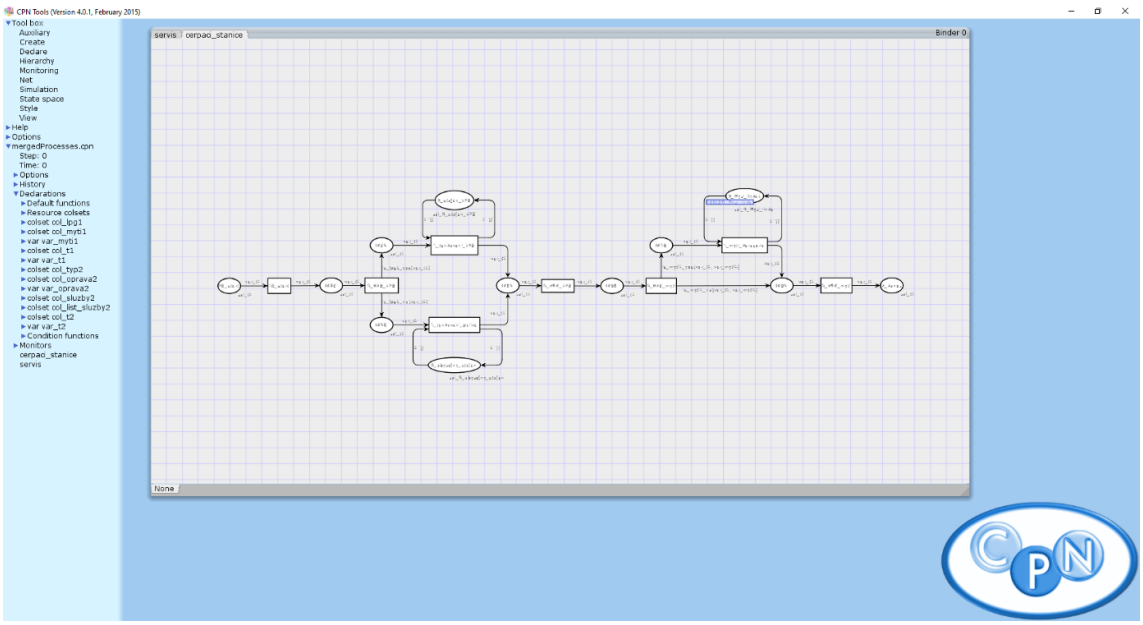
Metodika PetriBPMN se nezabývá pouze transformací individuálních procesů, ale poskytuje také možnost fúze více procesů do jediného výstupního modelu CPN. Tato fúze umožňuje spojit procesy, které mají společné prvky v podobě zdrojů, které sdílejí. Fúze funguje na principu inkrementálního přidávání transformovaných procesů. Na začátku je transformován jeden proces do modelu CPN-graf a postupně jsou pak přidávány další procesy. Každý z těchto procesů je oddělen pomocí stránek.

Výstupem této fáze je model CPN-graf popisující konečnou síť hierarchické barvené Petriho sítě. V příloze jsou opět přiloženy matice sousednosti popisující nově vzniklou datovou strukturu graf. Tato matice obsahuje hodnoty pouze dvojího druhu. Číselné hodnoty opět znamenají alokovaných/uvolněných zdrojů v podobě tokenů. Šipky zde znázorňují spojení elementů hranou, která může mít různé hranové výrazy. Ty jsou ovšem velice dlouhé a složité, a proto jsou zde vynechány.

4.3.4 Serializace transformovaného modelu

Posledním krokem metodiky je serializace transformovaného modelu CPN-graf do modelu CPN-xml, který je poté převeden do souboru XML pro práci v nástroji CPN Tools. Nástroj CPN Tools definuje formát souboru obdobně jako specifikace BPMN definuje formát souboru XML pro své soubory. Tím je serializace značně ulehčena a částečně automatizována. Výstupní soubor lze poté otevřít v nástroji CPN Tools viz Obrázek 18 a pracovat s modelem procesu již ve formalismu CPN.

Velký obrázek v příloze ukazuje modelový případ ve formalismu CPN. Sítě jsou pro přehlednost znázorněny společně v jednom obrázku. V nástroji CPN Tools jsou tyto dva procesy hierarchicky odděleny pomocí stránek ovšem se sdílenými definicemi barev, proměnných atd.

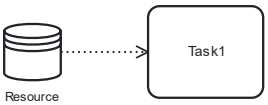
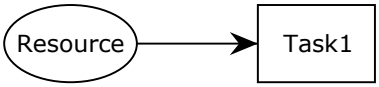
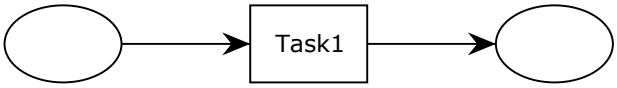
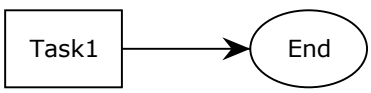


Obrázek 18: Výsledný model modelového případu v nástroji CPN Tools[23]

4.4 Mapování BPMN-light elementů do ekvivalentů v CPN

Oproti formalismu barvených Petriho sítí, obsahují BPMN modely větší množství různých elementů a je zapotřebí je proto mapovat na jejich ekvivalentní reprezentace v CPN pomocí míst a přechodů.

Tabulka 2: Mapování vybraných BPMN elementů do reprezentace v CPN[24]

BPMN element	Mapování v CPN
Datové úložiště – rezervoár zdrojů 	
Akvitita 	
Počáteční událost 	
Koncová událost 	

[23] vlastní zdroj
[24] vlastní zdroj

4.4.1 Datová úložiště

Nejjednodušším mapováním je mapování datových úložišť, která jsou jednoduše mapována jako míst. Toto místo může být jednoduchým místem anebo fúzním místem v závislosti na konkrétním použití rezervoáru zdrojů. Datové úložiště reprezentované více místy je doplněno o informaci fúzního setu.

Příkladem mapování datového úložiště je mapování rezervoáru zdrojů „*Myci_linka*“ v procesu servisu z modelového případu, kdy je rezervoár mapován jako místo s názvem fúzního setu. Fúzní set označuje toto místo jako fúzní místo, jelikož je zde zdroj „*Myci_linka*“ modelován i v druhém procesu čerpací stanice.

4.4.2 Aktivity

Mapování aktivit je opět velice jednoduchou záležitostí. V BPMN standardu představují aktivity aktivní prvek v diagramu a jsou proto mapovány na přechody, které rovněž představují aktivní prvek. Tabulka 2 zohledňuje i okolí přechodu a aktivita je zde proto znázorněna jako přechod obklopený dvěma místy.

4.4.3 Události

Dalším poměrně jednoduchým mapováním je mapování počáteční a koncové události. Počáteční událost je transformována jako jedno místo následované přechodem. Toto mapování bylo zvoleno z důvodu zachování pravidel BPMN, které dovoluje modelování datové asociace z počáteční události do datového úložiště. Pro transformaci to tedy znamená modelování dodatečného přechodu, který po svém odpálení může přesunout tokeny do míst reprezentujících datová úložiště. Koncová událost také umožňuje použití datových asociací. Na rozdíl od počáteční události je ale mapována pouze jako místo, protože před koncovou událostí je mapováním ostatních elementů zaručen výskyt přechodu. Díky tomu je možné modelovat datové asociace vstupujících do koncové události reprezentující výběr zdroje při ukončení procesu.

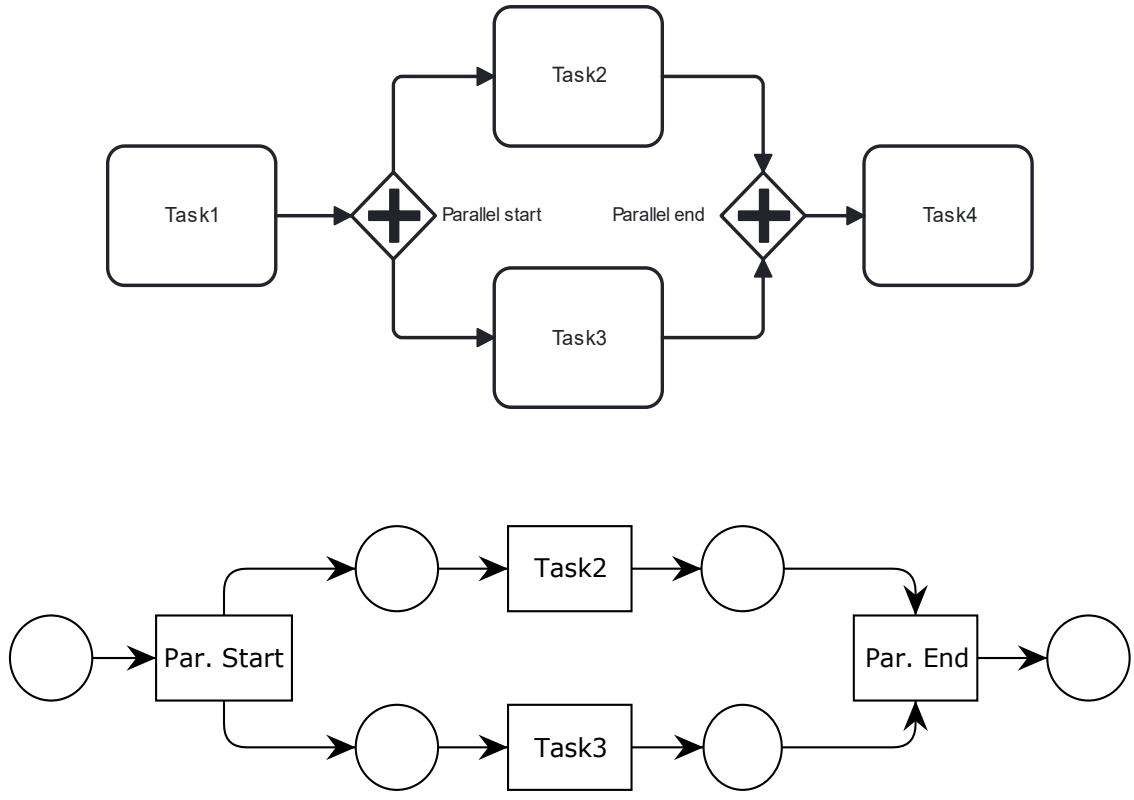
4.4.4 Sekvenční toky a datové asociace

Sekvenční toky a datové asociace jsou mapovány v Petriho sítích na hrany. U datových asociací se jedná o hrany obsahující jednoduchou inskripci popisující počet zdrojů (tokenů) se kterými je tímto datovým tokem manipulováno. Sekvenční toky mohou mít o něco složitější inskripce, zejména pokud se jedná o toky popisující podmínkové větvení. Tyto inskripce jsou popsány v následující kapitole mapování bran a také v kapitole 4.7.5 popisující jejich implementaci.

4.4.5 Brány

Brány jsou nejsložitějšími elementy pro mapování v procesu transformace modelů. V závislosti na typu brány je výsledek mapování zcela odlišný pro každý druh brány. Brány jsou obdobně jako aktivity aktivním prvkem modelu. Aktivní jsou z toho důvodu, že se přímo podílejí na větvení a rozhodování v procesu. Brány samotné jsou tedy mapovány na přechody. Výchozí a vstupní hrany jsou ovšem zcela odlišné pro každý druh, a to zejména z pohledu inskripce hranových výrazů.

