Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

Plzeň, 2018

Pavel Novotný

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

Bakalářská práce

Energetické modelování budov Fakulty aplikovaných věd

Vedoucí: Ing. Martin Střelec, Ph.D.
Vypracoval: Pavel Novotný A15B0546P

Plzeň, 2018

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 17.5.2018 .....

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. Dále děkuji Lukáši Soukupovi za spolupráci při řešení společné tématiky bakalářské práce. Rovněž děkuji celé katedře kybernetiky za podporu při zpracování této práce. V neposlední řadě bych rád poděkoval mým rodičům za jejich podporu po celou dobu studia.

Abstrakt

Cílem bakalářské práce je vytvoření společného datového úložiště pro všechny energetické datové zdroje budov Fakulty aplikovaných věd a nad ním vytvořit energetické analytics - konkrétně energetický model spotřeby. První část se věnuje analýze současných datových zdrojů - systému energetického hospodářství (EMS) a telemetrické soustavy. Následující část se zabývá návrhem a implementací společného datového úložiště v podobě relační databáze. Na tuto část navazuje propojení původních datových zdrojů s novou společnou databází. Po zpřístupnění datových zdrojů a jejich propojení s cílovou databází, je možné vystavět kontextový model uložený v podobě XML formátu. Na základě všech předchozích částí bylo možné vytvořit finální část práce: energetický model budov FAV. Tento model má potenciál pro uplatnění v oddělení provozu, kde by mohl být použit pro snížení nákladů spojených s nákupem energií z distribuční sítě.

Klíčová slova

energetické měření, energetická soustava, EMS - Energy Management System, Telemetrická soustava, databáze - MySQL, SQL - Structured Query Language, Java, XML - eXtended Markup Language, Matlab, kontextový model, energetický model, model typické denní spotřeby

Abstract

The bachelor's thesis goal is to design and implement a common data storage for all energy data sources present in the buildings of FAV and build energy analytics, such as energy demand model. The first part focuses on the analysis of current energy data sources present in the buildings of FAV. The next part is focused on design and implementation of a common data storage in form of a relation database. Further part focuses on creation of an interconnection between the current data storage and the newly implemented database. After these steps, the context model of Faculty applied sciences buildings was developed. The context model is written in an XML format standard. Next segment of thesis aims to energy demand model of the building of the Faculty applied sciences. This model should help the employees of Operation management department to improve their work and it also should help to optimize and reduce cost for electricity.

Key words

energy demand, energy grid, EMS - Energy Management System, Telemetry, database - MySQL, SQL - Structured Query Language, Java, XML - eXtended Markup Language, Matlab, context model, energy demang model, model of typical day demand

Obsah

Kapitola 1

Úvod

1.1 Motivace

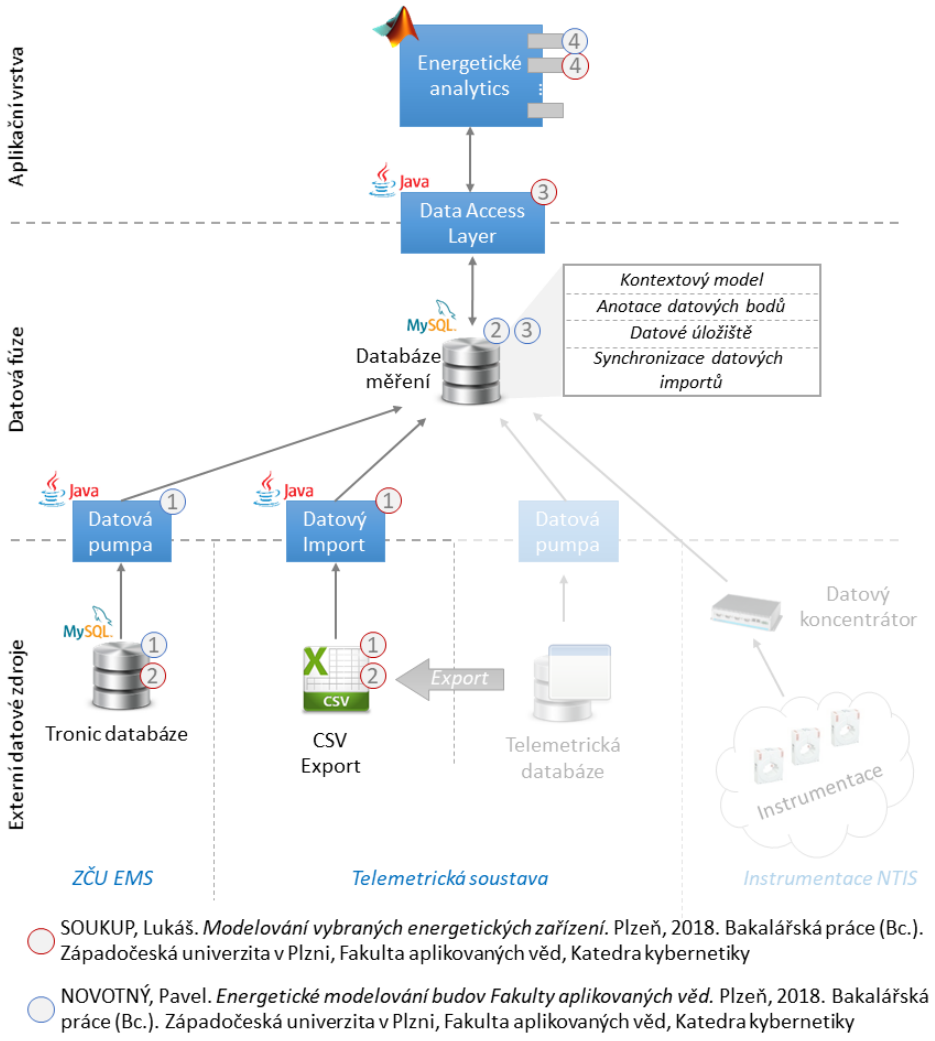
Bakalářská práce se zabývá analýzou a vývojem nástrojů pro zpracování energetických dat z budov areálu Fakulty aplikovaných věd ZČU. Práce byla motivována potřebou monitorování energií a energetických toků pro provozní účely. Pomocí vhodného monitoringu energií je možné řešit nejen technické úlohy jako například analýzu energetické náročnosti budov a celého areálu, ale i například sjednávat obchodní závazky jako je určení výkonových špiček pro kapacitní platby a jiné. Cílem bakalářské práce je vytvoření speciálních energetických analytics (tj. energetických modelů budov) založených na nově vyvíjeném systému pro sběr a zpracování dat. Výsledný monitorovací systém ulehčí a zefektivní kvalitu práce správců budov a pracovníků oddělení provozu a služeb ZČU a dále zpřístupní měřená data i pro studijní či výzkumné účely.

Cílem bakalářské práce je změnit stávající stav, kdy jsou paralelně provozovány dva separátní systémy sběru dat - telemetrická soustava a energy management system, které obsahují komplementární datové sady. Oddělené systémy poskytují pouze omezené možnosti pro výzkumně-výukové účely a limitují jejich použití pro efektivní monitoring energií. Nově vyvíjený systém bude sbírat a integrovat data z obou stávajících systémů a poskytovat konsolidovanou datovou základnu pro navazující aplikace.

Činnosti spojené s realizací bakalářské práce byly rozděleny do ucelených oblastí, které pokrývaly (I) analýzu a zpřístupnění dostupných datových zdrojů v areálu FAV, (II) návrh a implementaci vhodného datového úložiště pro ukládání a správu dat pro potřeby energetického modelování, (III) vytvoření vhodného kontextového modelu sbíraných datových bodů a (IV) vytvoření energetického modelu budov Fakulty aplikovaných věd. V uvedeném smyslu jsou členěny i jednotlivé kapitoly bakalářské práce.

1.2 Vymezení obsahu bakalářské práce

Celý systém pro sběr energetických dat, jejich zpracování a následné využití je rozsáhlé téma, které nelze pokrýt jednou bakalářskou prací. Proto byla celá tématika rozdělena na dvě bakalářské práce. Rozdělení vývojových aktivit je vidět na schématu 1.1. Předkládaná bakalářská práce zahrnuje aktivity označené modrou barvou.



Obrázek 1.1: Schéma systému pro sběr energetických dat

Schéma systému pro sběr energetických dat lze kategorizovat do tří oblastí - externí datové zdroje, datová fúze a aplikační vrstvy. Nejnižší vrstva s názvem *Externí datové zdroje* obsahuje datové zdroje třetích stran, které slouží k primárním sběru dat z instalovaných měřících systémů (kapitola 2). Konkrétně v této bakalářské práci byla řešena problematika zisku historických dat z databáze firmy Tronics s.r.o.. Z nejnižších datových vrstev jsou měřená data pomocí datových pump přenášena do společného datového úložiště. Způsob řešení přenosu dat je blíže diskutován v kapitole 2.

Na vrstvu externích datových zdrojů navazuje vrstva datové fúze. Zde jsou data z externích datových zdrojů zpracovávána do unifikovaného formátu implementovaném v *Databázi měření*, která obsahuje nejen měřená data, ale i anotaci datových bodů a relevantní datové modely (např. hierarchický kontextový model). Bližší popis databáze lze nalézt v kapitole 2.

Datové body získávané z externích datových zdrojů byly anotovány a pro potřeby modelování energetické spotřeby budov Fakulty aplikovaných věd byl vytvořen relevantní kontextový model uvedený v kapitole 3.

Nad vrstvou datové fúze je umístěna vrstva aplikační, která obsahuje koncové aplikace pracující nad daty z databáze měření. Prostřednictvím Data Access Layeru jsou získávána očištěná data pro potřeby navazujících analytických služeb (Analytics). Energetický model budov FAV představuje příkladové energetické analytics, které bylo vytvořeno při realizaci této bakalářské práce viz kapitola 4.

1.3 Použité technologie

Vzhledem k rozdílným softwarovým technologiím, využívaných při řešení bakalářské práce, bude v této kapitole popsán technology stack. Výrazná heterogenita softwarových nástrojů je důsledkem využití výhod jednotlivých technologií v různých kontextech (např. zpracování dat, vývoj modelů).

1.3.1 Java

Java je objektově orientovaný, interpretovaný programovací jazyk, který je ve světě hojně rozšířen díky jeho přenositelnosti, nezávislosti na architektuře a mimo jiné velké databázi knihoven. Jazyk Java byl vyvinut firmou Sun Microsystems roku 1995.
Mezi výhody jazyka Java se řadí například nezávislost na platformě, podpora objektově orientovaného programování a správa paměti (Java má tzv. garbage collector).[1]

Výhody pro využití v bakalářské práci:

- pokročilé nástroje pro získání a zpracování dat
- přenositelnost výsledného kódu

1.3.2 SQL

SQL (Structured Query Language) neboli standardizovaný strukturovaný dotazovací jazyk je dotazovací jazyk používaný především pro práci s relačními databázemi. Je založen na jazyce *SEQUEL*, který byl vyvinut firmou IBM, právě na základě potřeby vytvořit sadu příkazů pro popis a práci s relačními databázemi. [2] Základní části jazyka *SQL* pak jsou:

- *DDL* - určující definici dat neboli jak budou data ukládána a vztahy mezi nimi. Příklady:
  - *CREATE* vytvoření nových objektů
  - *ALTER* změny objektů
  - *DROP* odstraňování objektů
- *DML* - určující manipulaci s daty (výběr, vkládání, mazání,...). Příklady:
  - *SELECT* vybírání dat
  - *INSERT* vkládání dat
  - *UPDATE* změna dat
  - *DELETE* odstranění dat
- *DCL* - určující řízení dat neboli oprávnění pro manipulaci s daty. Příklady:

- *GRANT* přidělení práv
- *REVOKE* odnětí práv
- *COMMIT* potvrzení transakce
- *ROLLBACK* zrušení transakce

Pro potřeby vytvoření konsolidovaného datového úložiště energetických dat byl použit SQL databázový systém MySQL.

Výhody pro využití v bakalářské práci:

- efektivní ukládání a správa dat
- data mining možnosti (zisk informací z uložených dat)
- otevřenost poskytovaného databázového systému

1.3.3 XML

XML (eXtensible Markup Language) je značkovací jazyk vyvinut konsorciem World Wide Web Consorcium. Jde o standardizovaný formát pro výměnu informací a byl vyvinut za účelem jak transportu, tak i ukládání dat. Mezi je ho největší přednosti patří, že je čitelný jak strojově, tak i lidsky. Další jeho přednost je například snadná konverze do jiných formátů.
Základní syntaxe je velmi jednoduchá. Každý XML dokument musí mít hlavičku s verzí xml a kódováním. Dále pak musí mít jeden kořenový element. Neprázdné elementy musí být ohraničeny startovací a ukončovací značkou. Každý atribut (atributy jsou uvnitř elementů) musí být ohraničen uvozovkami. Jednotlivé elementy se nesmějí překrývat, mohou ale být vnořeny. Ke zpracování XML formátu je v této práci používán DOM (Document Object Model) přístup, který vytváří/parsuje XML na základě stromové struktury. [6]
Jednoduchý příklad XML formátu:

```
<?xml version="1.0" encoding="UTF-8" ? >
<book>
  <title>java book </title>
  <author>nick bore </author>
  <pages> 1020 </pages>
  <example>xml - valid xml file</example>
</book>
```

Obrázek 1.2: Příklad XML formátu [11]

Formát XML byl v bakalářské práci využit pro definici kontextového modelu. Výhody pro využití v bakalářské práci:

- jednoduchá datová struktura
- velice efektivní pro manuální úpravy
- otevřenost formátu

1.3.4 Matlab

Matlab je multiplatformní skriptovací programovací jazyk vyvinut firmou Mathworks a zároveň je Matlab označením i pro vývojové prostředí tohoto jazyka. Původně byl jazyk vyvinut především pro matematické výpočty a operace. Odtud vychází i jeho název Matlab = MATrix LABoratory. Postupem času se však velmi rozrostl a nyní obsahuje moduly a rozšíření například pro simulace mechanických soustav, biologických procesů, statistická měření, 3D grafy a mnoho dalších. Pro kybernetiku a robotiku je velice hojně využívaná součást Matlabu zvaná Simulink, ve které je možno vytvářet simulační modely systému, regulátory apod..

Pro potřeby bakalářské práce byl Matlab využit pro modelovací a simulační účely.

Výhody pro využití v bakalářské práci:

- rapidní vývoj softwarových prototypů (tj. modelů)
- možnost využití rozsáhlých toolboxů

Kapitola 2

Systémy pro sběr a uložení dat nasazené na ZČU

2.1 Přehled nasazených systémů

Celý komplex Západočeské univerzity, nacházející se na jihozápadní straně Plzně, je napojen na místní elektrickou distribuční síť několika transformátory s převodem 22kV/400V. Průměrné denní zatížení v letních obdobích přesahuje 1 MW. Denní zatížení nemá homogenní charakter, jelikož v jsou v areálu ZČU, kromě klasických kancelářských budov, také specializovaná pracoviště a laboratoře, které mají velmi nárazové odběry elektřiny.

