Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

Vedoucí: Ing. Martin Střelec, Ph.D.
Vypracoval: Jiří Louda - A14B0522P

Plzeň, 2017

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

Bakalářská práce

Návrh a implementace nástrojů pro zpracování modelů elektrických sítí ve formátu CIM

Vedoucí: Ing. Martin Střelec, Ph.D.
Vypracoval: Jiří Louda - A14B0522P

Plzeň, 2017

PROHLÁŠENÍ

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne *19.5.2017* .....

vlastnoruční podpis

Poděkování

Rád bych upřímně poděkoval zejména vedoucímu bakalářské práce Ing. Martinu Střelcovi, Ph.D. za jeho odborné vedení a ochotu, se kterou se mi věnoval při konzultacích. Rovněž bych rád poděkoval Ing. Richardu Lipkovi, Ph.D. za cenné rady, který mi poskytl zvláště v průběhu zpracovaní softwarového řešení. V neposlední řadě bych rád poděkoval mým rodičům za jejich podporu po celé době studia.

Abstrakt

Cílem bakalářské práce je zpracování modelů elektrické soustavy ve formátu Common Information Model (CIM), který využívá technologie XML - RDF. V teoretické části této bakalářské práce jsou popsány počátky přenosu elektrické energie, liberalizace tržního prostředí, charakteristika chytrých sítí Smart Grid. V navazující části je popsán standard CIM, který je využívá softwarové technologie UML, XML a RDF. Praktická část se zabývá načtením standardu CIM v programovacím jazyku Java, kde je využita technologie parsování Java Architecture for XML Binding (JAXB). Na závěr je uvedeno srovnání výsledků poskytnutých analýzou testovacích dat, které byly vytvořeny společností ENTSO-E.

Klíčová slova

elektrické sítě, přenos elektrické energie, Smart Grid, Common Information Model (CIM), Java Architecture for XML Binding (JAXB), Unified Modeling Language(UML), Extensible Markup Language(XML), Resource Description Framework (RDF)

Abstract

The bachelor's thesis focuses on the processing of power network model captured by the Common Information Model (CIM), which utilizes software technologies XML - RDF. The theoretical part of this bachelor's thesis describes origins of power transfer, liberalization of the market environment and basic characteristics of Smart Grid. The following section outlines the common information model defined by the standard IEC 61970 CIM standard that uses UML, XML, and RDF software. In the practical part, the software tool processing CIM format was implemented, where Java was selected as an effective programming language with numbers of parsing tools. Most promising parsing technology Java Architecture for XML Binding (JAXB) was utilized for CIM processing. The analysis and performance assessment was performed and included into the final part of the thesis. The software tools were validated on data provided by ENTSO-E.

Keywords

power network, power transmission, Smart Grid, Common Information Model (CIM),, Java Architecture for XML Binding (JAXB), Unified Modeling Language(UML), Extensible Markup Language(XML), Resource Description Framework (RDF)

Obsah

Kapitola 1

Úvod

Předkládaná bakalářská práce se zabývá zpracováním dat z oblasti energetiky a klade si za cíl vytvoření nástroje pro vstupně-výstupní operace související se zpracováním nově vznikajícího standardu Common Information Model (CIM). V bakalářské práci jsou diskutovány různé technologie pro zpracování formátu CIM, přičemž výsledné řešení je založeno na technologii JAXB. Vyvinutý nástroj pro zpracování CIM dat představuje rozhraní mezi vstupními daty a informační platformou Power Network Platform (PNP), která byla vytvořena na Katedře kybernetiky (KKY) Fakulty aplikovaných věd (FAV) jako nástroj pro načtení, zpracování a následné vyhodnocení dat z oblasti elektrických sítí.

Práce je rozdělena do sedmi kapitol včetně úvodní. Počáteční kapitoly poskytují technické informace ohledně aplikační oblasti, které jsou důležité k pochopení vyvinutého softwarového řešení popsáného v navazujících kapitolách.

Druhá kapitola se zabývá obecným popisem a historií přenosu elektrické energie.Ve třetí kapitole je popsána liberalizace tržního prostředí s elektřinou v Evropě.Čtvrtá kapitola se zabývá chytrými elektrickými sítěmi Smart Grid a jejím architektonickým modelem(SGAM). V závěru této kapitoly jsou popsány nevýhody dříve používaných formátů k přenosu informací mezi různými.. energetickými společnostmi a naopak výhody nového formátu CIM. Tento formát společně s podpůrnými informačními technologiemi UML, XML a RDF je podrobněji popsán v kapitole páté. Poslední kapitola pojednává o praktické části, která je především složena z implementace tříd potřebných pro načtení EQ, TP a SSH profilů. V závěru této kapitoly je uvedeno následné testování vytvořeného softwarového řešení. V závěrečné kapitole je zhodnocena bakalářská práce a dosažení vytčených cílů. Na úplném konci této bakalářské práce je uveden přehled použité literatury.

Kapitola 2

Počátky přenosu elektrické energie

Elektrická energie je v dnešní době využívána v rozsáhlé míře a zcela samozřejmě. Je využívána k funkci nespočetného množství zařízení v domácnostech, průmyslu a dalších oblastech. Stejně samozřejmě je také považována výroba a přenos elektrické energie. Do tohoto stavu však vedla velice dlouhá a složitá cesta. Historie průmyslového využití výroby a přenosu elektrické energie má počátky v 19. století.

První přenos elektrické energie proveden na kratší vzdálenost se uskutečnil jednoduchým vedením jednosměrného proudu nízkého napětí. Tento přenos realizoval ukrajinský inženýr a vynálezce Fjodor A. Pirockij v roce 1876.

Přenos elektrické energie na větší vzdálenost (175 km) provedl v roce 1891 ruský inženýr Michail Osipovič Dolivo-Dobrovolskij pomocí trojfázového střídavého proudu vysokého napětí mezi městy Laufen a Frankfurt nad Mohanem. První přenosy střídavého proudu velmi vysokého napětí se realizovaly v Evropě začátkem 20. století.[11]

Po skončení druhé světové války bylo zajištění dodávek elektřiny pro spotřebitele v Evropě povinností státu, tedy šlo v podstatě o veřejnou službou. Na tomto pojetí spočívala převažující forma uspořádání energetického odvětví v evropském hospodářství. Téměř v každé evropské zemi byla existence národní energetické společnosti, která byla většinou ve vlastnictví státu. Tato společnost vlastnila monopol na výrobu, distribuci i dodávku elektrické energie. Tento způsob se často promítl na zvýšené ceně.

Hlavní důvody pro zásah státu v tomto odvětví:

1. bezpečnost dodávek
2. složitost této komodity
3. neskladovatelnost
4. rovnováha mezi nabídkou a poptávkou

Kapitola 3

Liberalizace tržního prostředí s elektřinou v Evropě

Idea vedoucí k otevření energetického trhu ve členských zemích Evropské Unie pochází Římské smlouvy, zakládající Evropské hospodářské společenství (EHS), která byla podepsána roku 1957 v Říme. Mezi hlavní cíle této mezinárodní organizace byl volný pohyb osob, zboží, služeb a kapitálu. Liberalizace měla v rámci oddělení distribuční soustavy a prodeje energie přispět k odstranění monopolu, zvýšit počet hráčů na trhu, a tak povzbudit boj o zákazníka a následné snižování cen elektřiny a plynu. Ve skutečnosti šlo o rozdělení národních monopolů a stanovení jasných pravidel, která měla zajistit rovný a nediskriminační přístup k produktům všech společností, které se zabývají maloobchodním prodejem energie konečným uživatelům.

3.1 Etapy liberalizace

Proces liberalizace energetického trhu v Evropské unii byl zahájen na konci 90. let a probíhal ve formě přijímání právních aktů pro zajištění postupného otevírání vnitrostátních trhů ve snaze úspěšně řešit vznikající technické otázky a problémy právní a politické povahy.

3.1.1 První fáze

První fáze liberalizace trhu s energiemi začala v únoru 1997 přijetím směrnice 96/92/ES stanovující podmínku realizace ze strany členských států během následujících dvou let. Cílem první etapy liberalizace bylo vybudovat nejvyšší možnou úroveň konkurence a nejnižší potřebný objem kontroly. Částečné otevření energetického trhu se projevilo v tom, že každý stát mohl sám určit, kdy a které trhy budou otevřeny hospodářské soutěži. Jediným povinným kritériem v tomto ohledu byla realizace alespoň 35% roční spotřeby elektrické energie u koncových uživatelů na volném trhu. Další důležitou složkou první energetické směrnice bylo zahájení procesu oddělení

národních monopolů spojených s výrobou a prodejem elektřiny a plynu od sféry přenosu energie. V České republice v této souvislosti šlo o přijetí energetického zákona č.458/2000 Sb v roce 2000. Co se týče trhu s plynem, v červnu 1998 byla přijata směrnice 98/30/ES, která obsahovala obecná pravidla pro přemístění a skladování zemního plynu, jeho distribuce a spotřeby. Dotýkala se také problematiky organizace a fungování trhu s plynem a přístupu na tento trh.

3.1.2 Druhá fáze

V listopadu 2002 byl přijat druhý energetický balíček, jehož ustanovení byly zaměřeny především na zajištění rovného přístupu k sítím a další rozvoj konkurenčního prostředí. Tento energetický balíček obsahoval obsáhlá ustanovení týkající se další liberalizace energetického sektoru: obsahovala pravidla o pokračování rozdělení byznysu v monopolních společnostech, snížení jejich horizontální koncentrace, zavedení hospodářské soutěže pro velkoobchodní a maloobchodní dodávky, kontrolu přenosu a distribuce energie stejně jako řízení přístupu třetích stran k energetické infrastruktuře. S ohledem na rozdělení podniku bylo v této fázi minimálním požadavkem právní rozdělení provozovatelů přenosových systémů od provozovatelů, kteří prodávají energii koncovým uživatelům, a vytvoření regulačního orgánu na vnitrostátní úrovni v každé zemi.

3.1.3 Třetí fáze

V rámci třetího energetického balíčku, který vstoupil v platnost v červnu 2009, vznikl návrh zřízení celoevropského orgánu pro spolupráci v oblasti energetiky. Jednalo se o první krok na cestě ke vzniku regulátoru jednotného evropského trhu s elektřinou a plynem, bez něhož se fungování otevřeného trhu neobejde. Během třetí etapy došlo k účinnému oddělení výroby a prodeje energie od sítí pro její přenos, došlo k posílení nezávislosti regulačních orgánů a rozvoji spolupráce mezi státy a vnitrostátními regulačními institucemi pro řešení nadnárodních otázek. Dále bylo zajištěno harmonické fungování provozovatelů přenosových soustav energie pro efektivnější přeshraniční obchod a usilovalo se také o transparentnost trhu s energií a ochranu spotřebitele.

3.2 Výsledky liberalizačního procesu

Možnost volit svého dodavatele se pro různé kategorie spotřebitelů objevila se značným časovým odstupem, přičemž jako poslední toto právo obdržely domácnosti. U elektřiny šlo o následující mezníky otevření trhu:

- 2002 – zákazníci s roční spotřebou nad 40 GWh
- 2003 – zákazníci s roční spotřebou nad 9 GWh

- 2004 – zákazníci s průběhovým měřením spotřeby mimo domácností
- 2005 – koneční zákazníci mimo domácností
- 2006 – všichni koneční zákazníci včetně domácností

Po získání možností výběru dodavatele elektrické energie začal spotřebitel hrát aktivní roli. Několik desítek dodavatelských společností se v současné době aktivně podílí na trhu s elektřinou v ČR, což umožňuje spotřebitelům včetně domácností vybrat si z široké nabídky nebo v případě nespokojenosti se službou jednoduše nahradit jednoho dodavatele druhým. Největšími dodavateli v roce 2013 se staly společnosti ČEZ (3 500 000 odběrných míst), E.ON (1 200 000 odběrných míst) a Pražská energetika (PRE) s 700 000 odběrných míst.