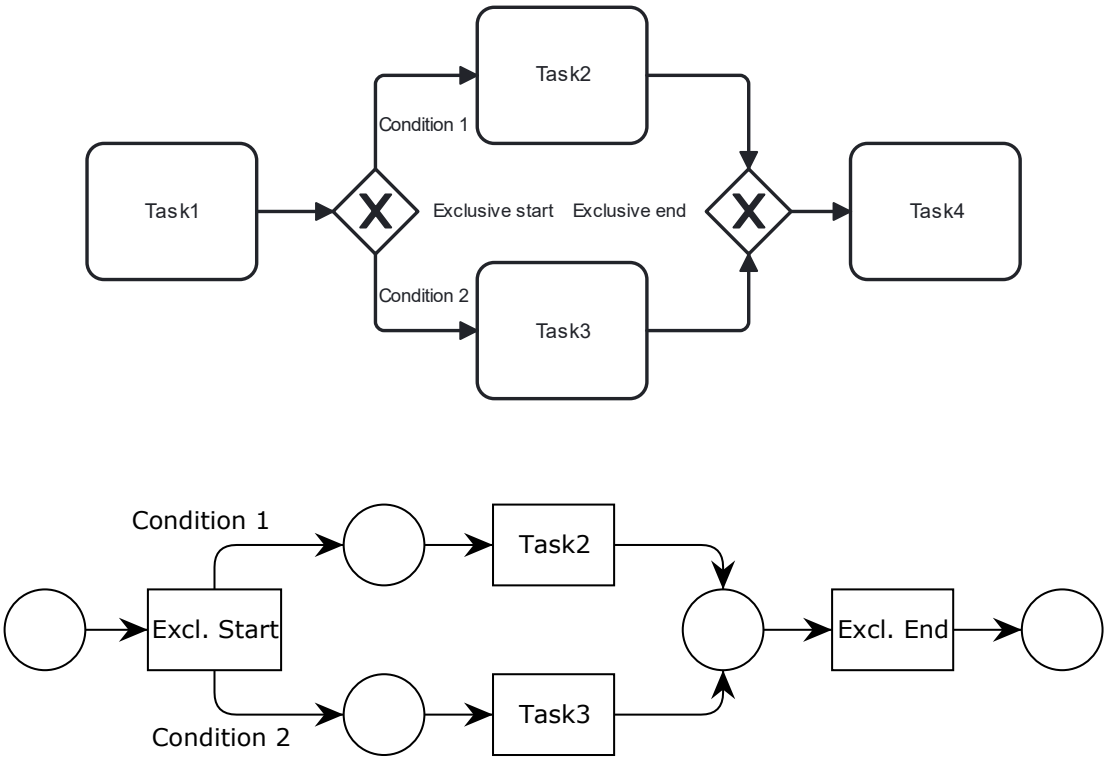
Nejjednodušším z nich je paralelní brána. Ta neobsahuje žádné podmínkové větvení nebo rozhodování. Token je po příchodu k paralelní bráně rozvětven do všech vycházejících hran. To velice ulehčuje mapování této brány z pohledu inskripcí výchozích hran, které obsahují jednoduchý výraz vázané proměnné, se kterou je vstupní token vždy svázán. Tímto mechanismem je tedy zajištěno paralelní spuštění všech vycházejících hran. Spojovací paralelní v podobě přechodu poté čeká na příchod všech tokenů pro zajištění synchronizace. To je umožněno stejným počtem předcházejících míst jako je počet výstupních hran z brány rozdělovací viz Obrázek 19.



Obrázek 19: Mapování paralelních bran do reprezentace v CPN[25]

[25] vlastní zdroj

Druhou branou je brána exkluzivní viz Obrázek 20. Ta už obsahuje podmínkové větvení, se kterým je aktivována pouze jedna vycházející hrana. Tato skutečnost lze zajistit inskripcemi, které jsou vzájemně se vylučující – exkluzivní. Hranové výrazy jsou v tomto případě tedy o něco složitější než u bran paralelních. Rozdíl je také ve spojovací bráně, která nemá ekvivalentní počet předcházejících míst s počtem hran vycházejících z rozdělovací brány. Tento fakt je z důvodu zajištění zmíněné exkluzivity. Ze všech výstupních hran rozdělovací brány je vybrána právě jedna, která je aktivována. Spojovací brána tedy čeká na jeden jediný token.



Obrázek 20: Mapování exkluzivních bran do reprezentace v CPN[26]

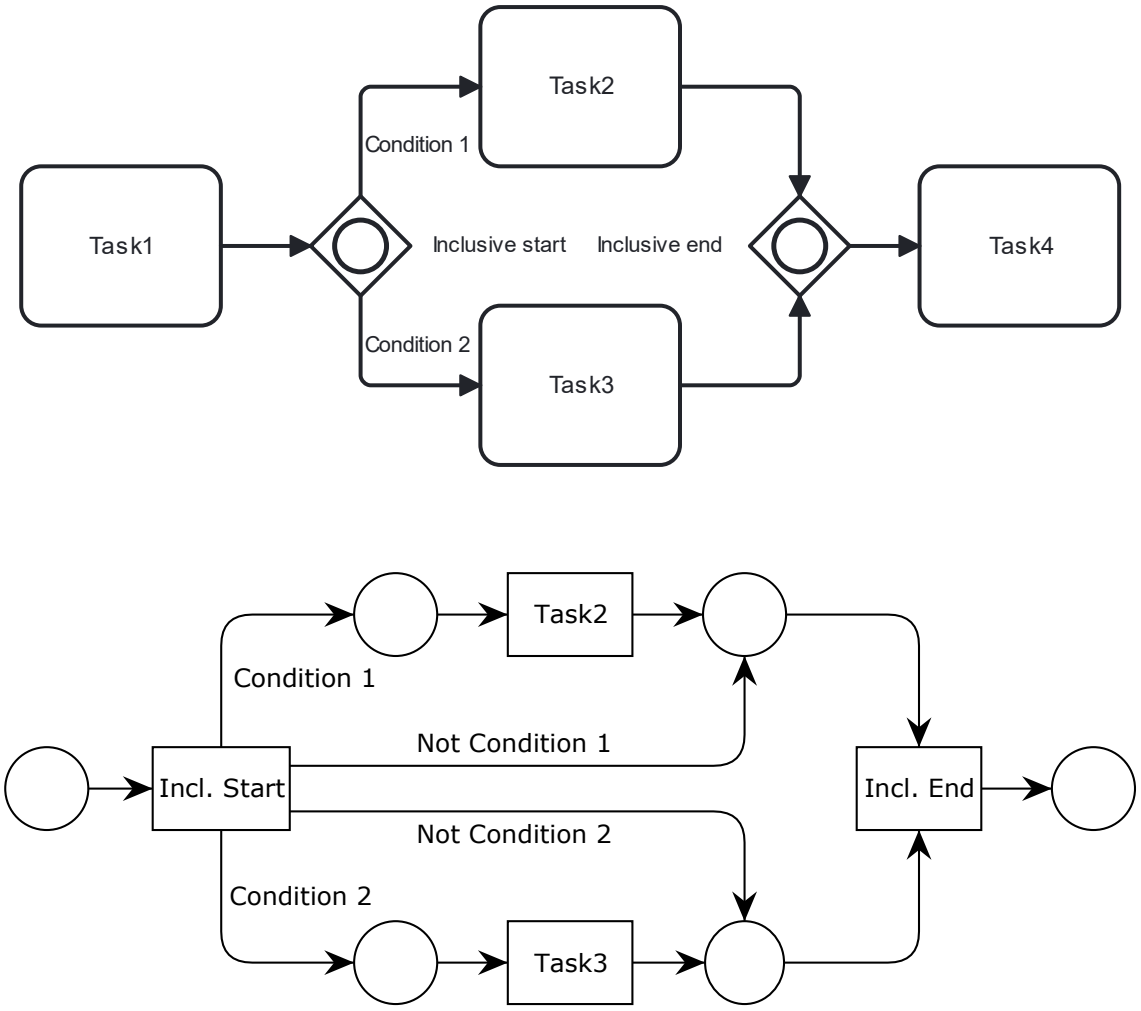
Nejkomplikovanější je mapování inkluzivní brány, která je na pomezí mezi bránou paralelní a exkluzivní. Sekvenční toky vycházející z rozdělovací brány mohou být aktivovány různě. Může být aktivován jeden či více z vycházejících toků. To představuje velice složitou implementaci takové brány. Řešením je zavedení alternativních toků, které obsahují negaci výrazu daného sekvenčního toku. Znamená to tedy, že pro každý vycházející tok je mapována její alternativní hrana, která je aktivována v případě, kdy podmínkový výraz původní hrany je vyhodnocen jako nepravda. Tato alternativní hrana znamená přeskočení celého původního toku. Toto přeskočení je docíleno způsobem, který ukazuje Obrázek 21. Alternativní tok daného

[26] vlastní zdroj

větvení vychází ze stejné rozdělovací brány, kde začíná tok původní, se kterým je spárován. Konec alternativního toku je v místě předcházející spojovací bránu. Těchto míst může být ovšem více. Je vybráno to místo, které se nachází na cestě původního přidruženého toku.

Chování konkrétního příkladu ze zmíněného obrázku je poté následující:

- Pokud je původní podmínka hrany *Condition 1,* resp. *Condition 2* vyhodnocena jako pravda, je vykonání přechodu *Task2*, resp. *Task3* je umožněno.
- Pokud je původní podmínka hrany *Condition 1,* resp. *Condition 2* vyhodnocena jako nepravda, je vybrána alternativní hrana *Not Condition 1,* resp. Not *Condition 2 a* vykonání přechodu *Task2*, resp. *Task3* je přeskočeno.



Obrázek 21: Mapování inkluzivních bran do reprezentace v CPN[27]

[27] vlastní zdroj

4.5 Použité technologie a nástroje pro implementaci

Pro vytvoření webové aplikace byl zvolen objektově orientovaný jazyk C# využívající framework .NET. Konkrétně byl použit .NET ve verzi 7, který je momentálně nejnovější vydanou verzí tohoto frameworku. Pro tvorbu webového uživatelského rozhraní byl zvolen webový framework Blazor, který je součástí ekosystému .NET. Rozhodujícím faktorem pro volbu této kombinace technologií byla jednoduchost použití a možnost tvorby celé webové aplikace pomocí jednoho jediného programovacího jazyka. Navíc umožnila tato kombinace vytvořit progresivní webovou aplikaci (PWA). Webové aplikace typu PWA využívají technologii WebAssembly[28], která poskytuje možnost běhu aplikace v samotném prohlížeči a také umožňují instalaci těchto aplikací přímo do zařízení pro práci i bez připojení k internetu.

Pro vytvoření pluginu validačních pravidel byl využit jazyk JavaScript, který je používán pro definování jednotlivých validačních pravidel.

Vývoj aplikace probíhal ve vývojovém prostředí Microsoft Visual Studio 2022 v edici Community, která je bezplatná a volně přístupná.

4.6 Implementace validačních pravidel

Nástroj Camunda Modeler i webový editor bpmn-js umožňují integrování validačních pravidel do samotného editoru. Pro Camunda Modeler je zapotřebí vložení pluginu obsahující validační pravidla. U webové verze bpmn-js je zapotřebí integrovat rozšíření *bpmn-js-bpmnlint* přímo do editoru při vývoji webové aplikace využívající bpmn-js editor. Tím jsou aktivována validační pravidla při vývoji v editoru a uživatel tím dostává zpětnou vazbu v podobě chybových a varovných hlášek reprezentující výsledek kontroly modelu pomocí těchto pravidel.

Rozšíření *bpmn-js-bpmnlint* je možné modifikovat a upravovat tak pravidla, která se mají při validaci modelu použít. V práci je použita podmnožina již existujících pravidel ale také nově vytvořená pravidla, která je zapotřebí integrovat do rozšíření *bpmn-js-bpmnlint.* Toto rozšíření bylo provedeno vytvořením nového pluginu pravidel s názvem *bpmnlint-plugin-cpn*. Tento plugin byl vytvořen ve formě npm balíčku, který je poté možné použít pro vytvoření modifikovaného validačního modulu pro použití v editoru Camunda Modeler i ve webovém editoru bpmn-js.

Plugin *bpmnlint-plugin-cpn* je velice jednoduchý plugin obsahující nově vytvořená pravidla a konfiguraci těchto pravidel. Konfigurace určuje, jaká pravidla jsou použita a zda při zjištění

[28] přenositelný strojový kód spustitelný na webových stránkách

porušení pravidla je chybová hláška zobrazena jako chyba či pouze jako varování. Plugin obsahuje celkem dvě konfigurace: „all“ a „cpn“. Konfigurace „all“ obsahuje všechna potřebná validační pravidla pro účely této práce a konfigurace „cpn“ pak obsahuje pouze nově vytvořená pravidla.

4.6.1 Dosavadní pravidla

Vývojáři editoru vytvořili set základních validačních pravidel, která validují nejběžnější problémy a kontrolují dodržení doporučených pravidel pro tvorbu přehledných modelů. Pro účely této práce byla přebrána část pravidel obsahující tato pravidla:

- end-event-required
- fake-join
- no-complex-gateway
- no-duplicate-sequence-flows
- no-gateway-join-fork
- no-implicit-split
- single-blank-start-event
- start-event-required
- superfluous-gateway

Názvy pravidel jsou většinou samo popisující kompletní dokumentaci těchto pravidel lze nalézt v oficiální dokumentaci. Pro účely modelování modelů splňující náležitosti nové sady BPMN-light bylo zapotřebí vytvořit dodatečná validační pravidla. Validační pravidla zejména zajišťují restrikci použitých elementů při modelování BPMN modelu.

4.6.2 Nově implementovaná pravidla

Tabulka 3 obsahuje seznam nově implementovaných pravidel, která jsou rozdělena do dvou skupin. Levá část tabulky obsahuje seznam pravidel, která kontrolují přítomnost zakázaných elementů v diagramu. Pravidla jsou vytvořena odděleně pro každou skupinu elementů pro jednodušší využití v jiných projektech. Při použití nepovoleného elementu je uživatel v editoru upozorněn chybovou hláškou. Pravidla pro restrikci elementů jsou opět samo vysvětlující a nejsou zde proto více popsána. V pravé části tabulky jsou pak pravidla pro kontrolu sémantiky a přítomnost doplňujících informací, které jsou zapotřebí pro metodiku PetriBPMN zejména v transformační fázi metodiky. Chybové hlášky jsou v tomto případě rozsáhlejší a popisují přesný problém v diagramu pro ulehčení korekce diagramu.

