UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Návrh modelu železniční sítě s využitím technologie Oracle Spatial Topology and Network Data Models

Emil Řezanina

Diplomová práce
2012

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2011/2012**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Emil Řezanina**
Osobní číslo:	**I10413**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Návrh modelu železniční sítě s využitím technologie Oracle Spatial Topology and Network Data Models**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

V teoretické části práce budou popsány technologie pro uchovávání multidimenzionálních dat a popis datových struktur vhodných pro reprezentaci vybraného segmentu železniční sítě.
V praktické části se diplomat zaměří na vybudování grafové reprezentace vybraného segmentu železniční sítě pomocí technologie Oracle Spatial Topology and Network Data Models.
V další fázi diplomant navrhne a implementuje testovací aplikaci, jež bude vizualizovat vybraný segment železniční sít včetně pohybu kolejových vozidel.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. MURRAY, Chuck, et al. Oracle Spatial : User's Guide and Reference 10g Release 2 (10.2) [online]. [s.l.] : Oracle, 2006 Dostupné z WWW:
2. MURRAY, Chuck, et al. Oracle Spatial : Topology and Network Data Models 10g Release 2 (10.2) [online]. [s.l.] : Oracle, 2005. Dostupné z WWW:

Vedoucí diplomové práce: **Ing. Jan Fikejz**
Katedra softwarových technologií

Datum zadání diplomové práce: **31. října 2011**
Termín odevzdání diplomové práce: **18. května 2012**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

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

V Pardubicích dne 15. listopadu 2011

Prohlášení autora

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

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

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 31. 7. 2012 Emil Řezanina

Poděkování

Rád bych touto cestou poděkoval vedoucímu mé diplomové práce Ing. Janu Fikejzovi za cenné rady a připomínky, za jeho čas, trpělivost a ochotu při řešení problémů vzniklých při zpracování této práce a také svojí rodině za morální podporu.

Anotace

V teoretické části práce jsou popsány technologie pro uchovávání multidimenzionálních dat a popis datových struktur vhodných pro reprezentaci vybraného segmentu železniční sítě.

Klíčová slova

Železniční síť, Oracle Spatial, Síťový data model, multidimenzionální data, prostorová data, Topologický data model.

Title

Design of a railway network model using the Oracle Spatial Topology and Network Data Models.

Annotation

The theoretical section describes technologies for multidimensional data storage and description of data structure suitable for the representation of the selected segment of the railway network.

Keywords

Railway network, Oracle Spatial, Network Data Model, Multidimensional data, spatial data, Topology Data Model.

Obsah

Seznam zkratek

TUDU	Traťový definiční úsek
GML	Geography Markup Language
DBMS	Database Management System
CLR	Common Language Runtime
SDO	Spatial data option
LRS	Linear Referencing System
MBR	Minimum Bounding Rectangle
DÚ	Definiční úsek
TUDU	Traťový definiční úsek
TDNU	Traťový definiční nadúsek
ADS	Abstraktní datová struktura
GIS	Grafický informační systém
DDL	Data Definiton Language
DML	Data Manipulation Language
OGIS	Open Geodata Interoperability Specification
FOI	Features of interest
API	Aplikační programovací rozhraní

Seznam obrázků

Seznam tabulek

1 Úvod

Cílem této diplomové práce je navrhnout model železniční sítě s využitím technologie Oracle Spatial Topology and Network Data Models.

V první části diplomové práce se primárně zaměřuji na popis vybraných technologií a datových struktur pro uchovávání multidimenzionálních dat. Jsou zde popsány technologie uložení prostorových dat v databázích Microsoft SQL Server a Oracle Database a v datových strukturách: 2D strom, Range strom a prioritní vyhledávací strom. Dále jsou také popsány možnosti reprezentace železniční sítě pomocí datové struktury typu graf.

Praktická část je zaměřena na vybudování grafové reprezentace vybraného segmentu železniční sítě pomocí technologie Oracle Spatial Network Data Model, vytvoření mapy sítě a popsání návrhu a implementace testovací aplikace, která bude pracovat s modelem železniční sítě.

Následně se diplomová práce podrobněji zabývá technologií Oracle s nadstavbou Spatial. Tato část je zaměřena především na popis technologie, prostorové datové typy a metadata a především na topologický a síťový datový modely. Jsou uvedeny i možnosti vizualizace prostorových dat pomocí nástroje MapViewer.

V praktické části byla použita diskrétní simulace pro simulování pohybu traťových vozidel.

V příloze je přiložena příručka k testovacímu programu.

2 Technologie pro uchovávání multidimenzionálních dat

Multidimenzionální data jsou taková data, která označují určité umístění v prostoru, nebo vícerozměrné informace o objektu. Multidimenzionální si můžeme představit jako *k*-dimenzionální, kde prefix *k* značí počet dimenzí. Dimenze jsou rozměry, které představují určitý pohled na daný objekt. Spolu všechny dimenze tvoří multidimenzionální klíč.

V této práci se zabývám prostorovými daty. Prostorová data jsou data o poloze, tvaru a vztazích mezi nimi. Nejčastější formou prostorových dat jsou souřadnice a topologie.

Máme dva základní způsoby reprezentace objektů a jevů reálného světa, resp. dva druhy prostorových dat.

- *Vektorová data* – reprezentace jako geometrické obrazce (body, čáry, polygony), které jsou určeny svými souřadnicemi a prostorovými vztahy mezi sebou (topologie).
- *Rastrová data* – vztahující se převážně k území jako celku. Skládají se z tzv. buněk, které spojitě pokrývají prostor a vytvářejí tzv. *mozaiku*. Buňky mohou mít tvar čtverce, trojúhelníku nebo šestiúhelníku a shodné nebo rozdílné rozměry (pravidelná, nepravidelná mozaika). Nejčastěji se používá pravidelná čtvercová mozaika. [1]

2.1 Prostorová data

Prostorová data nejčastěji slouží k popisu geometrických tvarů (např. bod, úsečka, polygon atd.) nebo jako lokalizace dat k určitému místu pomocí souřadnic. Obecně prostorové data mohou být *n*-rozměrné. Ať už se myslí souřadnice x a y (v 3D souřadnice x, y a z).

Umístění/poloha může například být část komerčních dat: organizace udržují seznamy zákaznických adres, vlastní majetek, dodávají zboží z a do skladů, řídí transportní toky mezi nimi a vykonávají mnoho dalších aktivit. V nekomerčních datech jsou prostorová data využívána podobně k lokalizaci objektů, vytváření sítí atd.

Popis polohy nemusí být ale jen statický - ve skutečnosti se polohy sledovaných objektů mohou v průběhu času měnit. Například vlak, který jede ze stanice A do stanice B, při jízdě mění svoji polohu. U takovýchto dat se vyplatí sledovat tuto skutečnost, díky které následně můžeme analyzovat průběh jízdy pro zjištění, jestli nenastalo třeba zpoždění, kolize na trase nebo detekování chyby, kdy dva vlaky jedou na stejné koleji proti sobě.

Všechny tyto informace o poloze mohou být uloženy, analyzovány a měněny mezi více systémy. Častým cílem pak je, aby práce s prostorovými daty byly co nejlevnější, nejrychlejší a nejspolehlivější. Mnoho systémů jsou navzájem propojeny přes Internet. Tak

může třeba koncový uživatel používat Internet pro přístup k systému a k dotazování na momentální status jeho parcel nebo zjištění, jestli vlak nemá zpoždění.

Obecně platí, že vztah mezi neprostorovými objekty a jejich odpovídající geometrií je možné vztáhnout na objekty založené na prostorových pojmech (blízko, daleko, překrývání, propojení atd.).

Další věc, kterou bychom měli zmínit u prostorových dat, jsou prostorové operace. Operace můžeme dělit na následující:

- **Skladiště prostorových dat** – ukládání dat ve vhodné formě v databázi. Například databázové systémy mohou mít typ geometrie k ukládání prostorových informací jako body, čáry, polygony a jiné druhy vektorových reprezentací. Systém také může mít typ sítě pro modelování sítí cest.
- **Analýzy vektorových prostorových dat**, která se typicky dělí na následující funkcionality.
    - **V dosahu** – operace identifikuje všechny prostorová data v určité vzdálenosti od místa dotazu.
    - **Obsahuje** – operace, která identifikuje všechny prostorová data, které obsahují oblast z dotazu.
    - **Nejbližší soused** – operace pro zjištění nejbližších sousedů k lokaci z dotazu.
    - **Vzdálenost** – operace, která vypočítává vzdálenost mezi dvěma prostorovými objekty.
    - **Buffer** – operace vytváří ohraničené zóny kolem prostorových dat.
    - **Překrytí** – tato operace překryje různé vrstvy prostorových dat.
    - **Vizualizace** – operace prezentuje prostorová data použitím map.
- **Analýzy síťových dat** – většina prostorových dat, jako jsou železniční sítě, mohou být také reprezentovány jako síťová data. Můžeme tedy provádět předchozí analýzu na těchto datech pomocí síťových funkcí než pomocí prostorových. [11]

Tato funkcionalita je základním kamenem pro GIS.

Při práci s prostorovými daty potřebujeme takové nástroje, které dovedou uchovávat tyto data a efektivně s nimi pracovat. Na výběr pro uchovávání prostorových dat můžeme použít soubory, databáze nebo datové struktury v paměti.

2.2 Uložení prostorových dat v souboru

Data jsou uložena v souborech v adresářové struktuře počítače. Tento postup není moc výhodný. Například uživatelé při editaci dat přistupují přímo k obsahu souborů. To znamená, že více uživatelů nesmí editovat tato data najednou. Příkladem souborového uložení může být datový formát *shapefile*, který používá společnost ESRI ve svém geografickém informačním systému ArcGIS. Celá datová vrstva (např. vrstva lesů, vodních toků atd.) je uložena v několika souborech:

- **soubor s prostorovými daty**,
- **soubor s atributovými daty** a
- **soubor s vyhledávacími indexy**.

Existují také formáty, které celou datovou vrstvu uchovávají v jediném souboru. Například soubory *.dxf, .dgn* programu MicroStation nebo formát GML. [1]

2.3 Uložení prostorových dat v databázi

V této kapitole jsem se zaměřil na uložení prostorových dat v objektově-relačních databázích, které obsahují podporu pro práci s prostorovými daty. Mezi zástupce těchto databázových systémů patří Microsoft SQL Server a Oracle Database.

2.3.1 Microsoft SQL Server

Práce s multidimenzionálními daty zahrnula firma Microsoft od verze Microsft SQL Server 2008.

Prostorová data jsou implementována jako CLR[1] rozšíření. Nejsou součástí serveru jako například datový typ *integer*. Jsou implicitní součástí serveru, takže není třeba řešit instalaci apod.

Využívají se dva datové typy, které pracují ve dvou dimenzích, a to:

- *geometry* – slouží k reprezentaci dat, jako např. bodů, čar, polygonů v rovině s omezením souřadnic (konečný prostor).
- *geografy* – slouží k reprezentaci stejných dat jak geometry, avšak na povrchu zeměkoule. [2]

[1] CLR – Komponenta virtuálního stroje v Microsoft .NET frameworku.

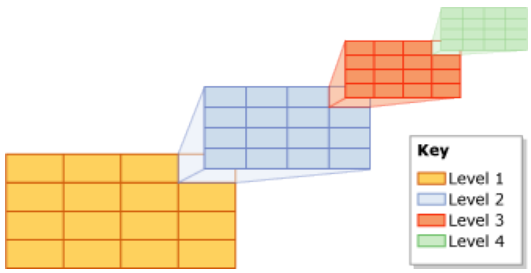
Dekompozice indexovaného prostoru

Server podporuje indexaci nad prostorovými daty, resp. nad sloupci s výše zmíněnými datovými typy. Index je tvořen standardním B stromem. Dekompozice je prováděna ve čtyřech úrovních pomocí mřížky a při tvorbě indexu je možné specifikovat, jak hustá mřížka bude na každé úrovni. Hustota mřížky je rozdělena v Tabulce 1.

Tabulka 1: Hustota mřížky. Zdroj: [autor]

Klíčové slovo	Konfigurace mřížky	Počet buněk
LOW	4x4	16
MEDIUM	8x8	64
HIGH	16x16	256

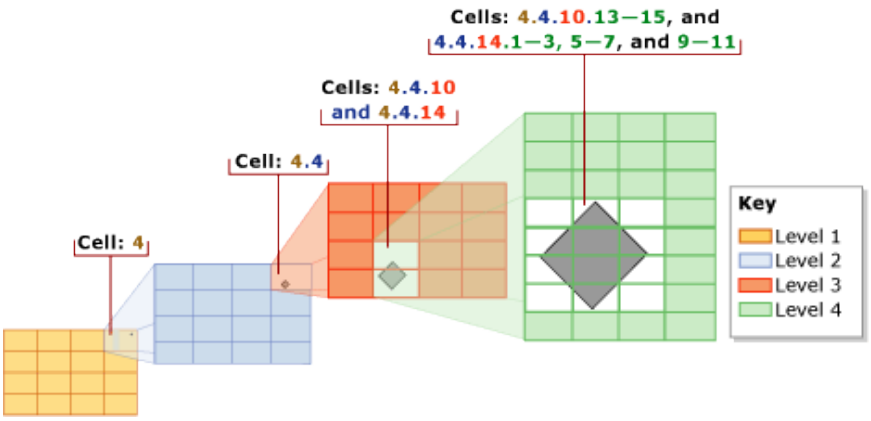
Způsob rozkladu prostoru v prostorovém indexu je zachycena na Obrázku 1.



Obrázek 1: MS SQL - rozklad prostoru v prostorovém indexu. Zdroj: [3]

Teselace

Jakmile je prostor rozdělen do mřížky, je třeba přečíst jednotlivé řádky a nastavit jim index. Tento proces, jinak teselace, prochází jednotlivé řádky a přiřazuje jim buňky, ve kterých objekt leží (nebo se jich dotýká). Tesalace využívá převádění obecného polygonu do trojúhelníkové, obdélníkové nebo jiné sítě. Rozložení buněk a tesalace objektu je zachycena na Obrázku 2.



Obrázek 2: MS SQL - rozložení buněk a tesalace objektu. Zdroj: [3]

Procházení se provádí procházením do šířky v jednotlivých úrovních. Aby se počet těchto buněk držel na rozumné hodnotě, existuje několik pravidel, která snižují tento počet.

- **covering rule** – pravidlo, když objekt nebo jeho část překrývá danou buňku. Zamezuje tesalaci na podrobnějších úrovních hierarchie pomocí záznamu v indexu.
- **cells-per-object rule** – pravidlo pro maximální počet buněk, které se pro určitý objekt mohou zaznamenat v indexu.
- **deepest-cell rule** – pravidlo, které určí nejhlubší buňku nebo buňky pro konkrétní objekt.

Schémata teselací v závislosti na typu objektu.

- **geometry grid tessallation** – výchozí schéma teselace pro datový typ geometry.
- **geography grid tessallation** – toto schéma teselace se vztahuje pouze na geography sloupec. [2][3]

2.3.2 Oracle Database

První pokusy přidání prostorových dat do databází se datuje k verzi Oracle 4. Ale velký zlom přinesl až Oracle 7 se SDO. SDO zavedl do databází především potřebný prostorový indexový systém, který umožňoval pracovat se šroubovitou spirálou přes 3dimenzionální prostor až k n-dimenzionálním prvkům. To také dovolilo vysoce účinnou kompresi výsledných dat vhodných pro datové uložiště a tak zrychlení vyhledávání a načítání. „Šroubovitý hyper-prostorový kód“, neboli HH code, zahrnoval formu prostorově plnící křivky. Od této chvíle byl již jen malý krůček k rozšíření Oracle Spatial, které se objevilo od verze Oracle 8. Oracle Spatial se stále vyvíjí.