Liberalizace přispěla ke zvýšení celkové efektivity energetického trhu, došlo k relativnímu snížení cen pro konečné spotřebitele, navíc pro ně vznikla možnost vstoupit do flexibilních kontraktů a využit nabídku doplňkových služeb dodavatelů. Po liberalizaci došlo k výraznému nárůstu komunikace a výměny informací mezi jednotlivými společnostmi, které se na trhu s elektřinou podílejí [1]. Což mělo za následek potřebu vytvoření vhodných softwarových řešení a vývoj nových informačních technologií pro efektivní výměnu informací. Tento fakt výrazně zdůraznil nástup nových chytrých zařízení integrovaných do tradičních elektrizačních soustav za účelem vytvoření chytrých sítí (Smart Grid).

Kapitola 4

SmartGrid

Výrazem „Smart Grid“, nebo také česky „inteligentní síť“, bývají označovány komunikační a energetické sítě, které umožňují regulovat výrobu a spotřebu elektrické energie v reálném čase. Výrazem „grid“ se označuje elektrická síť složená z přenosových a distribučních vedení, rozvoden, transformátorů a dalších síťových prvků, které dodávají elektřinu ze zdrojů elektrické energie (např. elektrárny) do domácností nebo firem.

Současná elektrická rozvodná síť byla postavena v devadesátých letech 20. století a vylepšena tím, jak technologie prochází každou dekádou. Dnes se skládá z více než 9 200 elektrických generátorů s více než 1 milionem megawattů generační kapacity napojených na více než 300 000 kilometrů přenosových vedení. Smart Grid představuje nebývalou příležitost přesunout energetický průmysl do nové éry spolehlivosti, dostupnosti a účinnosti, které přispějí k našemu hospodářskému a ekologickému růstu. Během přechodného období bude kritické provést testování, zlepšení technologií, vzdělávání spotřebitelů, vývoj norem a předpisů a sdílení informací mezi projekty.

4.1 Charakteristika chytrých sítí

Digitální technologie umožňují obousměrnou komunikaci mezi prvky elektrické sítě či mezi jednotlivými zainteresovanými stranami (tzv. stakeholdery) plnící například regulační úlohy. Smart Grid se skládá z ovládacích prvků, počítačů, automatizace a nových technologií. V tomto případě pracují tyto technologie s elektrickou sítí, aby digitálně reagovaly na rychle se měnící poptávku po elektrické energii. [10]

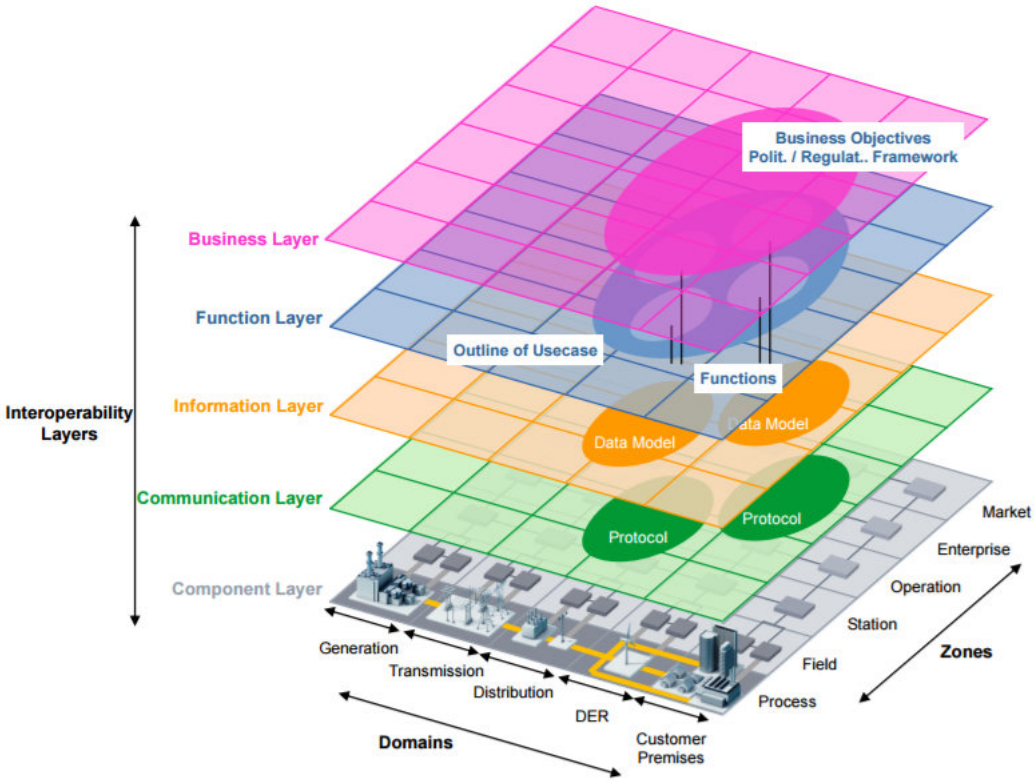
Hlavní výhody

- Efektivnější přenos elektřiny
- Rychlejší obnovení elektřiny po poruchách napájení
- Snížení provozních nákladů a nákladů na správu nástrojů a nakonec nižší náklady na energii pro spotřebitele

- Zvýšená integrace rozsáhlých systémů obnovitelných zdrojů energie
- Lepší integrace systémů pro výrobu energie vlastníka zákazníků, včetně systémů obnovitelné energie
- Zvýšená bezpečnost

4.2 Smart Grid Architecture Model (SGAM)

Energetické systémy a chytrá řešení z pohledu obecně přijatého modelu SGAM jsou rozděleny na dvě části. První část je spojena s fyzikálními zařízeními a elektrickými procesy, zatím co druhá se zabývá správou a zpracování informací. Fyzikální vrstva zachycuje výrobu, distribuci a spotřebu elektrické energie, která je na obrázku vyznačena dimenzí *Domains*. Oproti tomu datová vrstva poskytuje nástroje pro přenos, zpracování a zužitkování informací získaných z fyzické vrstvy. Informační vrstva se dále dělí do čtyř oblastí(*Interoperability Layers*), které budou vysvětleny dále v textu. Entity obsažené v jednotlivých vrstvách mohou být agregovány do složitějších celků, jak je na obrázku naznačeno dimenzí *Zones*. Z pohledu SGAM modelu se tato bakalářská práce zabývá informační podvrstvou (*Information Layer*).



Obrázek 4.1: Smart Grid Model

SGAM model je složen z pěti vrstev [5]:

- **Business Layer**
  Tato vrstva representuje obchodní pohled na výměnu informací týkajících se inteligentních sítí. SGAM je možné použít k mapování regulačních a ekonomických struktur, obchodních modelů zúčastněných stran na trhu.
- **Function Layer**
  Vrstva popisující funkce a služby včetně jejich vztahů z architektonického hlediska. Funkce jsou zastoupeny nezávisle na implementaci v daných aplikacích, systémech. V této vrstvě jsou především obsaženy funkční moduly řešící dílčí problematiky.
- **Information Layer**
  Informační vrstva popisuje informace, které se používají a vyměňují mezi funkcemi, službami a komponentami. Obsahuje informační objekty a základní datové modely.
- **Communication Layer**
  Hlavní funkcí komunikační vrstvy je zprostředkovávat datovou komunikaci mezi instalovanými technologiemi a aplikacemi prostřednictvím komunikačních protokolů a mechanismů pro vzájemnou výměnu informací mezi komponentami.
- **Component Layer**
  Tato vrstva má na starost fyzické rozdělení všech zúčastněných komponent inteligentní sítě. To zahrnuje aplikace, zařízení energetického systému, ochranné zařízení, síťovou infrastrukturu a jakýkoliv počítač.

4.3 Komunikace a datová výměna mezi jednotlivými entitami

Po deregulaci dodávek elektřiny státem pro spotřebitele v Evropě došlo k otevření trhu s energiemi. Energetické sítě začaly být vlastněny různými společnostmi a jejich provoz začal být ovlivňován velkým počtem odlišných zainteresovaných stran/společností (tzv. stakeholderů). Z tohoto důvodu došlo k nárůstu potřeby pravidelného vyměňování dat mezi těmito energetickými společnostmi. Každá z těchto společností používala k datové výměně své vlastní proprietární formáty, které byly velice odlišné. Tyto data popisují stavy objektů energetické sítě, jejich klíčové parametry a jiné potřebné informace ke správě energetických sítí.

Velký počet vlastních formátů používaných různými aplikacemi a odlišnými společnostmi vyžaduje obrovské množství programů k překládání importu a exportu dat z formátu jedné společnosti do formátu společnosti druhé.

Tento velký nárůst počtu aplikací a počtu výměn mezi společnostmi vyvolal požadavek k vytvoření společného formátu, který pokryje všechny oblasti výměny dat v energetické sféře.[8]

4.3.1 UCTE formát

Data jsou obsažena v standardizovaném souboru US ASCII s pevně danou strukturou a bez jakýkoliv řídících znaků. Data nesmí obsahovat diakritické znaménka. Soubor je rozdělen do těchto bloků:

- **Komentáře (C)**
- **Uzly (N)**
- **Vedení (L)**
- **Dvojvinuťový transformátor (T)**
- **Regulace dvojvinuťového transformátoru (R)**
- **Speciální popis dvojvinuťového transformátoru (TT)**
- **Výměna energie (E)**

Každý z těchto bloků je přestaven klíčovým řetězcem složeným ze dvou znaků "##"a znaku výše uvedeného v závorkách. [4]

4.3.2 CDF formát

Úplným názvem Common Data Model ve formátu IEEE. Jde o datový soubor s řádky obsahující maximálně 128 znaků. Řádky jsou seskupeny do sekcí s hlavičkami profilů. Datové položky jsou zadávány ve specifických sloupcích. Prázdné položky nejsou povoleny. V případě potřeby zadat prázdné položky se používá znak nuly. Položky s pohyblivou řádovou čárkou by měly mít explicitní desetinnou čárku. Implicitní desetinné čárky se nepoužívají. Tento formát se hojně využívá v akademické oblasti pro řešení typových úloh z oblasti energetiky. [6]

4.3.3 Omezení současných datových formátů

Výše zmíněné formáty mají omezené technické možnosti. Většinou popisují pouze nejzákladnější části energetické sítě s velmi omezenými možnostmi rozšíření. Navíc jsou data zapsána v pevně daným formátu, který neumožňuje výrazné modifikace či úpravy vedoucí k uspokojivému zachycení nových požadavků pro komunikaci v oblasti elektrických sítí.

4.3.4 Výhody CIM

Oproti technickým omezením výše běžně používaných datových formátů nabízí nově vznikající datový formát Common Information Model (CIM) řadu výhod [13]:

- **Rozsah (Pokrytí)**
  CIM velice podrobně popisuje základní i doplňující části energetické sítě. Proto je v současné podobě vyhovující pro většinu aplikací v oblasti elektrických sítí.