Tabulka 3: Implementovaná validační pravidla[29]

Pravidla pro restrikci elementů	**Dodatečná pravidla**
no-boundary-event	forking-conditions
no-business-rule-task	no-disconnected
no-call-activity-task	quantity-on-data-associations
no-collaboration	label-required
no-data-object-reference	no-duplicate-flownode-label
no-intermediate-catch-event	
no-intermediate-throw-event	
no-manual-task	
no-receive-task	
no-script-task	
no-send-task	
no-service-task	
no-sub-process-task	
no-transaction-task	
no-user-task	
no-non-blank-start-event	
no-non-blank-end-event	

V některých pravidlech je použita definice formátu zápisu identifikátorů. V CPN ML jazyce je nutné dodržet formát identifikátorů, který může obsahovat pouze alfanumerické znaky, apostrofy a podtržítka. Tato skutečnost je proto ve validačních pravidlech kontrolována pomocí regulárních výrazů. [26]

Prvním dodatečným pravidlem je pravidlo *forking-conditions*, které kontroluje přítomnost a formát podmínkových výrazů u sekvenčních toků, které tuto podmínku musejí mít. Jedná se o sekvenční toky vycházející z podmínkových větvení v podobě exkluzivní a inkluzivní brány. V případě, že všechny vycházející toky nemají potřebný podmínkový výraz, je uživatel na tuto skutečnost upozorněn. V opačném případě nastává kontrola formátu pravidel.

Zápis podmínkových výrazů může mít dva formáty v závislosti na vazbě podmínky. Pokud je podmínka svázána s tokenem putujícím diagramem je tato podmínka atributem tokenu, který je definován na samotném tokenu. Podmínkový výraz má pak formát: *název_atributu = hodnota*, kde *název_atributu a hodnota* mají formát identifikátoru.

Druhou variantou podmínkového výrazu je navázání podmínky na volnou proměnnou. Tím je umožněno oddělení podmínky od tokenu a mít tak možnost modelovat náhodnost v modelu. Tento podmínkový výraz má jiný formát. Položka *název_atributu* se již nenachází v podmínkovém výrazu nýbrž v názvu vstupujícího sekvenčního toku do dané brány a má zde význam názvu vázané proměnné. Vycházející sekvenční toky obsahují pouze podmínkový

[29] vlastní zdroj

výraz obsahující *hodnotu*, kterou může nabývat vázaná proměnná. Obě položky musí opět splňovat podmínky pro formát identifikátoru.

Druhým dodatečným pravidlem je pravidlo *no-disconnected*, které kontroluje elementy, které nemají vstupní a/nebo výstupní sekvenční toky. Pokud se jedná o události, ty musí mít alespoň vstupní (koncová událost) nebo výstupní (počáteční událost) sekvenční toky. Aktivity a brány musí mít jak vstupní, tak i výstupní sekvenční tok.

Třetí validační pravidlo kontrolující datové asociace je *quantity-on-data-associations,* které ověřuje přítomnost rozšiřujícího atributu (*extension properties*) s názvem „quantity“. Dále pak kontroluje formát tohoto atributu, který musí být celé nezáporné číslo reprezentující počet zdrojů spojených s touto asociací.

Čtvrté pravidlo – *label-required* kontroluje formát a přítomnost identifikátorů u elementů v diagramu. Výjimkou jsou paralelní brány a sekvenční toky, které se neangažují v podmínkovém větvení. S tímto pravidlem souvisí páté a poslední pravidlo – *no-duplicate-flownode-label,* které zakazuje přítomnost dvou stejných identifikátorů elementů z důvodu pozdějšího využití názvů elementů pro účely transformace modelů do CPN, které neumožňují více elementů se stejným identifikátorem.

4.6.3 Implementace validačního modulu

V práci je implementován validační modul jak pro webovou, tak i pro desktopovou verzi editoru. Oba tyto moduly jsou převzaty a pouze nastaveny pro použití konfigurace *all* pro aktivaci všech pravidel z vytvořeného pluginu *bpmnlint-plugin-cpn.* Toto nastavení se provádí editací souboru .bpmnlintrc, který obsahuje definice použitých pravidel. Pro oba moduly je tato konfigurace stejná viz Obrázek 22.

```
{
  "extends": [
    "plugin:cpn/all"
  ]
}
```

Obrázek 22: Obsah konfiguračního souboru .bpmnlintrc[30]

[30] vlastní zdroj

4.7 Implementace webové aplikace

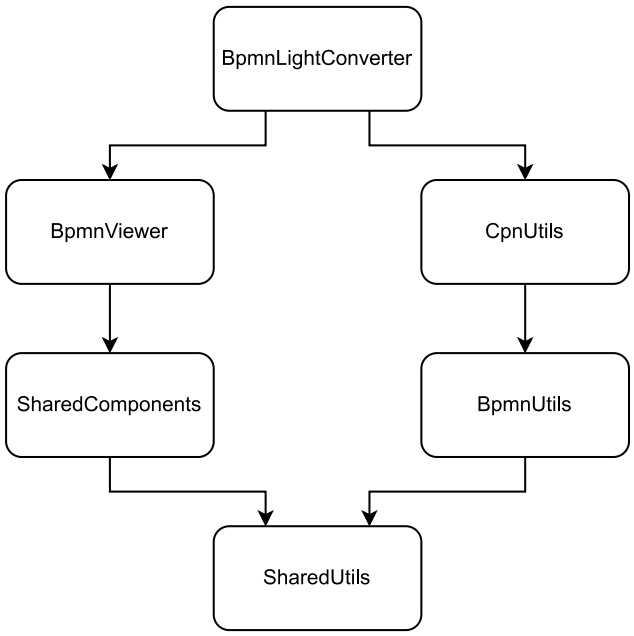
V této kapitole je popsána implementace nově vytvořené aplikace BPMN-light Converter. Nejprve je zde popsána struktura řešení webové aplikace a jednotlivé projekty řešení. Dále jsou zde popsány konkrétní implementace jednotlivých fází nově vytvořené metodiky PetriBPMN.

4.7.1 Struktura projektu

Aplikace BPMN-light Converter je rozdělena na celkem šest projektů. Obrázek 23 ukazuje závislosti mezi jednotlivými projekty v aplikaci.

Stěžejním projektem je projekt BpmnLightConverter, který obsahuje implementaci webového rozhraní samotné aplikace. Součástí webového rozhraní je také projekt BpmnViewer, který implementuje webovou verzi editoru BPMN souborů – bpmn-js. Projekt BpmnLightConverter využívá pro svou funkčnost také všechny ostatní projekty, které jsou implementovány v podobě knihovních projektů obsahující knihovní funkce a webové komponenty. Detailní popis a použití webového rozhraní je uveden v uživatelské příručce, která je přílohou této práce.

V následujících podkapitolách jsou popsány zejména projekty CpnUtils a BpmnUtils, které obstarávají celý proces transformace modelů. Projekty SharedComponents a SharedUtils obsahují komponenty, které jsou sdíleny mezi více projekty.



Obrázek 23: Diagram závislostí projektů[31]

[31] vlastní zdroj

4.7.2 Implementace deserializace a validace vstupních souborů XML

Celý proces metodiky transformace souborů začíná deserializací vstupních souborů XML. Deserializace byla implementována pomocí tříd, které vznikly vygenerováním z definic obsažených v souboru XSD, který je součástí oficiálního standardu BPMN. Vygenerováním vznikly třídy obsahující nutné popisující atributy a vlastnosti, které jsou zapotřebí pro deserializaci a serializaci souborů XML obsahující definice modelů BPMN procesů.

Dalším krokem je validace takto deserializovaných vstupních souborů. Validace je provedena pomocích identických validačních pravidel, která byla popsána v předchozích kapitolách, nyní ovšem implementované v jazyce C#.

Po validaci procesů je zapotřebí převést deserializované třídy do datové struktury graf. Struktura grafu je sestavena pomocí přiřazení sekvenčních toků a datových asociací k přidruženým elementům. Při tomto sestavení jsou navíc aktivity, brány a události opatřeny prefixem s podtržítkem přiřazeným k jejich názvu pro jejich snadné odlišení v konečné Petriho síti a již zmíněné matici sousednosti, která tuto grafovou strukturu reprezentuje.

Prefixy elementů jsou následující:

- T – aktivita
- R – datové úložiště
- S – počáteční událost
- E – koncová událost
- P – paralelní brána
- X – exkluzivní brána
- O – inkluzivní brána

Brány kromě zmíněného prefixu obsahují doplňující prefix určující jejich typ. Rozdělovací brány mají za prefixem přidán prefix BEG a spojovací pak prefix END. Například rozdělovací exkluzivní brána z modelového případu servisu s názvem „Typ“ má plný název ve tvaru „X_BEG_Typ“. Spojovací brány přebírají název brány z párové rozdělovací brány. V tomto konkrétním případě má spojovací brána název „X_END_Typ“.

Projekt BpmnUtils obstarává celou fázi deserializace, validace a transformace do grafové struktury, a také hledání párových bran, které je vysvětleno v následující kapitole. Tento projekt také obsahuje metody pro vytvoření výstupního CSV souboru obsahující matici sousednosti v podobě textového souboru.

4.7.3 Implementace hledání párových bran

Hledání párové rozdělovací brány k bráně spojovací je založeno na průchodu grafové struktury pomocí algoritmu prohledávání do hloubky. Každý procházený element obsahuje pomocnou datovou strukturu zásobník, která obsahuje rozdělovací brány, které byly doposud navštívené při průchodu grafu k danému elementu. Při návštěvě rozdělovací brány, je tato brána přidána do zásobníku a tento zásobník je přiřazen ke každému následovníkovi této brány. Pokud je navštívena brána spojovací, je navíc z tohoto zásobníku poslední brána odebrána. Odebraná brána je přiřazena jako párová brána k nově navštívené spojovací bráně. Tímto mechanismem je tedy nalezena nejbližší rozdělovací brána, která je předchůdcem aktuální spojovací brány.

4.7.4 Implementace transformace elementů

Transformace BPMN procesů začíná transformací jednoho procesu. Pokud je vybráno více procesů pro fúzi procesů, jsou tyto procesy postupně a inkrementálně přidávány k prvnímu transformovanému procesu. Každý z těchto procesů je transformací rozdělen do samostatných stránek v konečné hierarchické barvené síti., které sdílejí globální definice barev, proměnných, funkcí a jiných inskripcí. Z tohoto důvodu mají inskripce a definice, které nejsou zamýšlené ke sdílení mezi procesy, číselný sufix pro oddělení těchto definic.

Každý z procesů musí také mít vlastní definici množiny barev definující token. Tato množina barev je typu záznam, což umožňuje definovat různé atributy a vlastnosti tokenu. Deklarace proto musí být přiřazena až na konci, kdy je celý proces již transformován a jsou zjištěny všechny potřebné atributy tokenu. Před definicí barvy tokenu je proto potřeba průchodu celé grafové struktury procesu a postupně mapovat jednotlivé elementy a s nimi spojená podmínková větvení.

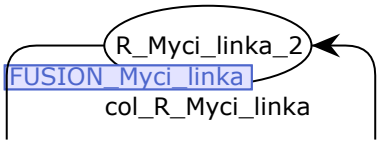
Před popisem transformace je nutné zavést pravidla tvorby definic. Definice v CPN jsou obdobně jako BPMN elementy rozděleny a odlišeny pomocí prefixů následovaných podtržítkem:

- col – množina barev
- col_list – množina barev typu list
- var – proměnná

Zavedením těchto prefixů je zjednodušena orientace v definicích, které mohou být u velkých modelů velice rozsáhlé a nepřehledné.

Prvním krokem transformace elementů je mapování datových úložišť reprezentující rezervoár zdrojů. Tento krok je potřeba provést před průchodem zbytku elementů procesu, které mohou obsahovat datové asociace odkazující se na tato úložiště. Množina barev transformovaných míst je dána názvem zdroje, například zdroj „Myci_linka“ z modelového případu má množinu barev „col_R_Myci_linka“. Tato množina je jednoduchého typu unit reprezentující jednoduchý element. Všechna ostatní místa v modelu mají množinu barev danou typem tokenu onoho procesu, tedy například „col_t1“ pro první z procesů.

Úložiště mohou být reprezentována více místy, tvořící fúzní set. V takovém případě jsou všechna místa jednoho setu opatřena doplňující informací o fúzním setu viz Obrázek 24. Příkladem fúzního místa je již zmíněný zdroj mycí linky z modelového případu.