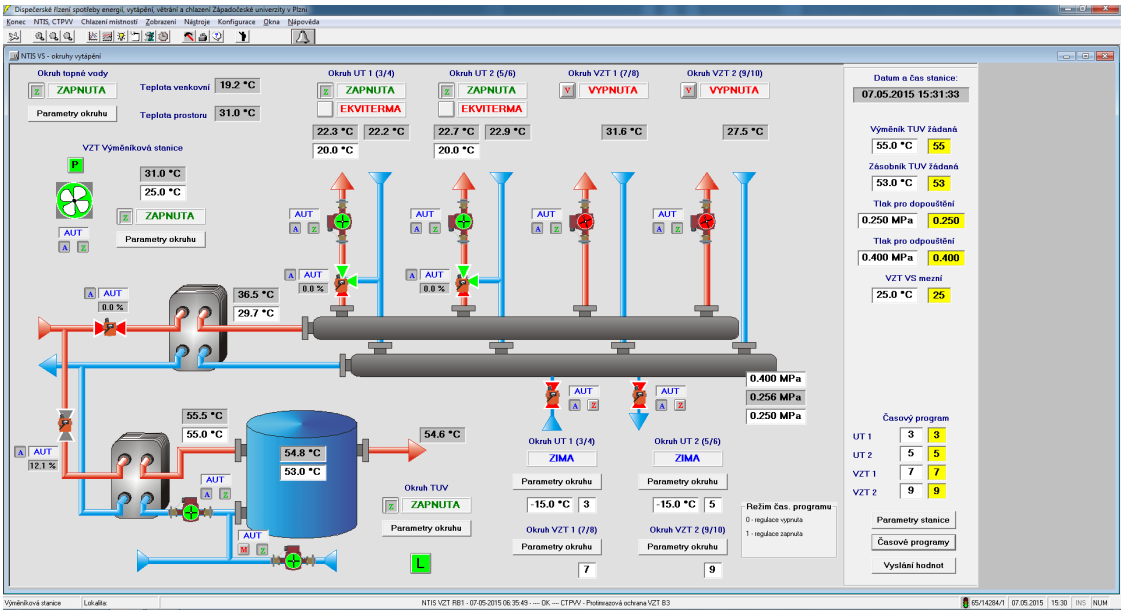
Do celkového odběru areálu ZČU je zahrnut vzduchotechnický systém (HVAC-topení, klimatizace, ventilátory), jehož odběr dosahuje až 680 kW. Tento systém má řiditelnou zátěž, díky tomu má potenciál ovlivnit řízení spotřeby celého areálu. Okamžitá spotřeba HVAC sytému je ovlivňována počasím a přítomností uživatelů. Největší část této energie je spotřebována chladícími zařízeními, čerpadly, výměníky vzduchu a rozvodným systémem.

V celém areálu ZČU jsou nasazeny dva systémy pro sběr a monitoring dat z energetických zařízení. První z nich (Telemetrická soustava) se stará o monitorování a sběr dat energetické spotřeby a to jak elektrické, tak tepelné. Druhý z nich je pak součástí systému energetického hospodářství (EMS - Energy Management System), který je zodpovědný za transformace a distribuce energií v rámci areálu ZČU.

2.2 Popis EMS ZČU

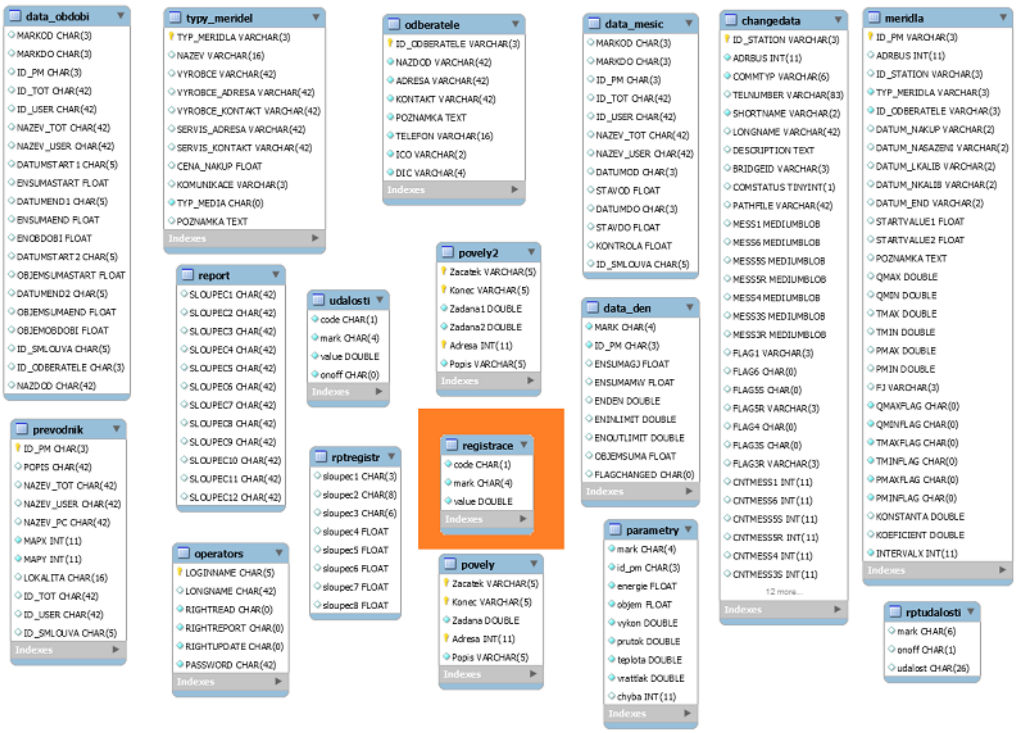
Při realizaci této bakalářské práce byla pozornost zaměřena na datové úložiště integrované v EMS systému ZČU. Jak bylo již v předchozí sekci zmíněno, v systému EMS se sbírají a monitorují data z procesů spojených s transformací energie a její distribuce v místní energetické síti. Například jsou sbírána data z vodních čerpadel, chladících zařízení a klimatizačních jednotek. EMS systém je zodpovědný

za dodržování tepelného komfortu a hygienických parametrů uvnitř budovy. EMS systém je provozován pracovníky oddělení provozu a služeb, kde je umístěn velín, ze kterého se řídí činnost a provoz energetických zařízení instalovaných v areálu ZČU.



Obrázek 2.1: Screenshot ze systému SCADA

Ve velíně jsou instalované uživatelské obrazovky tvořící SCADA systém, který na pozadí využívá databázi MySQL, kde jsou shromažďována vybraná data z energetických systémů a senzorů rozmístěných po celém areálu ZČU. Na obrázku 2.1 lze vidět ukázku jedné z obrazovek SCADA systému. Jedná se o okruhy vytápění v budově NTIS.



Obrázek 2.2: E-R schéma EMS databáze firmy Tronics s.r.o.

Na obrázku 2.2 je E-R schéma EMS databáze firmy Tronics s.r.o.. Přestože je tato databáze velmi rozsáhlá, je v této bakalářské práci využívána výhradně tabulka *registrace*, jelikož se do ní zapisují konkrétní měřené hodnoty. Tabulka *registrace* tvoří základní datový zdroj pro zisk energetických dat. Nezbytným dodatkem k databázi EMS a její tabulky *registrace* je i soubor s popisem jednotlivých datových bodů, ze kterých jsou hodnoty získávány. V *registraci* jsou pouze číselné identifikátory *code* jednotlivých bodů. Proto je pro kompletní kontextový popis nutný i soubor s anotací datových bodů.

2.3 Návrh datového úložiště pro potřeby energetického managementu

Typy datových uložišť

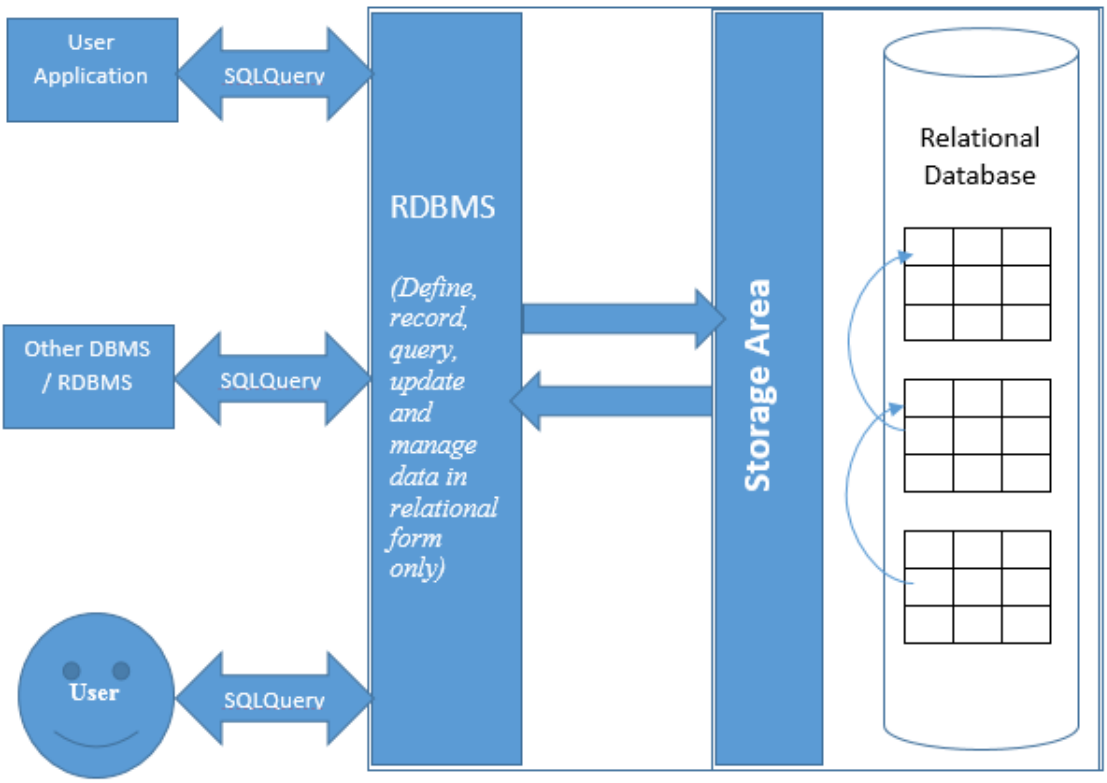
Existuje mnoho způsobů ukládání dat, mezi které například patří ukládaní dat do souborů (.csv, .json, .txt) či ukládání do databázových systémů. Výhody souborů jsou v jejich rychlosti a relativně snadné manipulaci. Na druhou stranu však dochází k problémům s konzistencí dat. Dalším problémem souborového přístupu je spolehlivost ukládání dat a velmi komplikovaná možnost přístupu k datům z více aplikací najednou.

Databázové systémy mají oproti souborovým systémům výhodu vrstvy Database Management System. Tato vnitřní vrstva nad samotnými daty se stará o konzistenci dat, přístupy jednotlivých uživatelů a správu nad daty jako takovými. Z důvodu výhod bezpečnosti, konzistence a spolehlivosti ukládání byl zvolen databázový systém jako vhodný způsob ukládání dat z různých původních datových zdrojů.

2.3.1 Databáze

Obecně se dá říci, že databáze je organizovaná kolekce dat. Existuje více typů organizace těchto dat a od toho odvozených typů databází. Mezi tyto typy patří například operační databáze, distribuční databáze a nejčastěji pak relační databáze. Relační databáze jsou databáze založeny na SQL(*Structured Query Language*).
Typickým znakem relačních databází je ukládání dat do tabulek, kdy jednotlivý záznam představuje jeden řádek s jedním až n sloupci. Dále je možno definovat vztahy a závislosti mezi různými tabulkami.

Nezbytnou součástí všech databází je *DBMS* (*Database Management System*).
$DBMS$ slouží jako vrstva mezi samotnými daty jako takovými, uloženými na disku a uživatelem nebo uživatelským programem. Tato vrstva se také stará o přístupy ke konkrétním datům, jejich komplexnost a souvislost. $DBMS$ také obstarává a zpracovává dotazy a požadavky uživatelů na záznamy v databázi. Speciálním případem $DBMS$ je *Relational Database Management System* užívaný pro relační databáze. Tento systém byl vytvořen v 70. letech ve firmě IBM. Na následujícím schématu je vidět způsob fungování $RDBMS$. [3]



Obrázek 2.3: Relační databázový systém [9]

Na obrázku 2.3 je znázorněno fungování relačního databázového systému. V levé části jsou příklady různých typů klientů, kteří mohou s databází pracovat. Klient vyšle do vrstvy *RDBMS* dotaz v podobě SQL query. Management system pak zpracuje dotaz a zkontroluje, jestli má klient potřebná oprávnění pro takovýto dotaz. Dále tato vrstva komunikuje přímo s uloženými daty. Získá data požadovaná v dotazu, zpracuje je podle požadavku a vrátí je zpět klientovi. *RDBMS* vrstva by se dala popsat jako prostředník mezi klienty a uloženými daty jako takovými. Spravuje přístupy, zajišťuje bezpečnost a tak dále.

Zřejmě nejdůležitější vlastnost *RDBMS* se skrývá pod zkratkou **ACID**. Tato zkratka představuje 4 vlastnosti, které musí splňovat databázové transakce, o které se stará právě *RDBMS*. Konkrétně se jedná o atomicitu, konzistenci, izolovatelnost a trvalost.

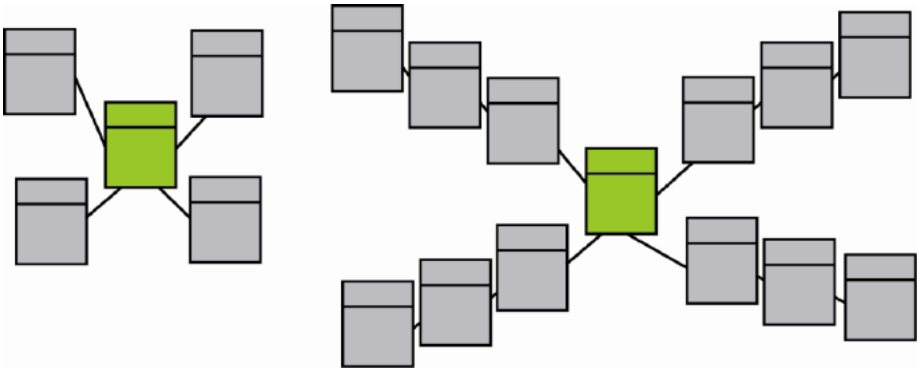
- Atomicita - databázová transakce by měla být dále nedělitelná.
- Konzistence - během transakce nesmí být porušena integrita dat.
- Izolovatelnost - v rámci transakce jsou operace uvnitř skryty před jakýmkoliv vnějším zásahem. Pokud dojde k takovémuto zásahu, musí být transakce vrácena tzv. *rollback*.
- Trvalost - změny, které byly úspěšně provedeny transakcí, jsou trvale uloženy v databázi.