- **Rozšiřitelnost**
  V případě potřeby formát umožňuje vytvářet nové třídy, aby bylo možné zachytit veškeré potřebné části energetické sítě. To zajišťuje konzistenci ve vyšších úrovních modelu. Rozšíření modelů v podobě tříd automaticky dědí obsah z výše uvedených tříd. Velice výhodné je to po stránce časové, jelikož se nemusí vytvářet nový model samostatně od samého začátku.

- **Interoperabilita**
  CIM (resp. norma IEC61970) je nově vznikající standard v oblasti datových výměn, který je postupně adoptován většinou zainteresovaných stran v oblasti elektrických sítí. Jednoduchost při vytváření nových tříd k popisu dalších částí energetické sítě výrazně podporuje vzájemnou spolupráci mezi různými dodavateli.

Kapitola 5

Common Information Model

5.1 Seznámení s CIM

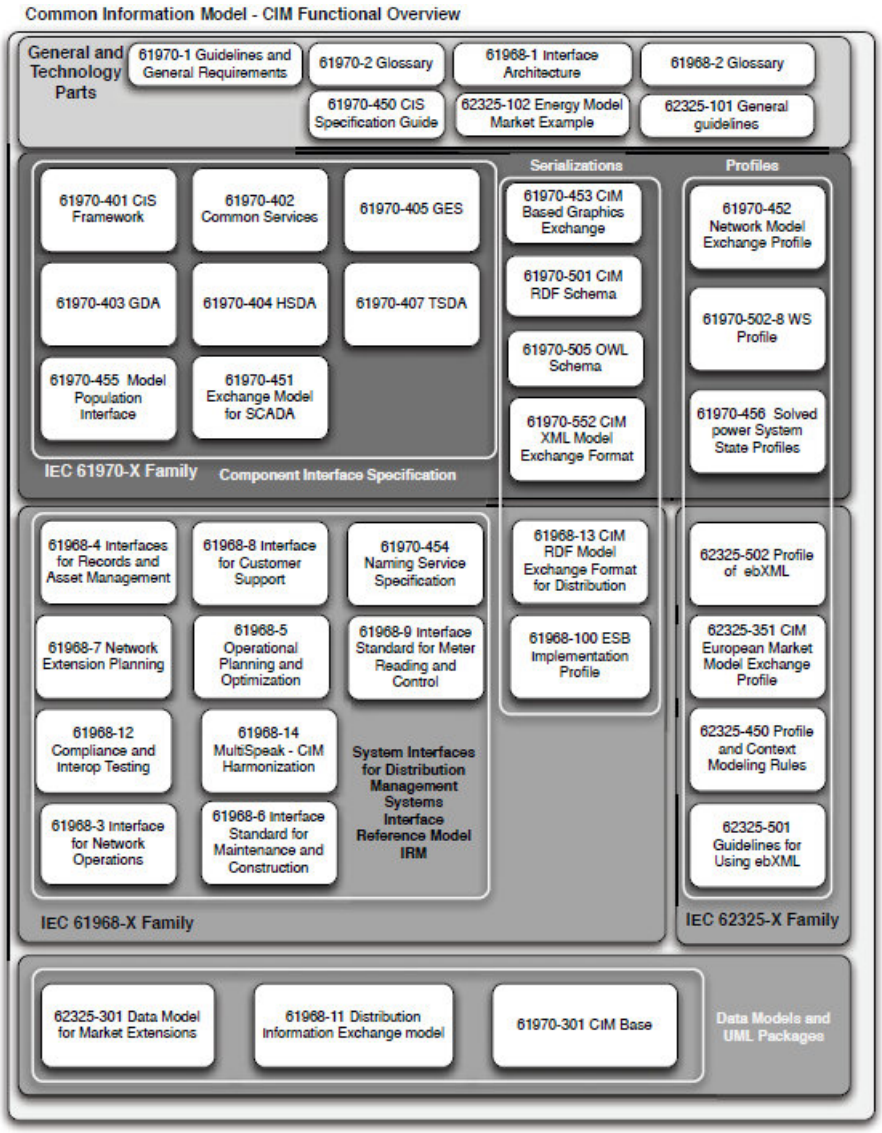
Common Information Model(CIM) je otevřený standard pro reprezentaci komponent energetického systému. Byl vyvinut společností Electric Power Research Institute(EPRI), která má sídlo v Severní Americe.

CIM reprezentuje tyto části po stránce grafického zobrazení(architektura CIM), které je potřebné pro přehled energetické sítě. Architektura CIM je založena na UML jazyku, tedy je objektově orientovaná. Koncepční schéma CIM definuje konkrétní množinu objektů a vztahů mezi nimi. Elementy energetické sítě jsou reprezentovány jako jednotlivé třídy CIM a všechny vztahy mezi nimi jsou dále reprezentovány asociačními vazbami. Podstatnou vlastností je také dědičnost, která umožňuje předávání vlastností z obecnějších elementů do specializujících popisující specifické části energetické sítě. CIM také obsahuje veškeré důležité informace o stavu jednotlivých částí elektrické sítě a poskytuje prostředky pro její aktivní kontrolu a management. [8, 9]

Schéma CIM pokrývá většinu současných prvků elektrické sítě a definuje společný základ pro jejich reprezentaci. Základní schéma CIM je velice snadno rozšiřitelné, díky čemuž je umožněno dodavatelům softwarových technologií implementovat specifické funkce rozšířením obecných elementů definovaných v základním schématu CIM. Formát CIM je tedy velice vhodný pro výměnu veškerých informací o různých částech energetické sítě v rámci jedné nebo více různých společností.

5.2 Popis CIM

Standard CIM definuje a zveřejňuje Distributed Management Task Force (DMTF). Celý formát CIM je velice rozsáhlý, je složen z mnoha bloků. Kompletní rozsah CIM je zobrazen na obrázku níže. V mé bakalářské práci se zabývám pouze blokem v pravém dolním rohu - *61970-301 CIM Base*, který jak z názvu vyplývá představuje elementární části popisu elektrické sítě.



Obrázek 5.1: Přehled všech funkčních částí CIM

Data relevantních částí CIM jsou uložena v příslušných datových souborech, které byly v rámci bakalářské práce zpracovány. [8, 9]

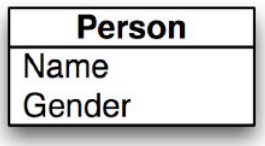
5.3 Základní technologie CIM

Pro specifikaci a uložení dat využívá formát CIM tři základní technologie z oblasti informačních systémů a to UML, XML a RDF, které budou blíže popsány v následujícím textu.

5.3.1 Unified Modeling Language

Jak již bylo uvedeno výše, formát CIM je specifikován pomocí modelovacího jazyka UML. Zatímco úplný popis jazyka UML převyšuje rozsah této práce, budou zde popsány základní principy, které poskytují užitečné prostředky k vizuální reprezentaci hierarchie objektů. Nejdůležitější aspekty UML pro popis CIM jsou popsány níže.

Třídy
Třída je základní konstrukční prvek objektově orientovaného programování sloužící jako předpis pro objekty, definující jejich vlastnosti a metody. Každá třída může mít své vlastní interní atributy a vztahy s jinými třídami. Každá třída může být instancována do libovolného počtu samostatných instancí, známých jako objekty, z nichž každý obsahuje stejný počet a typ atributů a vztahů. Každý tento objekt má ale své vlastní hodnoty.

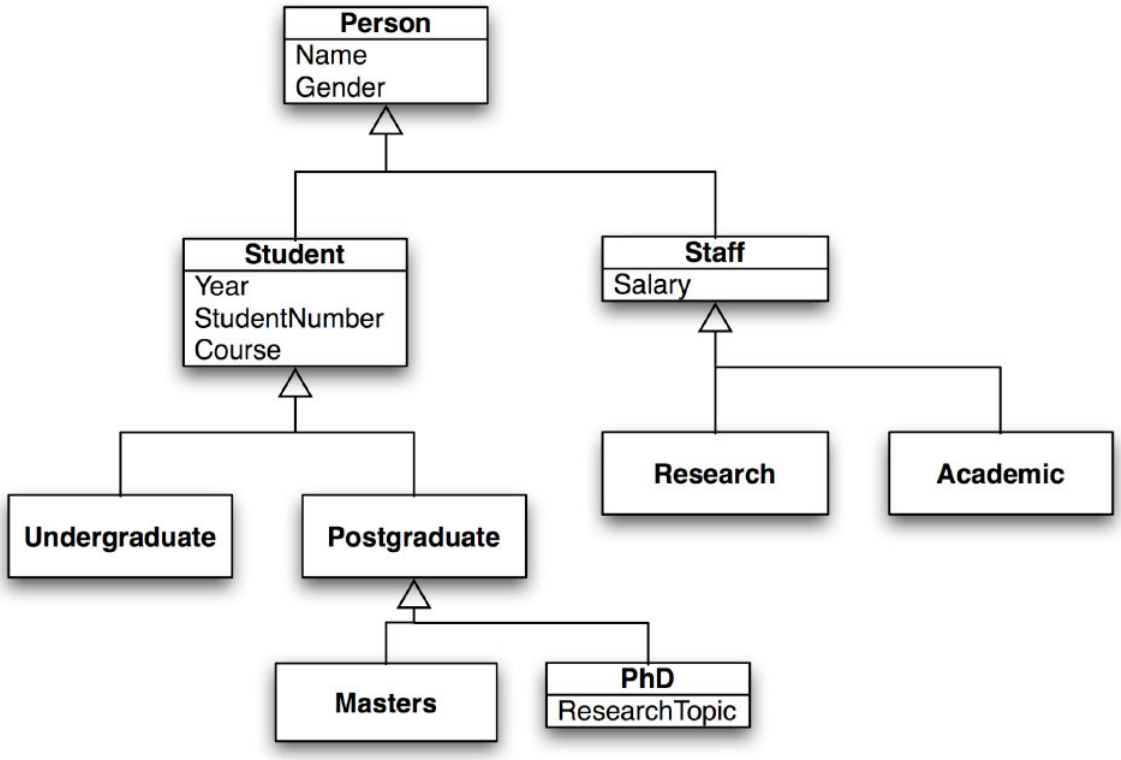


Obrázek 5.2: Příklad třídy

Jednoduchým příkladem třídy je třída Osoba (Person), která je znázorněná na obrázku 5.2, který byl převzat z [8]. Třída Osoba obsahuje dva základní atributy: Name (Jméno) a Gender(Pohlaví).

Příklad použití
Systém, který je tvořen osobami ve na nějaké univerzitě. Pro univerzitu obsahující 5 000 osob by systém vytvořil 5 000 instancí výše zmíněné třídy Osoba. Každá instance by obsahovala hodnotu pro Name(Jméno) a Gender(Pohlaví). V tomto případě by byl typ u obou atributů String(textový řetězec).

Dědičnost
Dědičnost je nástroj objektově orientovaného přístupu, který umožňuje specializovat obecnější třídy tak, že definuje novou (specializovanější) třídu jako podtřídu jiné obecnější třídy. Na základě zvolených pravidel podtřída dědí atributy svého rodiče, ale může také specifikovat své vlastní atributy.