Oracle Spatial můžeme chápat jako integrovaná množina funkcí a procedur, operátorů, podpůrných utilit mechanismů prostorového indexování a schématu geometrických datových typů, která umožňuje rychle a efektivně ukládat, přistupovat a analyzovat prostorová data v databázi Oracle. [4]

Oracle rozlišuje dvě verze podpory prostorových dat ve svých databázích.

- **Oracle Locator** – odlehčená verze Oracle Spatial a je poskytována zdarma u edicí databáze Standard a Enterprise.
- **Oracle Spatial** – licencovaná databázová možnost a je k dispozici pouze v Enterprise edici. Poskytuje funkce verze Locator včetně všech funkcí pro prostorové geometrické agregace, prostorovou analýzu a dolovací funkce, podporu LRS a geo-kódování. [5]

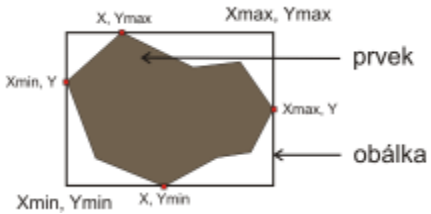
Oracle Spatial využívá objektový datový typ SDO_GEOMETRY, o kterém je pojednáno v kapitole 4.4.1.

Indexace

Ve verzích před Oracle 8 byl používán indexový systém HH code, který byl nahrazen R-stromem, quad-stromem, nebo kombinací obou struktur. Od verze Oracle 11g byla vylepšena a výrazně urychlena indexace v R-stromu, takže se doporučuje používat pouze tuto indexaci.

Obálka objektu

Důležitým pojmem, se kterým prostorové indexy pracují, je obálka prvku – MBR. Je určena minimálními a maximálními souřadnicemi obdélníku, do kterého se prvek vejde. Je rovnoběžný se souřadnicovými osami. Souřadnice obálky prvku se určí jako minimální a maximální X a Y souřadnice prvku. Obálku bodu tvoří bod sám. Viz Obrázek 3.



Obrázek 3: Obálka prvku. Zdroj: [2]

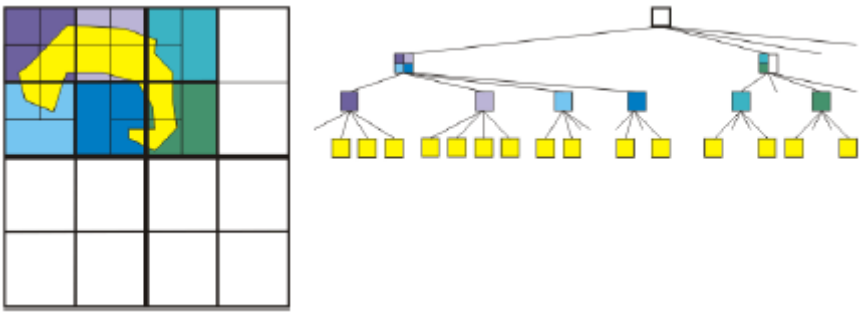
Použitím obálky se zrychluje vyhledávání prvků pomocí prostorového indexu. Na většinu prvků nám stačí porovnat se zadanými kritérii jen jejich obálku. U zbylých prvků je nutné porovnat i jejich souřadnicový popis, což je časově mnohem náročnější, než porovnat obálky. Obálka je určena pouze dvěma body na rozdíl od prvku, který může být složen z mnohem více bodů. [1]

Quad-strom index

Quad-strom je typ dlaždicového indexu[2], kde nestejně velké dlaždice vznikají rekurzivním dělením prostoru na kvadranty. Celá struktura je uložena ve stromu, kde každý uzel, který není listem, má čtyři potomky. Existuje více variant tohoto indexu.

Plošný quad-strom může sloužit pro indexování polygonových a liniových prvkových tříd nebo i pro komprimované ukládání rastrových dat (viz Obrázek 4). Prostor je dělen na pravidelné čtverce tak dlouho, dokud všechny neobsahují souvislou oblast – v případě rastru, nebo neaproximují co nejlépe daný polygon při předem určené hloubce dělení.

[2] Rozděluje prostor na pravidelnou čtvercovou síť (dlaždice) a v databázi je zaznamenána každá neprázdná dlaždice. Nemají-li prvky přibližně stejnou velikost, používají se dvě nebo tři úrovně dlaždic. [1]

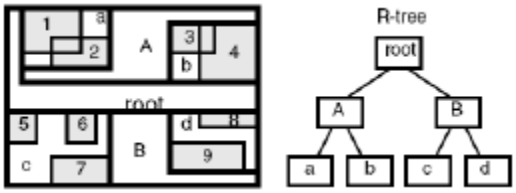


Obrázek 4: Plošný quad-strom index. Zdroj: [2]

Vyhodnocování prostorových dotazů pomocí quad-strom indexu probíhá jako prohledávání uzlů stromu vyhovujících obálce geometrie v dotazu od kořene k listům. [1]

R-strom index

R-strom, neboli region-tree, index seskupuje prvky aproximované svou obálkou podle jejich polohy do hierarchické stromové struktury. V listech stromu jsou uloženy ukazatele na prvky (geometrie i atributy prvků) a obálky prvků, uzly obsahují obálky všech obálek potomků, v kořenu je uložena obálka celého indexovaného prostoru. (Znázorněno na Obrázku 5.) Prvky (obálky prvků) mohou být obecně *n*-rozměrné.



Obrázek 5: R-strom index. Zdroj: [6]

Postup vyhodnocování prostorového dotazu pomocí R-stromu ve třech krocích.

1. Procházení stromu od kořene k listům. Jestliže obálka dotazu zasahuje do obálky uložené v aktuálním uzlu, prohledávání pokračuje směrem k potomkům uzlu. Výsledkem jsou obálky prvků (obsažené v listech stromu), do nichž zasahuje obálka dotazu.
2. Z prvků vybraných v prvním kroku se vyberou ty, do jejichž obálky zasahuje geometrie dotazu.
3. Z výsledku druhého kroku se vyberou ty prvky, do jejichž geometrie zasahuje geometrie dotazu. [1]

R-strom má výhodu oproti quad stromu v tom, že mu nevadí rozdílné velikosti obálek.

2.4 Datové struktury

V této podkapitole jsou popsány některé datové struktury, které lze použít pro uchovávání multidimenzionálních dat v paměti.

2.4.1 2D strom

Datová struktura 2D strom slouží pro uchovávání dat, která obsahují pouze prvky, které jsou definovány dvourozměrným klíčem. Když zobecníme 2D strom, získáme datovou strukturu *k*D strom, která umí pracovat až s *k*-dimenzionálním klíčem.

Struktura je založena na binárním stromu a využívá přitom principu jeho varianty – binárního vyhledávacího stromu, přičemž platí, že dělení do levých a pravých větví se určuje vždy podle jedné z hodnot tvořících klíč. Tyto jednotlivé klíče hodnot se vždy postupně opakují v cyklu. Konkrétně u 2D stromu je pravidlo na dělení nejprve podle první hodnoty a poté podle druhé hodnoty. Jako konkrétní příklad si můžeme představit souřadnice zeměpisné délky a šířky, kdy se nejprve aplikuje pravidlo podle hodnoty souřadnice X a následně podle hodnoty souřadnice Y. [7]

2.4.2 Range strom

Datová struktura, která obsahuje dva typy uzlů. První typ uzlů jsou listy stromů, které nesou informace. Tyto uzly jsou mezi sebou zřetězeny a seřazeny podle jedné z dimenzí multidimenzionálního klíče. Druhý typ vrcholů obsahuje informace o intervalu, ve kterém se nacházejí všichni potomci daného uzlu.

Celá struktura se skládá z jednoho hlavního stromu, který umožňuje hledání pouze podle první dimenze, a několika stromů druhé dimenze. Počet stromů druhé dimenze se rovná počtu navigačních vrcholů v hlavním stromu. Z navigačních vrcholů hlavního stromu lze do jednotlivých stromů druhé dimenze přeskočit. [7]

2.4.3 Prioritní vyhledávací strom

Struktura sloužící pouze k uchovávání dat s dvourozměrným klíčem. Struktura prioritního vyhledávacího stromu vychází z kombinace binárního vyhledávacího stromu a binární haldy. [7]

3 Topologie železniční sítě

V této kapitole se zaměříme na popis části topologie železniční sítě podle standardu ČD M12, která nás zajímá pro analýzu a následnou tvorbu modelu železniční sítě v praktické části diplomové práce.

Staničení

Lokalizační systém, který je odvozený z výsledků geodetických měření a projektu stavby kolejí. Umožňuje určení skutečné vzdálenosti kolmého průmětu charakteristického bodu daného objektu železniční dopravní cesty do osy koleje od předpisem stanoveného počátku.

Máme dvě traťové značky, které udávají staničení tratě (koleje).

- Kilometrovník – v délkovém intervalu 1 km.
- Hektometrovník – v délkovém intervalu 100 m.

Pojem „kilometrovník“ a „hektometrovník“ jsou nahrazovány jednotným pojmem „staničník“. [8]

Definiční úsek (DÚ)

Definiční úsek je základní evidenční a identifikační jednotka určená číselníkem definičních úseků. DÚ umožňuje rozčlenit železniční sít na menší části. Z evidenčního hlediska v sobě zahrnuje, kromě zemského povrchu i všechny nemovité objekty a zařízení, které do něj územně spadají. Ke skokům ve staničení může dojít pouze v hraničních bodech DÚ a to libovolně nebo uvnitř DÚ a to jenom směrem nahoru tak, aby byl údaj o staničení každého bodu v celém DÚ jedinečný. [8]

Traťový definiční úsek (TUDU)

TUDU je datový obraz DÚ (lokalizovaného fyzicky existujícího prostoru), který je reprezentován tabulkou dat. [8]

Traťový definiční nadúsek (TDNU)

TDNU je entita, která je tvořená jednoznačně určenou spořádanou množinou souvisle navazujících definičních úseků, které se mohou vzájemně dotýkat jak v podélném, tak příčném, nebo v obou směrech. [8]

Trať

Chápana jako jednoznačně dlouhodobě určená dopravní cesta mezi dopravami podle předpisu M12. [8]

Dopravně významné místo

Jednoznačně vymezená a identifikovaná část železniční sítě, která je charakterizovaná kolejovým rozvětvením nebo stanovištěm s kolejovým rozvětvením. Může být stanice, výhybka, odbočka, nákladiště atd. [8]

3.1 Datové struktury vhodné pro reprezentaci železniční sítě

Železniční síť si můžeme z pohledu modelování představit jako množinu staniček a traťových definičních úseků mezi nimi. To nás přivádí k datové struktuře graf, která se skládá z uzlů a vazeb mezi nimi.

Další možná datová struktura, která lze využít, je dvouúrovňový graf. Oproti datové struktuře graf se dvouúrovňový graf lépe využije při modelování TDNU.

3.1.1 Datová struktura graf

Datová struktura graf je zobecněním datové struktury síť. Tím přímo vybízí k použití v aplikaci pro modelování železničních, silničních, logických sítí atd.

Abstraktní datová struktura graf představuje heterogenní datovou strukturu, která pracuje s dvěma typy prvků – vrcholy a hranami. [9]

Možnosti implementace struktury grafu je možné rozdělit na následující typy:

- **vrcholově statické struktury** – operace `VložVrchol` a `OdeberVrchol` nejsou realizovatelné nebo jejich složitost není menší než O(*n*),
- **vrcholově dynamické struktury** – pro dynamickou práci s grafem se zaměřením na vrcholy,
- **hranově statické struktury** – operace `VložHranu` a `OdeberHranu` nejsou realizovatelné nebo jejich složitost není menší než O(*n*),
- **hranově dynamické struktury** – pro dynamickou práci s grafem se zaměřením na hrany. [9]

3.1.2 Datová struktura dvouúrovňový graf

Železniční síť je reprezentována vrcholy – hektometrovníky, které ve většině případů obsahují pouze dvě incidentní hrany (předchozí a následující hektometrovník). Díky tomu je možné dále optimalizovat datovou strukturu graf uchovávající tuto železniční síť. [10]

Tato implementace datové struktury tzv. dvouúrovňového grafu je lépe škálovatelná a využívá principu, kdy se do základního grafu popsaného o kapitolu výše umísťuj pouze vrcholy s jednou nebo více jak dvěma incidentními hranami. Vrchol, který má jenom jednu incidentní hranu, je umístěn na okrajích topologie železniční sítě nebo tam, kde končí koleje. Vrchol, který má více jak dvě incidentní hrany, je bod, v němž se spojuje nebo kříží více kolejových tratí. Tato reprezentace určuje super hrubou topologii železniční sítě.

Pro pochopení další části této práce musíme definovat nové dva pojmy:

- super hrana – hrana, která vznikne při vynechání vrcholů, které obsahují dvě incidentní hrany, a spojuje tedy vždy dvě dopravně významná místa,
- super vrchol – vrchol, který má v klasickém grafu dvě incidentní hrany a jeho pozice v dvouúrovňovém grafu je na super hraně.

Vrcholy s dvěma incidentními hranami jsou uloženy v druhotné tabulce vrcholů, která je podobná tabulce vrcholů v hlavním grafu. Rozšíření tohoto grafu spočívá hlavně v tom, že objekt reprezentující hranu nyní neobsahuje pouze klíč hrany a klíče vztažných vrcholů k této hraně, ale nyní v tomto případě také lineárně zřetězený seznam sub vrcholů uložených v pořadí, v jakém jsou umístěny ve skutečnosti na železniční síti, které tato super hrana zahrnuje.

Dvouúrovňový graf tedy oproti jednoúrovňovému grafu se skládá ze super hran a sub vrcholů. Dvouúrovňový graf pracuje na vyšší úrovni s vrcholy a super hranami a na nižší úrovni se super vrcholy a hranami. Super vrchol a hrana jsou elementární prvky super hrany. [10]

4 Oracle Spatial

Oracle Spatial je podpora prostorových dat od firmy Oracle. Je podporován většinou předních výrobců GIS. Oracle Spatial poskytuje SQL schémata a funkce, které slouží k ukládání, získávání, aktualizaci a dotazování na kolekce prostorových dat v databázi Oracle.

Oracle Spatial obsahuje schéma MDSYS, které popisuje ukládání, syntaxi a sémantiku podporovaných geometrických typů, prostorový indexový mechanismus, operátory, funkce a procedury pracující s prostorovými daty a ostatní operátory prostorových analýz, modely atd.

Oracle Spatial podporuje objektově relační model pro reprezentování geometrií. Tento model ukládá geometrie v Oracle nativním prostorovým datovým typem pro vektorová data `SDO_GEOMETRY`. [6]

V této kapitole se zaměřím na technologii Oracle Spatial. A to především na výhody, technologii, architekturu, funkcionalitu a na datový typ `SDO_GEMOETRY`.