Obrázek 24: Příklad fúzního místa[32]

Dalším krokem je průchod grafové struktury a postupné mapování elementů grafu. Průchod začíná u elementu počáteční události. Ta jako jediná je mapována na celkem dva elementy – místo a přechod. Pro tyto dva elementy je přidán dodatečný prefix pro jejich rozlišení. Pro nově vzniklé místo je použit prefix P a pro přechod T. Pro počáteční událost s názvem „Start“ je mapování v podobě místa „PS_Start“ následované přechodem „TS_Start“.

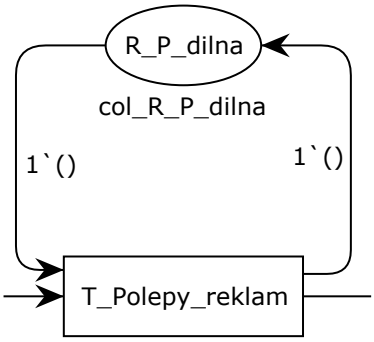
Mapování dalších elementů (aktivit, bran a koncové události) se řídí pravidly, která jsou popsaná v metodice PetriBPMN.

4.7.5 Implementace hranových výrazů

Při mapování elementů je zapotřebí vytvořit hrany spojující místa a přechody. Tyto hrany obsahují textové inskripce, které jsou různé v závislosti na typu toku, který hrana reprezentuje.

Nejjednodušším výrazem, který hrana může obsahovat je kvantita u datových asociací mezi místem zdroje a přechodem aktivity či události. V tomto případě je inskripcí výraz definující počet tokenů odebíraných/přidávaných z/do rezervoáru zdrojů. Tento výraz je ve formátu čísla následovaného znakem zpětného apostrofu zakončený kulatými závorkami. Obrázek 25 obsahuje příklad takové inskripce.

[32] vlastní zdroj



Obrázek 25: Příklad hranového výrazu u rezervoáru zdrojů[33]

Další velice jednoduchým hranovým výrazem je proměnná, která je typu množiny barev tokenu procesu. Výraz v podobě proměnné se vyskytuje na sekvenčních tocích, které nejsou součástí podmínkového větvení včetně vycházejících hran z rozdělovací paralelní brány. Pro první z procesů je hranový výraz „var_t1“, pro druhý „var_t2“ atd.

U podmínkových větvení jsou hranové výrazy nejsložitější. Existují celkem čtyři varianty, které mohou nastat v závislosti na druhu brány a typu vazby podmínky:

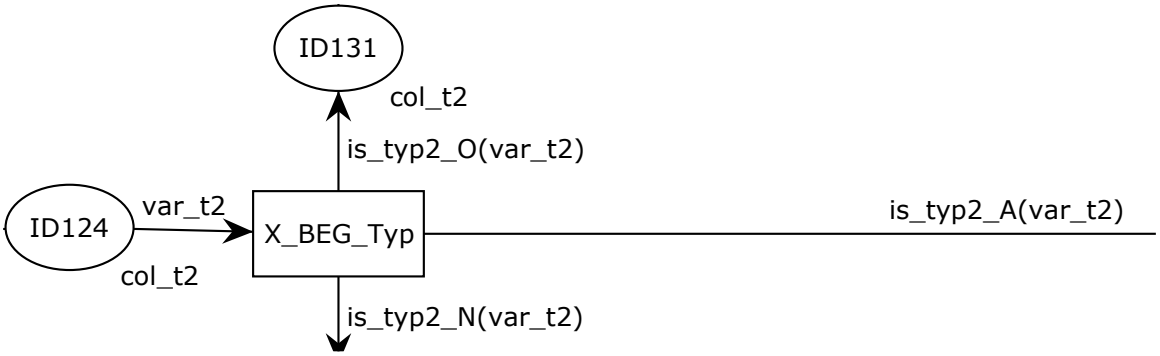
- exkluzivní brána s podmínkou na tokenu
- exkluzivní brána s podmínkou na volné proměnné
- inkluzivní brána s podmínkou na tokenu
- inkluzivní brána s podmínkou na volné proměnné

Všechny tyto varianty jsou implementovány formou vlastních definovaných funkcí pro zkrácení samotného hranového výrazu. Hranový výraz samotný poté obsahuje pouze volání této funkce pro lepší přehlednost modelu.

Prvním krokem pro definici funkce je určení, zda se jedná o hranu vycházející z exkluzivní či inkluzivní brány. Pokud se jedná o bránu exkluzivní, může zkoumaná podmínková proměnná či atribut tokenu nabývat pouze jedné jediné hodnoty v jednom okamžiku. Tento případ se nachází na první exkluzivní bráně v modelovém případu servisu, kterou zobrazuje zblízka také Obrázek 26. Funkce zde má název složený z prefixu „*is*“, následovaný názvem podmínkového atributu (typ), číselným sufixem pořadí procesu (2) a ukončený sufixem hodnoty podmínkového větvení (O, A nebo N). Tato funkce má pouze jeden vstupní parametr, který je typu volné proměnné tokenu pro navázání vstupního tokenu k vyhodnocení podmínkového větvení. Jeden parametr je zde v případě, že je zkoumaná podmínka atributem tokenu.

[33] vlastní zdroj

Obrázek 27 pak obsahuje deklaraci těchto volaných funkcí. V deklaraci je konkrétně porovnána hodnota atributu „typ2“ navázaného tokenu s podmínkovou hodnotou. Pokud je toto porovnání vyhodnoceno jako pravda, je hranou poslán jeden token, který byl navázán do proměnné. V opačném případě touto hranou není vyslán žádný token.



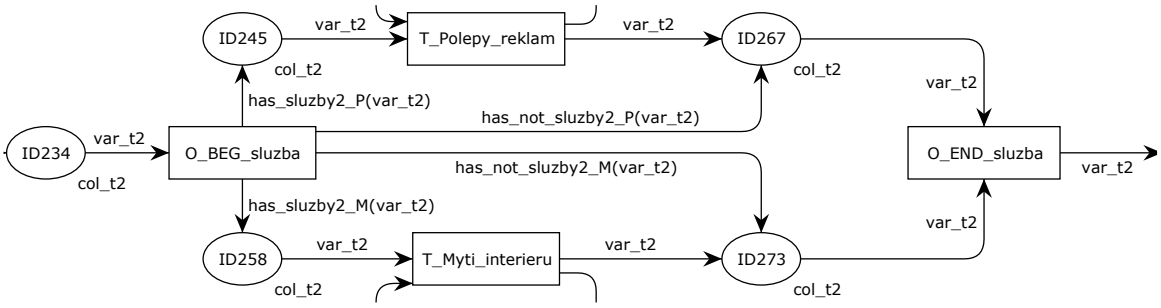
Obrázek 26: Transformovaná exkluzivní brána „Typ“ v modelovém případu servisu[34]

```
fun is_typ2_O(t2: col_t2) = if #typ2 t2 = typ2_O then 1`t2 else empty

fun is_typ2_A(t2: col_t2) = if #typ2 t2 = typ2_A then 1`t2 else empty

fun is_typ2_N(t2: col_t2) = if #typ2 t2 = typ2_N then 1`t2 else empty
```

Obrázek 27: Deklarace funkcí hranových výrazů exkluzivní brány vázaných na tokenu[35]

V případě brány inkluzivní, je potřeba počítat s více variantami. Proměnná či atribut může nabývat jedné či více hodnot v podobě listu, který může uchovávat jednu či více hodnot. Každá z těchto hodnot pak představuje splnitelnou podmínku jedné vycházející hrany. Inkluzivní brána tedy pracuje na principu výčtů a jedna vycházející hrana z inkluzivní brány je spojena s hodnotou výčtu. Příklad takové brány je opět v modelovém případu servisu, kterou v detailu ukazuje Obrázek 28. Oproti předchozímu příkladu je zde použit prefix „*has*“ pro rozlišení mezi funkcí určenou pro exkluzivní a inkluzivní brány. Dalším rozdíl je v samotném těle funkcí. U inkluzivní brány je zapotřebí zjistit, zda list uchovaný v atributu „sluzby2“ obsahuje podmínkovou hodnotu. K tomu je volaná pomocná funkce member, která má dva vstupní parametry. Prvním z nich je podmínková hodnota, u které se zjišťuje, zda je prvkem listu, který je zároveň druhým parametrem funkce viz Obrázek 29. Pokud je tento element prvkem listu, je tento výraz vyhodnocen jako pravda a hranou je umožněn tok tokenu.

[34] vlastní zdroj
[35] vlastní zdroj



Obrázek 28: Transformovaná inkluzivní brána „sluzba“ v modelovém případu servisu[36]

```
fun has_sluzby2_P(t2: col_t2) = if (member(sluzby2_P, #sluzby2 t2)) then
1`t2 else empty

fun has_sluzby2_M(t2: col_t2) = if (member(sluzby2_M, #sluzby2 t2)) then
1`t2 else empty
```

Obrázek 29: Deklarace funkcí hranových výrazů inkluzivní brány vázaných na tokenu[37]

Ať už v případě exkluzivní či inkluzivní brány, podmínková proměnná může být vázaná na volnou proměnnou, která je mimo token. V tomto případě jsou funkce hranových výrazů rozšířeny o dodatečný atribut, který je proměnnou s typem množiny barev tokenu procesu. Tato proměnná je při vyhodnocování hranových výrazů doplněna náhodnou hodnotou a není tak spojena se samotným vstupním tokenem.

4.7.6 Implementace alternativních toků

Již zmíněné alternativní toky u inkluzivních bran jsou implementovány pomocí negace funkcí hranových výrazů. To je docíleno jednoduchým nahrazením prefixu „is“ resp. „has“ za „is_not“ resp. „has_not“. Dále jsou v těle funkcí nahrazeny výrazy „if“ za „if not“ pro negaci celé podmínky.

Tyto alternativní toky jsou zde z důvodu zajištění proveditelnosti přechodu spojovací inkluzivní brány. Konkrétně v případě, který zobrazuje Obrázek 28, se jedná o místa s názvem „ID276“ a „ID273“, která musí obsahovat token pro zajištění proveditelnosti přechodu „O_END_sluzba“. Pokud by například hrana vedoucí z rozdělovací brány do místa „ID245“ byla vyhodnocena jako nepravda, token by touto „větví“ neprošel a místo „ID276“ by neobsahovalo potřebný token pro provedení přechodu „O_END_sluzba“. Proto je zapotřebí mít

[36] vlastní zdroj
[37] vlastní zdroj

alternativní trasu, která zajistí naplnění míst tokeny při vyhodnocení původního toku jako nepravda.

4.7.7 Implementace serializace výstupního XML souboru

Projekt CpnUtils, který implementuje transformaci BPMN modelů do CPN modelů, obstarává také poslední krok metodiky PetriBPMN, kterou je serializace grafové struktury popisující barvenou Petriho síť do konečného XML souboru. Navíc implementuje tvorbu matice sousednosti reprezentující transformovaný model procesu.

Serializace do souboru byla implementována obdobným způsobem jako deserializace BPMN procesu. Nástroj CPN Tools disponuje také specifikací XSD, která definuje strukturu souboru XML používaném v nástroji CPN Tools. Stejně jako v případě deserializace BPMN modelů zde byly vygenerovány potřebné třídy pro práci se soubory XML. Do těchto tříd je transformována grafová struktura Petriho sítě a poté uložena do výstupního souboru.

4.8 Simulace v CPNTools

Modelování procesů a jejich následné simulace hrají klíčovou roli při pochopení a optimalizaci složitých systémů. Simulace poskytují vhled do toho, jak různé součásti systému interagují při sdílení zdrojů při výskytu souběhu procesů soupeřících o tyto zdroje. Díky replikaci reálných scénářů nám simulace umožňují pozorovat a analyzovat dopad těchto souběhů na celkový výkon a chování systému.

Právě pro tyto účely se hodí transformované procesy ve formalismu Petriho sítě zejména z důvodu nedeterministického výběru provádění přechodů, pomocí kterých lze modelovat nedeterministické prvky reálných systémů. Pokud je v jednom okamžiku více přechodů v připraveném stavu, je výběr přechodu k provedení obecně považován za nedeterministický. Volba konkrétního přechodu, závisí na politice plánování implementované v simulačním nebo exekučním stroji.