Obrázek 5.3: Příklad dědičnosti

Na obrázku 5.3 je zobrazena reprezentace různých typů členů v rámci univerzitního systému (převzato z [8]). Tento diagram používá standardní symboly UML. Třída Student a Staff (Zaměstnanec) jsou podtřídami třídy Osoba. Student je stále člověk, obsahuje tedy atributy třídy Person(Osoba) Name(Jméno) a Gender(Pohlaví). Tyto dva atributy tedy zdědil z jeho rodičovské třídy Person(Osoba). Třída Student však ještě navíc obsahuje svoje vlastní atributy, kterými jsou Year(Rok studia), StudentNumber (Číslo studenta) a Course(Průběh). Podobně, pokud je někdo Staff (Zaměstnanec), je stále osobou, tedy získává opět atributy Name(Jméno) a Gender(Pohlaví) z jeho rodičovské třídy Person(Osoba). Vlastní atribut obsahuje pouze jeden, kterým je Salary(Plat).

Třída Student obsahuje dvě podtřídy Undergraduate(Bakalářské studium) a Postgraduate(Navazující studium). Třída Postgraduate(Navazující studium) se následně dědí Třídu Masters(Inženýrské/Magisterské studium) a PhD (Doktorandské studium). Třída PhD(Doktorandské studium) tedy obsahuje veškeré zděděné atributy z rodičovských tříd a ještě navíc obsahuje svůj vlastní atribut ResearchTopic(Výzkumné téma).

Třída Staff (Zaměstnanec) má také dvě podtřídy. První z nich je třída Research (Výzkumná) a Academic(Akademická). Obě z těchto tříd dědí atributy Salary(Plat) ze třídy Staff (Zaměstnanec) a Name(Jméno), Gender(Pohlaví) ze třídy Person(Osoba). [8, 9]

5.3.2 XML

Zpracování XML souborů je jednou z hlavních částí softwarového řešení této bakalářské práce. Jak již bylo výše zmíněno, CIM obsahuje veškeré důležité informace o stavu jednotlivých částí popisované energetické sítě. A právě o tato data jsou uložena v textových souborech pomocí technologie XML, která pro zapouzdření dat používá tagové značky. Díky XML-CIM je možné velmi jednoduše zapsat vnoření jednotlivých elementů.

XML je obecný značkovací jazyk, který byl vytvořen konsorciem World Wide Web (W3C). Tento jazyk se používá pro strukturované dokumenty a data. Významné části dokumentu jsou vyznačovány pomocí značek(tagů). V terminologii XML se jednotlivým označeným částem dokumentu říká elementy. Většině elementů odpovídají dva tagy – počáteční a ukončovací. Elementy do sebe mohou být navzájem vnořené a tím dle potřeby zachycovat strukturu informací uložených v dokumentu. Element může také obsahovat vlastní atributy.

Příklad použití tagů k označení elementů:

<tag>...Data...</tag>

Příklad tagu obsahující atributy:

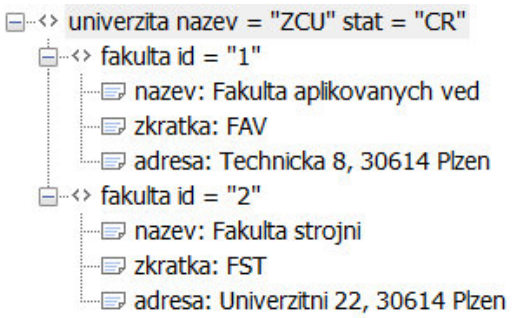
<tag atributJedna = "123" atributDva = "456"/>

```
<?xml version="1.0" encoding="windows-1250"?>
<univerzita nazev = "ZCU" stat = "CR">
  <fakulta id = "1">
    <nazev>Fakulta aplikovanych ved</nazev>
    <zkratka>FAV</zkratka>
    <adresa>Technicka 8, 30614 Plzen</adresa>
  </fakulta>
  <fakulta id = "2">
    <nazev>Fakulta strojni</nazev>
    <zkratka>FST</zkratka>
    <adresa>Univerzitni 22, 30614 Plzen</adresa>
  </fakulta>
</univerzita>
```

Obrázek 5.4: Jednoduchý příklad zápisu informací pomocí jazyka XML

Na obrázku 5.4 je zobrazen příklad jednoduchého souboru XML. Hlavní element(root element) představuje v tomto příkladě `univerzita`, která obsahuje atribut `nazev` a `stat`. Hlavní element `univerzita` má dva potomky(child element), kterými jsou v obou případech element `fakulta`. Tento potomek obsahuje atribut `id`, který je velmi důležitý, jelikož určuje unikátnost elementu `fakulta`. Element `fakulta` dále obsahuje tři potomky, kterými jsou `nazev`, `zkratka`, `adresa`.

Pokud bychom příklad XML z obrázku 5.5 rozepsali do stromu získali bychom tento zápis.



Obrázek 5.5: Stromová struktura předchozího příkladu

Obrázek 5.5 popisuje stromovou strukturu, která byla automaticky vygenerována programem PSPad [3]. Zde je zobrazena vizuální formou struktura XML dokumentu uvedeného výše. [8, 9]

5.3.3 RDF

U základního XML dokumentu neexistuje žádný způsob označení spojení mezi dvěma elementy, které nejsou vzájemně propojeni v rámci nějaké hierarchické struktury (např. nejedná se o dvojici rodič-potomek). Právě tento problém řeší Resource Description Framework (RDF), který rozšiřuje zdrojový dokument tak s ohledem jak na čitelnost tak i na strojové zpracování. Hlavní myšlenkou RDF je k popisovanému zdroji přiřadit výraz ve tvaru podmět – vlastnost – předmět (též subjekt – predikát – objekt). Pro tento výraz se také používá termín trojice. Jinak řečeno RDF popisuje zdroj, ten má nějaké vlastnosti a tyto vlastnosti mají odpovídající hodnoty. Přičemž podmět definuje, o jaký zdroj se jedná, vlastnost určuje jeho charakter a zároveň vyjadřuje vzájemný vztah mezi podmětem a předmětem.

Příklad potřeby použití RDF
Příklad je ukázán an na systému v knihovně obsahující název, jméno autora a pozici sekce a poličky, kde se daná kniha nalézá.

```
<?xml version="1.0" encoding="windows-1250"?>
<knihovna name="Knihovna města Plzně">
  <kniha nazev = "Harry Potter a Kámen mudrců" autor = "J. K. Rowlingová">
    <pozice sekce = "A" police = "2"/>
  </kniha>
  <kniha nazev = "Pán prstenů 1" autor = "J. R. R.Tolkien">
    <pozice sekce = "H" police = "5"/>
  </kniha>
  <kniha nazev = "Harry Potter a Tajemná komnata" autor = "J. K. Rowlingová">
    <pozice sekce = "A" police = "2"/>
  </kniha>
</knihovna>
```

Obrázek 5.6: Ukázka příkladu, kdy je potřeba použít RDF

Každý element `kniha` je obsažen v knihovně jako samostatný záznam. Kdybychom chtěli vytvořit souvislost mezi knihou *Harry Potter a Kámen mudrců* a *Harry Potter a Tajemná komnata*, abychom čtenáři naznačili, že je vhodné si přečíst nejprve první díl *Harry Potter a Kámen mudrců* než začínat druhým dílem, neexistuje standardní způsob, jak to udělat pomocí základních XML konstrukcí. Resource Document Framework (RDF) je schéma XML, které vyřeší výše zmíněný požadavek, jelikož je schopné definovat vztahy mezi XML uzly. Každému prvku přiřadí jedinečný atribut ID, který obsahuje prefix rdf. Tento prefix je definován pomocí namespace: *xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"* vždy na začátku RDF XML souboru. Přidání atributu *resource*(zdroj) elementu umožní vytvoření odkazu mezi elementy obsahující stejné ID.

Příklad použití RDF je ukázán na stejném příkladu jako na obrázku 5.6. U níže uvedeného příkladu knihovny je přidělen nový atribut ID, který má prefix *rdf*. K rozlišení elementů a atributů knihovny je potřeba vytvořit nový prefix *lib*, kterým označíme všechny ostatní elementy a atributy. V předchozím příkladě základní XML elementy žádný prefix neměly. Strukturu RDF souboru včetně nově definovaného prefixu specifikují tzv. jmenné prostory (namespace)[1]. Další změnou od předchozího příkladu je to, že byl přidán nový element `odkaz`, který obsahuje atribut `resource`. Tento atribut odkazuje na jiný element v dokumentu pomocí odkazu na jeho ID. Tedy na obrázku 5.7 kniha *Harry Potter a Kámen mudrců* odkazuje na knihu *Harry Potter a Tajemná komnata* pomocí atributu *resource*, který je roven *#_zaznam3*, což je *ID* knihy *Harry Potter a Tajemná komnata*.

Hlavnímu elementu(root element) RDF je přidán s atribut xmlns pro označení namespace a prefix.

```
<?xml version="1.0" encoding="windows-1250"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:lib="http://www.strath.ac.uk/libraries/2006/library-schema#">
  <lib:knihovna lib:name="Knihovna města Plzně">
    <lib:kniha lib:nazev = "Harry Potter a Kámen mudrců" lib:autor = "J. K. Rowlingová" rdf:ID="_zaznam1">
      <lib:pozice lib:sekce = "A" lib:police = "2"/>
      <lib:odkaz rdf:resource="#_zaznam3"/>
    </lib:kniha>
    <lib:kniha lib:nazev = "Pán prstenů 1" lib:autor = "J. R. R.Tolkien" rdf:ID="_zaznam2">
      <lib:pozice lib:sekce = "H" lib:police = "5"/>
    </lib:kniha>
    <lib:kniha lib:nazev = "Harry Potter a Tajemná komnata" lib:autor = "J. K. Rowlingová" rdf:ID="_zaznam3">
      <lib:pozice lib:sekce = "A" lib:police = "2"/>
      <lib:odkaz rdf:resource="#_zaznam1"/>
    </lib:kniha>
  </lib:knihovna>
</rdf:RDF>
```

Obrázek 5.7: Ukázka příkladu použití RDF

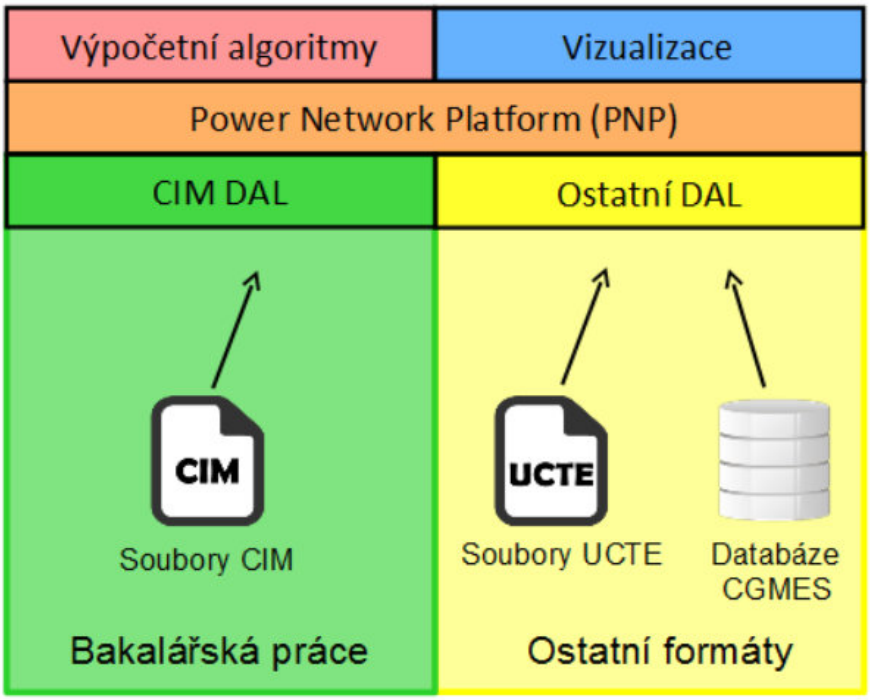
Výše zmíněný model RDF a specifické jazyky XML a UML jsou stěžejními informačními technologiemi pro práci s daty ve formátu CIM. [8, 9]