4.1 Výhody Oracle Spatial

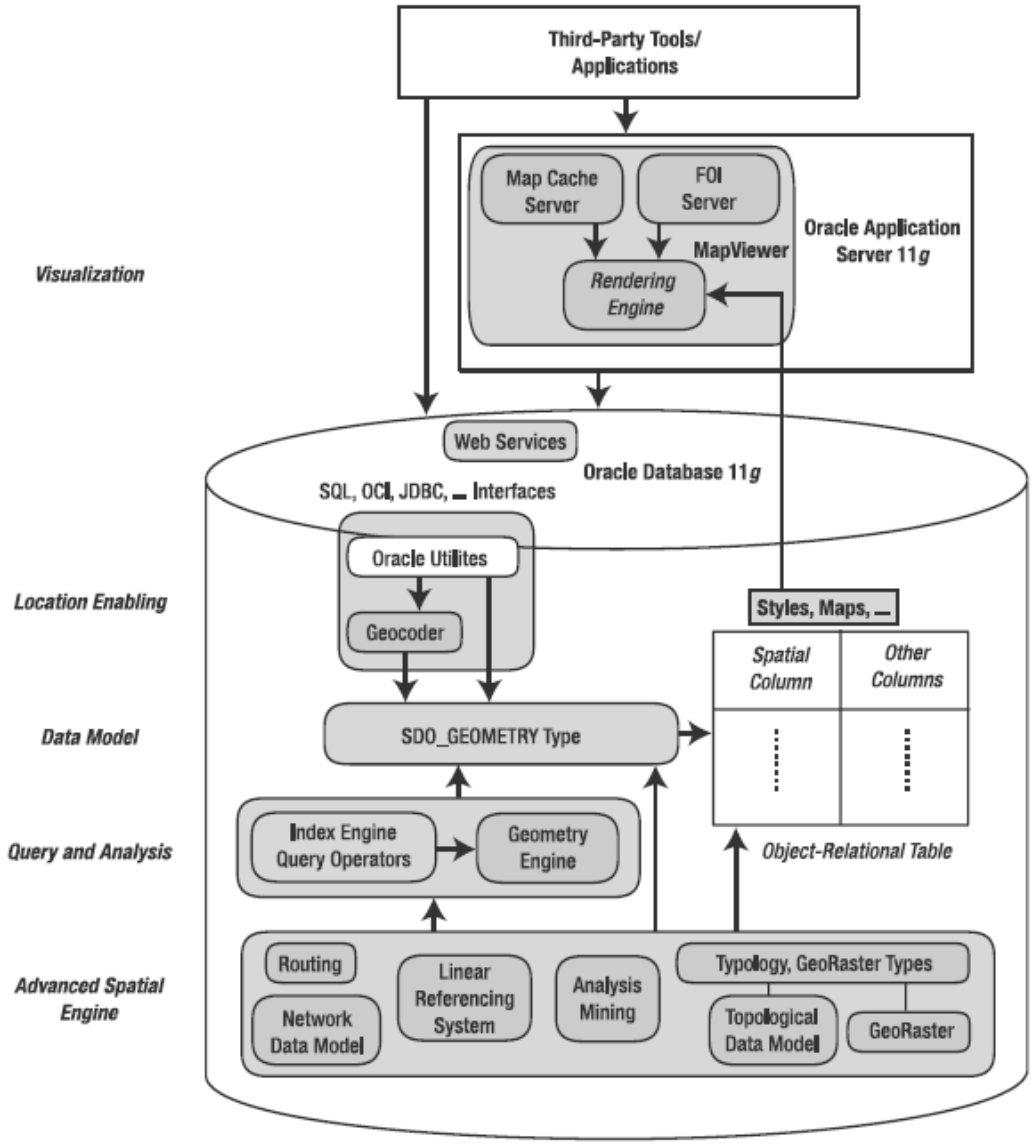
Hlavní výhody pro užívání Oracle Spatial jsou následující:

- Eliminuje potřebu duální architektury, protože všechna data mohou být uložena stejným způsobem. Jednotné ukládání dat znamená, že všechny typy dat (mapy, text a multimédia) jsou uloženy společně. Nemusejí být tedy ukládána odděleně.
- Používá SQL, standardní jazyk pro přístup k relační databázi. Díky tomu odstraňuje potřebu pro konkrétní jazyky, které by zpracovávali prostorová data.
- Je de facto standard pro ukládání a přístup k datům v Oracle a je plně podporován předními světovými prodejci geo-prostorových dat, nástrojů a aplikací, včetně NAVTEQ, Tele Atlas, Digital Globe, Autodesk a mnoho dalších.
- Poskytuje škálovatelnost, bezpečnost, integritu, obnovitelnost a pokročilé uživatelské funkce pro práci s prostorovými daty.
- Podpora pro hodně geometrických typu, jako jsou kruhy, oblouky, složité polygony, složené řetězce čar.
- Odstraňuje potřebu pro oddělení organizací zachovat prostorovou datovou infrastrukturu (hardware, software, podporu atd.). To eliminuje potřebu dalších zvláštních nástrojů a dovedností pro práci s prostorovými daty.

- Prostřednictvím aplikačního serveru umožňuje téměř jakékoliv aplikaci využít dostupnosti prostorových informací. Díky tomu se zmenší náklady a složitost prostorových aplikací. [6][11]

4.2 Oracle Spatial technology

Oracle Spatial technologie je rozdělena na dvě úrovně: Databázový server a aplikační server. Na Obrázek 6 jsou znázorněny jednotlivé složky, které tvoří balík Oracle Spatial technologie. Také ukazuje rozložení komponent v celých vrstvách databázového a aplikačního serveru. Základní komponenty, které jsou poskytovány jako součást Oracle 11g Database Server, obsahují modul uložiště, dotazové a analytické nástroje a další vylepšení. [11]



Obrázek 6: Komponenty Oracle Spatial technologie. Zdroj: [11]

4.2.1 Data Model

Oracle Spatial používá SDO_GEOMETRY datový typ pro ukládání prostorových dat v databázi Oracle. Uživatelé mohou definovat tabulky obsahující sloupce typu SDO_GEOMETRY, například pro uložení umístění GPS souřadnic stanic, vlaků, zákazníků atd. Informace je určena pomocí dvou částí: umístění s ohledem na nějaký původ a geometricky tvar.

- **Umístění** specifikuje, kde jsou data umístěna s respektováním dvou, tří nebo čtyř-dimenzionálním souřadnicovým prostorem. Například střed vlakové stanice Pardubice hlavní nádraží je umístěno na souřadnicích (15.75755563,50.03166914) v dvou-dimenzionálním „zeměpisné šířky a délky“ prostoru.

- **Tvar** specifikuje geometrickou strukturu dat. Bod, čára a polygon jsou příklady možných tvarů. Pro lepší přiblížení si můžeme říct, že střed vlakové stanice Pardubice hlavní nádraží má kromě svých souřadnic (15.75755563,50.03166914) také tvar typu bod. Traťový úsek je zase pro nás tvar čára a tvar vlakové stanice Pardubice hlavní nádraží je polygon propojených vícero bodů.

V některých aplikacích se můžeme setkat s tvary, které mohou být daleko komplexnější a můžeme také mít vícenásobné polygony nebo polygony s dírami. [11]

4.2.2 Location-Enabling

Uživatelé při vytváření tabulek mohou přidat sloupec SDO_GEOMETRY, který následně mohou naplnit daty použitím jedním ze standardních Oracle nástrojů, jako jsou SQL*Loader, Import a Export. Další možnou alternativou pro naplnění je využití gecoder komponent Oracle Spatial.

Jako příklad si můžeme vzít vytvoření tabulky pro vlaky, která bude obsahovat identifikační označení vlaku, název vlaku a aktuální umístění vlaku.

```
CREATE TABLE VLAKY
(
      id          NUMBER,
      nazev       VARCHAR2(24),
      umisteni    SDO_GEOMETRY
);
```

Pro naplnění vytvořené tabulky musíme při zadávání hodnoty sloupce umisteni použít konstruktor SDO_GEOMETRY objektu, o kterém pojednává kapitola 4.4.1.

V některých aplikacích nejsou pro prostorové informace explicitně k dispozici souřadnice. Místo toho jsou obvyklá pouze možná prostorová data jako poštovní adresy. Tyto údaje pak lze převést na SDO_GEOMETRY objekty použitím geocoder komponentu. **Geocoder** vezme poštovní adresu, konzultuje s vnitřní celostátní databází adres a míst, spočítá souřadnice zeměpisné délky a šířky pro danou adresu a vrátí zpět výsledek. Tento

proces je označován za geokódování v prostorové terminologii. Adresní/lokalizační databáze je často poskytována datovým dodavatelem třetí strany.[3] [11]

4.2.3 Query and Analysis

Uživatelé mohou pracovat se `SDO_GEOMETRY` daty pomocí dotazů a analýz. Komponenta dotazů a analýzy poskytuje jádro funkcionalit pro dotazování a analýzu prostorových geometrií. Tato komponenta má dvě dílčí části: Geometry Engine a Index Engine. Tyto součásti provádějí prostorové dotazy a analýzy. Například řeší, jaké dvě vlakové stanice jsou nejblíže k vlaku.

Geometry Engine poskytuje funkce pro analýzu, porovnání a manipulaci s geometriemi. Například jaké vlakové stanice jsou nejblíže k vlaku, počítat vzdálenosti mezi jednotlivými vlakovými stanicemi atd. Pro názornou ukázku můžeme použít dotaz, kterých 5 stanic je nejblíže k vlaku `'63321'`.

```
SELECT nazev
FROM
(
      SELECT S.nazev,
         SDO_GEOM.SDO_DISTANCE(S.umisteni, V.gps, 0.5) distance
      FROM stanice S, vlaky V
      WHERE V.nazev = '63321'
      ORDER BY distance
)
WHERE ROWNUM <= 5;
```

Ve vnitřní `SELECT` klauzuli si zjistíme vzdálenost jednotlivých stanic od vlaku `'63321'` pomocí funkce Geometry Engine `SDO_GEOM.SDO_DISTANCE`. Následně setřídíme vzestupně a pomocí vnější `SELECT` klauzuli si nastavíme výpis na pouhých 5 záznamů.

Index Engine slouží ke správě prostorových indexů a k funkcionalitám, které jsou prováděny na těchto indexech. V následujícím kódu si ukážeme, jak se ruší a vytváří index na umístění vlaků.

```
DROP INDEX vlaky_sid;
CREATE INDEX vlaky_sid ON vlaky(umisteni)
INDEXTYPE IS mdsys.spatial_index;
```

Na předchozím příkazu s klauzulí `CREATE` si můžeme všimnout části `INDEXTYPE`, která říká databázi, že má vytvořit prostorový index na `umisteni` sloupce, který je typu `SDO_GEOMETRY`. Použitím tohoto typu indexu dokáže Index Engine v Oracle Spatial ořezávat vzdálené řádky ze zpracovávaných dotazů a tím urychluje dotazy pro většinu aplikací. Index Engine poskytuje ekvivalentní funkce k Geometry Engine funkcím, označovaných jako operátory pro identifikování řádků v tabulce, která splňuje určitý přibližovací predikát. [11]

[3] Datový dodavatelé třetích stran adresných/lokalizačních databází jsou například NAVTEQ a Tele Atlas.

Například pokud potřebujeme vědět, jakých 5 stanic je nejblíže k vlaku '63321', tak použijeme následující příkaz.

```
SELECT S.nazev
FROM stanice S, vlaky V
WHERE V.nazev = '63321'
AND SDO_NN(S.umisteni, V.gps) = 'TRUE'
AND ROWNUM <= 5
```

V příkazu byl použit indexově založený operátor nazývaný `SDO_NN`, který slouží jako nejbližší soused. Operátor vrací setříděný seznam nejbližších stanic k vlaku 63321 s tím, že první stanice v seznamu je nejblíže k vlaku.

4.2.4 Visualization

Komponenty aplikačního serveru Oracle Spatial technologie zahrnují prostředky pro vizualizaci prostorových dat pomocí MapViewer nástroje. Tento nástroj bude popsán v kapitole 5.1.

4.2.5 Advanced Spatial Engine

Tato část se skládá z několika dílčích částí, které obstarávají sofistikované prostorové aplikace, jako jsou GIS. Každá z dílčích částí používá podkladový datový typ geometrie a Index a Geometry Engine funkčnost.

Systém lineárního odkazování usnadňuje překlad mílových značek na dálnici (nebo jiné lineární funkce) na geografický prostor souřadnic a naopak. Tato komponenta umožňuje uživatelům adresovat rozdílné segmenty lineárních geometrií, jako je například dálnice, aniž by ve skutečnosti odkazovali na souřadnice segmentu. Tato funkcionalita je užitečná třeba v dopravě.

Prostorová analýza a dolovací engine poskytuje základní funkcionalitu pro kombinování demografické a prostorové analýzy. Tato funkcionalita je užitečná při identifikování potencionálních míst pro zahájení nových obchodů založených na hustotě zákazníků a příjmech.

GeoRaster usnadňuje ukládání a získávání geografických obrázků pomocí jejich prostorových půdorysů a souvisejících metadat. GeoRaster definuje nový datový typ pro ukládání rastrových obrázku o geografických odkazovaných objektech. Tato funkce se využívá při řízení satelitních snímků.

Síťový datový model poskytuje datový model pro ukládání sítí v databázi Oracle. Základní síťové prvky jsou *uzly* a *spojení* mezi nimi. Tyto prvky mohou být asociovány s náklady a omezeními, například jako omezení rychlosti na určitých úsecích tratě. Ostatní funkce pak patří na výpočet nejkratších cest mezi dvěma místy v dané síti, nalezení nejbližších uzlů, jestli je uzel dostupný z jiného uzlu atd. Síťový datový model bude podrobněji popsán v kapitole 4.6.

Topologický datový model podporuje detailní analýzu a manipulaci s geometrickými prostorovými daty díky topologickým prvkům, jako jsou například uzly, hrany nebo plochy. V některých zemských aplikacích geometrie sdílejí svoje hranice, jako tomu je v případě vlastnictví hranic a vozovky, na které se nemovitost nachází. Oracle Spatial definuje nový datový typ na reprezentaci topologických prvků (jako třeba sdílený „úsek silnice“), které mohou být sdíleny mezi různými prostorovými objekty. Aktualizace sdílených prvků implicitně definuje aktualizace sdílených geometrických objektů. Obecně platí, že tato komponenta nám umožňuje editovat a manipulovat uzly a hrany bez narušení topologické sémantiky dané aplikace. [11]

4.3 Tolerance

Tolerance slouží k určení přesnosti práce s prostorovými daty. Definuje, kdy je ještě možné dva body umístěné od sebe brát za totožné. Tolerance je vždy kladné číslo. Minimální tolerance pro geodetická data je 0,05 a odpovídá 5 cm. Pro ostatní data je minimální tolerance 0,005 odpovídající 1/200 americké míle. [6]

4.4 Prostorové datové typy a metadata

Oracle Spatial obsahuje množinu datových typů, typových metod a operátorů, funkcí a procedur, které používají tyto typy. Geometrie je uložena jako objekt typu `SDO_GEOMETRY`.

SDO_GEOMETRY je objektový datový typ, který reprezentuje prostorová data. Tento datový typ v sobě uchovává informace o tvaru a umístění pro každý řádek tabulky. Nad `SDO_GEOMETRY` objektem můžeme vytvářet prostorový index a používat základní DDL (`CREATE, ALTER, DROP`) a DML (`INSERT, UPDATE, DELETE`) příkazy. [6]

V této kapitole se zaměříme na objektový datový typ `SDO_GEOMETRY` a na metadata, které využívá Oracle Spatial.

4.4.1 Datový typ `SDO_GEOMETRY`

V Oracle Spatial je geometrický popis prostorového objektu uložen v jednom řádku, v jednom sloupci objektového typu `SDO_GEOMETRY` v uživatelsky definované tabulce. Každá tabulka, která má sloupec typu `SDO_GEOMETRY`, musí mít jiný sloupec, resp. množinu sloupců, na kterých definuje unikátní primární klíč pro tabulku. Tabulky tohoto druhu jsou někdy označovány jako prostorové tabulky nebo tabulky prostorové geometrie.