2.3.2 MySQL

Pro tuto práci byla zvolena jako nejvhodnější MySQL databáze od společnosti Oracle. MySQL databáze byla zvolena pro svou jednoduchou obsluhu, dostupnost zdarma, rozšířenost a přehlednou dokumentaci. Součástí open-source balíčku od společnosti Oracle je také program MySQL Workbench, který byl použit pro návrh databáze a jejího E-R modelu.

Způsoby návrhu databází

Existuje více různých přístupů k návrhu databází. V tomto případě se rozhodovalo mezi takzvaným hvězdicovým schématem (star schema) a vločkovým schématem (snowflake schema).



Obrázek 2.4: Obecný příklad star schématu (vlevo) a snowflake schématu (vpravo) [10]

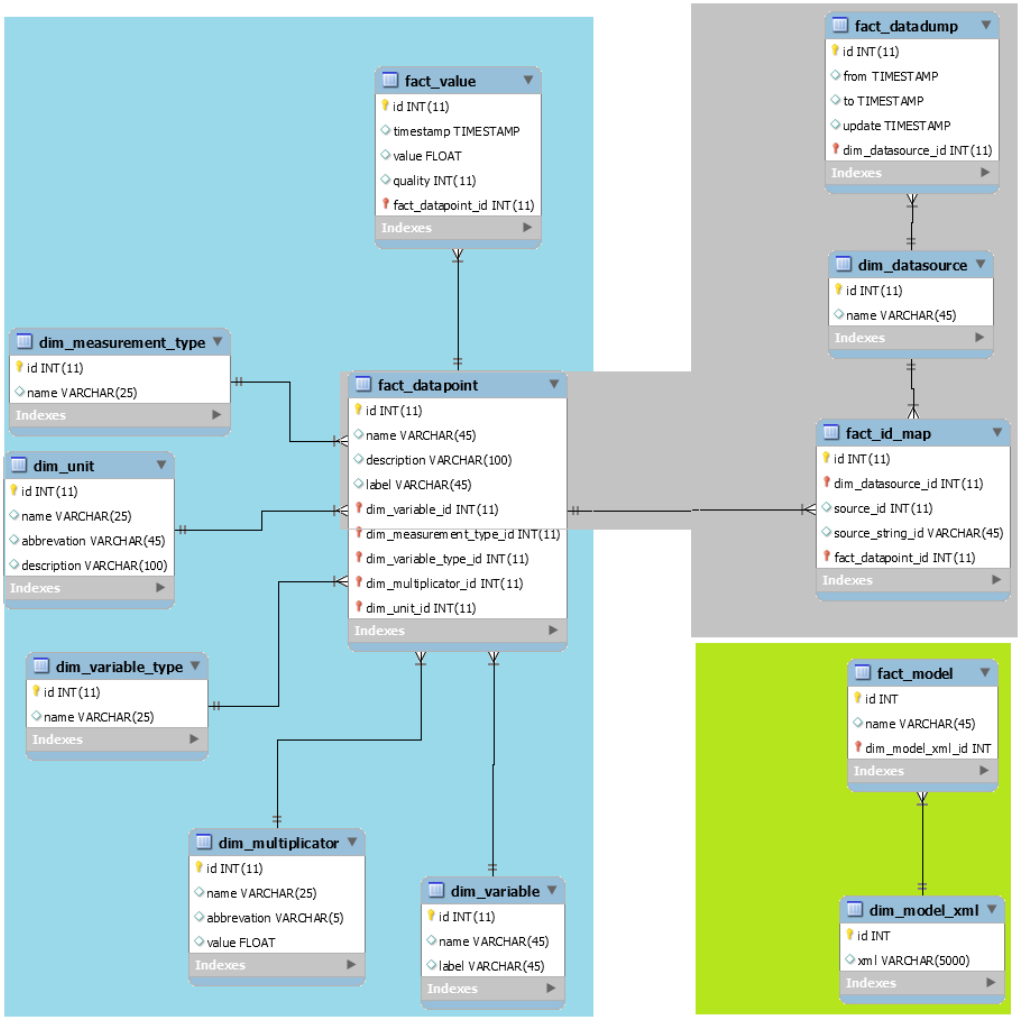
Snowflake i star schéma využívají faktických a dimenzionálních tabulek. Dimenzionální tabulky slouží pro ukládání statických dat společných pro všechny záznamy (tzv. dimenze). Faktické tabulky pak slouží pro ukládání jednotlivých záznamů jako takových s využitím statických informací z tabulek dimenzionálních.

Hlavní rozdílnost těchto dvou přístupů je v normalizaci a složitosti query dotazů. Ve schématu snowflake je každá dimenzionální tabulka přizpůsobena konkrétnímu typu záznamu. Tím pádem jsou data organizovanější a je zde menší riziko porušení integrity dat. Normalizované tabulky také zabírají méně datového místa. Další výhodou je snadnější údržba normalizovaných tabulek.
Složitost query dotazů je ale ve snowflake přístupu větší. Vzhledem k normalizaci musíme prostupovat hlouběji skrze dimenzionální tabulky, abychom se dostali ke konkrétním záznamům. Star přístup je z hlediska rychlosti query příkazů efektivnější, je u něj ale větší riziko vzniku problémů při získávání dat. [8]

Po zvážení všech možností byl pro návrh databázového systému zvolen snowflake přístup .

2.3.3 Databázový model

Databázový systém je jednou ze stěžejních částí této práce. Po zvážení předchozích bodů došlo k návrhu schématu této databáze. Jedná se o schéma typu snowflake, které obsahuje 12 různých datových tabulek. Tabulky mezi sebou mohou, ale nemusejí být svázány.
Databáze se dá rozdělit na tři části podle jejich účelu, samostatnosti a vzájemné provázanosti jednotlivých tabulek, viz následující schéma na obr.2.5.



Obrázek 2.5: Návrh snowflake schématu databáze

Základní modrá část obsahující energetická data sestává z dimenzionálních tabulek obsahujících informace o jednotkách veličiny, typu veličiny, proměnné a podobně, které se dále využívají při vytváření a zapisovaní konkrétních datových bodů. Tato hlavní část obsahuje faktické tabulky s jednotlivými datovými body (*fact_datapoint*) a všemi záznamy naměřených hodnot (*fact_value*). Vazbou typu 1:N je v tabulce *fact_value* mnoho hodnot pro různé časové okamžiky vždy svázáno s datovým bodem, ze kterého pocházejí. Tato provázanost je elementární vlastností, která je nutná pro následné využívání hodnot a jejich skládání do časových řad.

Šedá část schématu označuje takzvanou dumpovou část databáze, kde jsou uloženy nezbytné informace ke vkládání dat do databáze. Zároveň také obsahuje informace o posledním přelití dat, tudíž z ní můžeme zjistit i aktuálnost dat uvnitř databáze. Další podstatnou součástí této části je tabulka $fact\_id\_map$, kde jsou svázána jednotlivá *id* datových bodů uvnitř databáze s *id* datových bodů v jejich původních zdrojích. Této funkcionality se využívá například v kapitole *kontextový model*, kde je potřeba na základě původních *id* získat nová. V neposlední řadě je v této části také obsažena informace o původu vkládané informace (*dim_datasource*).

Poslední zelená část schématu je věnována segmentu databáze věnující se kontextovým modelům. Z této části je možno načíst libovolný kontextový model a dále jej zpracovávat. Více v kapitole *kontextový model*.

Detailní popis jednotlivých tabulek je k nalezení v přílohách.

2.4 Zpřístupnění datového zdroje

2.4.1 Anotace datových bodů

Pro potřeby vytváření energetických analytických služeb a zároveň také pro vytvoření nové databáze je potřeba datové body vhodně anotovat. V původní databázi EMS je však anotace nedostatečná, proto není možno ji do nové databáze vytvořit automaticky, ale je nutno anotaci udělat ručně. Zde bylo navázáno na dřívější aktivity katedry kybernetiky, kdy byla tato anotace již ručně vytvořena a uložena v podobě souboru *registr-ems*, který je ve formátu .csv. Soubor obsahuje přes 1800 anotovaných bodů, které zahrnují následující atributy: $id, multiplicator, description,$ $samplingperiod, variabletype, unit, medium, samplingtype$. Přičemž jednotlivé atributy mohou nabývat následujících hodnot.

- *id* - číselné id
- *description* - obecný popis datového bodu
- *sampling period* - vzorkování v hodnotách minut(zpravidla 10 minut)
- *unit* - jednotky hodnoty: $nespecifikovano, ^\circ C, \%, kPa$
- *variable* - veličina: $nespecifikovano, teplota, tlak, poloha$
- *variableType* - typ proměnné: $nespecifikovano, merena, ridici, virtualni$
- *samplingType* - typ měření: $nespecifikovano, rozdilove, prubehove$
- *medium* - médium: $nespecifikovano, voda, vzduch, glycol$

Zde je ukázka ze souboru *registr-ems*:

ID	multiplicator	name	description	sampling period [min]	unit	variable	variable type	unit	medium	sampling type
1	10	VS-Teplota primér přívodní		10	teplota	merena	°C	voda	prubezne
2	10	VS-Teplota primér vratná		10	teplota	merena	°C	voda	prubezne
3	10	VS-Teplota UT výstupní		10	teplota	merena	°C	voda	prubezne
4	10	VS-Teplota UT vratná		10	teplota	merena	°C	voda	prubezne
5	10	VS-Teplota UT sever		10	teplota	merena	°C	voda	prubezne
6	10	VS-Teplota UT jih		10	teplota	merena	°C	voda	prubezne
7	10	VS-Teplota venkovní		10	teplota	merena	°C	vzduch	prubezne
8	10	VS-Referenční prostor UT sever		10	teplota	merena	°C	voda	prubezne
9	10	VS-Referenční prostor UT jih		10	teplota	merena	°C	voda	prubezne
10	10	VS-Teplota výstup TUV		10	teplota	merena	°C	voda	prubezne
11	10	VS-Teplota chlazení výstupní		10	teplota	merena	°C	nespecifikovano	prubezne
12	10	VS-Teplota chlazení vratná		10	teplota	mereni	°C	nespecifikovano	prubezne
13	10	VS-Tlak v systému		10	tlak	merena	kPa	voda	prubezne
14	10	VS-Tlak diferenční		10	tlak	nespecifikovano	kPa	nespecifikovano	prubezne
15	10	VS-Výstup UT žádaná		10	nespecifikovano	ridici	nespecifikovano	voda	prubezne
16	10	VS-Teplota žádaná UT sever		10	teplota	ridici	°C	voda	prubezne
17	10	VS-Teplota žádaná UT jih		10	teplota	ridici	°C	voda	prubezne
18	10	VS-Výstup TUV žádaná		10	nespecifikovano	ridici	nespecifikovano	voda	prubezne
19	10	VS-Diferenční tlak žádaný		10	tlak	ridici	kPa	nespecifikovano	prubezne
21	10	SS01-Teplota UT radiátory		10	teplota	merena	°C	voda	prubezne
22	10	SS01-Teplota UT podlaha		10	teplota	merena	°C	voda	prubezne

Obrázek 2.6: Ukázka ze souboru, kde je uložena anotace datových bodů

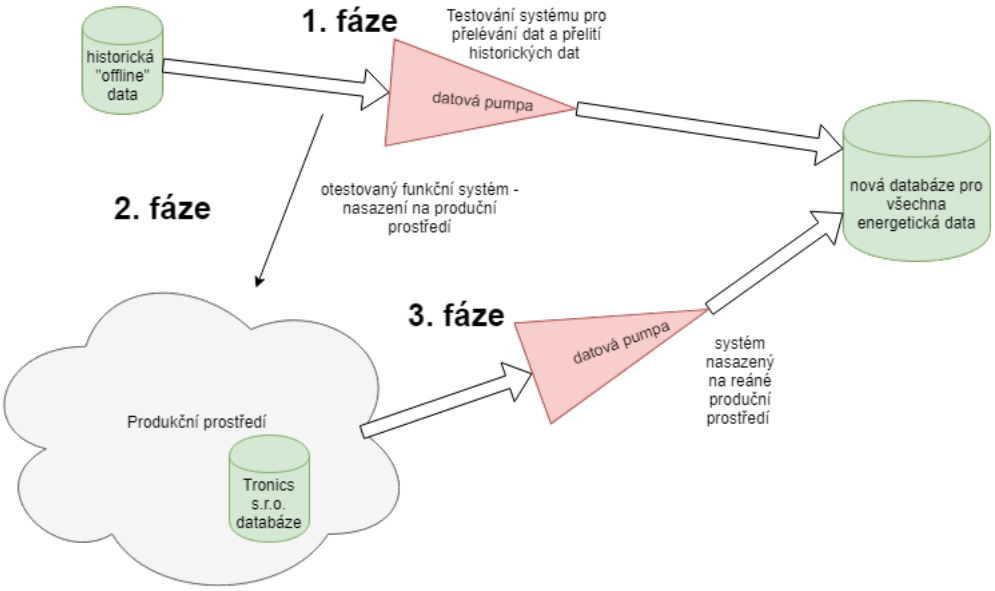
V zelené části souboru jsou data v původní databázi, v červené části jsou nově anotovaná data připravená pro následující automatický převod do nové databáze společné pro *EMS* i *Telemetrickou soustavu*.

2.4.2 Zisk měřených dat

Pro získávání měřených dat ze systému EMS byl vytvořen softwarový nástroj, který exportuje měření z původního EMS systému do nově vytvořené databáze. Celý nástroj je navržen tak, aby mohl být přímo nasazen na produkční prostředí a současný systém sběru energetických dat na ZČU. Jelikož je objem dat obrovský (1 rok obsahuje přibližně 100 milionů záznamů), není možné na produkčním prostředí přímo testovat navrhovaný systém. Z těchto důvodů bylo potřeba získat historická data v podobě záloh databáze, abychom mohli systém bezpečně testovat a abychom přeléváním historických dat nezahltili současný produkční systém. Po fázi testování a přelití historických dat bude systém nasazen zpět přímo do produkčního prostředí.

O datové úložiště EMS se stará firma Tronics s.r.o. Díky spolupráci jsme získali jejich historické databázové zálohy. Tyto zálohy jsou ve formě vnitřních formátů tabulek MySQL serveru. Díky jejich kompatibilitě s námi používaným MySQL serverem, jsme mohli takto offline získaná data vložit přímo do našeho MySQL serveru a dále s nimi pracovat v podobě relační databáze.