[1] umístěné na *http://www.strath.ac.uk/libraries/2006/library-schema#*

Kapitola 6

Softwarové řešení pro zpracování CIM

Formát CIM se stává novým standardem pro výměnu informací mezi rozdílnými systémy v oblasti energetiky. Laboratoř pokročilých energetických systémů (LAPS) Západočeské univerzity dlouhodobě vyvíjí pokročilé výpočetní nástroje a metody pro oblast přenosových a distribučních elektrických soustav. Pro urychlení vývojového cyklu a zvýšení obecnosti vyvíjených aplikací byla vyvinuta platforma Power Network Platform (PNP), která definuje informační model elektrické sítě a jejich prvků pro účely strojového zpracování. Tato platforma představuje unifikované rozhraní mezi jednotlivými vstupními datovými formáty jednotlivých dodavatelů a vyvíjenými nástroji a metodami v rámci LAPS. Tento přístup výrazně zvyšuje možnost využití vyvíjených nástrojů pro různé aplikační oblasti. V rámci bakalářské práce byly vyvinuty softwarové nástroje pro zpracování vybraných vstupních CIM souborů pro potřeby vytvoření informačního modelu v PNP platformě. Na obrázku 6.1 je zobrazeno zjednodušené schéma architektury zachycující vztah mezi PNP platformu, vstupně-výstupními vrstvami a vyvíjenými nástroji.

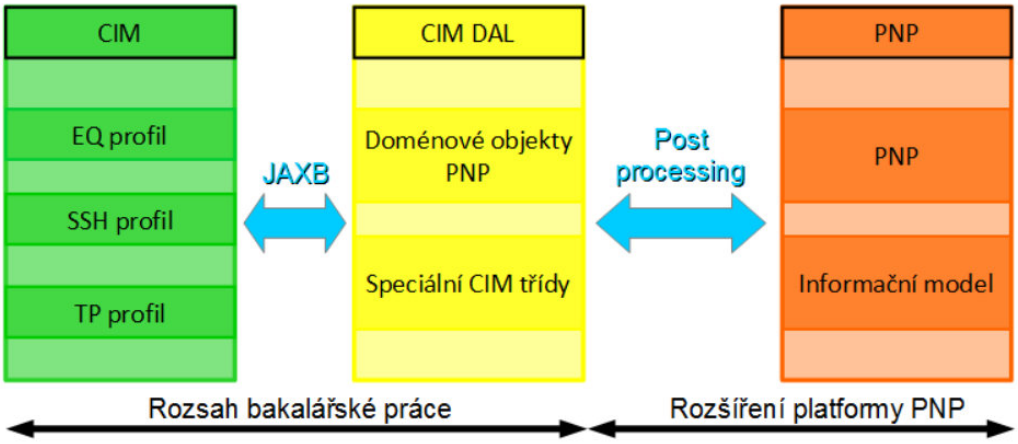


Obrázek 6.1: Zjednodušené schéma architektury

Nejnižší vrstvu tvoří jednotlivé datové zdroje, kde jedním z nich je CIM. Pomocí vstupně-výstupní vrstvy (DAL - Data Access Layer) jsou data zpracována do informačního modelu platformy PNP, který je tvořen datovými objekty a grafovými strukturami zachycující vlastnosti prvků sítě, vztahy mezi nimi a stav elektrické sítě. Na jednotný informační model PNP platformy navazují výpočetní algoritmy (např. Load Flow, estimace stavu a jiné), které na základě uložených informací provádějí specializované výpočty. Pomocí vizualizačních nástroje se může dále informační model PNP platformy s vypočítanými veličinami graficky zobrazit ve zvoleném uživatelském prostředí. Tato bakalářská práce se zabývá oblastí vstupních dat fe formátu CIM a vytvořením nástrojů pro jeho převod do PNP platformy (na obrázku naznačeno zeleně).

Rozsah bakalářské je přehledně zobrazen na obrázku 6.2. Zelená tabulka vlevo představuje formát CIM. K načtení tohoto formátu je potřeba načíst tří základní profily, kterými je EQ, SSH a TP. Tyto profily jsou detailněji vysvětleny níže. Načtení profilů zpracovává technologie JAXB, která nabízí metody pro konverzi XML dat na Java objekty a naopak. Právě tyto objekty representuje žlutá tabulka CIM DAL. Java objekty jsou rozděleny do dvou částí. První částí jsou Doménové objekty platformy PNP a druhou Speciální CIM třídy. K rozdělení do těchto dvou částí došlo na základě podrobné znalosti o CIM profilech a platformy PNP. Doménové objekty tvoří skupinu, která se vyznačuje tím, že existují identické elementy, které obsahuje platforma PNP i CIM. Druhou částí tvoří ostatní elementy, tedy elementy, které jsou obsaženy v profilech CIM, nikoliv v platformě PNP. Třetí a zároveň poslední tabulka PNP reprezentující platformu PNP vytvořenou na Katedře kybernetiky (KKY) Fakulty aplikovaných věd (FAV) jako nástroj pro načtení, zpracování

a následné vyhodnocení dat. Mezi tabulkami CIM DAL a tabulkou PNP se nachází modrá oboustranná šipka nesoucí název Post processing. Anglickým slovním spojením Post processing je v tomto případě myšlena navazující práce, na které je možné pokračovat v následujících projektech. Přesněji jde o rozsáhlý proces provádějící přenos veškerých dat z formátu CIM do této platformy.



Obrázek 6.2: Grafické znázornění rozsahu bakalářské práce

6.1 Testovací data

Vyvíjené softwarové řešení bylo testováno na připravených testovacích datech. Pro účely testování byly získány testovací data ze společnosti ENTSO-E, která obsahovala data elektrických sítí různých velikostí (celkem 5 typů elektrických sítí). Každý z modelů elektrických sítí obsahuje data rozdělená do několika souborů (resp. profilů), které jsou podrobněji popsány v kapitole níže.

- **Micro Grid** - Micro Grid reprezentuje nejmenší možnou sadu, která je především zaměřená na základní definici syntaxe a konektivitu. Díky své malé velikosti je velice vhodná pro testování a následnému ladění.
- **Micro Grid Error** - Jedná se o testovací síť založenou na konfiguraci Micro Grid Test Configuration. Tento soubor byl vytvořen pro ilustraci různých typů chybných případů, které byly poskytnuty během hodnocení různými společnostmi.
- **Mini Grid** - Modely Mini Grid jsou především určeny pro výpočet zkratu, tedy spojení mezi dvěma uzly elektrického obvodu, které mají odlišné napětí. U takového spojení může dojít k poškození obvodu.
- **Small Grid** - Small Grid modely slouží zejména k testování již výše zmíněné interoperability stejnosměrného proudu vysokého napětí. Další použití těchto modelů je k analýze zatížení.

- **Real Grid** - Real Grid model obsahuje detailní informace reálné elektrické soustavy většího rozsahu.

Real Grid data popisují vlastnosti daných uzlů a jejich větvení pro elektrickou soustavu většího rozsahu. Tento model je složen ze čtyř profilů, který mi jsou EQ, TP, SSH, SV. Pro ilustraci jsou tyto profily zobrazeny na obrázku 6.3, kde v prvním sloupci je název profilu, v druhém přípona a ve třetím velikost, který je uvedena v bajtech.

CGMES_v2.4.15_RealGridTestConfiguration_EQ_v2	xml	47 526 550
CGMES_v2.4.15_RealGridTestConfiguration_SSH_v2	xml	7 470 306
CGMES_v2.4.15_RealGridTestConfiguration_TP_v2	xml	6 382 989
CGMES_v2.4.15_RealGridTestConfiguration_SV_v2	xml	4 483 767

Obrázek 6.3: Ukázka profilů

Pro ilustraci rozsahu elektrické sítě obsažené v modelu Real Grid obrázek 6.4 obsahuje názvy a počty jednotlivých elementů, které jsou specifické pro jednotlivé profily. Všechny tyto elementy jsou implementovány v Jave pomocí tříd naznačených na níže uvedeném obrázku.

Class	# of Objects
ACLineSegment	7561
BaseVoltage	8
ControlArea	1
ControlAreaGeneratingUnit	1
CurrentLimit	32182
CurveData	2720
EnergyConsumer	6687
FossilFuel	138
GeneratingUnit	25
GeographicalRegion	1
HydroGeneratingUnit	727
HydroPump	2
LinearShuntCompensator	311
LoadResponseCharacteristic	6687
NuclearGeneratingUnit	59
OperationalLimitSet	17960
OperationalLimitType	6
PhaseTapChangerTable	9
PhaseTapChangerTablePoint	280
PhaseTapChangerTabular	9
PowerTransformer	1509
PowerTransformerEnd	3018
RatioTapChanger	1185
ReactiveCapabilityCurve	1347
RegulatingControl	1350
StaticVarCompensator	3
SubGeographicalRegion	7
Substation	4875
SvPowerFlow	8348
SvShuntCompensatorSections	311
SvTapStep	1194
SvVoltage	7359
Switch	1292
SynchronousMachine	1347
TapChangerControl	1194
Terminal	29072
ThermalGeneratingUnit	138
TopologicalIsland	131
TopologicalNode	7359
VoltageLevel	5577
WindGeneratingUnit	398

Obrázek 6.4: Počty zpracovávaných tříd v Real Grid

6.2 Typy profilů testovacích dat

Každá výše zmíněná datová sada obsahuje několik souborů (profilů) popisující jednotlivé aspekty zachycené datovým modelem CIM [12]:

- **DL** - Diagram Layout profile: profil obsahující data potřebná pro modelový diagram
- **DY**- Dynamics profile: profil, ve kterém jsou reprezentovány parametry nezbytné pro chování energetického systému
- **EQ** - Equipment profile: profil udávající stavy zařízení v elektrické síti
- **EQ_BD** - Boundary equipment profile: profil obsahující data, které specifikují vazbu výměnu dat mezi zařízeními
- **GL** - Geographical Location profile: profil obsahující GIS data
- **SSH** - Steady State Hypothesis profile: profil obsahující veškeré objekty potřebné pro výměnu vstupních parametrů, které jsou podstatné pro simulaci toku zatížení.
- **SV** - State Variables profile: profil obsahující všechny objekty potřebné k dokončení specifikace ustáleného stavu.
- **TP** - Topology profile: profil obsahující objekty topologie odkazují na odpovídající zařízení, které popisuje, jak je dané zařízení elektricky připojeno
- **TP_BD** - Boundary topology profile: profil obsahující data, které specifikují vazbu a výměnu dat mezi topologiemi

Tato bakalářská práce se zabývala převážně zpracováním profilů EQ, SSH a TP, které jsou popsány podrobněji dále.

EQ profil: Equipment profil zejména specifikuje prvky elektrické sítě, popisuje základní charakteristiky a elektrické připojení, které je vstupem pro topologické zpracování.