Oracle Spatial definuje objekt typu SDO_GEOMETRY jako:

```
CREATE TYPE sdo_geometry AS OBJECT
(
      SDO_GTYPE         NUMBER,
      SDO_SRID          NUMBER,
      SDO_POINT         SDO_POINT_TYPE,
      SDO_ELEM_INFO     SDO_ELEM_INFO_ARRAY,
      SDO_ORDINATES     SDO_ORDINATE_ARRAY
);
```

Oracle Spatial také definuje SDO_POINT_TYPE, SDO_ELEM_INFO_ARRAY a SDO_ORDINATE_ARRAY typy, které jsou použity v definici typu SDO_GEOMETRY následovně:

```
CREATE TYPE sdo_point_type AS OBJECT
(
      X     NUMBER,
      Y     NUMBER,
      Z     NUMBER
);
CREATE TYPE sdo_elem_info_array AS VARRAY (1048576) of NUMBER;
CREATE TYPE sdo_ordinate_array AS VARRAY (1048576) of NUMBER;
```

Ve zbývající části této kapitoly jsem se zaměřil na jednotlivé položky objektu SDO_GEOMETRY a metody s nimi spojenými.

SDO_GTYPE indikuje typ geometrie. Validní geometrické typy odpovídají těm, které jsou uvedeny v geometrickém objektovém modelu pro OGIS[4] jednoduché rysy SQL specifikace (s výjimkou povrchů). Číselné hodnoty se liší od těch zadaných v OGIS specifikaci, ale existuje přímá vazba mezi jmény a sémantikou, pokud existuje. SDO_GTYPE hodnota se skládá ze 4 číslic ve formátu *dltt*, kde:

- *d* označuje číslo dimenze (2, 3 nebo 4),
- *l* označuje lineární referenční měřítko dimenze pro tří-dimenzionální LRS geometrie, když dimenze (3 nebo 4) obsahuje hodnotu měřítka. Pro ne-LRS geometrie nebo pro přijmutí výchozího Spatial nastavení poslední dimenze jako měřítko pro LRS geometrii je 0.
- *tt* specifikuje geometrický typ (využívány čísla od 00 do 07). [6]

Pokud chceme geometrii typu bod v dvou-dimenzionálním prostoru, tak použijeme číslo 2001, kde 2 značí druhou dimenzi, 0 říká, že se nejedná o LRS geometrii, a 01 označuje, že se jedná o bod. Další geometrické typy jsou v

[4] OGIS – podrobná specifikace softwarového frameworku pro distribuovaný přístup k datům a geoprocessing zdrojům.

Tabulka 2 2.

Tabulka 2: Validní SDO_GTYPE hodnoty. Zdroj: [autor] (data převzata z [6])

Hodnota	Typ geometrie	Popis
dl00	UNKNOWN_GEOMETRY	Spatial ignoruje tuto geometrii.
dl01	POINT	Geometrie obsahující jeden bod.
dl02	LINE nebo CURVE	Geometrie obsahuje jeden řetězec přímky, který může obsahovat přímky nebo oblouky nebo oba.
dl03	POLYGON	Geometrie obsahující polygon s dírou nebo bez díry.
dl04	KOLEKCE	Geometrie je množina, která obsahuje všechny ostatní typy.
dl05	NÁSOBNÝ BOD	Geometrie má jeden nebo více bodů.
dl06	NÁSOBNÁ ČÁRA nebo NÁSOBÁ KŘIVKA	Geometrie má jeden nebo více linkových řetězců, resp. křivek.
dl07	NÁSOBNÝ POLYGON	Geometrie může mít násobné disjunktní polygony.

SDO_SRID atribut může být použit k identifikaci souřadnicového systému (prostorový referenční systém), který je spojený s geometrií.[5] Pokud je SDO_SRID nastavený na *null*, tak geometrie nemá přiřazený souřadnicový systém. Pokud SDO_SRID je nastavený na jinou hodnotu než *null*, tak musí obsahovat jednu z hodnot z tabulky SDO_COORD_REF_SYS ve sloupci SRID a tato hodnota musí být vložena do SRID sloupce v pohledu USER_SDO_GEOM_METADATA. Všechny geometrie v geometrickém sloupci musí mít stejnou SDO_SRID hodnotu.[6]

SDO_POINT atribut je definován použitím SDO_POINT_TYPE objektového typu, která má atributy X, Y a Z, které jsou typu NUMBER. Pokud pole SDO_ELEM_INFO a SDO_ORDINATES jsou obě prázdné, tak SDO_POINT obsahuje X a Y souřadnice bodu. Pokud nejsou nastavené na *null*, tak je tento atribut ignorován.

SDO_ELEM_INFO atribut je definován použitím pole čísel proměnné délky. Tento atribut definuje, jak mají být interpretovány údaje v atributu SDO_ORDINATES. SDO_ELEM_INFO se skládá minimálně z 1 trojice čísel SDO_STARTING_OFFSET (pro určení místa v SDO_ORDINATES, kde začínají pro tuto trojici hodnoty), SDO_ETYPE

[5] SRID – Spatial Reference Identifier je celočíselná hodnota, která popisuje typ souřadnicového systému používaná prostorovým enginem. Pokud chceme vytvářet mapu, potřebujeme uvést SRID. Databázová tabulka MDSYS.CS_SRS obsahuje kompletní seznam různých souřadnicových systémů.

(indikuje typ prvku) a SDO_INTERPRETATION (pracuje v závislosti na hodnotě SDO_ETYPE. Buď slouží pro složené prvky k tomu, aby řekl, kolik bude dalších trojic, nebo jako interpretující atribut určuje pořadí souřadnic pro prvek, jak mají být interpretovány. Například u hranice polygonu můžeme ručit pořadí propojení přímých nebo obloukových segmentů.). Hodnoty, které mohou být v atributu SDO_ELEM_INFO jsou v Tabulce 3.

Tabulka 3: Hodnoty v SDO_ELEM_INFO. Zdroj: [autor] (data převzata z [6])

SDO_ ETYPE	**SDO_ INTERPRETATION**	**Význam**
0	(jakákoliv hodnota)	Typ nulového prvku. Používán pro model geometrických typů, které nejsou podporovány v Oracle Spatial.
1	1	Bodový typ.
1	0	Orientace pro orientovaný bod.
1	*n* > 1	Bodové seskupeni s *n* body.
2	1	Řetězec čáry, který má vrcholy spojené přímými úsečkami.
2	2	Řetězec čáry vytvořen z propojené sekvence oblouků. Každý oblouk je popsaný pomocí trojice souřadnic, určujících začáteční bod oblouku, bod na oblouku a koncový bod oblouku. Pro příklad, když máme řetězec skládající se z 5 údajů, tak 1,2 a 3 údaj jsou body pro první oblouk, 3,4 a 5 jsou údaje pro druhý oblouk.
1003 nebo 2003	1	Jednoduchý polygon, který obsahuje body propojené přímými úsečkami. Musíme specifikovat pro každý bod všechny vertexy a poslední bod musí být stejný jako začáteční bod.
1003 nebo 2003	2	Polygon skládající se z propojené sekvence oblouku, které jsou uzavřené. Každý oblouk je popsán trojicí čísel, jak to bylo u řetězce čary skládající se ze sekvence propojených oblouků.
1003 nebo 2003	3	Obdélníkový typ. Ohraničený obdélník, který je popsán pouze dvěma body: dolní levý a horní pravý.

1003 nebo 2003	4	Kruhový typ. Popsaný třemi různými body, které leží na obvodu kruhu.
4	*n* > 1	Řetězec čáry, který je tvořen skupinou přímých úseček a oblouků. Pomocí *n* se stanový počet sousedících dílčích prvků, které tvoří řetězec čáry. Následné trojice tedy odpovídají popisu jednotlivých řetězců čáry (musí mít SDO_ETYPE 2). Platí, že poslední bod dílčího prvku je první bod následujícího dílčího prvku, takže nemusí být opakovány.
1005 nebo 2005	*n* > 1	Polygon, který je tvořen skupinou bodů propojených přímými úsečkami a oblouky. Opět platí stejné jako u předchozí možnosti.

SDO_ORDINATES atribut je definován jako pole čísel proměnné délky, který ukládá souřadnice, které tvoří hranici prostorového objektu. Hodnoty v poli jsou řazeny podle dimenzí. Například u hranice polygonu, který se skládá ze čtyř dvou-dimenzionálních bodů, pole vypadá `{ x1, y1, x2, y2, x3, y3, x4, y4 }`.[6]

Na dalších dvou příkladech budou popsány konstruktory `SDO_GEOMETRY` na bodovou geometrii a na geometrii čáry.

```
INSERT INTO VLAKY VALUES
(
      1,
      '63321',
      SDO_GEOMETRY
      (
            2001,
            8307,
            SDO_POINT_TYPE (15.75, 50.03),
            NULL,
            NULL
      )
);
```

Na příkladu je insert do tabulky vlaku a to přímo vlak '63321', u kterého nás zajímá až třetí položka. Jedná se o konstruktor `SDO_GEOMETRY`, který odpovídá bodu na mapě. Když si vezeme postupně jednotlivé položky: 2001 značí dvou-dimenzionální bod, 8307 je pro nás souřadnicový systém zeměpisné délky a šířky, následujícím údajem je `SDO_POINT`, kde jsme použili konstruktor se souřadnicemi 15.75 a 50.03. `SDO_ELEM_INFO` a `SDO_ORDINATES` není vyplněno, protože se jedná o bod.

```
INSERT INTO RAILWAY_LINK$
(
LINK_ID,LINK_NAME,START_NODE_ID,END_NODE_ID,
LINK_TYPE,ACTIVE,LINK_LEVEL,GEOMETRY,COST)
VALUES
(
     4163,
     'L4163',
     2766,
     2761,
     null,
     'Y',
     1,
     MDSYS.SDO_GEOMETRY
     (
      2002,
      8307,
      null,
      MDSYS.SDO_ELEM_INFO_ARRAY(1,2,1),
      MDSYS.SDO_ORDINATE_ARRAY(15.2366397221883,50.2433197222816,15.23794
27777396,50.2429883331723)),
      1
);
```

Na příkladu je příkaz na vložení řádku do tabulky `RAILWAY_LINK$`, která slouží k ukládání linek mezi body železniční sítě. Zaměříme se na sloupec `GEOMETRY` a na jeho konkrétní konstruktor. Prvního hodnota konstruktoru 2002 oznamuje, že se jedná o dvoudimenzionální linku, 8307 značí, že se použije souřadnicový systém zeměpisné délky a šířky, `SDO_POINT` není vyplněno, protože se nejedná o bod, `SDO_ELEM_INFO` obsahuje předpis trojčísel `(1, 2, 1)`, z toho první 1 oznamuje, že začínáme pracovat s hodnotami v `SDO_ORDINATES` od prvního čísla, 2 znamená, že se jedná o řetězec čáry a druhá jednička znamená, že se jedná o přímku, sloupec `SDO_ORDINATES` obsahuje souřadnice ve tvaru `{x1, y1, x2, y2 }`.

4.4.2 Pohledy geometrických metadat

Geometrické metadata popisující dimenze, horní a dolní hranice a toleranci v každé z dimenzí jsou uložena v globální tabulce patřící MDSYS (kterou uživatelé nemohou upravovat přímo). Každý Spatial uživatel má následující pohledy, které jsou dostupné v jeho schématu:

- **`USER_SDO_GEOM_METADATA`** obsahující metadata informace pro každou prostorovou tabulku patřící uživateli (schématu). Je to pouze pohled, který můžeme upravovat a umožňuje vkládat Spatial uživatelům metadata týkající se prostorových tabulek.

- **`ALL_SDO_GEOM_METADATA`** obsahují metadata informace pro každou prostorovou tabulku, na které uživatel může provádět `SELECT`.

Spatial uživatelé jsou odpovědný za vyplnění těchto pohledů. Pro každý prostorový sloupec musí být vložen příslušný řádek v USER_SDO_GEOM_METADATA pohledu. Oracle Spatial zajišťuje, že ALL_SDO_GEOM_METADATA pohled je také upraven, aby odrážel řádky, které jsme vložili do USER_SDO_GEOM_METADATA.

Každý pohled metadat má následující definici:

```
(
      TABLE_NAME  VARCHAR2(32),
      COLUMN_NAME VARCHAR2(32),
      DIMINFO     SDO_DIM_ARRAY,
      SRID        NUMBER
);
```

TABLE_NAME sloupec obsahuje jméno tabulky, která obsahuje sloupec typu SDO_GEOMETRY. Prostorová tabulka nemůže být indexově organizovaná tabulka, pokud chceme vytvořit prostorový index na prostorovém sloupci.

COLUMN_NAME sloupec obsahuje název sloupce typu SDO_GEOMETRY.

DIMINFO sloupec je pole objektového typu s proměnnou délkou, seřazené podle dimenzí a má jeden vstup pro každou dimenzi.

SRID sloupec obsahuje jednu věc z následujících: SRID hodnotu pro souřadnicový systém pro všechny geometrie ve sloupci nebo hodnotu NULL, pokud není specifický souřadnicový systém spojen s geometriemi.

Navíc v pohledu ALL_SDO_GEOM_METADATA je sloupec OWNER, který slouží k identifikování schématu, který vlastní tabulku specifikovanou ve sloupci TABLE_NAME.

Následující podmínky se vztahují na názvy schémat, tabulek a sloupců, které jsou uloženy v Oracle Spatial pohledech metadat:

- Názvy musí obsahovat pouze písmena, číslice a podtržítka. Například název nesmí obsahovat mezeru, apostrof, uvozovky nebo čárku.
- Všechny písmena v názvech jsou převedena na velká písmena před tím, než jsou uložena v pohledech geometrických metadat nebo před zpřístupněním tabulek. [6]

4.4.3 Pohledy prostorových indexů

Existují dvě sady pohledů prostorových indexů metadat pro každé schéma (uživatele): *xxx*_SDO_INDEX_INFO a *xxx*_SDO_INDEX_METADATA, kde *xxx* může být USER nebo ALL. Tyto pohledy jsou pro uživatele pouze pro čtení. Uživatelé je vytvoří, ale jsou spravovány Spatial indexovým rutinou.

xxx_SDO_INDEX_INFO pohledy obsahují základní informace okolo prostorových indexů. USER_SDO_INDEX_INFO obsahuje indexové informace pro každou

prostorovou tabulku vlastněnou uživatelem a ALL_SDO_INDEX_INFO obsahuje indexové informace pro každou prostorovou tabulku, na které uživatel může provádět SELECT. USER_SDO_INDEX_INFO a ALL_SDO_INDEX_INFO pohledy obsahují stejné sloupce, jak je zobrazeno na Tabulce 4, s výjimkou, že USER_SDO_INDEX_INFO pohled neobsahuje SDO_INDEX_OWNER sloupec.