Záloha EMS databáze jako taková je pro potřeby splnění této práce nedostatečná. Dále musel být navržen nástroj, který bude převádět data z původní EMS databáze firmy Tronics s.r.o. do nově implementované databáze společné pro EMS i telemetrickou soustavu. Tento nástroj byl pojmenován *datová pumpa*. Znázornění postupu při vývoji datové pumpy:



Obrázek 2.7: Jednoduché schéma popisující fáze vývoje nástroje pro zisk měřených dat

Na schématu 2.7 lze vidět, že vývoj datové pumpy probíhal ve třech fázích. V první fázi se jednalo především o návrh, testování a plné zprovoznění systému datové pumpy a offline přelévání historických dat. Ve druhé fázi došlo k nasazení systému datové pumpy na produkční prostředí. Ve třetí fázi dochází k plnému běhu systému v produkčním prostředí.

Datová pumpa

Datová pumpa slouží k přenesení dat ze spodní externí datové vrstvy do vyšší databázové vrstvy, na které je možno stavět energetické analytics. Systémy pro správu dat jsou dodávány společnostmi Tronics s.r.o. a Data-Ing s.r.o. Systém pumpy je navržen za účelem spojení externích datových zdrojů do jedné databáze pod plnou kontrolou ZČU. Systém datové pumpy je navržen tak, aby měl společné rozhraní pro jakýkoliv externí zdroj a díky této vlastnosti může být datová pumpa použita pro libovolný existující i budoucí zdroj energetických dat. Rozhraní a koncová část pumpy společná pro jakýkoliv zdroj dat byla implementována a navržena ve spolupráci s Lukášem Soukupem [4]. Tato práce se konkrétně specializuje na datovou pumpu a její napojení na EMS databázi.

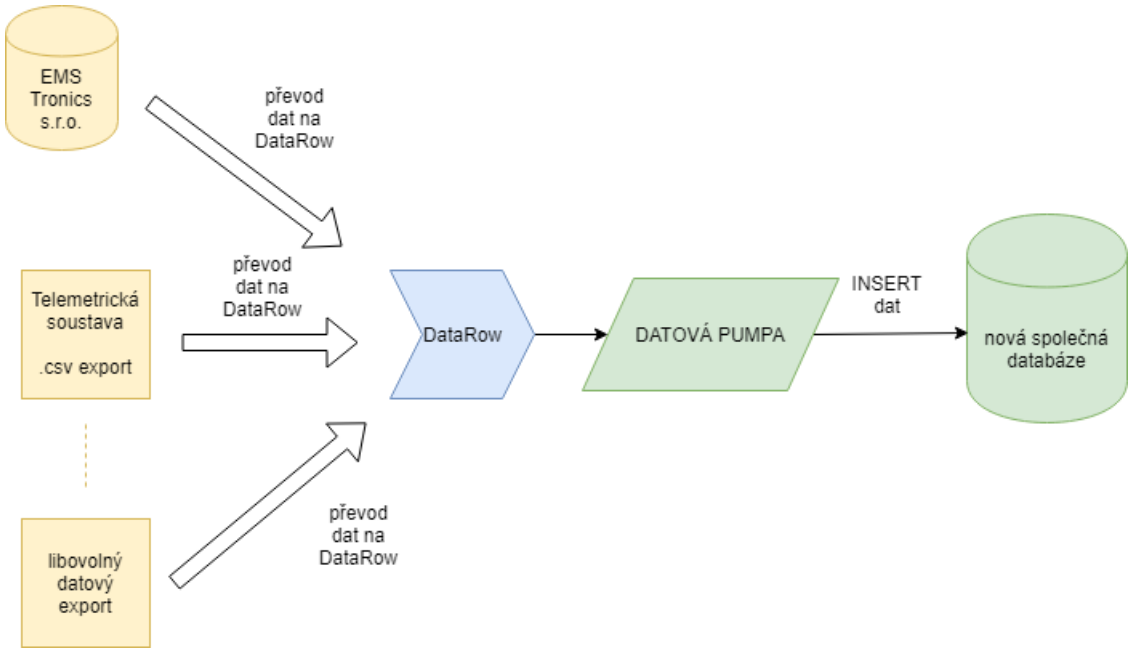
Rozhraní datové pumpy funguje na základě třídy *DataRow*, která obsahuje *ArrayListy* s hodnotami $timestamp, quality, value, id\_datapoint$. V této podobě se dají do datové pumpy vkládat data z libovolného zdroje. Třída *DataRow*:

```
package DataRow;
import java.sql.Timestamp;
```

```
import java.util.ArrayList;

public class DataRow {
    public ArrayList<Timestamp> timestamp = new
       ArrayList<>();
    public ArrayList<Integer> quality = new
       ArrayList<>();
    public ArrayList<Float> value = new ArrayList<>();
    public ArrayList<String> id_datapoint = new
       ArrayList<>();
}
```

Třída *DataRow* představuje způsob uložení dat podporovaný datovou pumpou. Při načítání z datového zdroje se z jednoho záznamu vyberou čtyři základní údaje ($timestamp, quality, value, id\_datapoint$), které se uloží do příslušných ArrayListů pod stejným indexem. Třída *ArrayList* je implementací datového typu seznam, ale má dynamicky se měnící velikost.

Na obrázku 2.8 je uvedeno rozhraní datové pumpy a její napojení na libovolný externí zdroj.

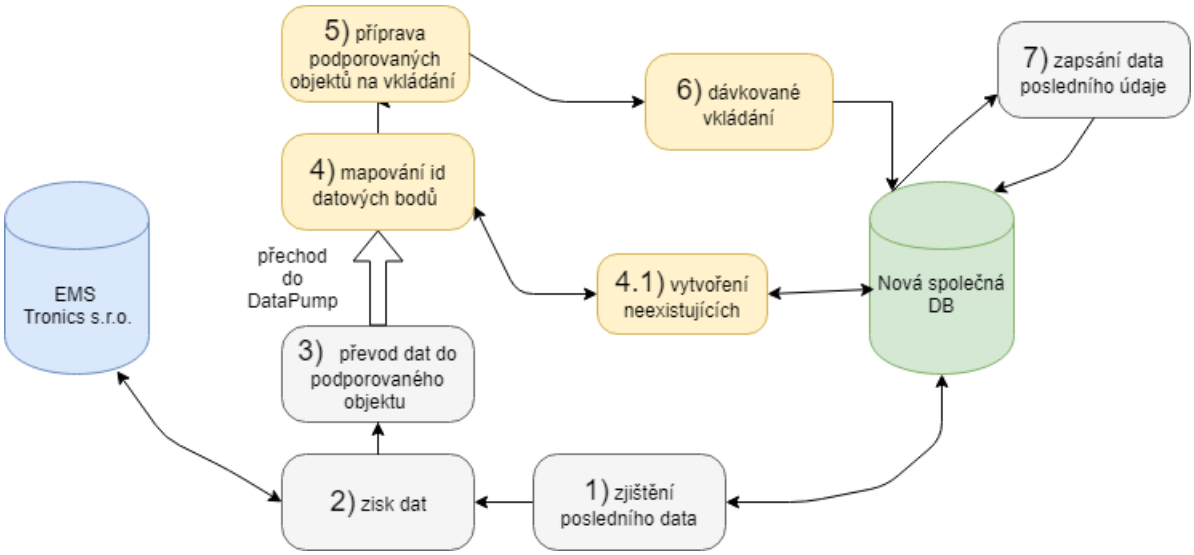


Obrázek 2.8: Schéma rozhraní datové pumpy

Na obrázku 2.8 se v levé části nachází spodní datová vrstva, kde jsou energetická data spravována firmami třetích stran. V této fázi jsou data ještě rozdělena samostatně podle jejich původu (EMS, telemetrická soustava a případně další). Po získání těchto dat dochází k jejich úpravě (znázorněno šipkami) na společný datový formát, který podporuje datová pumpa. Přesněji se jedná o třídu *DataRow*. Po převodu na společný formát jsou data z jednotlivých datových zdrojů poslána do společné části

datové pumpy. V této sekci společné datové pumpy se data zpracují a upraví pro novou databázi společnou pro všechny původní datové zdroje. V poslední fázi dojde k jejich vložení do společné cílové databáze.

Schéma funkčnosti části převádějící EMS data přes datovou pumpu do nové cílové databáze:



Obrázek 2.9: Schéma postupných kroků přenesení dat s EMS databáze do nové cílové databáze

Na obrázku 2.9 je znázorněno v jednotlivých krocích, jak dochází k převodu EMS dat do nové společné cílové databáze. Zelené objekty představují jednotlivé databáze, obdélníkové objekty jsou kroky nezbytné k převodu dat. Kroky se dají rozdělit podle jejich příslušnosti. Šedé kroky jsou specifické jen pro pumpování EMS dat, žluté kroky se vztahují ke společné části datové pumpy, tím pádem jsou stejné pro pumpování dat z libovolného datového zdroje.
Popis jednotlivých kroků pumpování EMS dat je následovný. V prvním kroku dojde dotazem na cílovou databázi ke zjištění data posledního záznamu uvnitř této databáze (tabulka *fact_datadump*). Tento časový údaj představuje počáteční datum nového bloku dat, který chceme převést. Z důvodu velkého množství dat je převod nutné provádět po týdenních blocích. V kroku číslo dvě dojde napojením na EMS databázi k zisku týdenního datového bloku z tabulky *registrace*. V následujícím kroku je nutné převést načtená data do společného datového objektu, který je podporován společnou částí datové pumpy. Po tomto převodu dojde k vložení dat do společné části datové pumpy. V prvním kroku datové pumpy (celkově krok číslo 4), je nutno namapovat příchozí data do nové databáze. Jelikož jsou *id* datových bodů v cílové databázi jiná než v databázi zdrojové, nachází se v cílové databázi tabulka *fact_id_map*, kde jsou obě *id* uložena a je díky tomu možné přemapovat stará *id* na nová. Pokud je datový bod pro cílovou databázi nový, dojde k jeho vytvoření do tabulky *fact_datapoint* a namapování jeho nového *id* do *fact_id_map*, aby se již při ukládání dalších hodnot pro tento datový bod ukládala data pod správným *id* a nevytvářel se duplikátní datový bod. V pátém kroku podle obrázku 2.9 dochází k

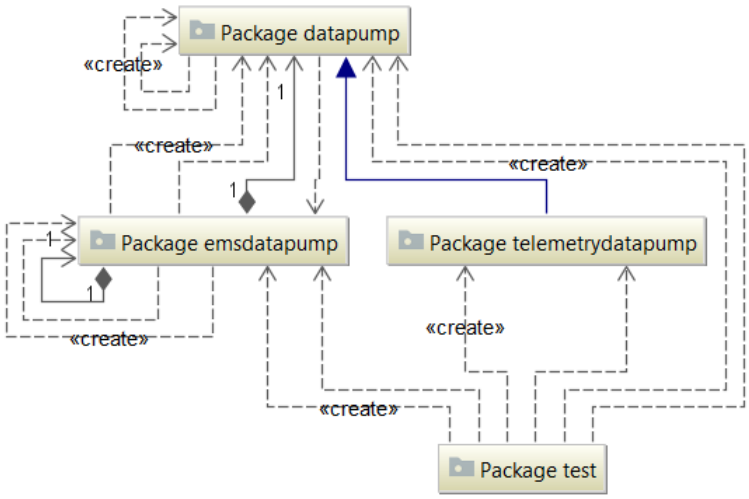
přípravě příchozích hodnot s již přemapovanými *id* na vložení do cílové databáze. V posledním kroku datové pumpy (krok 6) dochází k dávkovanému vkládání týdenního bloku dat do cílové databáze. Dávkované vkládání bylo zvoleno z důvodu bezpečnosti přenosu, větší rychlosti a snazšího odchytávání chyb během vývoje. Konstanta pro dávkované vkládání byla určena na 10 000 řádků pro jednu dávku. Po ukončení vkládání přichází poslední krok specifický pro EMS data, a to sice uložení data posledního nového záznamu do cílové databáze (tabulka *fact_datadump*), aby byla hotova příprava na další vkládání. Těmito kroky lze shrnout fungování převodu EMS dat do cílové databáze.

Ukázka kódu z implementace napojení na EMS Tronics s.r.o. databázi a převod dat na *DataRow* formát. Konkrétně se jedná o třídu *SourceDatabaseConnector*.

```
package emsdatapump;
import datapump.DataRow;

import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;

public class SourceDatabaseConnector {
    String url;
    String user;
    String pw;
    DataRow mydata = new DataRow();


    public SourceDatabaseConnector(String url, String
       user, String pw) {
        this.url = url;
        this.user = user;
        this.pw = pw;
    }
    /**Returns data  of 7 days from old db
     * */
    public DataRow readDatapointsID(String dateFrom,
       String dateTo) {
        int code;
        String mark;
        String value;
        String query = "SELECT * FROM db_old.registrace
           WHERE mark >= " +dateFrom+" and mark < "+
           dateTo;
        try (Connection con =
           DriverManager.getConnection(this.url,
           this.user, this.pw);
```

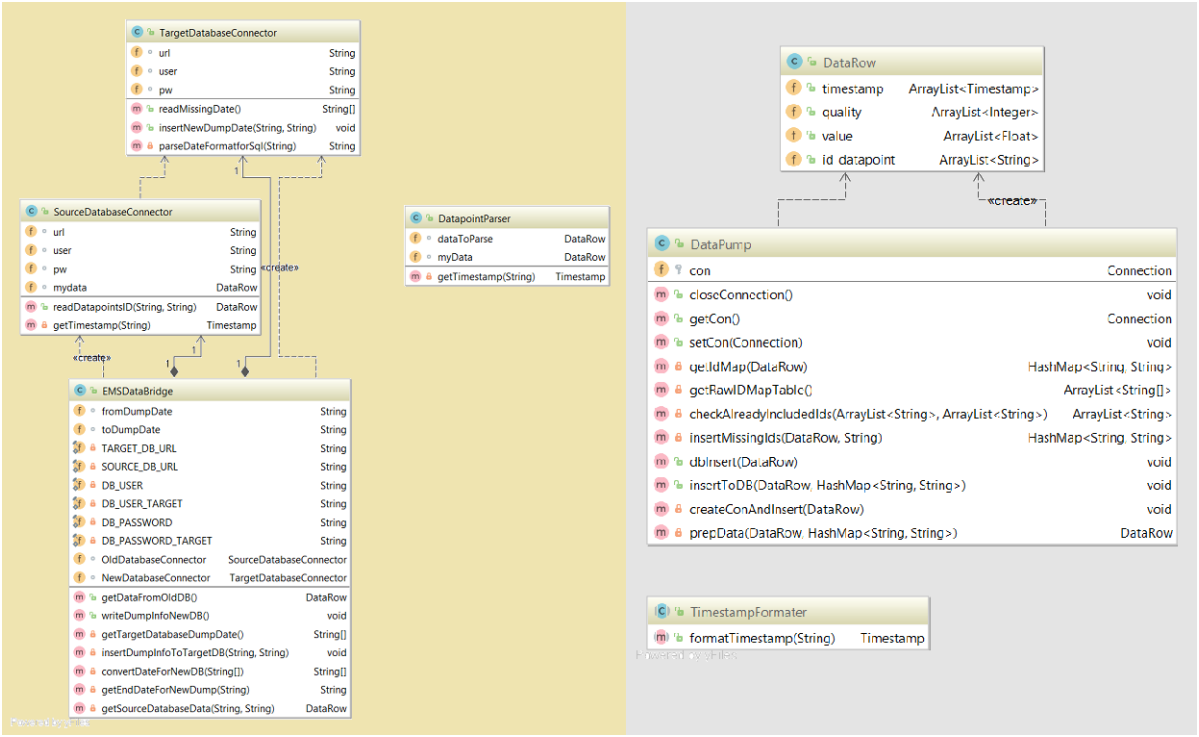
```
                Statement st = con.createStatement();
                ResultSet rs = st.executeQuery(query))
            {
                while (rs.next()) {
                    code = rs.getInt(1);
                    mark = rs.getString(2);
                    value = rs.getString(3);
                    mydata.value.add(Float.parseFloat(value));
                    mydata.id_datapoint.add(Integer.toString(code));
                    mydata.timestamp.add(getTimestamp(mark));
                    mydata.quality.add(0);
                }
                con.close();
            } catch (SQLException ex) {
                Logger lgr = Logger.getLogger(
            TargetDatabaseConnector.class.getName());
                lgr.log(Level.SEVERE, ex.getMessage(), ex);
            }

            return mydata;
        }
            .
            .
            //support methods
            .
            .
}
```

Následuje UML schéma datové pumpy na úrovni balíčků.