4.8.1 Runtime

Zavedením času do simulace Petriho sítí poskytují časované sítě přesnější a realističtější přístup k modelování procesů, které zahrnují časově závislé události, zpoždění, termíny a časově závislé chování procesů. Integrace času do simulací Petriho sítí umožňuje zlepšit strategie přidělování zdrojů a plánování. Díky modelování dostupnosti zdrojů a zohlednění událostí závislých na čase umožňují časované sítě vyhodnocovat různé algoritmy plánování a politiky

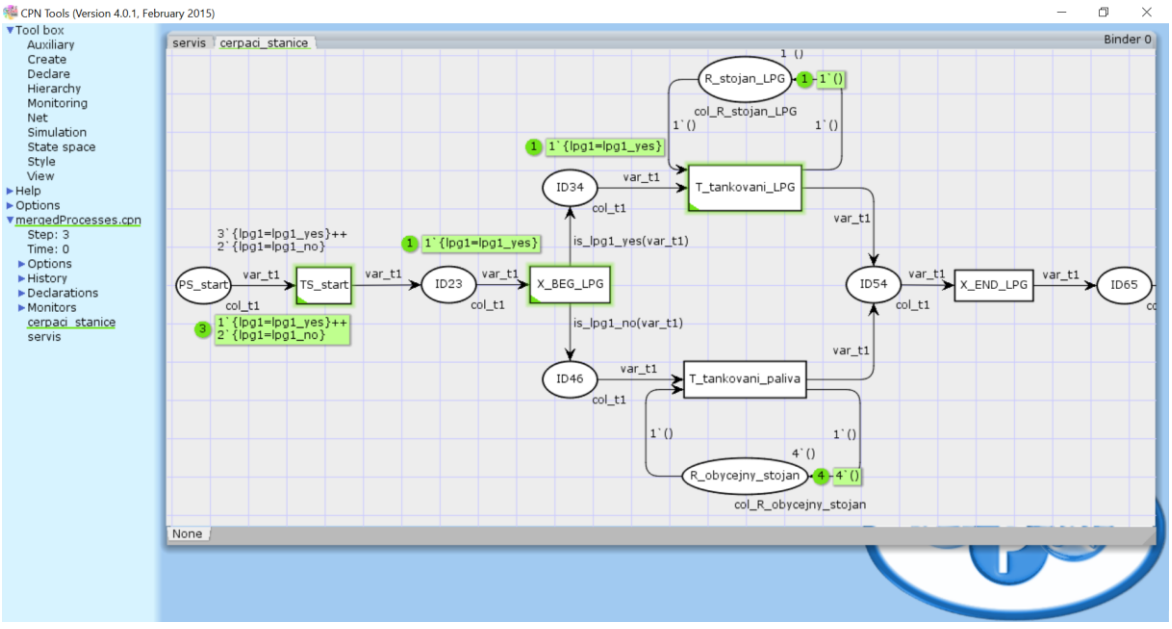
přidělování zdrojů. Tato schopnost podporuje rozhodovací procesy tím, že poskytuje možnost popisu vlivu času na využití zdrojů a výkonnost systému.

Zavedením času je také možné zkoumat a analyzovat výkonnostní metriky za běhu procesu. Tyto metriky mohou zahrnovat propustnost, dobu odezvy, využití prostředků a další relevantní ukazatele výkonnosti. Analýzou těchto ukazatelů je možné vyhodnotit výkonnost systému, identifikovat oblasti neefektivity a optimalizovat model CPN s cílem zvýšit celkovou výkonnost.

Výkonnost systému lze také zkoumat z pohledu teorie front. Pomocí Petriho sítí je možné modelovat fronty, definovat rychlost příchodu, dobu obsluhy a kapacitu fronty a simulovat chování systému front. Umožňují analýzu výkonnosti měřením metrik z teorie front, jako je délka fronty, čekací doba a propustnost.

4.8.2 Sběr statistik v nástroji CPN Tools

Nástroj CPN Tools umožňuje provádění simulací dvojího druhu: interaktivní a automatické. V interaktivní simulaci je průběh simulace manuálně prováděn a řízen uživatelem. Uživatel si může manuálně zvolit, který přechod a s jakým navázáním proměnných bude odpálen. Jednotlivé kroky jsou pak graficky znázorněny v grafickém uživatelském prostředí v reálném čase viz Obrázek 30.



Obrázek 30: Ukázka interaktivní simulace v nástroji CPN Tools[38]

[38] vlastní zdroj

V případě automatické simulace je uživatelem pouze určen hraniční počet provedených kroků případně jiné ukončovací kritérium. Simulátor poté automaticky provede simulaci modelu využitím nedeterministického výběru událostí a přechodů při provádění simulace. Uživateli je poté zobrazen konečný stav simulace. [21]

Při použití automatické simulace umožňuje nástroj CPN Tools také sběr statistik provedených simulací pomocí reportů. Exportovaný report obsahuje průběh simulace kroků provedených simulátorem. Nástroj také umožňuje export reportu výkonnostní analýzy pomocí sběru dat během automatické simulace. Data shromážděná během simulací často zahrnují podrobnosti, jako jsou velikosti front, časové informace a zatížení jednotlivých komponent. Tato data se shromažďují pomocí monitorů, které umožňují uživatelům určit, kdy se mají data shromažďovat v různých krocích automatizovaných simulací a jaká konkrétní data se mají shromažďovat. [21]

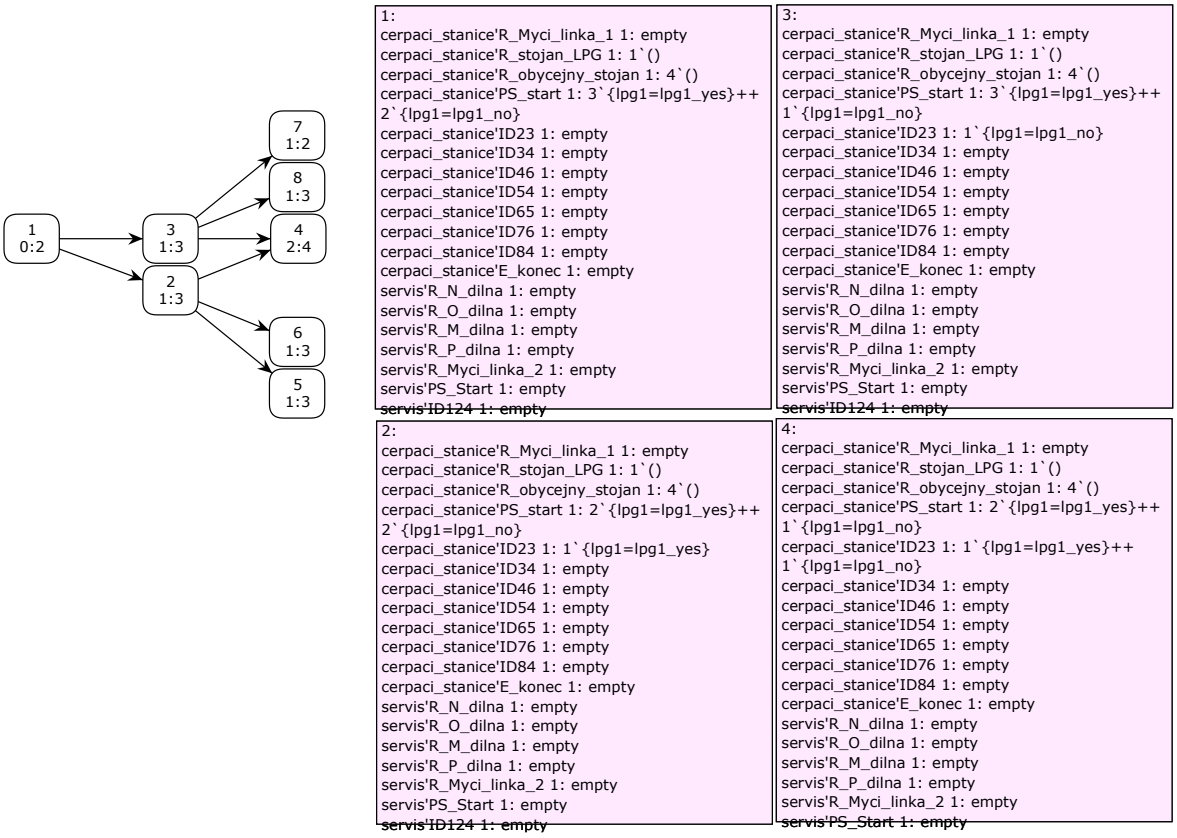
4.8.3 Analýza stavového prostoru

Ještě před samotným provedením simulace je možné využít analýzy stavového prostoru. Analýza poskytuje informaci o stavech, které mohou během procesu nastat. Využívá k tomu generátor stavového prostoru, který generuje graf dosažitelnost. Tento orientovaný graf se skládá z vrcholů reprezentující stav v podobě značení. Hrany pak představují přechody mezi stavy. Tento graf lze spočítat a zkonstruovat zcela automaticky a umožňuje analyzovat různé vlastnosti chování modelu. Příkladem takových vlastností je například minimální a maximální počet tokenů nacházejících se v místech, konečné stavy a dosažitelnost jednotlivých stavů z počátečního značení. Tímto mechanismem je tedy možné zkoumat stavy uváznutí (deadlock a livelock) či stavy vyhladovění. Nástroj CPN Tools navíc poskytuje možnost exportu reportu tohoto druhu. [21], [27]

Analýza stavového prostoru má ovšem i svá úskalí, a to zejména při použití u větších rozsáhlejších modelů. U rozsáhlých modelů, které obsahují mnoho elementů nastává problém exploze stavového prostoru. Stavový prostor systému představuje všechny možné kombinace stavů, kterých může systém během svého provádění dosáhnout. S rostoucím počtem elementů, procesu roste velikost stavového prostoru exponenciálně, takže jej nelze plně prozkoumat nebo analyzovat. Možným řešením je rozdělení modelu do hierarchické Petriho sítě například pomocí zmíněné metody párování bran a rozdělení modelu do více oddílů. [27]

Obrázek 31 obsahuje ukázku části grafu stavového prostoru, který byl vygenerován pro modelový případ čerpací stanice s počátečním značením, které ukazuje Obrázek 30. V obrázku

je obsažena pouze část, jelikož by celý graf popisující celý stavový prostor byl velice rozsáhlý. Celkový počet vrcholů a hran je viditelný v ukázce části reportu, který obsahuje Obrázek 32.



Obrázek 31: Ukázka části grafu stavového prostoru[39]

```
Statistics
------------------------------------------------------------------------

  State Space
     Nodes:  4320
     Arcs:   15792
     Secs:   2
     Status: Full
  Scc Graph
     Nodes:  4320
     Arcs:   15792
     Secs:   1
```

Obrázek 32: Ukázka části reportu analýzy stavového prostoru[40]

[39] vlastní zdroj
[40] vlastní zdroj

ZÁVĚR

Cíle práce, které byly definovány v úvodu a zadání práce byly splněny. Nejprve zde byl popsán teoretický úvod do problematiky modelování podnikových procesů a také představeny oba formalismy BPMN a CPN. Dále zde byla představena sada elementů BPMN-light redukující standard BPMN. Poté zde byla popsána nově vytvořená metodika PetriBPMN zabývající se transformací a fúzí podnikových procesů vytvořených v novém formalismu BPMN-light do cílového formalismu barvené Petriho sítě. V neposlední řadě zde byla popsána implementace webové aplikace, která využívá novou metodiku pro konverzi XML souborů vstupních modelů do výstupního souboru XML. Nakonec zde byly popsány možnosti tvorby analýz a simulací výsledného modelu transformované barvené Petriho sítě.

Navzdory dosaženým výsledkům však existují oblasti, které by bylo možné v budoucím vývoji zlepšit nebo rozšířit. Dosavadní řešení se zabývá pouze modelováním evoluce procesů bez sledování časové složky (je tedy uplatněna metoda Monte Carlo). Pro popis dynamických systému by bylo zapotřebí doplnit model procesu o časovou složku. Zavedením časovaných barvených Petriho sítí by bylo možné provádět simulace a analýzy v plném rozsahu a zkoumat tak dodatečné charakteristiky procesů.

POUŽITÁ LITERATURA

[1] WESKE, Mathias. *Business Process Management: Concepts, Languages, Architectures*. Third edition. Berlin: Springer, 2019. ISBN 978-3-662-59432-2. Dostupné také z: https://doi.org/10.1007/978-3-662-59432-2.

[2] DUMAS, Marlon, Marcello LA ROSA, Jan MENDLING a Hajo A. REIJERS. *Fundamentals of Business Process Management* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013 [cit. 2023-04-13]. ISBN 978-3-642-33142-8. Dostupné z: doi:10.1007/978-3-642-33143-5.

[3] HOLT, Jon. *Pragmatic Guide to Business Process Modelling*. 2nd Edition. London: BCS The Chartered Institute for IT, 2009. ISBN 978-1-906124-12-0.

[4] LAGUNA, Manuel a Johan MARKLUND. *Business Process Modeling, Simulation and Design*. 3rd edition. London: Chapman & Hall, 2018. ISBN 9780429673337.

[5] VAN DER AALST, Wil. *Process Mining: Data Science in Action* [online]. Second Edition. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016 [cit. 2023-04-19]. ISBN 978-3-662-49850-7. Dostupné z: doi:10.1007/978-3-662-49851-4.