Tabulka 4: Sloupce v xxx_SDO_INDEX_INFO. Zdroj: [autor] (data převzata z [6])

Název sloupce	Datový typ	Účel
SDO_INDEX_OWNER	VARCHAR2	Vlastník indexu.
INDEX_NAME	VARCHAR2	Jméno indexu.
TABLE_NAME	VARCHAR2	Jméno tabulky obsahující sloupec, nad kterým je vytvořen index.
COLUMN_NAME	VARCHAR2	Název sloupce, na kterém je vytvořen index.
SDO_INDEX_TYPE	VARCHAR2	Obsahuje QTREE (pro quadtree index) nebo RTREE (pro R-tree index).
SDO_INDEX_TABLE	VARCHAR2	Název tabulky prostorových indexů.
SDO_INDEX_STATS	VARCHAR2	Zastaralý, vyhrazené pro použití Oracle.

xxx_SDO_INDEX_METADATA pohled obsahuje detailnější informace ohledně metadat prostorových indexů. USER_SDO_INDEX_METADATA obsahuje indexové informace pro všechny prostorové tabulky vlastněné uživatelem a ALL_SDO_INDEX_METADATA obsahující indexové informace pro všechny prostorové tabulky, na které uživatel může provádět SELECT. USER_SDO_INDEX_METADATA a ALL_SDO_INDEX_METADATA pohledy obsahují stejné sloupce, s kterými se zde nebudeme zabývat.[6]

4.5 Topology Data Model

Topologický datový model Oracle Spatial umožňuje pracovat s daty jako uzly, hranami a plochami v topologii. Můžeme ukládat informace kolem topologických prvků a geometrických vrstev v Oracle Spatial tabulkách a pohledech metadat. Potom můžeme provádět určité Spatial operace odkazující na topologické prvky, například vyhledávání které řetězce (jako ulice) mají nějakou prostorovou interakci se speciální polygonovým objektem (jako park).

[6] Definici xxx_SDO_INDEX_METADATA naleznete ve zdroji [6].

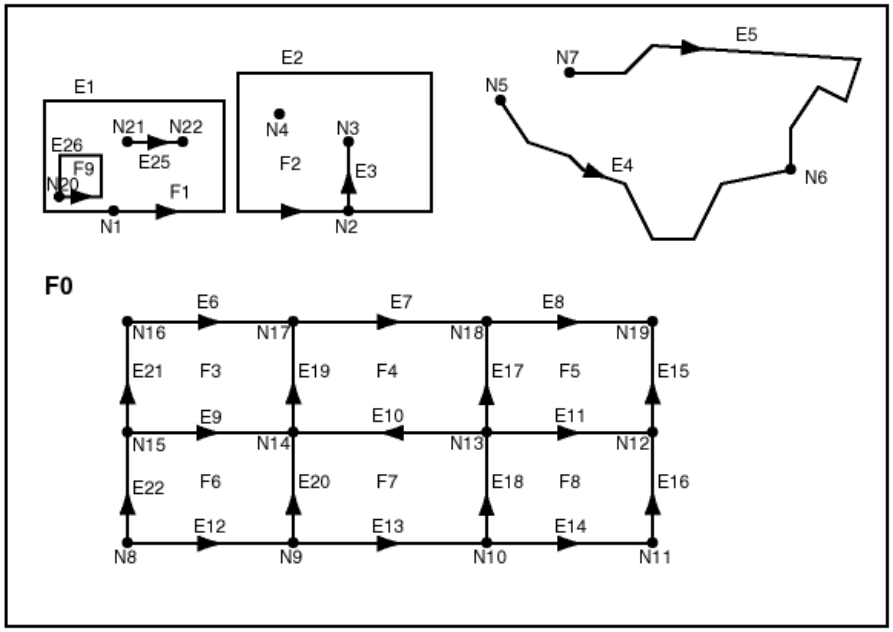
Topologie je obor matematiky zabývající se objekty v prostoru. Topologické vztahy obsahují vztahy, jako *obsahuje*, *uvnitř*, *pokrývá*, *je pokrýván*, *dotýká se* atd. Topologické vztahy se nezmění, když je souřadnicový prostor deformován jako zkroucení nebo roztahování. Příklady vztahů, které nejsou topologické, jsou třeba *délka*, *vzdálenost mezi* a *rozlohy*. Základní prvky v topologii jsou.

Uzly, reprezentované bodem, mohou být izolované nebo mohou být použity jako hranice hrany. Dvě nebo více hran se mohou setkávat v neizolovaném uzlu. Uzel má pár souřadnic s ním spojené, které popisují prostorové umístění pro daný uzel. Příkladem geografických prvků, které mohou být reprezentovány uzly, mohou být začáteční a koncové body ulic, zajímavá místa nebo třeba letiště (pokud je mapa v malém měřítku).

Hrany jsou ohraničeny dvěma uzly, začátečním uzlem a koncovým uzlem. Hrana má přiřazený geometrický objekt, obvykle souřadnicový řetězec, který popisuje prostorovou reprezentaci hrany. Hrana může mít několik vrcholů tvořící řetězec čáry, obloukový řetězec nebo kombinaci. Příkladem geografických prvků, které mohou být reprezentovány hranami, mohou být segmenty ulic a řek.

Pořadí souřadnic dává **směr** hraně. Směr je důležitý vymezující topologický vztah. Pozitivní směr souhlasí s orientací základní hrany a negativní směr obrací tuto orientaci.

Plocha, reprezentovaná polygonem, má odkaz na jednu směrovou hranu jeho vnější hranice. Pokud nějaký ostrovní uzly nebo hrany jsou přítomny, tak má také odkaz na jednu směrovou hranu na hranici každého ostrova. Příkladem geografických prvků, které mohou být reprezentovány plochami, mohou být parky, jezera, kraje a státy.



Obrázek 7: Zjednodušená topologie. Zdroj: [13]

Na Obrázek 7 je znázorněna zjednodušená topologie, která je popsána následujícím způsobem.

- E prvky (E1, E2 atd.) jsou hrany, F prvky (F0, F1 atd.) jsou plochy, N prvky (N1, N2 atd.) jsou uzly.
- F0 (nulová plocha) je vytvořena pro každou topologii. Je to univerzální plocha obsahující všechno ostatní v topologii. Není tam žádná geometrie spojená s univerzální plochou.
- K dispozici jsou uzly vytvořené pro každou bodovou geometrii a pro každý začáteční a koncový uzel hran. Například plocha F1 má pouze jednu hranu (uzavřená hrana), E1. Hrana má stejný uzel jako začáteční a koncový uzel (N1).
- Ostrovní vrchol je uzel, který je izolovaný v ploše. Například uzel N4 je ostrovní uzel v ploše F2.
- Ostrovní hrana je hrana, která je izolovaná v ploše. Například hrana E25 je ostrovní hrana v ploše F1.
- Informace kolem topologických vztahů jsou uloženy ve speciálních informačních tabulkách hran, ploch a uzlů. [13]

4.5.1 Topologické geometrie a vrstvy

Topologická geometrie (také označována jako *rys*) je prostorová reprezentace objektu reálného světa. Například *Hlavní ulice* by mohla být názvem topologické geometrie. Geometrie je ukládána jako sada topologických prvků (uzlů, hran a ploch). Každá topologická geometrie má unikátní ID (přiřazené Spatial, když záznam je importován nebo načten) spojené s ním.

Vrstva topologické geometrie je kolekce topologických geometrií speciálního typu. Například *Ulice* by mohla být vrstvou topologické geometrie, která obsahuje *Hlavní ulice* topologické geometrie. Každá vrstva topologické geometrie má unikátní ID (přiřazené Spatial) s ní spojené. Data pro každou vrstvu topologické geometrie jsou uložena v **tabulce rysů**. Například rysová tabulka pojmenovaná `MESTSKE_ULICE` může obsahovat informace kolem všech topologických geometrií (jednotlivé cesty a ulice) ve vrstvě *Ulice* topologických geometrií.

V některých topologiích vrstvy topologické geometrie (rysové vrstvy) mají jednu nebo více rodič-potomek vztahů v **topologické hierarchii**. To znamená, že vrstva na vrchní úrovni se skládá z rysů v jeho podřízené vrstvě, která je v hierarchii níže. Vrstva potomka může obsahovat rysy ve vrstvě jeho potomka jako další vrstva pod ní atd. Nejnižší úroveň v hierarchii je úroveň 0.

Pokud vrstvy topologické geometrie v topologii mají tento hierarchický vztah, je to daleko více efektivní mít vrstvy modelu hierarchické, než když specifikujeme všechny topologické geometrie na jedné vrstvě.

K modelování vrstev topologické geometrie jako hierarchické se specifikuje vrstva potomka v `CHILD_LAYER_ID` parametru při volání

SDO_TOPO.ADD_TOPO_GEOMETRY_LAYER proceduru k přidání rodičovské vrstvy topologické geometrie. Přidání nejnižší úrovně (úroveň 0) vrstvy topologické geometrie je první. [13]

4.5.2 Tabulky topologického data modelu

Pro používání Spatial topologických vlastností musíme nejdříve vložit data do speciálních tabulek hran, uzlů a ploch, které jsou vytvořeny Spatial, když vytváříte topologii.

Spatial automaticky spravuje tabulky vztahů pro každou topologii (<název_topologie>_RELATION$)[7], která je vytvořena poprvé, kdy rysová tabulka je spojena s topologií (při zavolání SDO_TOPO.ADD_TOPO_GEOMETRY_LAYER procedury, která specifikuje topologii).

Pro každou topologii jsou vytvořeny tabulky, které začínají na <název_topologie>_. (například <topologie>_EDGE$). Tyto tabulky jsou následující.

- **Hranová informační tabulka** sloužící k ukládání informací o hranách a název tabulky končí na EDGE$.
- **Uzlová informační tabulka** sloužící k ukládání informací o uzlech a název tabulky končí na NODE$.
- **Plošná informační tabulka** sloužící k ukládání informací o plochách a název tabulky končí na FACE$.
- **Tabulka vztahů**. Při práci s objekty topologie Spatial automaticky udržuje informace o jednotlivých objektech v těchto tabulkách. Název takovýchto tabulek končí na RELATION$.
- **Tabulka historie** slouží k zachycení vkládacích a mazacích operací a název tabulky končí na HISTORY$. [13]

4.5.3 Topologické datové typy

Hlavní datový typ spojený s topologickým datovým modelem je SDO_TOPO_GEOMETRY, který popisuje topologickou geometrii. SDO_TOPO_GEOMETRY typ má několik konstruktorů a jednu členskou funkci.

[7] <název_topologie> je specifikován při zavolání SDO_TOPO.CREATE_TOPOLOGY proceduře jako název topologie, kterou vytváříme.

SDO_TOPO_GEOMETY je objektový datový typ, sloužící k popisu topologické geometrie. Je ukládán v jednom řádku jednoho sloupce v uživateli definované tabulce. Objektový datový typ SDO_TOPO_GEOMETRY je definován jako:

```
CREATE TYPE sdo_topo_geometry AS OBJECT
(
      tg_type     NUMBER,
      tg_id       NUMBER,
      tg_layer_id NUMBER,
      topology_id NUMBER
);
```

Popis jednotlivých atributů je popsán v Tabulka 5.

Tabulka 5: Atributy SDO_TOPO_GEOMETRY typu. Zdroj: Autor (data převzata z [13])

Atribut	Vysvětlení
TG_TYPE	Typ geometrické geometrie: 1= bod, 2 = řetězec čáry, 3 = polygon nebo vícenásobný polygon, 4 = heterogenní kolekce.
TG_ID	Unikátní ID číslo (generováno Spatial) pro topologickou geometrii.
TG_LAYER_ID	ID číslo pro vrstvu topologické geometrie, do které topologická geometrie patří.
TOPOLOGY_ID	Unikátní ID číslo (generováno Spatial) pro topologii.

Každá topologická geometrie v topologii je unikátně identifikována pomocí kombinace jeho TG_ID a TG_LAYER_ID hodnoty. [13]

4.5.4 Přístupy pro editování topologických dat

Pro editování topologie můžeme použít PL/SQL nebo Java API. Obě možnosti v Oracle Spatial používají v paměti topologickou mezipaměť, konkrétně TopoMap objekt.

- Pokud použijeme PL/SQL API, můžeme buď explicitně vytvářet a používat vyrovnávací paměť nebo povolit Spatial vytvářet a používat vyrovnávací paměť automaticky.
- Když použijeme Java API, musíme explicitně vytvořit a použít vyrovnávací paměť.

TopoMap objekt je spojen s mezipamětí v paměti, která je spojena s topologií. Když se explicitně vytvoří a používá mezipaměť pro úpravu topologie, musíme vytvořit TopoMap objekt, který bude spojen s topologií, načíst všechny nebo některé topologie do vyrovnávací paměti, editovat objekty, pravidelně aktualizovat topologii pro čtení změn v databázi, potvrzovat změny prováděné v mezipaměti a čistit mezipaměť.

TopoMap objekt může být v aktualizovatelném módu nebo módu pouze pro čtení. To nastavujeme pomocí hodnoty `ALLOW_UPDATES` parametru, když voláme `SDO_TOPO_MAP.LOAD_TOPO_MAP` proceduru.

S TopoMap objektem pracujeme přes balíčky funkcí a procedur, které tvoří část PL/SQL API pro model prostorových topologických dat, a které jsou:

- **`SDO_TOPO`** – Tento balíček obsahuje podprogramy pro vytváření a práci s topologiemi.
- **`SDO_TOPO_MAP`** – Tento balíček obsahuje podprogramy vztahující se k editování topologií. Tyto podprogramy používají TopoMap objekt. Buď ten, který jsme dříve vytvořili, nebo ten, který vytvořil Spatial implicitně. [13]

4.6 Network Data Model

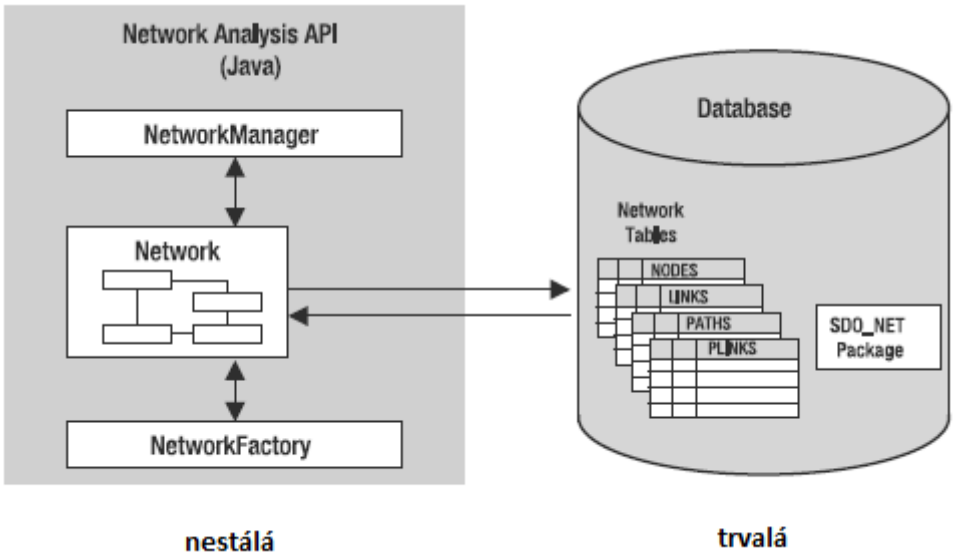
V některých aplikacích mohou být objekty modelovány jako uzly a vazby v síti. Síťový model obsahuje logické informace jako například vztahy připojení mezi uzly a vazbami, směry vazeb a náklady spojené s uzly a vazbami. S logickými síťovými informacemi můžeme analyzovat síť a odpovídat na otázky. Mnoho z nich se týká cest, výpočtů a sledování. Dále možnost logické síťové informace, kde bereme prostorovou informaci jako lokaci uzlů a geometrie vazeb, které mohou být spojovány s logickou sítí. Tato informace vám může pomoci modelovat logickou informaci (jako náklady na trasu, protože fyzická délka může být přímo počítána ze své prostorové reprezentace).