SSH profil: Steady State Hypothesis profil obsahuje všechny objekty potřebné pro výměnu vstupních parametrů, aby mohly provádět výpočty toku zatížení.

TP profil: Topology profil obsahuje všechny topologické objekty odkazují na odpovídající zařízení popisující, jak je elektricky propojeno. V tomto profilu je specifikována topologie elektrické sítě, přičemž samotná topologie je výsledkem analýzy zpracování topologie sítě.

6.3 Volba technologie softwarového řešení

Pro implementaci softwarového řešení zpracovávající CIM byl použit programovací jazyk Java, který obsahuje řadu vyspělých technologií pro zpracování XML dat.

6.3.1 Jazyk Java

Jazyk Java je objektově orientovaný programovací jazyk, který vyvinula firma Sun Microsystems. Java poskytuje mnoho možností k parsování souboru XML, kde hlavními uvažovanými technologiemi k softwarovému řešení bakalářské práce byly DOM, SAX a JAXB.

6.3.2 DOM

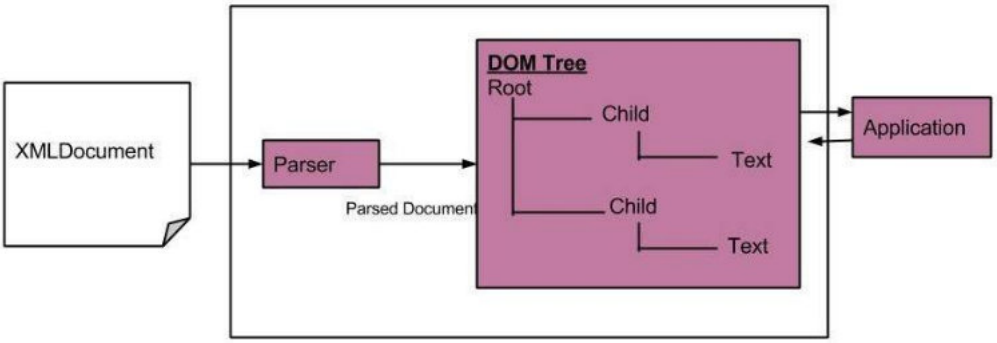
Document Object Model je aplikační programovací rozhraní (API) pro dokumenty HTML a XML. Definuje logickou strukturu dokumentů a způsob přístupu a manipulace s dokumentem. DOM definuje objekty, vlastnosti a metody pro přístup ke všem prvkům XML.
DOM je rozdělen na 3 různé části:

- **Core DOM** - model pro jakýkoli strukturovaný dokument
- **XML DOM** - model pro XML dokumenty
- **HTML DOM** - model pro HTML dokumenty

Jelikož moje bakalářská práce se pouze zabývá zpracování dokumentu XML, je níže pouze rozepsána část XML DOM.

Jak již bylo výše zmíněno, dokumenty XML obsahují elementy, které jsou zapsány ve stromové struktuře. XML DOM je programovací rozhraní popisující právě tyto elementy a vztahy mezi nimi. Vzhledem k tomu, že XML DOM poskytuje také rozhraní API, které umožňuje vývojáři přidávat, upravovat, přesouvat nebo odstraňovat elementy v libovolné části stromové struktury.

Níže je zobrazeno schéma struktury DOM (převzato z [14]), které popisuje, že program pro parsování dokumentu vyhodnocuje XML dokument jako strukturu DOM procházením přes každý uzel.



Obrázek 6.5: XML DOM Parser

Výhody XML DOM

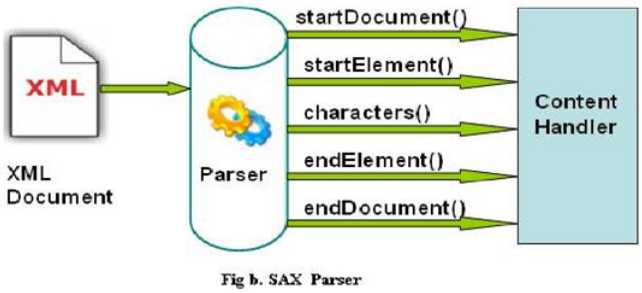
- **Jednoduché hledání konkrétních informací**
  Informace v XML DOM jsou uspořádány v hierarchii, která umožňuje vývojáři rychlé hledání konkrétních informací.
- **Modifikovatelnost**
  XML DOM je dynamický a poskytuje vývojáři prostor pro přidávání, úpravu, přesunutí nebo odstranění uzlů/elementů v libovolné části stromu.

Nevýhody XML DOM

- **Paměť**
  Parser uchovává celou stromovou strukturu XML v paměti. V případě zpracování velkého souboru XML je tento způsob velice nevýhodný, jelikož celý program zůstává po celou dobu v paměti.
- **Rychlost**
  Díky většímu využití paměti je rychlost nižší než SAX i JAXB.

6.3.3 SAX

Simple API for XML poskytuje nástroje pro zpracování parsování XML dokumentů s rozhraním API. SAX poskytuje mechanismus pro čtení dat z dokumentu XML, který je alternativou k technologiím DOM nebo JAXB. SAX je velice specifický. Na rozdíl od DOM nevytváří úplné zobrazení dokumentu. Parser SAX pracuje postupně od začátku do konce na jednotlivých částech dokumentu XML a vydává události k parsování. Tyto události jsou předány správci událostí, které poskytují přístup k obsahu dokumentu. Právě díky tomuto způsobu dochází k výrazném zrychlení rozdělení XML do jednotlivých částí. Zmiňovaný způsob pomocí postupných volání událostí, je zobrazen na obrázku 6.6 (převzat z [2]).



Obrázek 6.6: XML SAX Parser

Výhody SAX

- **Vyšší rychlost zpracování**
  Jelikož Parser pracuje vždy s částí celého dokumentu dochází k výraznému zrychlení zpracování vstupních sourobů.
- **Nízká paměťová náročnost**
  Neuchovává celou stromovou strukturu XML v paměti nýbrž pouze část dokumentu.

Nevýhody SAX

- **Specifická práce s dokumentem**
  Dokument je zpracováván sekvenčně, při čtení se nelze vracet.
- **Rozdělení dat**
  Data jsou rozdělena na části podle událostí.

6.3.4 JAXB

JAXB je další z uvažovaných možností k parsování XML dokumentu. Celým názvem Java Architecture for XML Binding. Tento algoritmus poskytuje rychlý a pohodlný způsob pro spojení XML schéma a reprezentaci v jazyku Java. Architektonické schéma technologie JAXB je zobrazeno na obrázku 6.7 [7]. Technologie JAXB obsahuje několik základních komponent, přičemž této bakalářské práci byly využity následující prvky technologie JAXB:

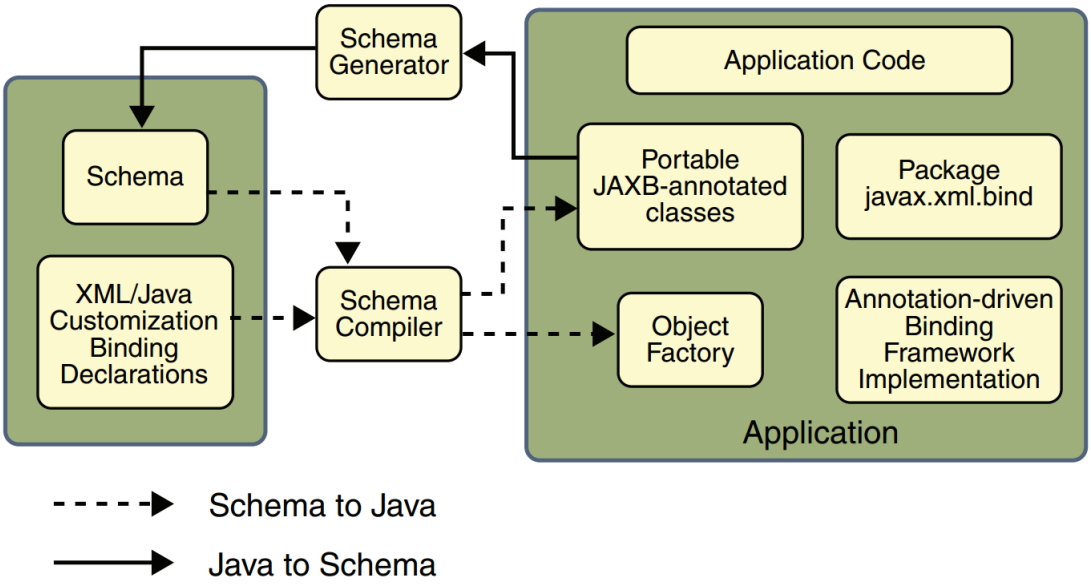
- **Schema compiler (Kompilátor schémat)**
  Spojuje zdrojové schéma na odvozené elementy v programu. Vazba mezi elementy je popsána pomocí jazyku XML.
- **Schema generator (Generátor schémat)**
  Tato část mapuje soubor existujících elementů vytvořených v programu s odvozeným schématem. Mapování je popsáno programovými anotacemi, které jsou zobrazeny na obrázku 6.7.

- **Binding runtime framework (Modul pro řízení spojení mezi bloky)**
  Poskytuje operace unmarshalling (čtení) a marshalling (sestavení), které obsahují metody pro přístup, manipulaci a ověřování obsahu XML buď pomocí schémat odvozených nebo existujících elementů vytvořených v programu.

- **XML/Java Customization Binding Declarations**
  Tato část představuje vstupní soubory CIM, která jsou následně zpracovány.

- **Application Code**
  Část JAXB, která obsahuje veškerý kód kromě části popisující vytvořené třídy. Tato část obsahuje JAXB metody umožňující načtení a následné zpracování CIM dokumentu.

- **Portable JAXB annotated classes**
  Jde o anotované třídy vytvořené vývjářem. Anotace je nutná k popisu tříd a atributů, které chceme načíst.



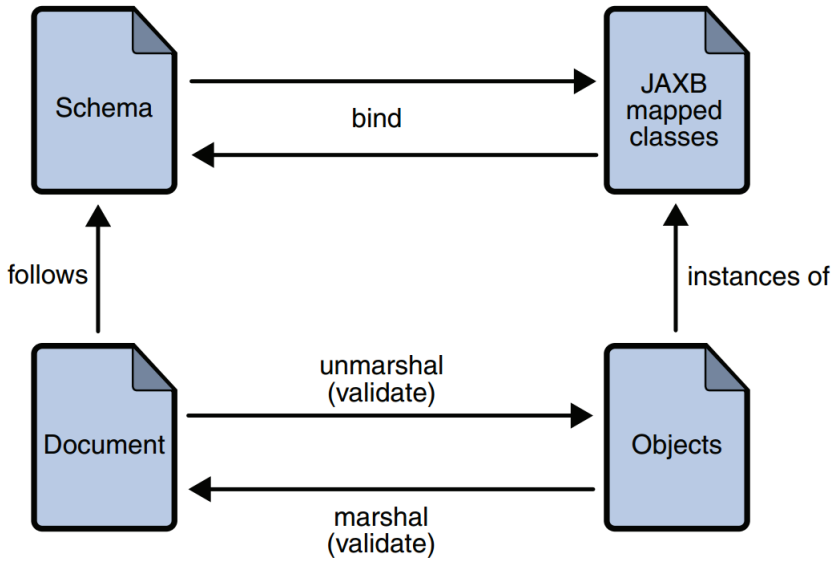
Obrázek 6.7: Architektonický přehled JAXB

Na obrázku 6.8 [7] je zobrazeno funkcionální propojení mezi jednotlivými částmi JAX, kde propojení těchto částí probíhá v těchto krocích:

1. **Dokument**
   Nejprve máme XML dokumenty, který je potřeba načíst. Tyto dokumenty jsou označeny na obrázku 6.8 jako *Document*.