Obrázek 2.10: UML schéma balíčků

Na obrázku 2.10 lze vidět UML schéma balíčků programu datové pumpy pro dva datové zdroje. Balíček *datapump* je společný pro oba datové zdroje, obsahuje společnou část napojení na cílovou databázi a dávkové vkládání. Balíčky *emsdatapump* a *telemetrydatapump* jsou balíčky zpracovávající data z původních zdrojů a transformují je pro datovou pumpu jako takovou. Balíček *test* obsahuje spouštěcí třídy, které obsahují metodu *main*.



Obrázek 2.11: Diagram tříd části pumpující EMS data

Na obrázku 2.11 lze vidět diagram tříd uvnitř balíčku *emsdatapump* (levá strana) a *datapump* (pravá strana), které dohromady pumpují data ze zdroje EMS dat do cílové databáze. Na obrázku jsou zobrazeny i metody a atributy jednotlivých tříd a jejich závislosti v rámci balíčků.

Funkčnost pumpování dat by se dala zjednodušeně popsat následujícím způsobem:

- Dotaz třídy *TargetDatabaseConnector* na poslední dump databáze, který úspěšně proběhl → zisk počátečního data pro nový dump do cílové databáze.
- *SELECT* z původní EMS databáze v rozsahu dat za období jednoho týdne počínajícího dnem získaným v předchozím kroku → zisk přibližně 2 až 3 milionů záznamů pro přesun do cílové databáze (třída *SourceDatabaseConnector*).
- Zpracování data loadu třídou *DatapointParser* → zisk dat připravených pro zápis do cílové databáze.

- Kontrola zda nová databáze obsahuje informace o datových bodech, které jsou obsaženy v data loadu a jejich případný zápis do nové databáze.
- *INSERT* data loadu do nové databáze.
- Zapsání informací o provedeném dumpu do nové databáze.

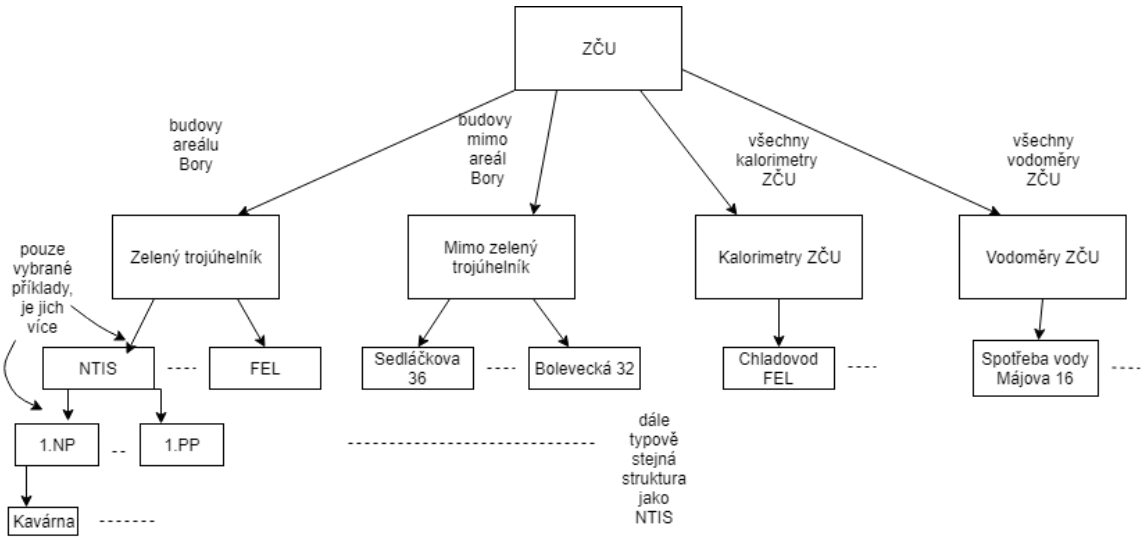
Kapitola 3

Kontextový model energetických dat

3.1 Obecný kontextový model

Kontextový model specifikuje souvislosti mezi jednotlivými datovými body uloženými v konsolidované databázi. Kontextů jako takových může být mnoho jako například zachycení vztahů řídících a řízených jednotek v podobě stavových modelů, které určují vztahy nejen mezi vstupy a výstupy systému, ale i mezi jeho stavovými veličinami.

V této práci byl využit kontextový model, který definuje hierarchickou strukturu. Model tedy určuje vztahy mezi nadřazenými a podřazenými entitami, díky tomu se dají snadno provádět operace jako například agregace jednotlivých částí. Hierarchický model byl zvolen i díky jeho systematičnosti odpovídající struktuře budovy (stromová struktura). Tímto se myslí například vztah budova - patra - místnosti. Na následujícím schématu je vidět stromová struktura zachycující organizaci datových bodů měřených v areálech ZČU.



Obrázek 3.1: Struktura budov/datových bodů ZČU

3.2 Hierarchický kontextový model ZČU

Pro vytvoření kontextového modelu budovy je potřeba zachytit strukturu budovy, její jednotlivé části a vztahy mezi nimi. Tato struktura je obsažena v databázi spravované firmou Data-Ing s.r.o., ze které nám byla poskytnuta historická data měření. K vytvoření kontextového modelu byl dodán a použit export jejich databáze v podobě CSV souborů. Každá část budovy v exportu databáze obsahuje své vlastní *id*, jméno a *id* svého rodiče. Dále je možno se přes cizí klíče dotázat na datapointy příslušící jednotlivým prvkům v této struktuře a získat tak jejich jméno a *id* nezbytné pro vytvoření kompletního modelu.

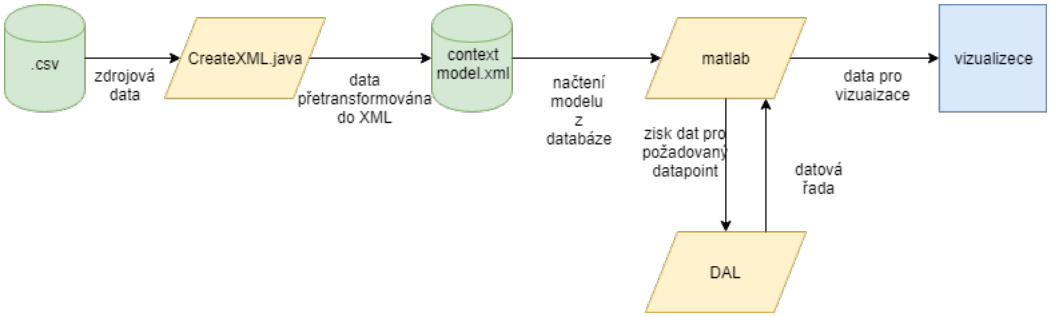
V rámci návrhu způsobu uložení kontextového modelu byly zvažovány dvě možnosti (JSON, XML) jak uložit různé prvky a vztahy mezi nimi. Výhody souborů standardu typu JSON jsou ve snazším zpětném zpracovávání a menší prostorové náročnosti na úložiště. Výhodami XML souborů jsou na první pohled přehledná a viditelná stromová struktura a jasně daná hierarchie tagů. Z důvodu názornosti a přehlednosti bylo zvoleno uložení do formátu standardu XML.

Standard XML byl blíže popsán v sekci 1.3. Kontextový model byl navržen se strukturou dvou typů tagů *collection* a *datapoint*. Tagy *collection* označují celek, který může obsahovat další *collection* nebo *datapoint*. Tag *datapoint* je tag obsahující jméno *datapointu* a jeho *id*. Za tagem *datapoint* už poté nemůže následovat žádný další *datapoint* ani *collection*. Zde je příklad uložení prvního patra budovy NTIS.

```
<collection name = "NTIS">
  <collection name = "1NP">
    <collection name = "Fasáda">
      <datapoint name = "Teplota" id = "f17cd3cd-8b57-4de5-ae66-c3f83293c440" />
    </collection>
    <collection name = "Šatna">
      <datapoint name = "Teplota" id = "219fb0ad-b04e-4729-a9f3-99d66be1c1a6" />
    </collection>
    <collection name = "Kavárna">
      <datapoint name = "Spotřeba el. energie" id = "33e58d22-bbc2-4e4e-b28e-96d942529830" />
      <datapoint name = "Teplota" id = "7f3b4c8e-5089-4a45-8b70-648723a97415" />
    </collection>
  </collection>
```

Obrázek 3.2: Datové body v 1. nadzemním podlaží budovy NTIS

Na příkladu zachyceném na obrázku 3.2 lze dobře vidět hierarchii uvnitř kontextového modelu. Nad vrstvou *collection NTIS* jsou ještě další vrstvy oblastí celé ZČU, pro tuto ukázku je to však nepodstatné. *Collection NTIS* představuje vrstvu na úrovni budov. Budovy se dále dají dělit na patra - *collection 1NP*. Každé patro pod sebou má své další části, jako například místnosti nebo vnější měření (*collection Fasáda, collection Šatna* a tak dále). Pod touto vrstvou se vyskytují samotné datové body se svými *id* viz *datapoint Teplota, datapoint Spotřeba el. energie* a tak dále. Na následujícím schématu se nachází znázornění postupu vytváření a zpracovávání kontextového modelu.



Obrázek 3.3: Diagram vytvoření a zpracování kontextového modelu

Pro potřeby vytvoření kontextového modelu byl v jazyce Java implementován program *CreateXML*. Tento program načte CSV soubor, který obsahuje vztahy mezi objekty ZČU. Následně pomocí rekurzivního vyhledávání zjistí všechny závislosti typu rodič-dítě. Po zjištění všech závislostí dojde opět pomocí rekurze k vygenerování kontextového modelu všech budov a datových bodů ZČU. Následující zpracování kontextového modelu bylo implementováno v jazyce Matlab, kvůli jeho optimalizaci pro vizualizace a možnosti volání libovolných knihoven vytvořených v jazyce Java. V Matlabu se do datového typu *struct* načte kontextový model z XML. Datový typ *struct* v sobě obsahuje informace o vnitřních vztazích typu rodič - dítě. Díky těmto informacím lze pak pomocí rekurze vyhledávat libovolný datový bod uložený v kontextovém modelu včetně jeho *id*. Pomocí *id* se z Data Access Layeru (pozn. DAL vytvořen Lukášem Soukupem [4]) získává odpovídající datová sada, která lze následně vizualizovat.

3.3 Detailní popis vytváření kontextového modelu

V jazyce Java byl vytvořen program, který automaticky hierarchický kontextový model vytváří z CSV záloh databáze firmy Data-Ing s.r.o. Jeho základními kroky jsou:

- Načtení souboru objekt.csv.
- Pomocí rekurzivního volání se spárují všechny entity, které jsou ve vztahu rodič a dítě.
- Načtou se soubory prvek.csv a vlastnost.csv, které dohromady dávají provázanost *id* jednotlivých datových bodů v databázi a prvků, ve kterých se nacházejí. Tyto dva soubory se načítají do dvou *HashMap*, pomocí jichž se přiřadí prvkům konkrétní *id* vlastnosti. *Id* vlastnosti je totožné s *id* v cílové databázi v tabulce $fact\_id\_map$.
- Soubor prvek-objekt.csv je ve své podstatě reprezentací databázové vazby M:N mezi instancemi objektů a prvků. V tomto kroku se objektům přiřadí jeden nebo více prvků, které jim odpovídají. Prvky obsahují informace o *id* datového bodu ze souboru vlastnost.csv.
- Opět rekurzivním voláním se do předpřipravené XML struktury vkládají objekty pod tagem *collection*, pokud jsou koncové a mají pod sebou datové body, vloží se do struktury tyto body pod tagem *datapoint* s atributy *name* a *id*.
- Celá takto vytvořená struktura se uloží i s předpřipravenou XML hlavičkou do context-model.xml. Takto vytvořený hierarchický kontextový model splňuje i vlastnosti validního XML souboru.

Zde je uveden příklad z kódu pro vytváření kontextového modelu. Přesněji se jedná o rekurzivní metodu na vytváření těla XML souboru pro uložení hierarchického kontextového modelu.

```
public static String createStructure(Objekt objekt,
   String preambule,CSVCollection csvCollection,
   String xml_string) {
      try {
          xml_string += (preambule + "<collection name
             = \"" + objekt.getNazev() + "\">\n");
          for (Prvek prvek : objekt.getPrvekList()) {
              for (Vlastnost vlastnost :
                 prvek.getVlastnostList()) {
                  xml_string += (preambule + "  " +
                     "<datapoint name =
                     \""+vlastnost.getNazev()+"\" id =
                     \"" + vlastnost.getId() + "\"
                     />\n");                  }
          }
```

```
            // Loop over children
            for (Objekt child : objekt.getPotomci()) {
                xml_string = createStructure(child,
                   preambule + "  ", csvCollection,
                   xml_string);
            }
            xml_string += (preambule +
               "</collection>\n");
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        return xml_string;

    }
```

Kapitola 4

Energetický model budov FAV

4.1 Obecný popis

Energetický model budov Fakulty aplikovaných věd je zaměřen na modelování spotřeby elektrické energie a byl vytvořen na základě hierarchického kontextového modelu budov FAV popsaného v kapitole 3. Kontextový model budovy v sobě obsahuje strukturu a hierarchii datových bodů včetně elektrických, tudíž je velice vhodný pro vytvoření energetického modelu.