Možnosti síťového modelování Spatial zahrnuje objektové schéma a API. Schéma objektů zahrnuje metadata a síťové tabulky. Rozhraní zahrnuje na straně serveru PL/SQL API (SDO_NET balíček) pro vytváření, řízení a analyzování sítí v databázi a střední vrstvu (na straně klienta) Java API pro síťovou analýzu. [13]

Síťová podpora v Oracle Database 10g je složená z následujících prvků:

- Data model pro ukládání sítí uvnitř databáze jako sada síťových tabulek. To je *trvalá* kopie sítě.
- SQL funkce pro definování a správu sítě (`SDO_NET` balíček).
- Funkce síťové analýzy v jazyce Java. Java API pracuje na kopii sítě načtené z databáze. To je *nestálá* kopie sítě. Výsledky analýz (jinými slovy výpočet síťových cest) a síťové změny mohou být zapsány zpět do databáze.
- Funkce síťové analýzy v PL/SQL (`SDO_NET_MEM` balíček). Toto API je takový „obal“ nad Java API, které se pak vykonává uvnitř databáze. Tato technika stále užívá nestálou kopii sítě, která je nyní načtena na Javě založené paměti uvnitř databáze. [11]

Jednotlivé rozdělení prvků je znázorněno na Obrázku 8.



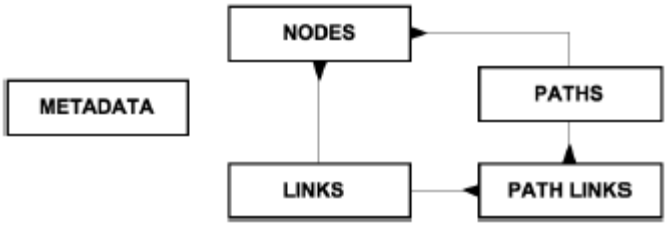
Obrázek 8: Oracle Network Data Model. Zdroj: [11]

4.6.1 Síťové tabulky

Síť je definovaná použitím dvou tabulek: tabulkou uzlů (*node* table) a tabulkou vazeb (*link* table). Musíme poskytovat ty tabulky s pořádnou strukturou a obsahem pro modelování sítě.

Síť může také mít tabulku cesty (*path* table) a tabulku vazby cesty (*path link* table). Tyto tabulky jsou volitelné a jsou plněné výsledky analýz poskytovaných v Java API, jako například nejkratší cesta mezi dvěma uzly.

Na Obrázek 9 lze vidět vztahy mezi tabulkami, které popisují síť.



Obrázek 9: Hlavní síťové tabulky. Zdroj: [11]

Máme velkou míru flexibility v členění tabulek – některé sloupce mohou být pojmenovány jakýmkoliv způsobem (sloupce geometrie a ceny jsou zvlášť) a jejich pořadí není důležité. Můžeme také zahrnout jiné sloupce, abychom drželi další informace.

Aktuální pojmenování tabulek, které ustanovují síť a jejich strukturu, jsou definovány v oddělených tabulkách metadat nazývaných `USER_SDO_NETWORK_METADATA`, které se upravují stejným způsobem jako základní prostorová data (`USER_SDO_METADATA`). Především to je místo, kde specifikujete název sloupce v tabulkách uzlů a vazeb, které definují hodnotu nákladů nebo název sloupce geometrie. To nám také dovoluje definovat různé sítě nad stejnou sadou tabulek použitím

různých nákladových sloupců – například jedna síť založená na vzdálenosti a jiná založená na cestovních časech.

Máme tedy několik technik pro definování datových struktur pro síť. Jedna možnost je zavolání jednoduché procedury, která udělá „všechno“. Vytvoří veškeré tabulky se standardními názvy a naplní metadata. Jiná možnost je vytvořit všechny tabulky ručně (nebo vytvořit pohledy nad existujícími tabulkami) a vyplnit metadata ručně. Dále jsou poskytnuty funkce k tomu, aby ověřily, že datové struktury jsou platné a souvislé. [11]

4.6.2 Koncepce síťového datového modelu

Síť je typ matematického grafu, který zachycuje vztahy mezi objekty používající konektivitu. Konektivita může nebo nemusí být založena na prostorové blízkosti. Například: když dvě města jsou na opačných stranách jezera, nejkratší cesta založená na prostorové blízkosti (přímka přes střed jezera) není relevantní, když chcete jet z jednoho města do jiného. Místo toho naleznete nejkratší dojezdové vzdálenosti. Potřebujete informace o konektivitě silnic a křižovatek a kolem „nákladů“ individuálních vazeb.

Síť se skládá z množiny uzlů a vazeb. Každá vazba (někdy také nazývaná hrana nebo segment) specifikuje dva uzly. Síť může být orientovaná nebo neorientovaná, ačkoli vazby a cesty typicky mají orientaci.

Zde jsou některé klíčové pojmy vztahující se k modelu datové sítě.

- **Uzel** reprezentuje objekt zájmu.
- **Vazba** reprezentuje vztah mezi dvěma uzly. Vazba může být směrová (to znamená, že má směr) nebo nesměrová (to znamená, že nemá směr).
- **Cesta** je střídavá posloupnost uzlů a vazeb, začíná a končí uzly, a obvykle se uzly a vazby v ní objevují pouze jednou.
- **Síť** je množina uzlů a vazeb. Síť je orientovaná, pokud obsahuje vazby, které jsou orientované, a sít je neorientovaná, pokud obsahuje vazby, které jsou neorientované.
- **Logická síť** obsahuje informace o konektivitě ale ne geometrické informace. To je model používaný pro síťovou analýzu. Logickou síť lze považovat za orientovaný graf nebo neorientovaný graf, v závislosti na aplikaci.
- **Prostorová síť** obsahuje informace o konektivitě a geometrické informace.
  V prostorové síti uzly a vazby jsou `SDO_GEOMETRY` geometrické objekty bez LRS informace (**SDO síť**) nebo s LRS informací (**LRS síť**) nebo `SDO_TOPO_GEOMETRY` objekty (síť geometrické topologie).

  V LRS síti každý uzel obsahuje hodnotu geometrie ID a hodnotu měřítka a každá vazba obsahuje hodnotu geometrie ID a koncovou hodnotu měřítka a hodnota geometrie ID v každém případě odkazuje na `SDO_GEOMETRY` objekt s LRS informací. Prostorová síť může být orientovaná nebo neorientovaná v závislosti na aplikaci.

- **Rys** je objekt zájmu v síťové aplikaci, která je spojená s uzlem nebo vazbou. Například v transportní síti rysy obsahují východy a křižovatky (mapovány do uzlů), dálnice a ulice (mapovaný do vazeb).
- **Náklad** je ne negativní číselný atribut, který může být spojen s vazbami nebo uzly pro počítání **minimálních nákladů cesty**, která je cestou s minimálními celkovými náklady z počátečního uzlu do koncového uzlu. Můžeme specifikovat nákladový faktor, jako například doba jízdy nebo jízdní vzdálenost pro vazby v síťových metadatech.
- **Dosažitelné uzly** jsou všechny uzly, které můžou být dosaženy z daného bodu. **Dosažené uzly** jsou všechny uzly, které můžou dosáhnout daný bod.
- **Stupeň** uzlu je číslo vazeb na (to znamená událost na) uzel. **In-stupeň** je číslo příchozích vazeb, a **out-stupeň** je počet odchozích vazeb.
- Síťové **omezení** jsou omezení definovaná na síťovou analýzu výpočtů. (Například, když trasy jízdy se musí skládat z dálnic a hlavních silnic).
- **Kostra grafu** ze spojeného grafu je strom (to znamená graf bez cyklů), který spojuje všechny uzly v grafu. Vazební orientace jsou ignorovány v kostře grafu. **Minimální náklad kostry grafu** je kostra grafu, která spojuje všechny uzly a má minimální celkové náklady. [13]

4.6.3 Síťová hierarchie

Některé síťové aplikace vyžadují reprezentaci na různých úrovních abstrakce. Například dva hlavní procesy mohou být reprezentovány jako uzly s vazbami mezi nimi na nejvyšší úrovni abstrakce, a každý hlavní proces může mít několik podřízených procesů, které jsou reprezentovány jako uzly a vazby na nižší úrovni.

Síťová hierarchie umožňuje reprezentovat síť s několika úrovněmi abstrakce přiřazením hierarchické úrovně pro každý uzel. (Vazby nejsou přiřazeny hierarchické úrovni a vazby mohou být mezi uzly v stejné hierarchické úrovni nebo v různých úrovních.) Nejnižší (nejdetailnější) úroveň v hierarchii je úroveň 1 a následně vyšší úrovně jsou číslované 2, 3 atd.

Uzly v sousedících úrovních síťové hierarchie mají rodič/potomek vztahy. Každý uzel na vyšší úrovni může být nadřazený uzel pro jeden nebo více uzlů na nižší úrovni. Každý uzel na nižší úrovni může být potomek uzel jednoho uzlu na vyšší úrovni. Sourozenecké uzly jsou uzly, které mají stejný nadřazený uzel.

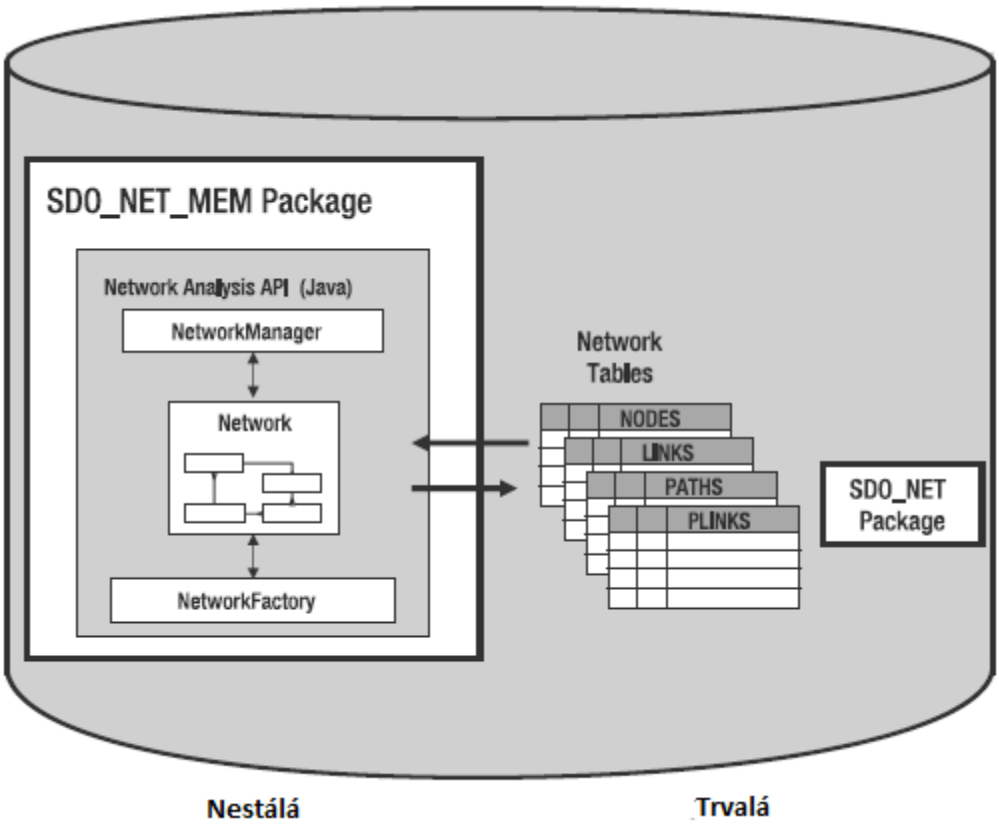
Vazby mohou také mít rodič/potomek vztahy. Nicméně protože vazby nejsou přiřazeny k hierarchické úrovni, není nutný vztah mezi vztahy odkazů rodič/potomek a úrovněmi hierarchie sítě. [13]

4.6.4 Síťové upravování a analýzy používáním síťového paměťového objektu

Tato sekce popisuje, jak vykonat síťové operace upravování a analýzy použitím **síťového paměťového objektu**, který je mezipaměť ve virtuální paměti. Můžeme nahrávat síť nebo hierarchickou úroveň sítě do síťového paměťového objektu, provádět operace na síťových objektech v paměťovém objektu a pak buď zahodit veškeré změny, nebo zapsat změny do sítě v databázi.

Více síťových paměťových objektů může existovat v době určené pro specifickou síť, ale pouze jedna může být aktualizovatelná. Všechny ostatní musí být jen pro čtení. Pro lepší výkon, pokud plánujete používat síťový paměťový objekt pouze na získání informací nebo provádění operací síťové analýzy, učiňte síťový paměťový objekt pouze pro čtení.

V síťovém data modelu PL/SQL API podprogramy v `SDO_NET` balíčku operují na síti v databázi a podprogramy v `SDO_NET_MEM` balíčku operují na síťovém paměťovém objektu ve vyrovnávací paměti. Pro některé operace síťových úprav (jako přidávání uzlů, vazeb nebo cest) můžeme použít buď `SDO_NET` nebo `SDO_NET_MEM` procedury. Balíček `SDO_NET` obsahuje funkce a procedury pro správu sítě. Balíček `SDO_NET_MEM` obsahuje podprogramy, které tvoří část PL/SQL API pro Spatial síťový data model. Tento balíček, který implementuje funkce k dispozici prostřednictvím Java API, obsahuje podprogramy vztahující se k editaci a analýze sítě. Balíčky jsou zachyceny na Obrázku 10.



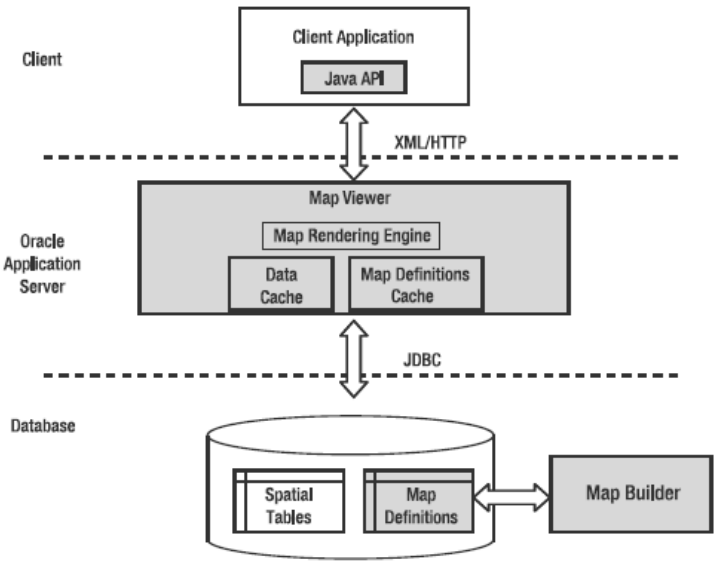
Obrázek 10: PL/SQL a Java APIs. Zdroj: [11]

Balíček SDO_NET_MEM jde rozdělit do seskupení podprogramů podle určitého účelu.

- SDO_NET_MEM.NETWORK_MANAGER podprogramy k vytváření a rušení síťových paměťových objektů a provádění analýz. Podprogramy odpovídají Java třídě *NetworkManager*.
- SDO_NET_MEM.NETWORK podprogramy slouží k přidávání a mazání uzlů, vazeb a cest. Podprogramy odpovídají Java třídě *Network*.
- SDO_NET_MEM.NODE podprogramy pracující s nastavením a výpisem atributů pro uzly. Podprogramy odpovídají Java třídě *Node*.
- SDO_NET_MEM.LINK podprogramy pracující s nastavením a výpisem atributů pro vazby. Podprogramy odpovídají Java třídě *Link*.
- SDO_NET_MEM.PATH podprogramy pracující s nastavením a výpisem atributů pro cesty. Podprogramy odpovídají Java třídě *Path*. [11]

5 Oracle Application Server

Oracle Spatial architekturu si můžeme rozdělit na dvě části: Oracle databázi, která byla podrobněji popsána v kapitole 0, a Oracle aplikační server (viz Obrázek 11).