2. **Namapování tříd** Tato část popisuje namapování JAXB tříd pomocí výše zmíněných anotacích, Mapování je vytvořeno mezi strukturou dokumentu XML a JAXB třídy. Tato část je reprezentována jako *JAXB mapped classes*.

3. **Vytvoření objektu**
   Poslední částí je vygernerování JAXB objektů na základě namapování tříd s XML dokumentem. Vytvoření objektu zajišťuje metoda *unmarshalling*. Načtená datá se z objektu získají pomocí přístupových funkcích(getterů).



Obrázek 6.8: XML JAXB - spojení mezi hlavními částmi

Jak bylo výše zmíněno, hlavními metodami poskytující JAXB je *unmarshalling*(čtení) a *marshalling*(vytváření) XML dokumentů . Z tohoto důvodu jsou obě metody vysvětleny více podrobně.

Unmarshalling

Jde o jednoduchý přístup k XML dokumentu poskytující klientské aplikaci převedení dat ze struktury XML do Java objektů odvozených z JAXB.

Tato metoda byla přímo použitá v mojí bakalářské práci, proto je detailněji popsána v následujícími odstavci. Ukázka této metody je zobrazena na obrázku 6.9

Tato metoda je založena na vytvoření objektu JAXBContext a volání metody pro načtení dokumentu. Objekt JAXBContext poskytuje vstupní rozhraní API JAXB a udržuje informace o vazbách mezi jazyky XML a Java. Jedním ze způsobů vytvoření instancí JAXBContext je volání statické metody *newInstance*. Z JAXBContext je získán objekt Unmarshaller, který funguje jako ovladač, který řídí zpracování textu XML pro vytvoření ekvivalentní sady Java objektů . V objektu rdf jsou uloženy načtená data z XML dokumentu. Tato data jsou dále volány pomocí přístupových funkcí (tzv. getterů).

```
public T unmarshall(String filename, boolean verbrose)
{
    T rdf = null;
    try {
        JAXBContext jc = JAXBContext.newInstance(Class.forName(rdfClassName));
        Unmarshaller ums = jc.createUnmarshaller();
        rdf = (T) ums.unmarshal(new File(filename));

        if (verbrose)
            rdf.verbrose();

    } catch (Exception e) {
        e.printStackTrace();
    }
    return rdf;

}
```

Obrázek 6.9: Java Unmarshalling

Marshalling
Metoda Marshalling poskytuje klientské aplikaci možnost převést strom odvozených JAXB objektů zpět do struktury XML. Ve výchozím nastavení používá Marshaller při generování dat XML kódování UTF-8.

Klientské aplikace nebudou vyžadovat ověření stromu obsahu Java před zařazením. Neexistuje také požadavek, aby byl strom s obsahem Java platný vzhledem k jeho původnímu schématu, který je zpřístupní zpět do dat XML.

Výhody JAXB

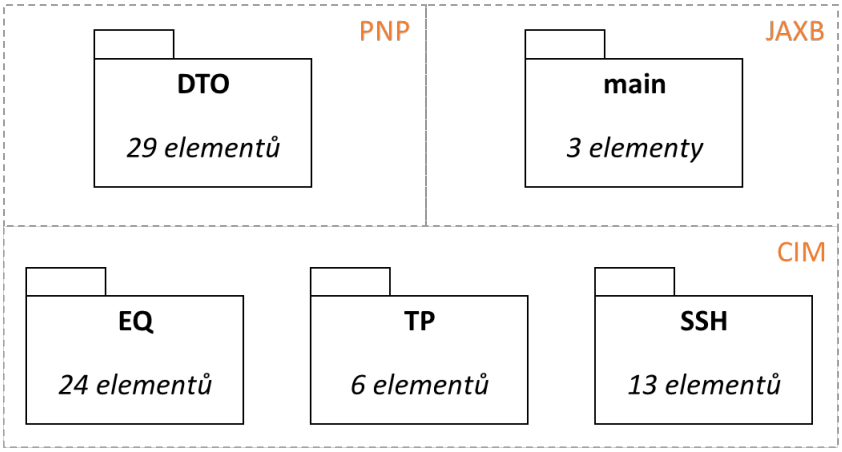
- **DTD (Document Type Definition)**
  Použití JAXB zajišťuje platnost zpracovávaného XML dookumentu.
- **Nízká paměťová náročnost**
  Vygenerovaná stromová struktura vytvořená pomocí JAXB zabírá méně místa než strom vytvořený pomocí DOM
- **Jednoduchost**
  Po provedení přípravných kroků je použití technologie JAXB v konkrétní aplikaci velmi jednoduché.

Nevýhody JAXB

- **DTD (Document Type Definition)**
  Není možné použití pro zpracování obecného XML. Dokument je zpracováván sekvenčně, při čtení se nelze vracet.
- **Neintuitivní přípravné práce**
  Je potřeba provést časově náročné a neintuitivní přípravné kroky před začátkem parsování proto, aby technologie JAXB poznala jaký strom je potřeba zkonstruovat.

6.4 Popis architektury softwarového řešení

Softwarové řešení vyvinuté v této bakalářské práce je rozděleno do následujících částí.



Obrázek 6.10: Diagram balíčků (Package Diagram)

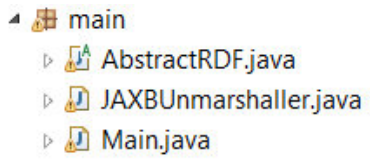
Jednotlivé části softwarového řešení mají následující význam:

- **PNP** - Platforma PNP obsahující package (balíček) *DTO*, který je složen z 29 doménových tříd.
- **JAXB** - Část obsahující package *main*, který zahrnuje 3 elementy / třídy, které slouží pro spuštění a koordinaci běhu komponent zpracovávající jednotlivé profily.
- **CIM** - Balíček CIM obsahuje třídy, které byly specielně navrhnuté pro zpracování souborů CIM.

V následujícím textu budou popsán nejprve balíček main, který spadá do části JAXB a následně budou popsány třídy a balíčky ve vztahu k jednotlivým načítaným profilům.

6.4.1 Třídy pro spuštění a koordinaci běhu komponent zpracovávající CIM

Na obrázku 6.11 je zobrazen balíček main, který obsahuje třídy pro spuštění a koordinaci běhu jednotlivých softwarových komponent zpracovávající vybrané profily CIM.



Obrázek 6.11: Main package

AbstractRDF: je abstraktní třída, obsahující metodu verbrose(). Třída je využívána jako rozhraní pro třídy typu RDF, které obsahují metody pro zpracování EQ, TP a SSH profilů. Třídy vytváří instance objektůJava - List pro třídu svázanou pomocí technologie JAXB. Například do objektu List - ACLineSegment se uloží všechny elementy ACLineSegment, které jsou uloženy v celém XML dokumentu definující EQ profil.

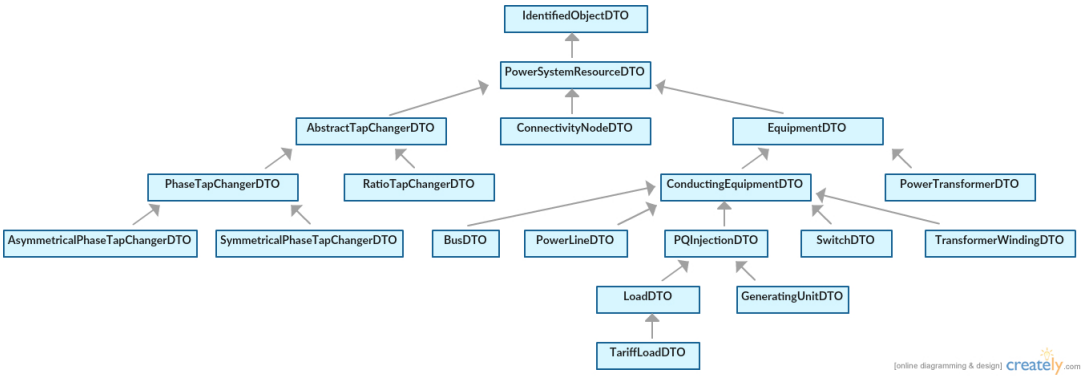
JAXBUnmarshaller: zapouzdřuje unmarshalling jednotlivých tříd, který je podrobně vysvětlen v kapitole 6.3.4.

Main: je hlavní třída řídící výsledné načtení XML dokumentů. Tato část obsahuje spuštění celého programu sekvenčně i pomocí vláken.

6.4.2 Třídy pro načítání EQ profilu

Třídy pro načítání EQ profilu jsou rozděleny do dvou částí:

- **Doménové třídy(DTO)**
  Tyto třídy byly vytvořené na základě analýzy elementů v platformě PNP a formátem CIM. Na diagramu tříd níže jsou vyobrazeny třídy, které obsahuje platforma PNP i standard CIM. Modře vyznačené bloky znázorňují elementy, které v Javě jsou reprezentovány jednotlivými třídami. Šedé šipky reprezentují dědičnost mezi těmito elementy. V odstavci pod obrázkem 6.12 je detailněji popsán diagram tříd.



Obrázek 6.12: Diagram tříd doménových elementů

Diagram tříd (Class Diagram) zobrazený na obrázku 6.12 slouží pro zachycení vztahů mezi jednotlivými doménovými třídami. Každý z těchto elementů obsahuje specifické atributy, které pro přehlednost nejsou na obrázku 6.12 uvedeny. Díky dědičnosti získávají zděděné elementy veškeré atributy, které obsahují jejich rodičovské elementy. Veškeré informace of všech CIM elementech jsou sepsány ve Standardu CIM.

Implementace DTO objektů v Javě

Jak již bylo výše zmíněno, každý z elementů zobrazených na obrázku 6.12 je reprezentován třídou. Každá třída obsahuje specifické parametry, které ji definují. Tyto třídy, které reprezentují části platformy PNP byly vytvořeny Katedrou Informatiky Fakulty aplikovaných věd. Úkolem v DTO části pro EQ profil bylo vytvoření veškerých JAXB anotacích. Příklad anotace je zobrazen na ukázce Java kódu(obrázek 6.12).

```java
package DTO;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

@XmlAccessorType(XmlAccessType.FIELD)
@XmlRootElement(name = "ACLineSegment")
public class PowerLineDTO extends ConductingEquipmentDTO {

    @XmlElement(name = "ACLineSegment.r", namespace = "http://iec.ch/TC57/2013/CIM-schema-cim16#")
    protected Double resistance;

    public Double getResistance() {
        return resistance;
    }
```

Obrázek 6.13: Příklad implementace PowerLineDTO

Příklad třídy PowerLineDTO

Na obrázku 6.13 je zobrazena implementace elementu ACLineSegment ve formátu CIM. Element ACLineSegment představuje element PowerLineDTO v platformě PNP. Jak již bylo výše zmíněno, při vytváření tříd je nutné anotovat veškeré elementy, které chceme z dokumentu XML získat. Abychom mohli anotovat požadované elementy k namapování XML schéma je potřeba nejprve naimportovat JAXB balíčky, které tuto anotaci umožňují. Dalším krokem je nastavení XmlAccessorType na XmlAccessType.FIELD, což umožní namapování podlé hledaného řetězce. To znamená, že po tomto nastavení JAXB prochází XML dokument a když narazí na řetězec, který je roven anotaci. V příkladě níže je anotována třída PowerLineDTO a její atribut resistance. Třída je reprezentována anotací XmlRootElement a atribut resistance anotací XmlElement. Anotace tohoto elementu je složena z přesného názvu a namespacu pomocí kterého se odkazujeme na daný prefix (viz kapitola RDF). Na závěr je vygenerována metoda getResistance(), kterou voláme při vypisování požadované hodnoty.