[6] MYLER, Harley R. *Fundamentals of Engineering Programming with C and Fortran* [online]. Cambridge: Cambridge University Press, 2012 [cit. 2023-04-16]. ISBN 9780521629508. Dostupné z: doi:10.1017/CBO9781139175029.

[7] CHOUDHARY, Reema, Nauman RIAZ a Jacopo SOLDANI. A business process re-engineering approach to transform business process simulation to BPMN model. *PLOS ONE* [online]. 2023, 18(3) [cit. 2023-04-12]. ISSN 1932-6203. Dostupné z: doi:10.1371/journal.pone.0277217.

[8] CHAPIN, Ned. Flowcharting With the ANSI Standard: A Tutorial. *ACM Computing Surveys* [online]. 1970, **2**(2), 119-146 [cit. 2023-04-17]. ISSN 0360-0300. Dostupné z: doi:10.1145/356566.356570.

[9] ČSN ISO 5807. *Zpracování informací. Dokumentační symboly a konvence pro vývojové diagramy toku dat, programu a systému, síťové diagramy a diagramy zdrojů systému*. Praha: Český normalizační institut, 1996, 28 s. Třídící znak 369011.

[10] ARLOW, Jim, Ila NEUSTADT a Bogdan KISZKA. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2008. ISBN 978-80-251-1503-9.

[11] MAGYAR, Gabor, Gabor KNAPP, Wita WOJTKOWSKI, W. Gregory WOJTKOWSKI a Jože ZUPANČIČ, ed. *Advances in Information Systems Development* [online]. Boston, MA: Springer US, 2007. ISBN 978-0-387-70760-0. Dostupné z: doi:10.1007/978-0-387-70761-7.

[12] MENDLING, Jan. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, XX, 194 [cit. 2023-04-17]. Lecture Notes in Business Information Processing. ISBN 978-3-540-89223-6. ISSN 1865-1348. Dostupné z: doi:10.1007/978-3-540-89224-3.

[13] MENDLING, Jan a Markus NÜTTGENS. EPC markup language (EPML): an XML-based interchange format for event-driven process chains (EPC). *Information Systems and e-Business Management* [online]. 2006, **4**(3), 245-263 [cit. 2023-04-18]. ISSN 1617-9846. Dostupné z: doi:10.1007/s10257-005-0026-1.

[14] VON ROSING, Mark, August-Wilhelm SCHEER a Henrik VON SCHEEL. *The complete business process handbook: body of knowledge from process modeling to BPM*. Waltham, MA: Morgan Kaufmann, 2015. ISBN 9780127999593.

[15] WHITE, Stephen A. a Derek MIERS. *BPMN modeling and reference guide: understanding and using BPMN : develop rigorous yet understandable graphical representations of business processes*. Lighthouse Point: Future Strategies, c2008. ISBN 9780977752720.

[16] DESEL, Jörg a Javier ESPARZA. *Free Choice Petri Nets* [online]. New York: Cambridge University Press, 2009 [cit. 2023-04-25]. ISBN 9780521465199. Dostupné z: doi:10.1017/CBO9780511526558.

[17] LIU, Guanjun. *Petri Nets: Theoretical Models and Analysis Methods for Concurrent Systems* [online]. Singapore: Springer Nature Singapore, 2022 [cit. 2023-04-25]. ISBN 978-981-19-6308-7. Dostupné z: doi:10.1007/978-981-19-6309-4.

[18] *Business Process Model and Notation (BPMN)*. Version 2.0.2. Milford: Object Management Group, 2013. formal/2013-12-09. Dostupné také z: https://www.omg.org/spec/BPMN/2.0.2.

[19] BPMN coverage. *Camunda Platform 8 Docs* [online]. Berlin: Camunda Services, c2023 [cit. 2023-04-29]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/bpmn-coverage/.

[20] REISIG, Wolfgang. *Understanding Petri Nets* [online]. 1. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013 [cit. 2023-05-03]. ISBN 978-3-642-33277-7. Dostupné z: doi:10.1007/978-3-642-33278-4.

[21] JENSEN, Kurt a Lars M. KRISTENSEN. *Coloured Petri Nets: Modelling and Validation of Concurrent Systems* [online]. 1. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009 [cit. 2023-05-05]. ISBN 978-3-642-00283-0. Dostupné z: doi:10.1007/b95112.

[22] . POPOVA-ZEUGMANN, Louchka. *Time and Petri Nets* [online]. 1. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013 [cit. 2023-05-03]. ISBN 978-3-642-41114-4. Dostupné z: doi:10.1007/978-3-642-41115-1.

[23] GIRAULT, Claude a Rüdiger VALK. *Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications* [online]. 1. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003 [cit. 2023-05-05]. ISBN 978-3-642-07447-9. Dostupné z: doi:10.1007/978-3-662-05324-9.

[24] RAMADAN, Mohamed, Hicham ELMONGUI a Riham MANSOUR. BPMN Formalisation using Coloured Petri Nets. In: *ResearchGate* [online]. Berlin: ResearchGate, 2011 [cit. 2023-05-09]. Dostupné z: https://www.researchgate.net/publication/268364122_BPMN_Formalisation_using_Coloured_Petri_Nets.

[25] DECHSUPA, C., W. VATANAWOOD a A. THONGTAK. Transformation of the BPMN Design Model into a Colored Petri Net Using the Partitioning Approach. *IEEE Access* [online]. 2018, **6**, 38421-38436 [cit. 2023-05-10]. ISSN 2169-3536. Dostupné z: doi:10.1109/ACCESS.2018.2853669.

[26] CPN ML identifier. In: *CPN Tools: A tool for editing, simulating, and analyzing Colored Petri nets* [online]. Eindhoven: Eindhoven University of Technology, 2018 [cit. 2023-05-09]. Dostupné z: https://cpntools.org/2018/01/09/cpn-ml-identifier/.

[27] DECHSUPA, C., W. VATANAWOOD a A. THONGTAK. Hierarchical Verification for the BPMN Design Model Using State Space Analysis. *IEEE Access* [online]. 2019, **7**, 16795-16815 [cit. 2023-05-15]. ISSN 2169-3536. Dostupné z: doi:10.1109/ACCESS.2019.2892958

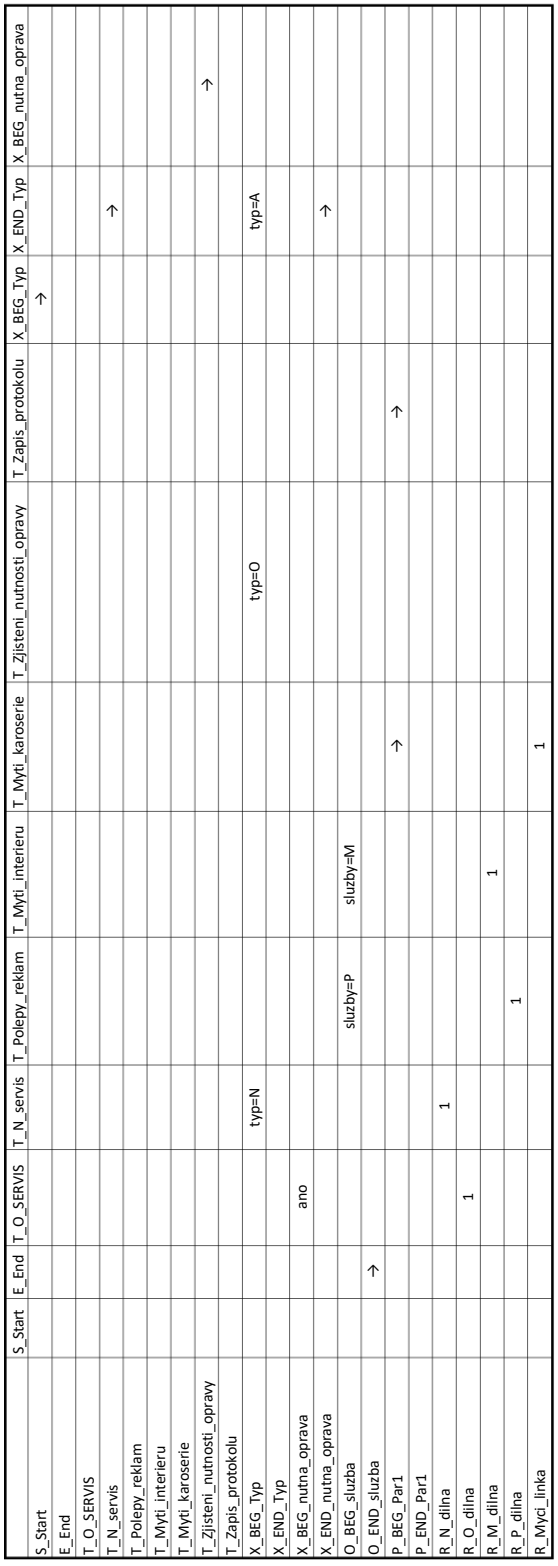
PŘÍLOHY

PŘÍLOHA A – ZDROJOVÉ KÓDY

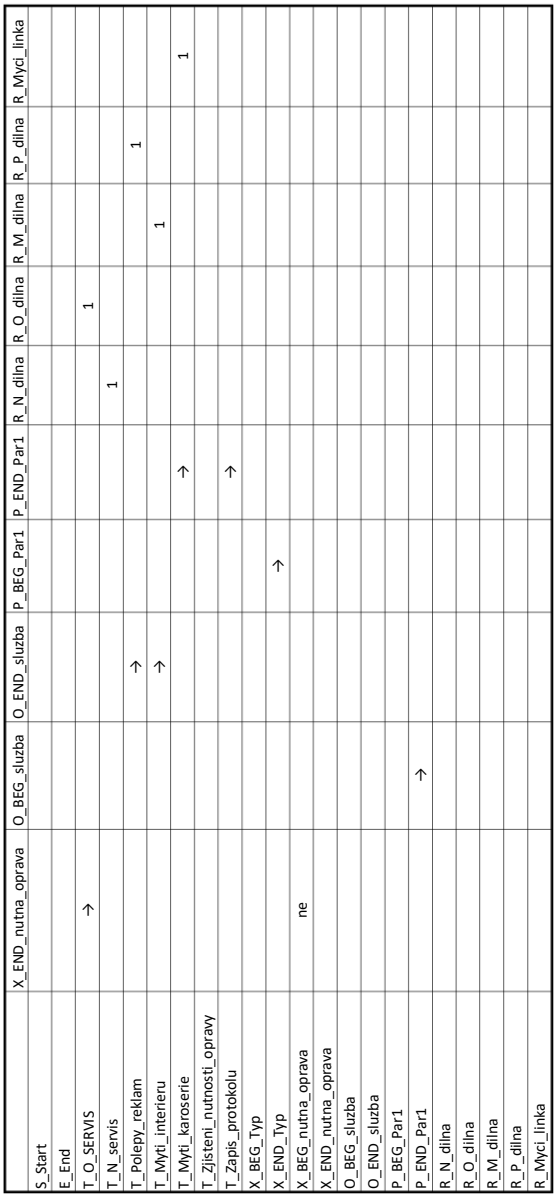
K práci jsou přiloženy zdrojové kódy vypracované aplikace.

PŘÍLOHA B – MATICE SOUSEDNOSTI BPMN MODELU SERVISU

Příloha obsahuje matici sousednosti BPMN modelu modelového případu autoservisu. Vysvětlení symbolů a hodnot matice se nachází v textu diplomové práce v kapitole 4.3.2.