Obrázek 11: Oracle MapViewer architektura. Zdroj: [11]

V předchozích kapitolách jsme si ukázali, jak se pracuje s prostorově založenými dotazy a jak manipulovat s prostorovými objekty v Oracle databázi. Pokud bychom chtěli prostorová data vizualizovat, je možné využít vizualizační nadstavby Oracle MapViewer.

MapViewer je komponenta na straně serveru, která umožňuje vytvářet mapy čtením vhodných databázových pohledů a tabulek. Vrací mapy do klientských aplikací v příslušných formátech. Každá mapa je sestavena použitím jedné nebo více vrstev nebo témat. Každé téma reprezentuje logické seskupení geografických prostorových rysů, jako jsou například silnice, adresy zákazníků, řeky atd. Tyto rysy jsou vykreslovány speciálními styly. [11]

5.1 Oracle MapViewer

Oracle MapViewer je čistě Java komponenta na straně serveru zahrnuta v Oracle aplikačním serveru. MapViewer komponenta je poskytována od Oracle Application Server 10g. MapViewer slouží k vykreslování prostorových dat, která jsou uložena v `SDO_GEOMETRY` sloupcích v Oracle tabulkách, jako zobrazení map. Tato hlavní součást, znázorněná na Obrázek 11, se skládá z:

- *Mapově vykreslovacího enginu běžícího v Oracle aplikačním serveru*. Vykreslovací engine je vystaven jako servlet, který zpracovává žádosti zaslané klientskými aplikacemi, načte správné informace z prostorových tabulek a vytvoří mapy v různých grafických formátech (GIF, PNG, JPEG

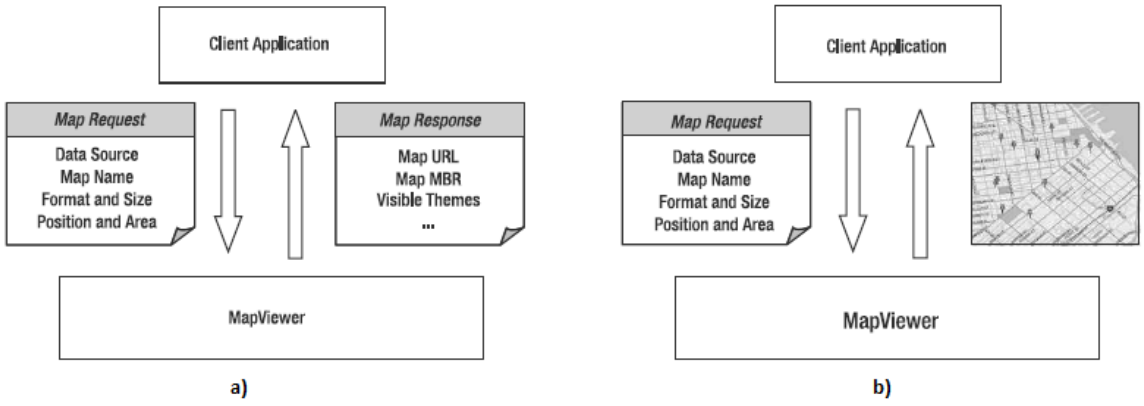
nebo SVG), které pak vrátí klientovi. Kromě jádra mapovacího servletu, MapViewer server nabízí mapový server pro vyrovnávací paměť a FOI[8].

- *Mapové definice*. Mapové definice jsou ukládány v databázi. Je to místo, kde jsou popsány mapy. Jedná se o tabulky, které se používají pro vykreslování map (barva, tloušťka čáry a písmo) atd.

- *Série rozhraní pro API*. Tyto API umožňují přistupovat k vlastnostem MapVieweru z různých vývojových prostředí aplikací. Tato rozhraní API obsahují XML, Java, PL/SQL a JavaScript (Ajax) rozhraní. Java API obsahuje JSP značky pro usnadnění začlenění map v JSP.

- *Grafický Map Builder nástroj*. Program, který pomáhá se správou mapových definic uložených v databázi.

Klientské aplikace se dotazují MapViewer servletu přes http pomocí žádost/odpověď modelu. Tyto požadavky a odpovědi jsou kódovány v XML. Java klienti mohou použít Java API, která se stará o výstavbu a zasílání XML žádostí, stejně jako čtení a parsování XML odpovědí, viz Obrázek 12 a). Alternativou je přímé zasílání obrázku mapy klientským aplikacím namísto URL generované mapy, viz Obrázek 12 b). [11]



Obrázek 12: MapViewer žádost/odpověď tok a tok se zasíláním obrázků. Zdroj: [11]

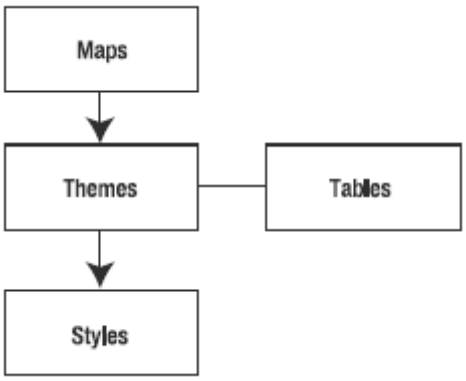
[8] FOI jsou prvky na mapě, s kterými uživatel může pracovat. Bývají zobrazovány jako obrázky ale také i jako geografické prvky. Jsou dynamické a se statickými základními mapami umožňují vytvářet webové mapovací aplikace.

5.1.1 Oracle Maps

Oracle Maps je název pro sady technologií poskytovaných MapViewerem, která umožňuje stavět vysoce dynamické a vysoce výkonné mapové aplikace. MapViewer architektura s Oracle Maps používá tyto hlavní součásti:

- *Server mapové mezipaměti* – ukládání generovaných map do mezipaměti. Mapy jsou ukládány dlaždicově pomocí souboru, které jsou sdíleny mezi všemi uživateli, pro rychlejší prohlížení map.
- *FOI sever* – sever, který čte a vykresluje dynamické vlastnosti pro prostorové tabulky. Například stanice, která je zobrazena jako obrázek budovy a po kliknutí zobrazí navíc další údaje.
- *Ajaxově založený JavaScript mapovací klient API* – JavaScript knihovna poskytuje všechny funkce potřebné aplikacemi k zobrazení a interakci s mapami s vysoce dynamickým způsobem: hladkým tažením a přibližováním, informační tipy, informační okna, dotazy a výběry.

Mapa je vytvářena z jednoho nebo více témat, které odkazují na styly. Téma je založené na tabulce. Na Obrázek 13 můžeme vidět tyto vztahy.



Obrázek 13: Prvky mapové definice. Zdroj: [11]

Styly popisují, *jak* informace mají na mapě vypadat: barvy, značky atd. Témata definují informace, které se objeví na mapě a jaký využívají styly na zobrazení. Různé mapy mohou používat stejná témata a různá témata mohou používat stejné styly. Téma může používat hodně různých stylů. Tak různá témata mohou být založena na stejné tabulce. Tabulka může být i pohled nebo dynamicky definovaný SQL dotaz.

Definice jsou uloženy v databázi v sadě tří slovníkových pohledů: `USER_SDO_MAPS`, `USER_SDO_THEMES` a `USER_SDO_STYLES`. Podobně jako všechny slovníkové pohledy přicházejí ve třech variantách: `USER`, `ALL` a `DBA`. Vztahy mezi jednotlivými prvky nejsou implementovány pomocí referenčního integritního omezení, ale spíše MAP definice obsahuje seznam témat, a TÉMA se vztahuje k seznamu stylů. Tyto seznamy jsou kódovány v definičních prvcích pomocí XML notací.

Můžeme přímo upravovat tabulky, kde jsou uloženy mapové, stylové a témat definice. Požadavkem však je, že rozumíte precizně jak struktuře tabulek tak i XML syntaxi. Lepší je využít nástroj mapových definic nazývaný Map Builder, který je poskytován MapViewerem. Tento nástroj bude popsán v následující kapitole. [11]

5.1.2 Oracle Map Builder

Oracle Map Builder je samostatná aplikace, která umožňuje vytvářet a spravovat mapovací metadata (informace o stylech, tématech a základních mapách), které jsou uloženy v databázi. Například užití tohoto nástroje k vytváření stylů nebo modifikování definice stylu. Kromě zpracování metadat nástroj poskytuje rozhraní pro zobrazení metadat (například vidět, jak se styl čáry objeví na mapě) a také prostorové informace.

Při vytváření, upravování a mazání informací o tématech, stylech a mapách by se měl používat Oracle Map Builder, místo přímé změny metadat pohledů MapViewer. V případě jakýchkoliv změn provedených mimo Oracle Map Builder, jako například příkazy SQL, by se mělo aktualizovat připojení k databázi v Oracle Map Builderu pro zobrazení aktuálních položek. [12]

6 Vybudování grafové reprezentace segmentu železniční sítě

6.1 Úvod

Hlavním cílem DP bylo vytvořit grafovou reprezentaci vybraného segmentu železniční sítě pomocí technologie Oracle Spatial Topology and Network Data Models. Segment byl zvolen podle poskytnutých dat pokrývající Pardubický a Kralohradecký kraj, které byly poskytnuty jako data ve formě SQL souborů.

Oracle Spatial nabízí mnoho prostorových funkcionalit, které lze použít při tvorbě grafu, který obsahuje prostorová data. Jako možnosti se nabízí hned dvě části Oracle Spatial a to:

- topologický datový model a
- síťový datový model.

Topologický datový model se spíše hodí při práci s územními daty, kdy modelujeme rozsáhle územní plochy jako krajiny, města atd. Tento model umožňuje nad svými topologickými geometriemi používat vlastní funkcionality pro zjištění interakce mezi svými objekty.

Síťový datový model se hodí například pro reprezentaci silničních sítí, sítí metra, počítačových sítí ale také železničních sítí. Síťový datový model poskytuje síťovou reprezentaci železniční sítě, kdy si můžeme převést body hektometrovníků na uzly a traťové úseky mezi nimi na vazby. Na síti dále můžeme vytvářet jednotlivé trasy pro vlaky nebo používat funkcionality tohoto modelu, jako například hledání nejkratší cesty mezi stanicemi.

Pro lepší vybudování grafové reprezentace segmentu železniční sítě se nám více hodí síťový datový model.

Síťový datový model umožňuje vytvořit jeden z následujících typů sítě:

- prostorovou síť s ne-LRS `SDO_GEOMETRY` objekty,
- prostorovou síť s LRS `SDO_GEOMETRY` objekty,
- prostorovou síť s `SDO_TOPO_GEOMETRY` objekty a
- logickou síť.

Pro naši potřebu využiji prostorovou síť s ne-LRS `SDO_GEOMETRY` objekty, protože máme data se souřadnicemi zeměpisné délky a šířky.

6.2 Zvolení struktury

Než vůbec jsem mohl začít budovat graf železniční sítě, musel jsem si zvolit, kterou strukturu bude mít železniční síť. Struktury, které se nabízeli, byly jednoúrovňová síť a hierarchická síť, která umožňovala rozdělit železniční síť na dvě úrovně a to abstraktní a

detailnější úroveň. Bohužel tato možnost při vizualizaci sítě ještě není podporována v Oracle Database 11g MapViewerem. Jako kompromis byla zvolena první možnost jednoúrovňové sítě s tím, že bude vymodelována ještě druhá **abstraktní síť** pro vizualizaci sítě do testovací aplikace.

Jednoúrovňová síť, dále označována jako **detailnější síť**, se bude skládat z bodu hektometrovníků a vazeb mezi nimi. V dalších kapitolách budování grafu se tedy budeme zabývat pouze touto sítí.

Abstraktní síť slouží k znázornění modelu segmentu železniční sítě na abstraktní úrovni. Místo zobrazení bodů hektometrovníků a hran mezi nimi používáme super hrany a super uzly. Pro vybudování této sítě bylo nejdříve nutné zjistit, které uzly hektometrovníků mají k sobě více než 2 vazby. Pokud uzel má více než 2 vazby nebo pouze 1, tak byl zvolen jako super uzel. Super hrana zase je hrana, která spojuje super uzly mezi sebou podle hran mezi vypuštěnými uzly.

Rozdíl mezi abstraktní a detailní sítí je znázorněn na Obrázek 14.

Obrázek 14: Rozdíl mezi detailní (a) a abstraktní (b) sítí. Zdroj: [autor]

6.3 Postup vybudování grafu

V této části se zaměříme na hlavní kroky pro vybudováním grafu pomocí síťového data modelu Oracle Spatial, tyto kroky jsou následující:

- vytvoření sítě,
- naplnění sítě,
- validace sítě,
- vložení informací o síti do tabulky metadat a
- vytvoření prostorového indexu.

6.3.1 Vytvoření sítě

Pro vytvoření sítě se nejdříve potřebuje rozhodnout, jak danou síť vytvoříme. Máme dvě možnosti: vytvořit síť za pomoci Spatial nebo vytvořit síť manuálně. Pro naše

potřeby je jedno, jakou možnost použijeme, protože obě možnosti mají stejný výsledek, jaký potřebujeme. Ve zbytku této kapitoly si tedy ukážeme pouze první možnost.

Pro vytvoření naší sítě použijeme proceduru:

```
EXECUTE SDO_NET.CREATE_SDO_NETWORK('RAILWAY', 1, FALSE, FALSE);
```

Procedura SDO_NET.CREATE_SDO_NETWORK vytvoří prostorovou síť obsahující ne-LRS SDO_GEOMETRY objekty. Parametry procedury jsou: parametr NETWORK, který je název vytvářené sítě, parametr NO_OF_HIERARCHY_LEVELS nastavující počet hierarchických úrovní pro vazby v síti, parametr IS_DIRECTED nastavuje, jestli je síť orientovaná a parametr NODE_WITH_COST umožňuje vytvořit sloupec pro cenu, pokud není specifikován, tak se bere standardně hodnota *false*. V našem případě se síť jmenuje RAILWAY, obsahuje pouze jednu hierarchickou úroveň pro vazby v síti, má vypnutou orientaci (není teda směrová) a nemá vygenerovaný nákladový sloupec v tabulce uzlů.

Procedura SDO_NET.CREATE_SDO_NETWORK má dva možné formáty. První formát je kratší a vytváří implicitně potřebné tabulky. Tento formát byl použit v našem případě a vytvořil tabulky RAILWAY_LINK$, RAILWAY_NODE$, RAILWAY_PATH$ a RAILWAY_PLINK$. Druhý formát procedury je komplexnější, protože oproti prvnímu formátu ještě zahrnuje přiřazení síti jednotlivé tabulky a jejich sloupce s geometriemi a náklady.

6.3.2 Naplnění sítě

Před naplněním sítě si nejdříve musíme doplnit do vygenerovaných tabulek chybějící sloupce, potřebné pro železniční síť a import dat ze souborů. Jedná se hlavně o sloupec TUDU v tabulce uzlů.

V následující fázi bude proveden import dat do tabulky RAILWAY_NODE$ a RAILWAY_LINK$.

6.3.3 Validace sítě

Pro ověření sítě můžeme použít SDO_NET.VALIDATE_NETWORK funkci v následujícím tvaru:

```
SELECT SDO_NET.VALIDATE_NETWORK('RAILWAY') FROM DUAL;
```

VALIDATE_NETWORK vrací *true*, pokud je síť validní, a vrací *false*, když není validní. Funkce kromě detekce, jestli síť a její tabulky existují, kontroluje geometrii ve sloupcích typu SDO_GEOMETRY v tabulkách uzlů a cest, jestli jsou vyplněny a jestli každý začáteční a koncový uzel pro každou vazbu existuje v tabulce uzlů.