- **Speciální CIM objekty**
  Jedná se o třídy, které nejsou součástí platformy PNP. Tyto třídy byly vytvořeny speciálně pro zpracování určitých aspektů formátu CIM. V rámci analýzy formátu CIM byly identifikovány potřebné elementy, které platforma PNP přímo neobsahuje, ale pro soubor CIM jsou důležité. Všechny tyto třídy jsou podtřídami rodičovské třídy IdentifiedObject, ze které dědí tyto atributy:

  – originalID: unikátní označení každého elementu, který v tomto případě představuje atribut rdf:ID.

  – name: název daného elementu.

  – aliasName: zkrácená forma celého názvu.

  – description: popis určitého elementu.

  Seznam speciálně vytvořených tříd pro načítání EQ profilu:

Obrázek 6.14: Třídy EQ profilu

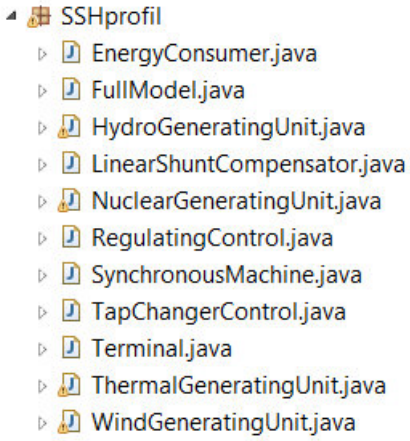
6.4.3 Třídy pro načítání TP profilu

TP profil není součástí PNP platformy a následující třídy byly specielně vytvořeny pro načítání topologie uložené ve formátu CIM . Balíček TP obsahuje následující tyto třídy:

Obrázek 6.15: Třídy TP profilu

6.4.4 Třídy pro načítání SSH profilu

SSH profil stejně jako u TP profilu balíček SSH obsahuje třídy specielně vytvořené pro načítání CIM formátu. SSH profil je složen z těchto tříd:



Obrázek 6.16: Třídy SSH profilu

6.5 Analýza výkonnosti softwarového řešení

Analýza proběhla po načtení vybraných testovacích dat. Výsledky jsou uvedeny pro největší z testovaných XML dokumentů.

6.5.1 Real Grid

V rámci zpracování testovací sady Real Grid byl zpracován následující počet elementů:

- **EQ profil:** 123 445
- **TP profil:** 36 432
- **SSH profil:** 43 786

Celkem: 203 663

Jak je patrné EQ profil představuje největší část zpracování, což je dáno počtem a variabilitou zpracovávaných CIM objektů. Počet zpracovávaných elementů se projeví na časové náročnosti zpracování CIM profilu, jak bude ukázáno v následujícím textu.

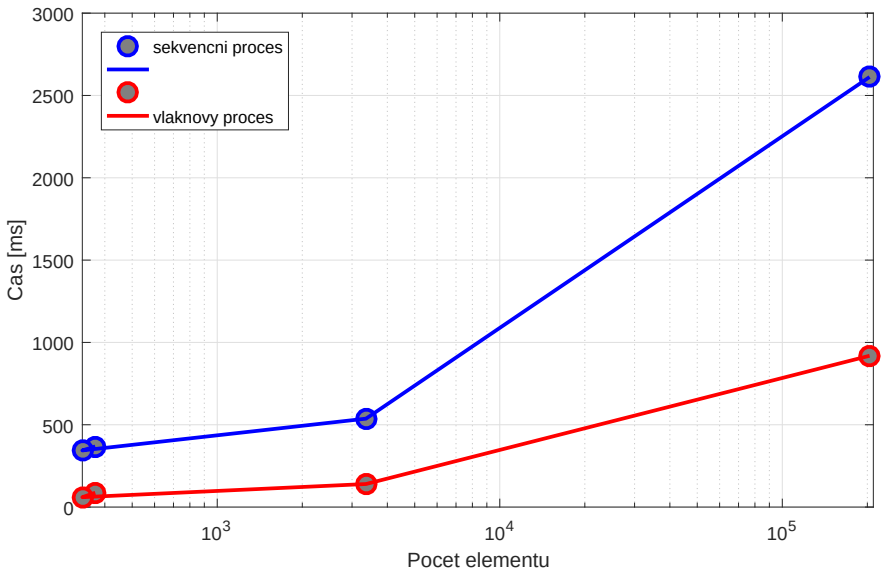
Pro vyhodnocení časové náročnosti byly testovány dva způsoby volání výpočetních komponent - sekvenční a vláknový. V průběhu testování byly změřeny následující časy:

- **Sekvenční:** 2261 ms
- **Vláknový:**
  - **EQ:** 909 ms
  - **TP:** 393 ms
  - **SSH:** 247 ms

Podle očekávání je vláknový přístup výrazně rychlejší a celková doba zpracování vybraných souborů je rovna době zpracování EQ profilu.

6.5.2 Vyhodnocení škálovatelnosti softwarového řešení

Na obrázku 6.17 je vyobrazena časová závislost zpracování vybraných CIM profilů na počtu elementů. Na svislé ose je vynesen výpočetní čas a na vodorovné ose je zobrazen počet elementů. Pro lepší názornost je vodorovná osa v logaritmickém měřítku. Na obrázku jsou vyneseny dva grafy časové náročnosti pro sekvenční a vláknové zpracování.



Obrázek 6.17: Vyhodnocení nacteni pomoci JAXB

Jak je z obrázku patrné, výpočetní náročnost zpracování vybraných CIM profilů vykazuje polynomiální nárůst výpočetního času s ohledem na počet elementů.

Kapitola 7

Závěr

Cílem této práce bylo vytvoření softwarových nástrojů pro zpracování dat elektrických sítí ve formátu CIM. Bakalářská práce se skládala ze dvou hlavních částí - teoretické a praktické.

V teoretické části byla popsána problematika počátků přenosu elektrické energie a následný proces liberalizace trhu s energiemi, který mimo jiné přinesl zvýšenou potřebu komunikace a výměny informací mezi jednotlivými stranami (tzv. stakeholdery). V další části bakalářské práce je popsána problematika komunikace a datové výměny mezi různými energetickými společnostmi a jsou diskutovány různé datové formáty využívané jednotlivými společnostmi. Práce se především zabývala v praxi používanými formáty UCTE a CDF. Byly identifikovány hlavních nevýhody současných formátů, které by měly být odstraněny zavedením unifikovaného formátu CIM, jehož analýza a zpracování vybraných částí bylo těžištěm bakalářské práce. Formát CIM využívá především technologie UML, XML a RDF, které jsou zde vysvětleny jednak obecně, tak také na ilustračních příkladech.

Praktická část spočívala ve vývoji softwarové řešení pro zpracování vybraných profilů (souborů) formátu CIM. Pro implementaci byl zvolen programovací jazyk Java, který byl vybrán z důvodu rozsáhlé nabídky technologií poskytující nástroje pro zpracování XML-RDF dokumentu. Byla provedena rešerše těchto technologií - DOM, SAX a JAXB, které jsou podrobně popsány v textu práce . K výslednému softwarovému zpracování byla vybrána technologie Java Architecture for XML Binding (JAXB).

Po zvolení této technologie následovala část zabývající se přímým zpracováním formátu CIM. Zpracování bylo vytvořeno v programovacím jazyku Java, kde správnost funkčnosti programu byla testována na sadě testovacích dat, které byly poskytnuty společnosti ENTSO-E. Testovací sada obsahovala data popisující elektrické sítě a byla dále rozdělena do podsložek (profilů), které popisovaly různé části elektrické sítě. V této bakalářské práci se zpracovávaly vybraní profily EQ, TP a SSH. Byly vytvořeny a ověřeny softwarové komponenty zpracovávající jednotlivé profily a koordinační nástroje pro spuštění a řízení celého procesu zpracování dat.

Na závěr byla vytvořena analýza zkoumající časovou náročnost zpracování CIM souborů, při které byly testovány dva způsoby volání výpočetních komponent - sek-

venční a vláknový. Detailní analýza reprezentující jednotlivé profily byla provedena na největší z analyzovaných energetických sítí. Výsledky analýzy ukázaly, že použití technologie JAXB je velice vhodné pro zpracování CIM souborů a má vysoký potenciál pro praktické využití.

V bakalářské práci se neřešilo následné zpracování načtených dat do informačního modelu elektrické sítě využívaného platformou PNP. Vzhledem k rozsahu nebyla detekce chyb a datových nekonzistencí součástí vyvinutého softwarového řešení. V dalším vývoji vyvinutých softwarových nástrojů by měly být implementovány komponenty řešící zmíněné problémy.

Literatura

[1] Tomáš Bartoš. Liberalizace energetického trhu v evropské unii a vývoj cen elektrické energie v členských zemích. Master's thesis, Mendelova univerzita v Brně, 2012.

[2] SAP Corporation. Java mapping concepts (dom and sax), 2009. https://wiki.scn.sap.com/wiki/pages/viewpage.action?pageId=78053807.

[3] Jan Fiala. Pspad editor, 2016.

[4] Union for the Coordination of the Transmission of Electricity. Ucte data exchange format for load flow and three phase short circuit studies. 2007.

[5] CEN-CENELEC-ETSI Smart Grid Coordination Group. Smart grid reference architecture, November 2012.

[6] W. Group. Common format for exchange of solved load flow data. *IEEE Transactions on Power Apparatus and Systems*, PAS-92(6):1916–1925, November 1973.

[7] Eric Jendrock, Jennifer Ball, Debbie Carson, Ian Evans, Scott Fordin, and Kim Haase. *Java(TM) EE 5 Tutorial, The (3rd Edition) (The Java Series)*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2006.

[8] Alan W. McMorran. An introduction to iec 61970-301 & 61968-11: The common information model, January 2007.

[9] J. Simmins. Intelligrid common information model primer. Technical report, Electric Power Research Institute, 3420 Hillview Avenue Palo Alto, CA 94304-1338 USA, October 2013.

[10] Jakub Slavík. Víte, co to je a jak funguje smart grid?, March 2013. http://www.proelektrotechniky.cz/vzdelavani/22.php.

[11] Kateřina Teplá. Historie dálkového přenosu elektrické energie, July 2016. http://cz.energy-hub.cz/pro-energy/13258-historie-dalkoveho-prenosu-elektricke-energie.

[12] Mathias Uslar, Michael Specht, Sebastian Rohjans, Jörn Trefke, and José M. González. *The Common Information Model CIM: IEC 61968/61970 and 62325 - A practical introduction to the CIM*. Springer-Verlag Berlin Heidelberg, 2012.

[13] Wikipedia. Common information model (computing), 2004. https://en.wikipedia.org/wiki/Common_Information_Model_(computing).

[14] www.tutorialspoint.com. Xml dom - overview, 2010. https://www.tutorialspoint.com/dom/xml_dom_overview.htm.