Existuje velké množství typů modelů pro potřeby energetického modelování, například jsou to modely: stavové, armax, first principle a podobně. V této práci byl zvolen přístup založený na typických denních diagramech. Tento model se také nazývá typický denní průběh spotřeby (zkratka TDD). Tento přístup byl zvolen pro jeho výpočetní jednoduchost a z důvodu znalosti a dostupnosti kompletních dat, ze kterých je model sestavován. Především díky znalosti kompletních dat se nemusí využívat k sestavení modelu například složitých rekurzivních výpočtů u stavového modelu a je možné sestavit model typických denních diagramů.

Typický denní diagram spotřeby

TDD model je založený na znalosti veškerých dat získaných měřením elektrické spotřeby za dané časové období, ze kterých jsou vypočteny střední hodnoty a směrodatné odchylky.

Nejprve dojde k výběru dat za určité období. Následně dochází k výpočtu středních hodnot přes všechny časové okamžiky pro zvolené období. Vektor těchto středních hodnot pak udává výslednou křivku modelu typické denní spotřeby. Pro úplnost pravděpodobnostního popisu se obdobným způsobem vypočítává směrodatná odchylka. Směrodatná odchylka zahrnuje variabilitu vstupních dat a částečně nepřesnost modelu. Směrodatná odchylka tedy vypovídá o hodnotách, jakých může ve skutečnosti spotřeba v jednotlivých dnech dosahovat. K výpočtu středních hodnot a směrodatných odchylek závislých na časovém okamžiku t a vybraném období k se používají následující

vzorce.

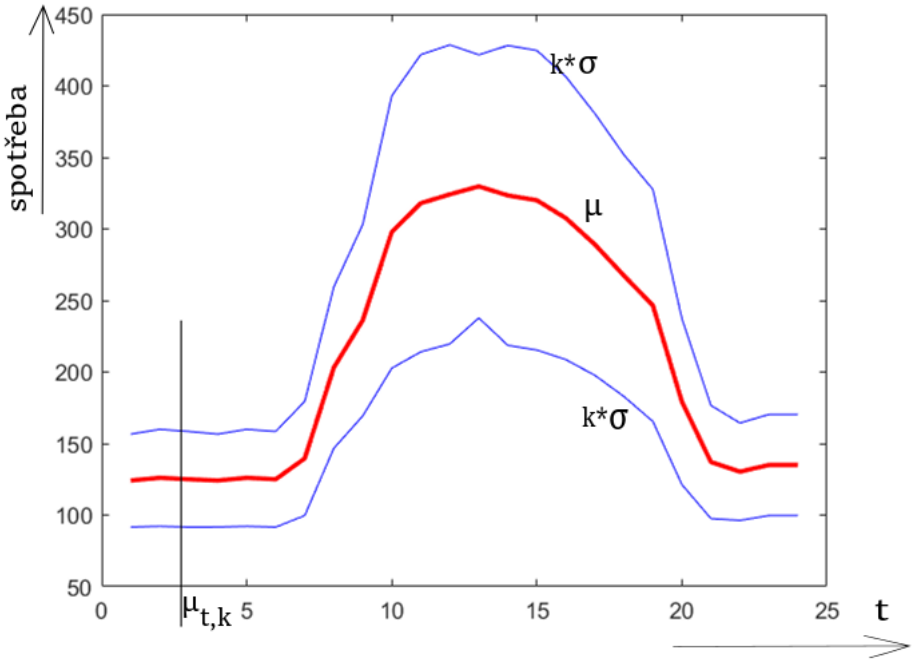
$$\mu_{t,k} = \frac{1}{n_{t,k}} \sum_{i=1}^{n_{t,k}} x_{t,k,i} \ , \tag{4.1}$$

kde $t \in \{0, 1, \ldots, 23\}$, $k \in \{\{jaro, pracovni\}, ..., \{zima, nepracovni\}\}$ a i je index vzorku v datové sadě, který představuje první až n-tou hodnotu ve sloupci.
Vzorec pro výpočet směrodatné odchylky σ je:

$$\sigma_{t,k} = \sqrt{\frac{1}{n_{t,k} - 1} \sum_{i=1}^{n_{t,k}} |x_i - \mu_{t,k}|^2} \ , \tag{4.2}$$

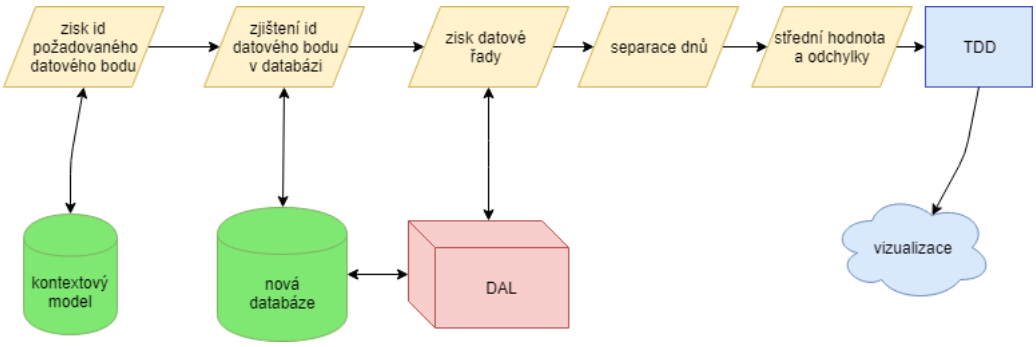
kde $\mu_{t,k}$ je střední hodnota pro daný časový okamžik, zbytek indexů je totožný jako při výpočtu střední hodnoty.

Na základě vypočtených středních hodnot a směrodatných odchylek je možno vytvořit energetický model spotřeby typu TDD. Vektor středních hodnot dává střední průběh denní spotřeby a směrodatná odchylka při použití $\pm 2\sigma$ dává pásmo, které pokrývá 95% výskytu hodnot. Na následujícím grafu je obecná ukázka modelu TDD.



Obrázek 4.1: Obecný TDD model

Na obrázku 4.1 je znázorněn obecný TDD model. Svislá osa vyznačuje hodnotu spotřeby. Vodorovná osa určuje čas, v tomto případě se jedná o hodiny v rámci jednoho dne. Modré křivky jsou kvantily, které ohraničují pásmo výskytu hodnot v rozmezí velikosti kvantilu. Velikost kvantilu je daná konstantou k. Červenou barvou je vyznačený průběh střední hodnoty spotřeby v rámci požadovaného typu dne. Hodnota $\mu_{t,k}$ je střední hodnota spotřeby v konkrétním čase a kategorii.
Na obrázku 4.2 je znázorněno schéma generování energetického modelu typu TDD.



Obrázek 4.2: Kroky při generování TDD modelu

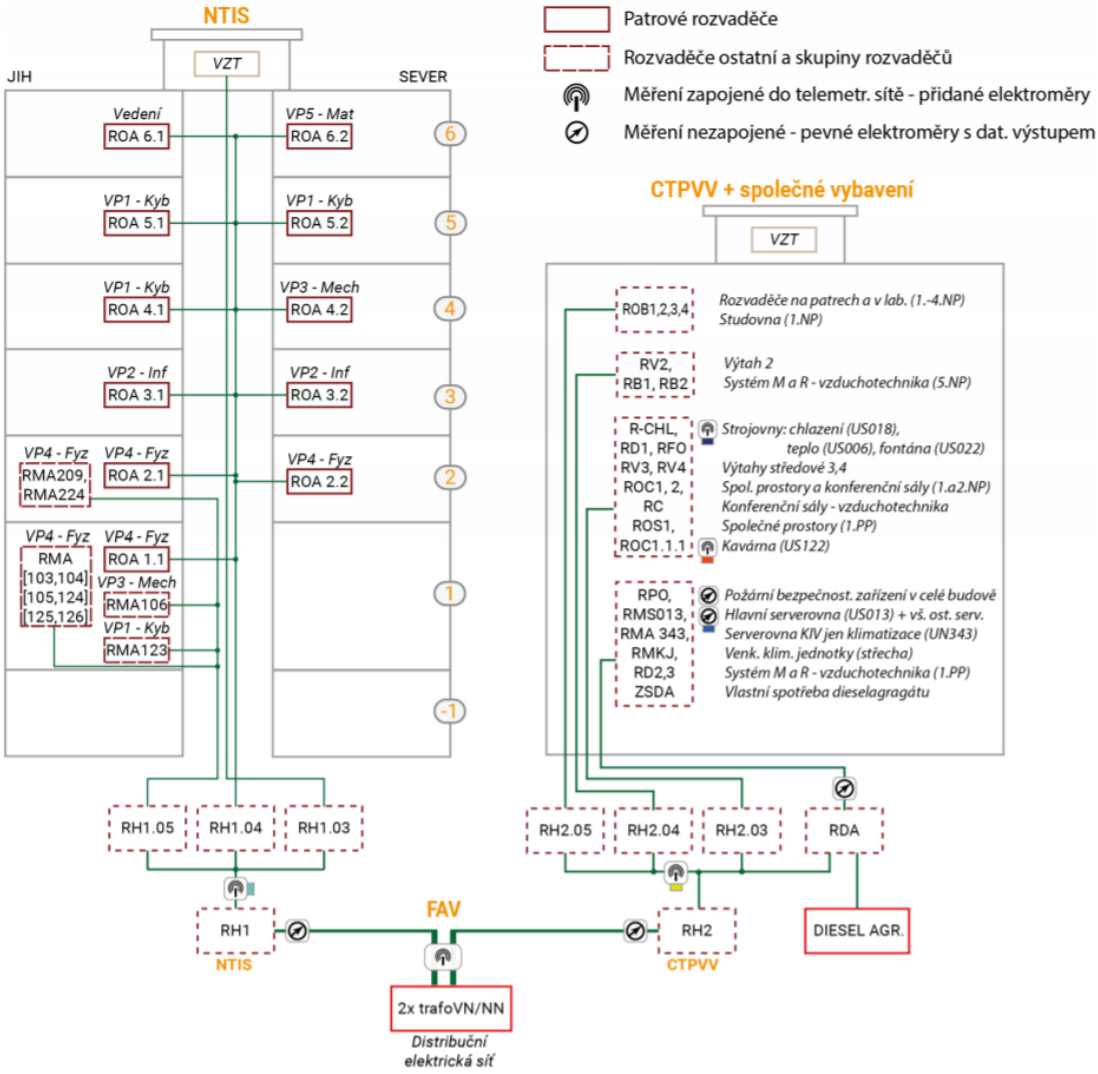
V prvotní fázi je zjišťováno, pomocí vyhledávání dle jména, umístění chtěného datového bodu v kontextovém modelu a *id* požadovaného datového bodu. V další fázi dochází k zisku *id* tohoto bodu z nově vytvořené společné databáze energetických dat pomocí mapovací tabulky $fact\_id\_map$. Po zisku správného *id* následuje požadavek na zisk datové řady za zvolené období s volitelnou periodou. Tento požadavek zpracovává vrstva *Data Access Layer - DAL* [4]. *DAL* vrací dvourozměrnou matici, kde první řádek obsahuje časové údaje pořízení záznamu a druhý řádek sestává z naměřených hodnot. Tato matice se dále zpracovává podle časových kategorií.

Kategorie byly z hlediska logiky energetické spotřeby zvoleny ve dvou úrovních. První úroveň se rozděluje na čtyři roční období - jaro, léto, podzim, zima. Tato úroveň byla zavedena proto, aby bylo vidět, zda je patrný rozdíl ve vlivu ročních období na spotřebu elektrické energie. Druhá úroveň dělí dny podle toho, zda jsou pracovní či nepracovní. K tomuto dělení došlo za účelem zjištění rozdílu spotřeby ve dnech, kdy je budova téměř prázdná a ve dnech, kdy je lidmi plně využívána. Po specifikaci požadovaného dělení vytvořený program zpracuje příchozí datovou řadu z *DAL* a rozdělí ji do matice tak, že jednotlivé řádky představují data pro jednotlivé dny. Pro modelování byla zvolena hodinová perioda, tím pádem má každý řádek matice 24 sloupců.

V předposlední fázi dochází k výpočtu střední hodnoty a směrodatné odchylky. Tyto dvě hodnoty jsou počítány pro dané časové okamžiky přes všechny dny. Dalo by se říci, že se ve své podstatě počítají z každého jednotlivého sloupce matice popsané výše. Po všech předešlých krocích a operacích jsou data pro energetický TDD model připravena a je možné je vizualizovat.

4.2 Rozložení měření spotřeb

V budovách Fakulty aplikovaných věd se v rozvodné elektrické síti nachází více přístupných míst pro měření spotřeby elektrické energie. Na následujícím obrázku je znázorněno rozložení těchto přístupných míst. [5]

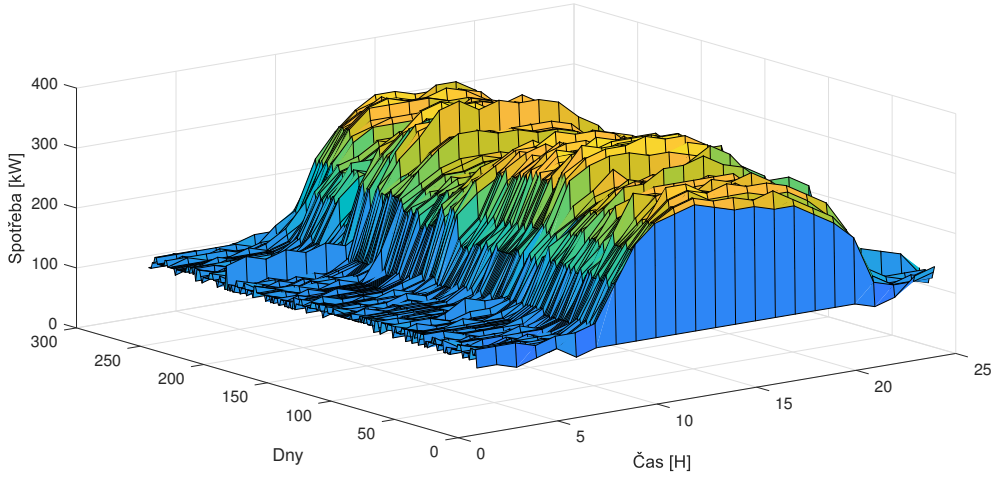


Obrázek 4.3: Znázornění rozložení měření elektrické spotřeby [5]

Na schématu 4.3 lze vidět rozložení zapojení rozvodné sítě na úrovni propojení rozvaděčů. Dále je možno říci, že budova NTIS je do vnější distribuční sítě napojena přes hlavní rozvaděč RH1. Rozvaděč RH1 napájí v budově NTIS především kancelářské prostory. Rozvaděč RH2 určený pro budovu CTPVV napájí kromě jejích kanceláří a prostor také společné prostory a zázemí budov FAV. Tímto se myslí například kavárna, studovna, konferenční sály a společné prostory.