	S_Start	E_End	T_O_SERVIS	T_N_servis	T_Polepy_reklam	T_Myti_interieru	T_Myti_karoserie	T_Zjisteni_nutnosti_opravy	T_Zapis_protokolu	X_BEG_Typ	X_END_Typ	X_BEG_nutna_oprava
S_Start										→		
E_End												
T_O_SERVIS												
T_N_servis											→	
T_Polepy_reklam												
T_Myti_interieru												
T_Myti_karoserie												
T_Zjisteni_nutnosti_opravy												→
T_Zapis_protokolu												
X_BEG_Typ				typ=N				typ=O			typ=A	
X_END_Typ												
X_BEG_nutna_oprava			ano									
X_END_nutna_oprava											→	
O_BEG_sluzba					sluzby=P	sluzby=M						
O_END_sluzba	→											
P_BEG_Par1							→		→			
P_END_Par1												
R_N_dilna				1								
R_O_dilna			1									
R_M_dilna						1						
R_P_dilna					1							
R_Myci_linka							1					



	X_END_nutna_oprava	O_BEG_sluzba	O_END_sluzba	P_BEG_Par1	P_END_Par1	R_N_dilna	R_O_dilna	R_M_dilna	R_P_dilna	R_Myci_linka
S_Start										
E_End										
T_O_SERVIS	→						1			
T_N_servis						1				
T_Polepy_reklam			→						1	
T_Myti_interieru			→					1		
T_Myti_karoserie					→					1
T_Zjisteni_nutnosti_opravy										
T_Zapis_protokolu					→					
X_BEG_Typ				→						
X_END_Typ										
X_BEG_nutna_oprava	ne									
X_END_nutna_oprava										
O_BEG_sluzba										
O_END_sluzba										
P_BEG_Par1										
P_END_Par1		→								
R_N_dilna										
R_O_dilna										
R_M_dilna										
R_P_dilna										
R_Myci_linka										

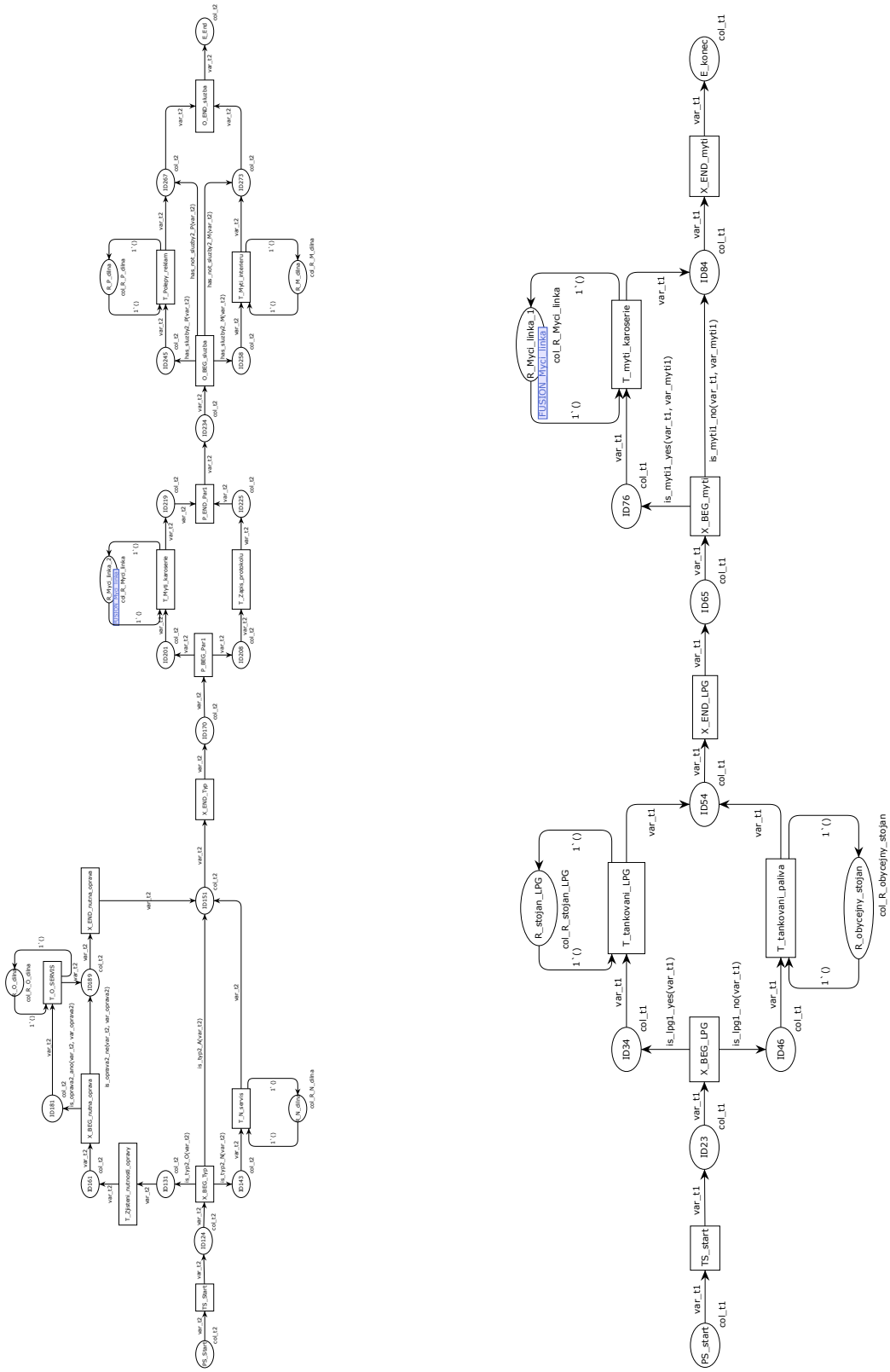
PŘÍLOHA C – MATICE SOUSEDNOSTI BPMN MODELU ČERPACÍ STANICE

Příloha obsahuje matici sousednosti BPMN modelu modelového případu čerpací stanice. Vysvětlení symbolů a hodnot matice se nachází v textu diplomové práce v kapitole 4.3.2.

	S_start	E_konec	T_tankovani_LPG	T_tankovani_paliva	T_myti_karoserie	X_BEG_LPG	X_END_LPG	X_BEG_myti	X_END_myti	R_Myci_linka	R_stojan_LPG	R_obycejny_stojan
S_start						→						
E_konec												
T_tankovani_LPG							→				1	
T_tankovani_paliva							→					1
T_myti_karoserie									→	1		
X_BEG_LPG			lpg=yes	lpg=no								
X_END_LPG								→				
X_BEG_myti					yes				no			
X_END_myti		→										
R_Myci_linka					1							
R_stojan_LPG			1									
R_obycejny_stojan				1								

PŘÍLOHA D – MODELOVÝ PŘÍPAD VE FORMALISMU CPN

Příloha obsahuje obrázek transformovaného modelu kompletního modelového případu popsaného v kapitole 4.2 ve formalismu CPN.



PŘÍLOHA E – UŽIVATELSKÁ PŘÍRUČNA VYTVOŘENÉ APLIKACE

K práci je přiložena uživatelská příručka k vytvořené aplikaci BPMN-light converter.

Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Uživatelský manuál aplikace BPMN-light Converter

Bc. David Le

2023

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

Uživatelský manuál poskytuje návod použití webové aplikace BPMN-light Converter, která je součástí diplomové práce na téma: Transformace BPMN modelů technologických procesů do modelů využívajících barvené Petriho sítě. V manuálu jsou popsané jednotlivé části aplikace a samotný princip používání aplikace. Aplikace umožňuje uživatelům vytvářen či importovat již vytvořené BPMN modely, které poté může konvertovat do souboru XML pro práci v nástroji CPN Tools.

1 APLIKACE

1.1 Nasazení a spuštění aplikace

Aplikace BPMN-light Converter je vytvořena jako statická webová progresivní aplikace, která umožňuje její použití na libovolném zařízení s prohlížečem podporujícím spouštění těchto aplikací.

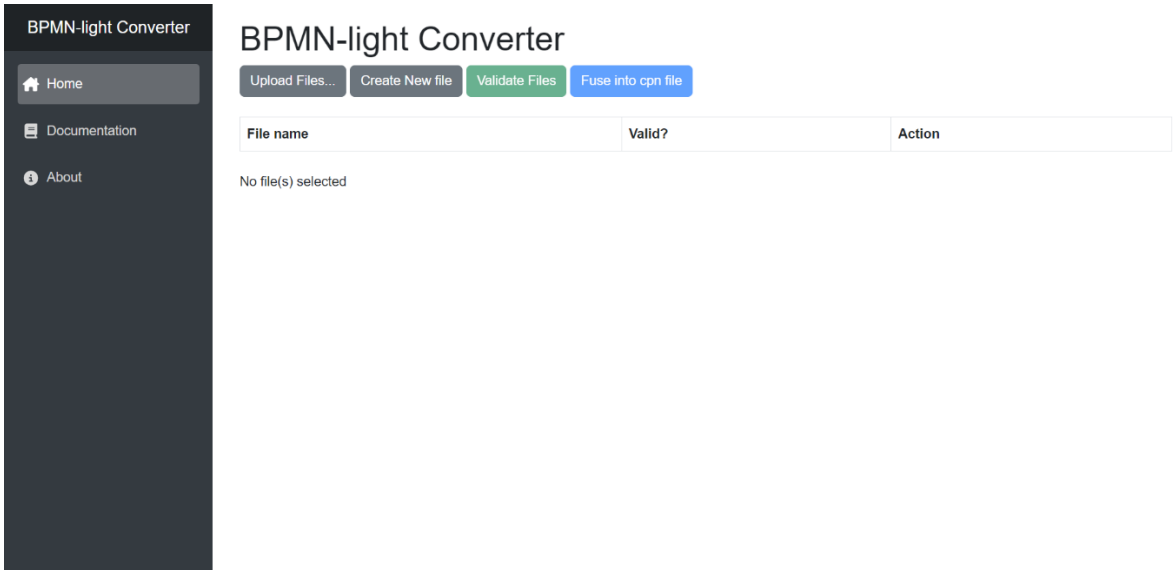
Pro spuštění je ale zapotřebí aplikaci nejprve sestavit a poté nasadit na webový server. Webový server musí podporovat zabezpečený protokol HTTPS (například cloudová platforma Azure Static Web Apps) nebo lze aplikaci spustit pouze lokálně (například pomocí aplikace XAMPP Výsledkem sestavení jsou statické soubory, které je zapotřebí přesunout do kořenového adresáře webu na webovém serveru.

Aplikaci je poté možné použít přímo ve webovém prohlížeči nebo nainstalovat na zařízení pro práci offline bez nutnosti připojení k internetu.

1.2 Části aplikace

Aplikace obsahuje pro navigaci hlavní menu, které je velice jednoduché a skládá se pouze ze tří tlačítek odkazující na různé části aplikace:

- Home – Návrat uživatele na domovskou stránku, kde se zároveň nachází samotná část pro práci se soubory.
- Documentation – odkaz na kopii této uživatelské příručky
- About – stránka se základními informacemi o aplikaci



Obrázek 1: Hlavní stránka aplikace BPMN-light Converter

2 POUŽITÍ APLIKACE

2.1 Tvorba vstupních BPMN-light souborů.

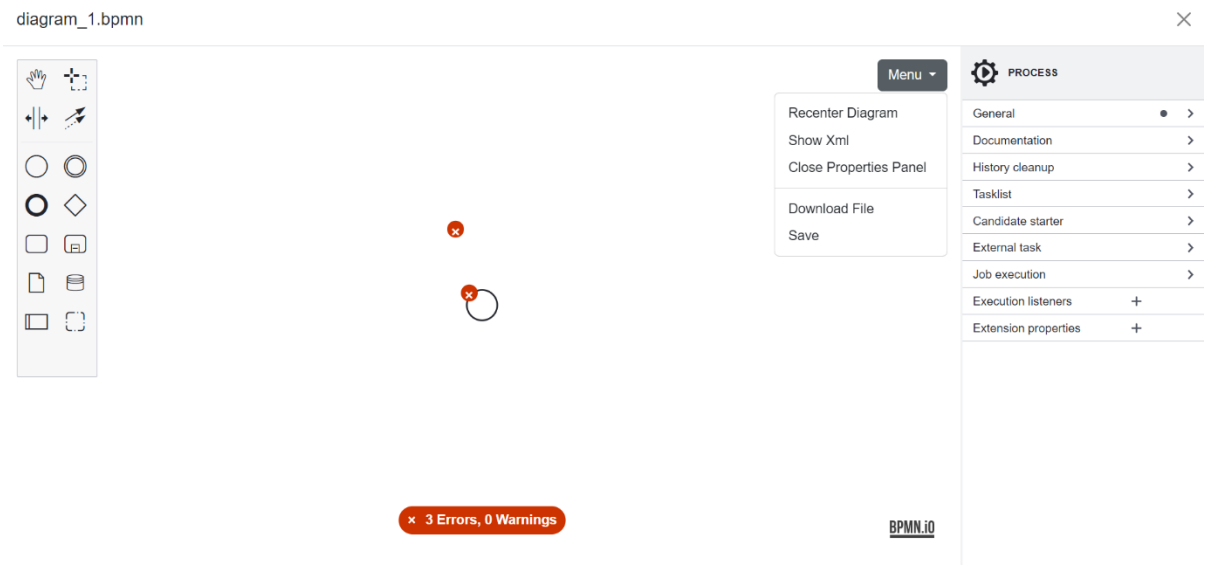
Prvním krokem je vložení výchozích BPMN-light modelů procesů. To může být provedeno dvěma způsoby.

Prvním způsobem je nahrání již vytvořených modelů uložených na lokálním zařízení pomocí dialogového okna pro nahrání souborů. Pro nahrání souborů je zapotřebí kliknout na tlačítko „Upload Files“ a poté vybrat z aktuálního zařízení soubory pro nahrání do webové aplikace.