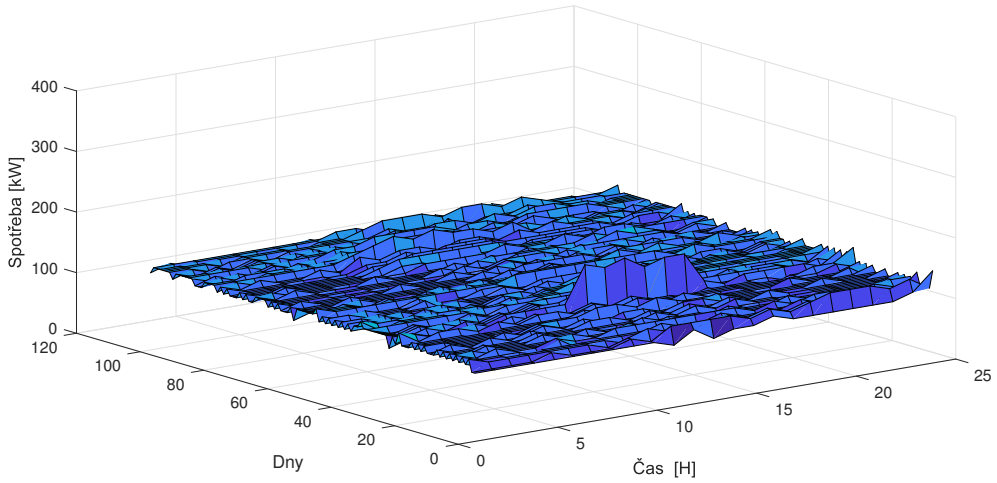
4.3 TDD modely

Modely jsou generovány na datech z roku 2016 získaných z měřícího místa před fakturačním elektroměrem NTIS *(RH1)* a rozvaděčem *RH2*, který napájí společné prostory v budovách FAV (kavárna, chodby, a podobně) a výukové místnosti v části CTPVV. Jedná se tedy o spotřebu veškerých objektů a příslušenství FAV. Data se získávají vždy za určité roční období a dále se dělí na pracovní a nepracovní dny. Po zisku dat přes vrstvu DAL dochází k transformaci dat do matice denních spotřeb. V matici denních spotřeb jsou data rozdělena podle řádků do jednotlivých dnů a vzhledem k zisku dat v hodinových periodách má matice rozměry *počet dnů/24*. Matice dnů pro pracovní a nepracovní dny v celém jednom roce vypadá následovně. Zde jsou zobrazena data z roku 2016.



Obrázek 4.4: Zobrazení spotřeb pro pracovní dny roku 2016

Na grafu 4.4 lze vidět za sebou ve směru osy dnů naskládané údaje o spotřebě v jednotlivých dnech. Pro zisk TDD modelů se následně provádí výpočet střední hodnoty a směrodatné odchylky. Jelikož se jedná o graf pracovních dnů, je velmi dobře zřetelný vliv pracovní doby na velikost spotřeby.



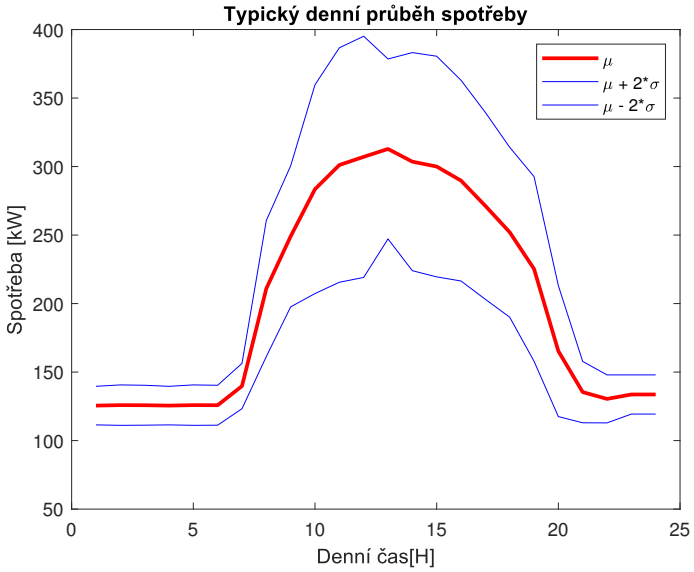
Obrázek 4.5: Zobrazení matice dnů pro nepracovní dny roku 2016

Na grafu 4.5 lze vidět za sebou ve směru osy dnů naskládané údaje o spotřebě v jednotlivých nepracovních dnech. Pro zisk TDD modelů se následovně provádí výpočet střední hodnoty a směrodatné odchylky. Na rozdíl od pracovních dnů se v tomto grafu nepracovních dnů nevyskytuje rozdíl ve spotřebě v pracovní a nepracovní době.

4.3.1 Případová studie energetického modelu FAV

V následující grafech jsou zobrazeny TDD modely budov FAV ve vybraných obdobích. Nejprve je uveden obecný popis TDD, poté jsou uvedeny TDD modely pro vybraná období.

Na grafu 4.6 je zobrazen ukázkový průběh spotřeby v pracovních dnech v modelu typické denní spotřeby. Červenou barvou je vyznačena střední hodnota μ pro modelované období, tedy průměrná spotřeba v daném časovém okamžiku. Modrou barvou jsou vyznačeny okraje pásma, které určuje 95% kvantil rozptylu, kde se hodnoty mohou vyskytovat $(\mu \pm 2\sigma)$.

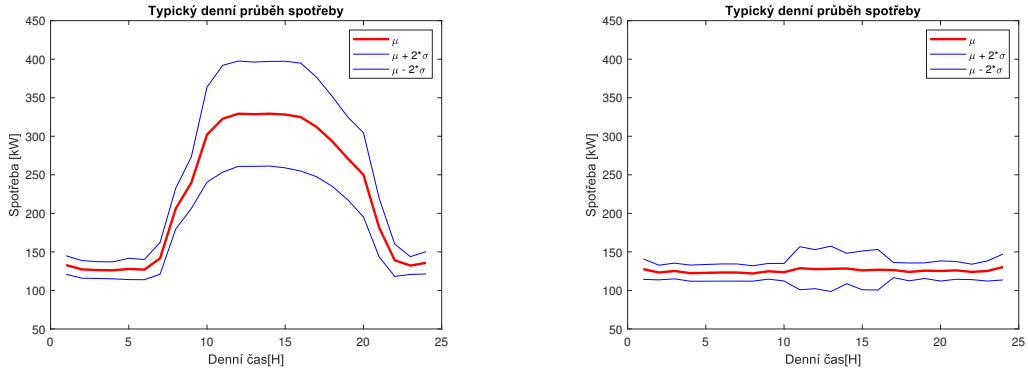


Obrázek 4.6: Typický denní průběh spotřeby pro pracovní dny roku 2016

Tento popis je společný pro všechny následující grafy zobrazující TDD.

4.3.2 Jaro

V případě modelování jarního období volíme data v rozmezí od 20.3. do 21.6. Po výpočtu středních hodnot a směrodatných odchylek pro matici dnů je možno data zobrazit v TDD formátu.



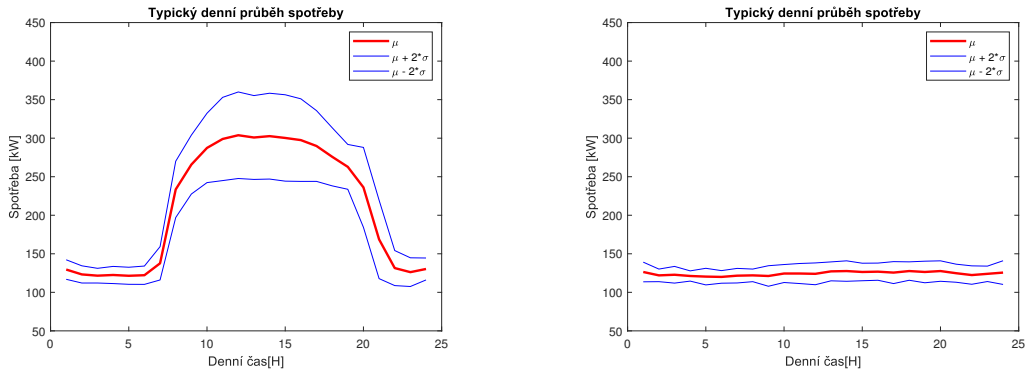
(a) pracovní den (b) nepracovní den

Obrázek 4.7: TDD model spotřeby budovy NTIS pro jarní období

V jarním období v nepracovních dnech je průměrná denní spotřeba 125.4333 kW a v maximu dosahuje 130.4000 kW. V pracovních dnech je průměr 222.3232 kW a v maximu dosahuje spotřeba 329.2424 kW.

4.3.3 Léto

Pro modelování letního období volíme data od 21.6. do 23.9.



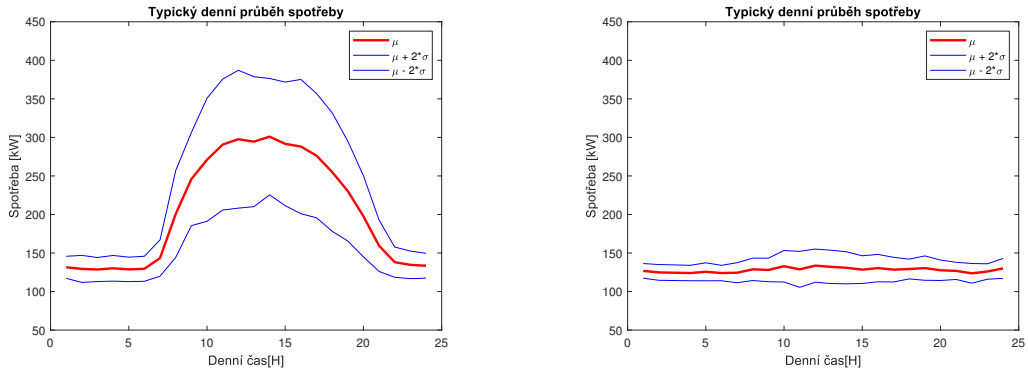
(a) pracovní den (b) nepracovní den

Obrázek 4.8: TDD model spotřeby budovy NTIS pro letní období

V letním období v nepracovních dnech je průměrná denní spotřeba 124.2667 kW a v maximu dosahuje 127.6000 kW. V pracovních dnech je průměr 212.0956 kW a v maximu dosahuje spotřeba 303.8235 kW.

4.3.4 Podzim

V případě modelování podzimního období volíme data od 23.9. do 21.12.



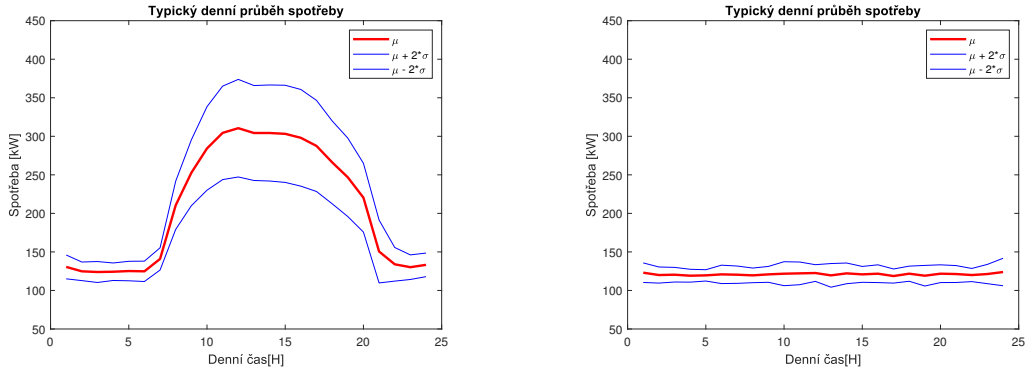
(a) pracovní den (b) nepracovní den

Obrázek 4.9: TDD model spotřeby budovy NTIS pro podzimní období

Na podzim během nepracovních dnů je průměrná denní spotřeba 127.9000 kW a v maximu dosahuje 133.6000 kW. V pracovních dnech je průměr 205.2910 kW a v maximu dosahuje spotřeba 300.9524 kW.

4.3.5 Zima

Pro modelování zimního období volíme data od 21.12. do 20.3.



(a) pracovní den (b) nepracovní den

Obrázek 4.10: TDD model spotřeby budovy NTIS pro zimní období

V zimě je během nepracovních dnů průměrná denní spotřeba 120.9420 kW a v maximu dosahuje 123.9130 kW. V pracovních dnech je průměr 209.8140 kW a v maximu dosahuje spotřeba 310.5357 kW.

4.4 Vyhodnocení

Na základě předchozí sekce lze říci, že spotřeba je relativně v malé míře ovlivněna ročním obdobím. Naopak je vidět markantní rozdíl v pracovních a nepracovních dnech, kdy v nepracovních dnech je spotřeba velmi nízká a konstantní. V pracovních dnech dochází v době mezi zhruba 7. a 19. až 20. hodinou k prudkému nárůstu spotřeby elektrické energie kvůli využívání budovy lidmi. Celkově nejvyšší průměrná i nejvyšší maximální spotřeba je během pracovních dnů v jarním období. Překvapivě je nejvyšší průměrná i maximální spotřeba v nepracovních dnech v jiném období, konkrétně se jedná o podzimní období.

4.4.1 Porovnání dat z let 2016 a 2017

V následujících tabulkách se nachází porovnání spotřeb elektrické energie budov FAV v letech 2016 a 2017. Hodnoty spotřeb vychází z výše vytvořených TDD modelů. V tabulce 4.1 se nachází porovnání hodnot v pracovních dnech ve vybraných obdobích. V tabulce 4.2 jsou uvedeny hodnoty v nepracovních dnech ze stejných období. Hodnoty uvedené v tabulkách jsou v kilowattech a ve formátu maximální/průměrná spotřeba během dne v určeném období.

Tabulka 4.1: Porovnání pracovních dnů 2016/2017

	jaro	**léto**	**podzim**	**zima**
2016	329.24/222.32	303.82/212.01	300.95/205.29	310.54/209.81
2017	321.97/215.64	289.70/203.67	317.78/213.86	312.86/210.12

Tabulka 4.2: Porovnání nepracovních dnů 2016/2017

	jaro	**léto**	**podzim**	**zima**
2016	130.40/125.43	127.60/124.27	133.60/127.90	123.91/120.94
2017	130.00/124.75	129.23/124.73	136.40/132.50	130.91/127.54

Z tabulek 4.1 a 4.2 je možno vyčíst, že hodnoty se v jednotlivých letech poměrně liší. Období maximálních spotřeb však zůstává pro oba roky stejné. V obou letech je spotřeba v nepracovních dnech maximální v podzimním období. Pro pracovní dny se jedná o jarní období.