2.1.1 Integrovaný editor

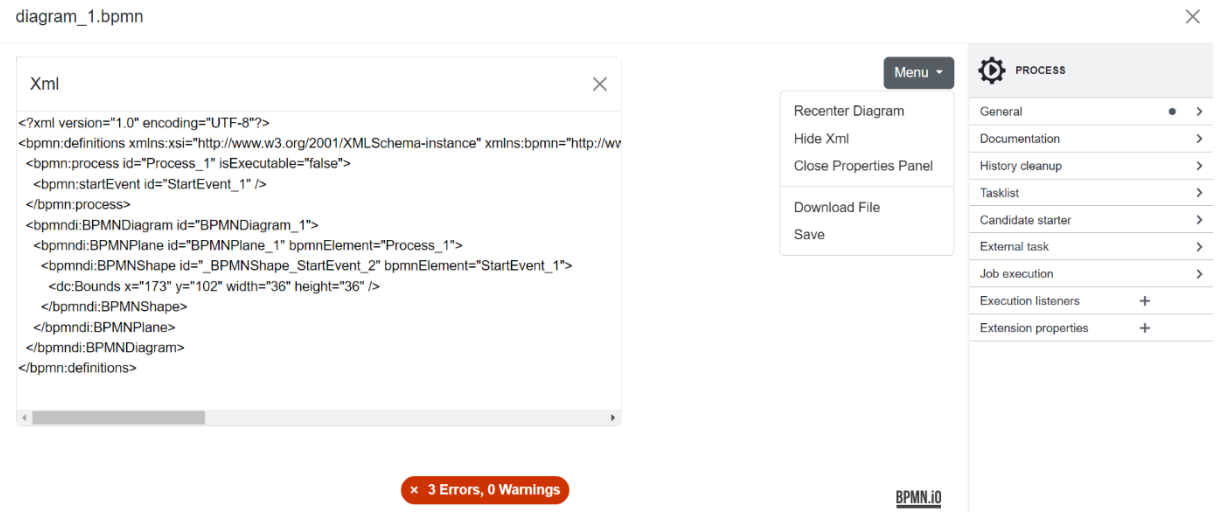
Druhou možností je vytvoření nových BPMN-light modelů. To lze provést přímo v aplikaci kliknutím na tlačítko „Create New File“. Tím se otevře integrovaný editor, pomocí kterého lze vytvářet validované modely ve formalismu BPMN-light. Editor obsahuje všechny potřebné prostředky pro tvorbu modelů. Obrázek 2 obsahuje náhled integrovaného editoru.

Editor obsahuje v levé části paletu prvků BPMN, pro vkládání do diagramu. V pravé části pak obsahuje postranní panel, který umožňuje úpravu názvu a jiných doplňujících vlastností vybraného elementu. Dále je zde rozbalovací tlačítko „Menu“, pomocí kterého lze provádět další akce jako je například re centrování aktuálního diagramu nebo skrytí postranního panelu. Editor také umožňuje uložení aktuálního diagramu v aplikaci kliknutím na položku „Save“ nebo dokonce stažení modelu na lokální zařízení po stisknutí položky „Download File“.



Obrázek 2: Integrovaný editor bpmn-js v aplikaci BPMN-light Converter

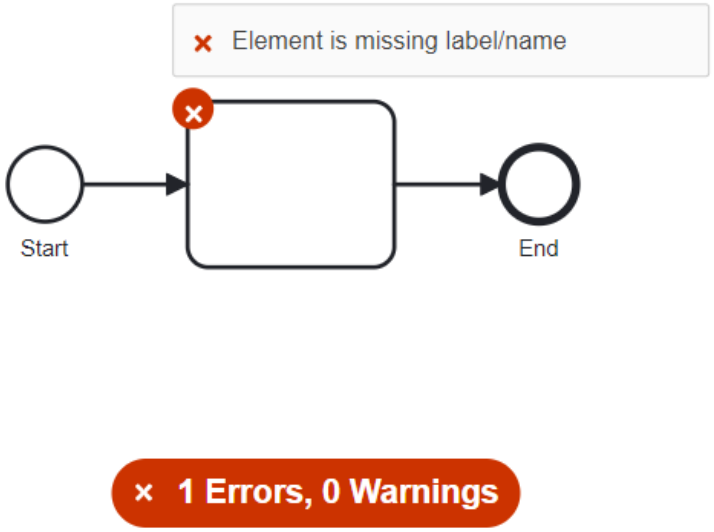
Aktuální obsah souboru, který bude stažen/uložen je možné si zobrazit po kliknutí na položku „Show Xml“. Náhled příkladového modelu ukazuje Obrázek 3.



Obrázek 3: Okno s obsahem souboru aktuálního modelu

Integrovaný editor navíc umožňuje přímo při modelování validaci modelů, který je právě editován. Tím je uživateli poskytnuta zpětná vazba o validaci modelu a o krocích, které je potřeba vykonat pro modelování validního modelu. Po najetí kurzorem na daný element jsou zobrazeny chybové hlášky svázané s daným elementem. Po vyřešení všech problémů, je barva indikátoru počtu chyb na spodní straně obrazovky změněna na zelenou.

Validační pravidla, která je zapotřebí dodržet jsou popsána v textu diplomové práce.

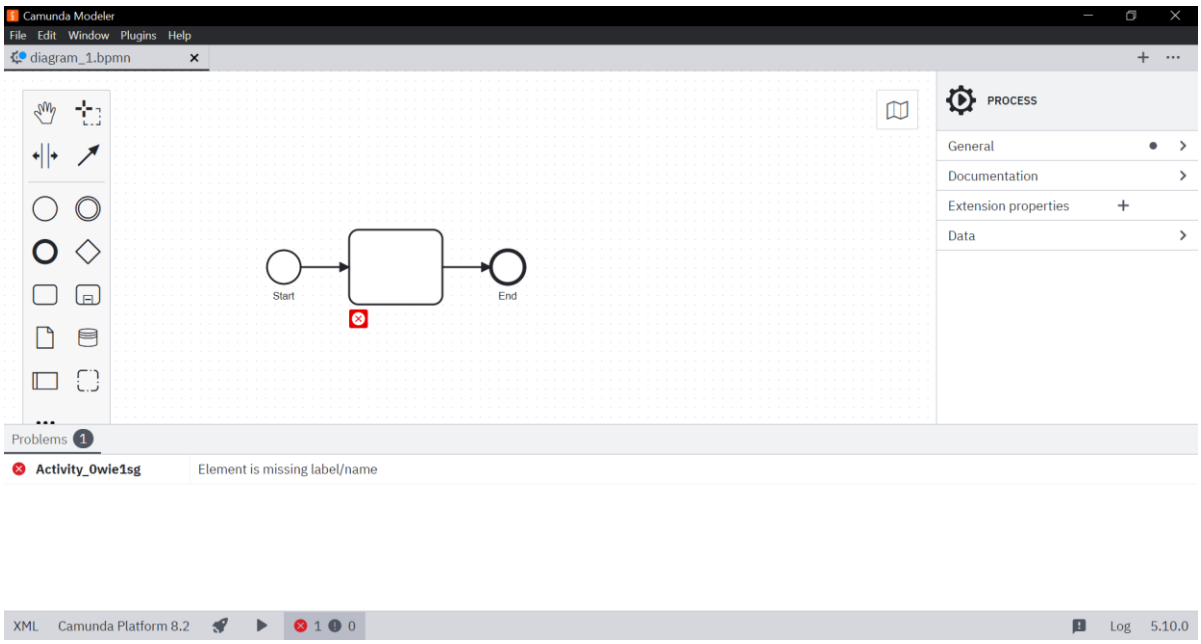


Obrázek 4: Ukázka validace nevalidního modelu v editoru

Obrázek 5: Ukázka validace validního modelu v editoru

2.1.2 Editor Camunda Modeler

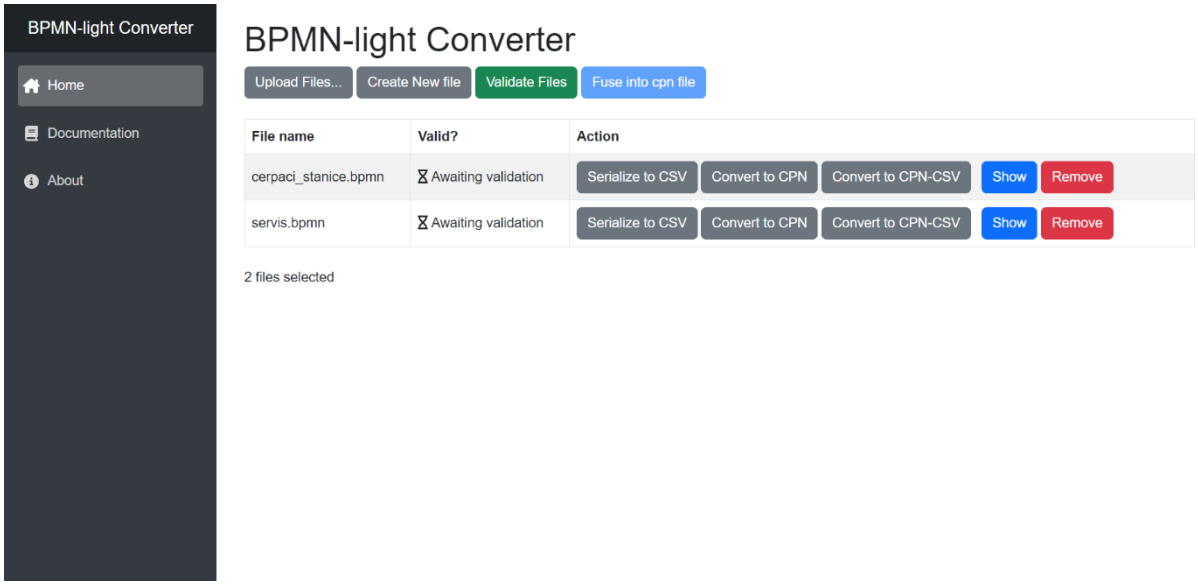
Alternativou k integrovanému editoru je použití desktopové verze editoru Camunda Modeler. Ten ovšem neobsahuje validační pravidla potřebná pro tvorbu validních BPMN-light modelů. Proto je zapotřebí začlenění pluginu s validačními pravidly do aplikace. To se provede zkopírováním složky vytvořeného pluginu „camunda-modeler-custom-linter-rules-plugin“ do složky „resources/plugins“, která se nachází ve složce obsahující editor Camunda Modeler. Plugin je po spuštění aplikace aktivován automaticky a uživatel je obdobně jako v integrovaném editoru informován o validitě modelu.



Obrázek 6: Desktopový editor Camunda Modeler s validačním pluginem

2.2 Validace souborů

Po vytvoření nebo nahrání souborů jsou tyto soubory vypsány v tabulce. Soubory před transformací a dalším zpracováním je zapotřebí validovat. Validace může být spuštěna pro jednotlivý soubor odděleně při výběru jedné z akcí ze sloupce „Action“ (pochpoitelně kromě akce Remove). Může být také spuštěna pro všechny soubory najednou kliknutím na tlačítko „Validate Files“.



Obrázek 7: Nahrané soubory před validací

Po spuštění validace je u jednotlivých souborů zobrazen výsledek validace. Pokud jsou všechny soubory úspěšně validovány a mají stav „Valid“, je poté možné kliknout na tlačítko „Fuse into cpn file“, které provede transformaci a fúzi procesů do jednoho XML souboru obsahující transformovanou barvenou Petriho síť. Pokud není nějaký soubor úspěšně validován, je změněn stav souboru na „Not valid“. Soubor lze poté otevřít v integrovaném editoru pomocí tlačítka „Show“ a vyřešit validační problémy v modelu procesu.

2.3 Transformace souborů

Transformace vybraných souborů je možná pomocí již zmíněného tlačítka „Fuse into cpn file“. Tímto jsou všechny transformovány všechny procesy zobrazené v tabulce. Transformace jednotlivých souborů je možná pomocí tlačítka „Convert to CPN“, které provede transformaci daného souboru BPMN do XML souboru obsahující model CPN.

Jednotlivé soubory je možné také tlačítkem „Serialize to CSV“ serializovat do souboru csv, který obsahuje matici sousednosti elementů výchozích BPMN souborů.

Tlačítko „Convert to CPN-CSV“ obsahuje také matici sousednosti ve formátu textového souboru csv. Matice je ovšem vytvořena z transformované barvené Petriho sítě vybraného BPMN modelu. Formát obou csv souborů je popsán v textu diplomové práce.

2.4 Práce s transformovanými soubory

Transformované soubory s příponou cpn ve formátu XML souboru je možné použít pro práci v nástroji CPN Tools. Tyto soubory není potřeba dále nijak upravovat a lze je přímo otevřít v nástroji CPN Tools.

Transformace modelů je zcela automatizovaná a výsledný model barvené Petriho sítě obsahuje všechny potřebné náležitosti jako jsou deklarace barev a funkcí a definice míst, přechodů a hran. Pro účely simulací a druhotných analýz je ovšem vhodné model doplnit o dodatečné informace jako je například časová složka, počáteční značení a další inskripce, které výchozí BPMN-light model neobsahuje.