Kapitola 5

Závěr

Cílem této práce bylo vytvoření jednotného systému pro sběr energetických dat a následně vybudovat nad tímto systémem energetické analytics. V rámci energetických analytics se tato práce věnovala energetickému modelu budov Fakulty aplikovaných věd. Práce se skládala ze 4 hlavních částí - analýzy současných datových zdrojů a návrhu společného datového úložiště, zpřístupnění původních datových zdrojů a jejich propojení s novým společným úložištěm, vytvoření kontextového modelu budov ZČU a vytvoření energetického modelu budov Fakulty aplikovaných věd.

V části analýzy současných datových zdrojů byla zjištěna současná situace uložení měřených energetických dat. Energetická data jsou rozdělena na dvě samostatné části, o které se starají dvě různé firmy. Energy management system spravuje firma Tronics s.r.o., telemetrickou soustavu spravuje firma Data-Ing s.r.o. Rozdělení dat je z praktického hlediska velmi nevýhodné a vznikla tak potřeba vytvoření společného datového úložiště. Během návrhu datového úložiště došlo k rozhodnutí použít relační databázi postavenou na serveru firmy MySQL. MySQL bylo zvoleno kvůli své rozšířenosti a dostupnosti zdarma. K návrhu databáze byl využit snowflake přístup.

Původní datové zdroje a nově vytvořená společná databáze jsou však dva separátní systémy, proto vznikla potřeba vytvořit softwarový nástroj (datová pumpa), který bude přelévat data z původních datových zdrojů do nové vytvořené společné databáze. Tento systém byl navržen tak, aby se v koncovém použití dal napojit přímo na produkční prostředí a bude tím pádem možné udržovat společnou databázi stále aktuální. Návrh této pumpy proběhl ve více fázích. Nejprve byl návrh testován na offline datech a až následně proběhlo nasazení na produkční prostředí. Datová pumpa má společné rozhraní, je do ní tak možné vložit data z libovolného datového zdroje a vkládat je tak do společné databáze.

V databázi firmy Data-Ing s.r.o., která spravuje data z telemetrické soustavy, byla analyzována kostra vztahů mezi jednotlivými objekty uvnitř ZČU. Byl vytvořen program automaticky generující z této struktury hierarchický kontextový model všech objektů ZČU včetně jejich datových bodů. Datové body dávají od elektrické spotřeby přes spotřebu vody až po teplotu. Tento hierarchický kontextový model je zachycen ve struktuře podle standardu formátu XML. XML byl vybrán pro jeho

dobrou čitelnost člověkem i strojem. Ve struktuře hierarchického kontextového modelu budov ZČU jsou všechny budovy ZČU včetně jejich vnitřní struktury.

Poslední část práce je věnována energetickému modelování. Konkrétně je zaměřena na energetický model budov Fakulty aplikovaných věd. Energetické modelování je částí nadstavby energetických analytics, které jsou postaveny nad navrženou společnou databází energetických dat. Pro tuto práci byl zvolen model typického denního průběhu spotřeby, kdy jsou vypočítávány střední hodnoty a směrodatné odchylky pro konkrétní časové okamžiky přes všechny dny ve zvoleném období. Pro porovnání různých vlivů vnějšího a vnitřního prostředí budovy na spotřebu elektrické energie, byly zvoleny časové segmenty, pro které se tvoří model TDD. Pro vnější vlivy byly vytvořeny časové segmenty podle ročních období. Pro vlivy vnitřní se pak časové segmenty dělí na pracovní a nepracovní dny, aby byl vidět vliv uživatelů budovy na její spotřebu.

Na základě energetického modelování může oddělení provozu a služeb ZČU například modifikovat své smlouvy o dodávce elektrické energie. Tato modifikace vede k optimalizaci kapacitních plateb a tím pádem ke snížení nákladů na elektrickou energii. Toto modelování lze na základě této práce provést pro libovolnou budovu ZČU, tudíž je možné optimalizovat a snížit náklady pro celou ZČU.

Literatura

[1] Herout Pavel. *Učebnice jazyka Java - 5. vydání.* Kopp, 2010.

[2] Anthony Molinaro. *SQL.* Computer press, 2005.

[3] Ramez Elmasri, Shamkant B. Navathe. *Fundamentals of Database Systems.* Addison-Wesley, Boston, 2011.

[4] Soukup Lukáš. *Modelování vybraných energetických zařízení, Bakalářská práce.* Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Plzeň, 2018.

[5] Ing. Martin Střelec Ph.D., Ing. Libor Jelínek Ph.D., Ing. Miloš Fetter, Ing. Milan Štětina, Ing. Roman Pišl. *Měření rozložení spotřeb elektrické energie v budovách NTIS a CTPVV.* Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Plzeň, 2016.

[6] W3Schools.com: XML Tutorial, *https://www.w3schools.com/xml/default.asp*

[7] MathWorks: Matlab Documentation, *https://www.mathworks.com/help/matlab/index.html*

[8] Emil Drkušic: Star Schema vs. Snowflake Schema, *http://www.vertabelo.com/blog/technical-articles/data-warehouse-modeling-star-schema-vs-snowflake-schema*

[9] sqlrelease.com: DBMS, RDBMS and SQL Server, *http://www.sqlrelease.com/wp-content/uploads/2015/05/RDBMS1.png?x43477*

[10] UMapIT: Merging the datacube paradigm with an occurrence-based approach to support on-demand web mapping, *https://www.researchgate.net/profile/Yvan_Bedard/publication/22724669/figure/fig4/AS:302384581496843@1449105510838/Star-schema-left-vs-Snowflake-schema-right.png*

[11] w3webtutorial.com: Valid Xml File, *http://www.w3webtutorial.com/xml/valid-xml-file-document.php*

Seznam obrázků

PŘÍLOHY

FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

Dokumentace databáze energetických dat

Pavel Novotný

2018

Úvod

Dokument popisuje databázi vytvořenou za účelem skladování energetických dat měřených v budovách FAV ZČU v Plzni. Pro návrh bylo použito vločkové (snowflake) schéma.

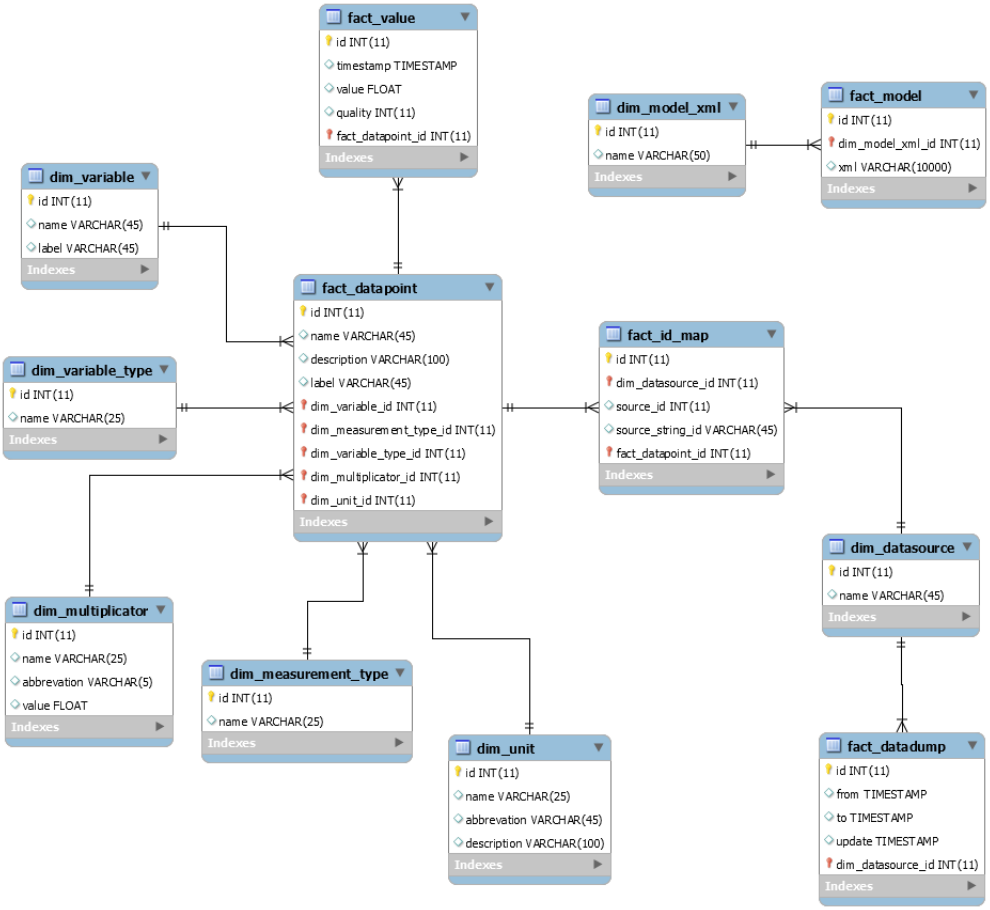


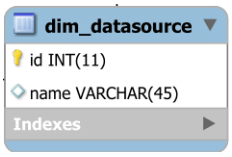
Schéma se dá rozdělit na tři části
1) - mapování: tabulky *dim_datasource*, *fact_id_map*, *fact_datadump*
2) - modely: tabulky *dim_model_xml*, *fact_model*
3) - měření: zbylé tabulky

Dimenzionální tabulky

Dimenzionální tabulky slouží k ukládání číselníku/enumerátorů.
Jednotlivé tabulky:

dim_datasource

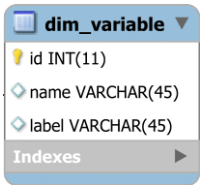
Tabulka obsahuje zdroje, ze kterých jsou načítána data.



id - unikátní identifikátor
name - název daného datového

dim_variable

Tabulka s veličinami v databázi.



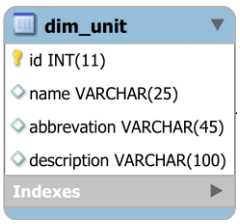
id - unikátní identifikátor
name - jméno dané proměnné
label - název dané proměnné

Příklad:

	id	name	label
	1	unspecified	USNP
	2	temperature	TEMP

dim_unit

Tabulka obsahující fyzikální jednotky.



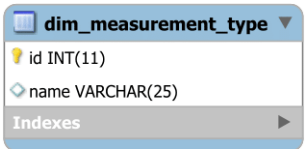
id - unikátní identifikátor
name - jméno dané jednotky
abbrevation - zkratka dané proměnné
description - popis dané proměnné

Příklad:

id	name	abbrevation	description
1	unspecified	UNSP	unknown or unspecified unit
2	celsius	°C	temperature in celsius

dim_measurement_type

Tabulka obsahující informace o typu měření.



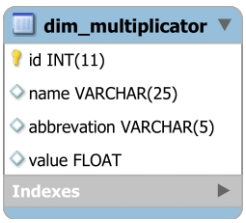
id - unikátní identifikátor
name - jméno typu měření

Příklad:

id	name
1	unspecified
2	continuous

dim_multiplicator

Tabulka s využívanými multiplikátory.



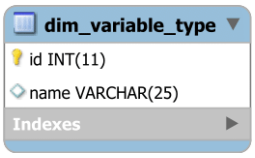
id - unikátní identifikátor
name - jméno multiplikátoru
abbrevation - zkratka multiplikátoru
value - konkrétní číselná hodnota multiplikátoru

Příklad:

id	name	abbrevation	value
1	mega	M	1000000
2	kilo	k	1000

dim_variable_type

Tabulka obsahující typy proměnných.



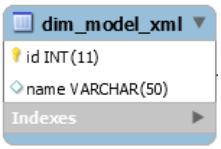
id - unikátní identifikátor
name - jméno typu proměnné

Příklad:

id	name
1	unspecified
2	control
3	meassured

dim_model_xml

Tabulka obsahující jména typů modelů.



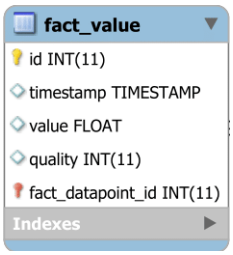
id - unikátní identifikátor
name - jméno typu modelu

Faktické tabulky

Faktické tabulky slouží k ukládání konkrétních datových hodnot.
Jednotlivé tabulky:

fact_value

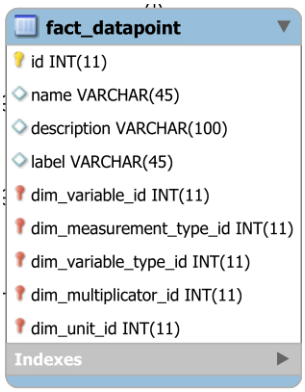
Tabulka do které jsou zapisovány měřené hodnoty.



id - unikátní identifikátor
timestamp - časový údaj vzniku měřené hodnoty
value - konkrétní naměřená hodnota
quality - číselná hodnota udávající kvalitu měřené hodnoty
$fact\_datapoint\_id$ - cizí klíč odkazující na původ měřené hodnoty

fact_datapoint

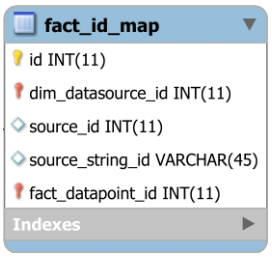
Tabulka popisuje zdroje měřených signálů.



id - unikátní identifikátor
name - jméno data pointu
description - popis konkrétního data pointu
label - název konkrétního data pointu
dim_variable_id - cizí klíč odkazující na proměnnou měřené hodnoty
dim_measurement_type_id - cizí klíč odkazující na typ měření
dim_variable_type_id - cizí klíč odkazující na proměnnou měřené hodnoty
dim_multiplicator_id - cizí klíč odkazující na multiplikátor měřené hodnoty
dim_unit_id - cizí klíč odkazující na jednotku měřené hodnoty

fact_id_map

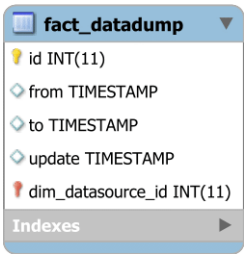
Vzhledem k velikosti dat načítaných z původních databází, musíme načítání rozdělit na menší části podle času, ze kterého pochází. K mapování toho načítání slouží právě tato tabulka.



id - unikátní identifikátor
$dim_datasource\_id$ - cizí klíč odkazující na datový zdroj
source_id - číselné id zdroje
source_string_id - string id zdroje
$fact\_datapoint\_id$ - cizí klíč odkazující na datapoint

fact_datadump

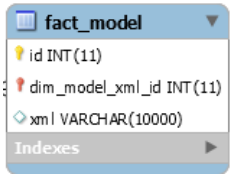
Tabulka obsahující údaje o načtených dumpech do databáze.



id - unikátní identifikátor
$from$ - časový údaj počátku posledního dumpu
to - časový údaj konce posledního dumpu
update - časový údaj dalšího naplánovaného dumpu
dim_datasource_id - cizí klíč odkazující na tabulku s datovými zdroji

fact_model

Tabulka obsahující kontextové modely ve formátu XML.



id - unikátní identifikátor
dim_model_xml_id - cizí klíč odkazující na tabulku se jmény modelů
xml - model uložený v xml formátu