6.3.4 Nastavení metadat sítě a prostorového indexu

V poslední fázi vytvoření prostorové sítě s ne-LRS SDO_GEOMETRY objekty musíme vložit informace o každém sloupci typu SDO_GEOMETRY do pohledu USER_SDO_GEOM_METADATA a vytvořit prostorové indexy nad těmito sloupci.

Pro zaznamenání sloupců s geometrií z tabulek RAILWAY_NODE$ a RAILWAY_LINK$ použijeme následující příkazy:

```
INSERT INTO USER_SDO_GEOM_METADATA
VALUES (
  'RAILWAY_NODE$',
  'GEOMETRY',
  SDO_DIM_ARRAY(
    SDO_DIM_ELEMENT('X', 0, 20, 0.00005),
    SDO_DIM_ELEMENT('Y', 0, 60, 0.00005)
  ),
  8307
);

INSERT INTO USER_SDO_GEOM_METADATA
VALUES (
  'RAILWAY_LINK$',
  'GEOMETRY',
  SDO_DIM_ARRAY(
    SDO_DIM_ELEMENT('X', 0, 20, 0.00005),
    SDO_DIM_ELEMENT('Y', 0, 60, 0.00005)
  ),
  8307
);
```

U každého sloupce GEOMETRY v tabulkách nastavíme dimenze na hodnoty: souřadnice X od 0 do 20 a souřadnice Y od 0 do 60. Tolerance u obou dimenzí je stanovena na 0.0005.

V poslední řadě vytvořím prostorové indexy u každé tabulky na sloupec s SDO_GEOMETRY typem.

```
CREATE INDEX RAILWAY_NODES_IDX ON RAILWAY_NODE$(GEOMETRY)
INDEXTYPE IS MDSYS.SPATIAL_INDEX;
CREATE INDEX RAILWAY_LINKS_IDX ON RAILWAY_LINK$(GEOMETRY)
INDEXTYPE IS MDSYS.SPATIAL_INDEX;
```

Prostorový index u tabulky RAILWY_NODE$ na sloupci GEOMETRY je pojmenovaný jako RAILWAY_NODES_IDX a prostorový index u tabulky RAILWAY_LINK$ na sloupci GEOMETRY je pojmenovaný jako RAILWAY_LINKS_IDX. Při vytvoření indexů jsou vytvořeny tabulky pro jednotlivé prostorové indexy. Indexové tabulky jsou specifikované v pohledu USER_SDO_INDEX_METADATA.

6.4 Struktura grafu

V této části si popíšeme strukturu sítě, kterou jsme vytvořili v předchozích fázích. Strukturu si rozdělíme podle jednotlivých fází. Hlavní čtyři tabulky byly vytvořeny při použití procedury `SDO_NET.CREATE_SDO_NETWORK`:

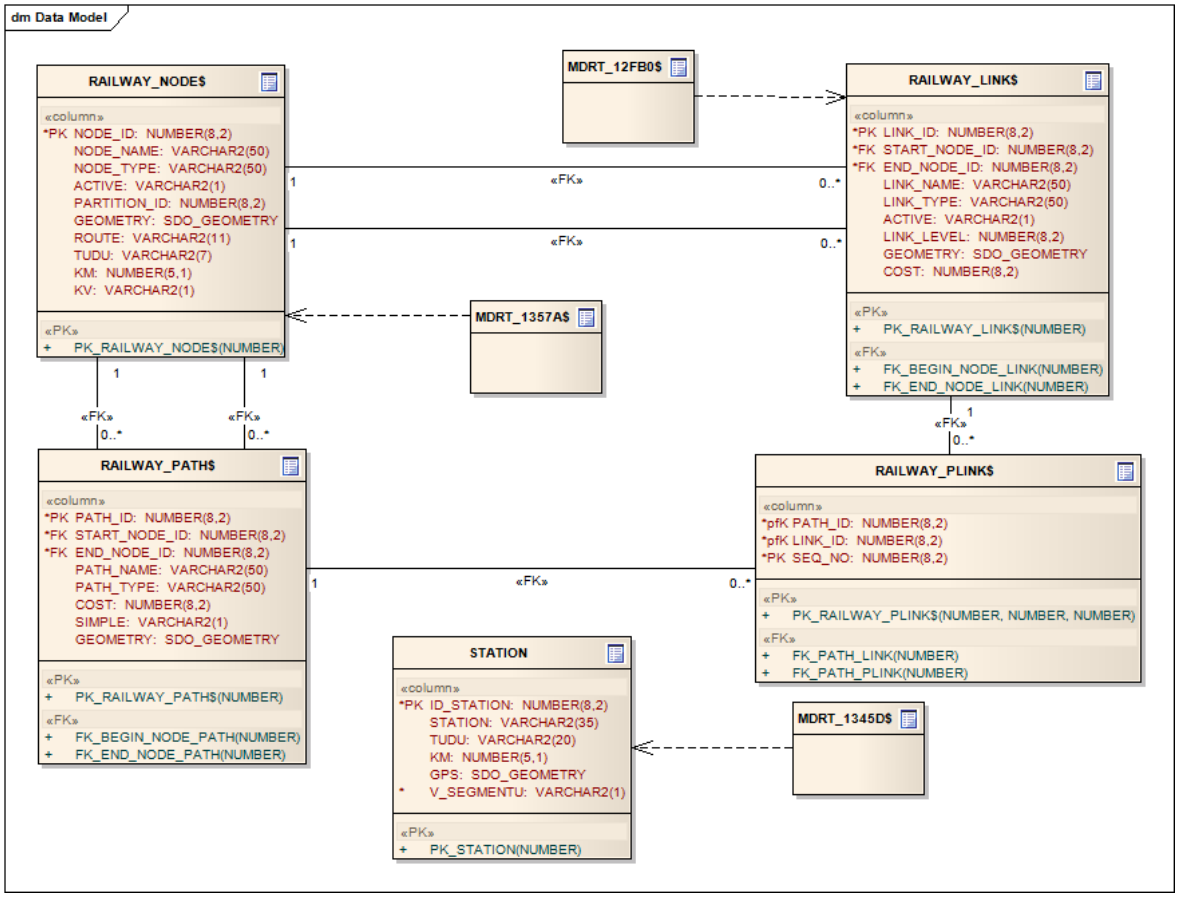
- `RAILWAY_NODE$`,
- `RAILWAY_LINK$`,
- `RAILWAY_PATH$` a
- `RAILWAY_PLINK$`.

Při vytváření prostorových indexů se nám vytváří speciální indexové tabulky, které jsou označovány jako MDRT tabulky s kódem (např. `MDRT_12FB0$`). Přiřazení mezi speciálními indexovými tabulkami a indexy jsou uloženy v pohledu `USER_SDO_INDEX_METADATA`.

Poslední tabulka, která je využívaná, ale není přímo spojená s grafem, je tabulku `STATION`, o které je pojednáno v kapitole 6.4.6.

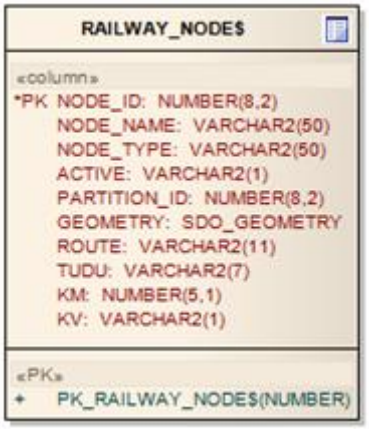
6.4.1 UML Diagram

UML diagram tabulek železniční sítě je zachycen na Obrázku 15.



Obrázek 15: UML diagram tabulek železniční sítě. Zdroj: [autor]

6.4.2 Tabulka `RAILWAY_NODE$`



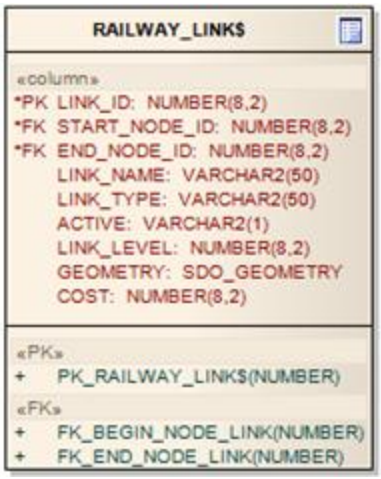
Obrázek 16: UML diagram tabulky `RAILWAY_NODE$`. Zdroj: [autor]

Tabulka `RAILWAY_NODE$` (viz Obrázek 16) obsahuje údaje o bodech hektometrovníků. Struktura tabulky se skládá z vygenerovaných sloupců a sloupců navíc pro import:

- vygenerované sloupce:
  - `NODE_ID` – identifikační číslo uzlu. Jediný parametr, který je při vytváření tabulky uzlů pro síťový datový model povinný,
  - `NODE_NAME` – jméno uzlu, které vzniklo ze spojení ID uzlu a prefixu „N“. Například pro uzel s ID 432 bude použito jméno „N432“,
  - `NODE_TYPE` – určení typu uzlu, v našem případě není tato položka vyplněna,
  - `ACTIVE` – obsahuje příznak „Y“, když je uzel aktivní, nebo příznak „N“, když je uzel neaktivní. Pokud je uzel neaktivní, tak síť vynechává tento uzel ve svých analýzách,
  - `PARTITION_ID` – identifikační číslo oddílu, které není zatím využíváno,
  - `GEOMETRY`[9] – geometrie uzlu, s kterou pracuje Oracle Spatial a používá se na vykreslování uzlu,
- sloupce, které byly vloženy do tabulky pro import:
  - `ROUTE` – je nepoužívaný sloupec, který byl vytvořen pro potřeby importu,
  - `TUDU` – označení TUDU, kterému uzel náleží,
  - `KM` – číslo určující kilometr, na kterém se uzel nachází,
  - `KV` – označení, zda je hektometr umístěn na koleji (K) nebo na výhybce (V).

[9] Sloupec `GEOMETRY` není povinný, protože síťový datový model může být i logický, kde nepracujeme s prostorovými daty.

6.4.3 Tabulka `RAILWAY_LINK$`

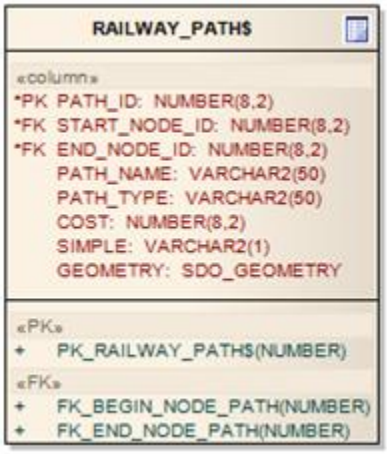


Obrázek 17: UML diagram tabulky `RAILWAY_LINK$`. Zdroj: [autor]

Tabulka `RAILWAY_LINK$` (viz Obrázek 17) slouží k ukládání vazeb mezi jednotlivými uzly železniční sítě. Tyto vazby tvoří traťové úseky. Tabulka se skládá z následujících sloupců:

- `LINK_ID` – identifikační číslo vazby. Tento údaj je při vytváření tabulky `RAILWAY_LINK$` povinný,
- `START_NODE_ID` – identifikační číslo uzlu, z kterého je vedena vazba. Tento údaj je povinný,
- `END_NODE_ID` – identifikační číslo uzlu, do kterého je vedena vazba. Tento údaj je povinný,
- `LINK_NAME` – jméno vazby, které vzniklo ze spojení ID uzlu a prefixu „L“. Například pro vazbu s ID 3506 bude použito jméno „L3506“,
- `LINK_TYPE` – určení typu vazby, v našem případě není tato položka vyplněna,
- `ACTIVE` – obsahuje příznak „Y“, když je vazba aktivní, nebo příznak „N“, když je vazba neaktivní. Pokud je vazba neaktivní, tak síť vynechává tuto vazbu v analýzách,
- `LINK_LEVEL` – prioritní úroveň pro vazbu. Tato hodnota je využívaná pro síťové analýzy. Vazba s nejvyšší prioritní úrovní může být spíše použita. Pro nás mají všechny vazby stejnou prioritu,
- `GEOMETRY` – geometrie vazby, s kterou pracuje Oracle Spatial a používá se také na vykreslování vazby,
- `COST` – hodnota nákladu spojená s danou vazbou. Pro nás počítáme náklady jako délka trati v kilometrech.

6.4.4 Tabulka `RAILWAY_PATH$`

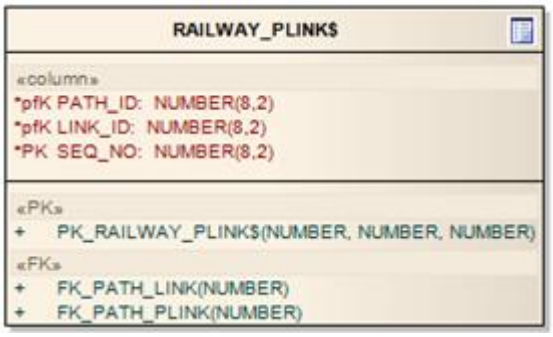


Obrázek 18: UML diagram tabulky `RAILWAY_PATH$`. Zdroj: [autor]

Tabulka `RAILWAY_PATH$` (viz Obrázek 18) slouží k ukládání cest na síti. Cestu si můžeme vytvořit sami nebo nechat vygenerovat algoritmem nejkratší cesty. Tabulka má následující tvar:

- `PATH_ID` – identifikační číslo cesty. Tento údaj je při vytváření tabulky `RAILWAY_PATH$` povinný,
- `START_NODE_ID` – identifikační číslo uzlu, kterým trasa začíná. Povinný údaj,
- `END_NODE_ID` – identifikační číslo uzlu, kterým trasa končí. Povinný údaj,
- `PATH_NAME` – pojmenování cesty,
- `PATH_TYPE` – uživatelsky definovaný typ cesty, v našem případě nevyplněn,
- `COST` – celkové náklady na cestu,
- `SIMPLE` – když obsahuje hodnotu „Y“, je cesta jednoduchá, pokud každá vazba v cestě je projetá pouze jednou. Pokud obsahuje hodnotu „N“, je cesta komplexní a vazby v cestě mohou být projety vícekrát,
- `GEOMETRY` – sloupec obsahující geometrii cesty.

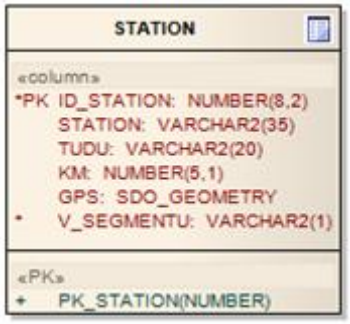
6.4.5 Tabulka `RAILWAY_PLINK$`



Obrázek 19: UML diagram tabulky `RAILWAY_PLINK$`. Zdroj: [autor]

Tabulka RAILWAY_PLINK$ (viz Obrázek 19) slouží pro zaznamenání použitých vazeb a jejich pořadí v cestě. Tabulka obsahuje sloupce: PATH_ID, které je identifikační číslo cesty, LINK_ID je identifikační číslo použité vazby a SEQ_NO říká v jakém je pořadí daná vazba v cestě.

6.4.6 Tabulka STATION



Obrázek 20: UML diagram tabulky STATION. Zdroj: [autor]

Tabulka STATION obsahuje záznamy jednotlivých železničních stanic na území České Republiky. Tabulka je zobrazena na Obrázku 20. Tuto tabulku používáme hlavně pro aplikaci, kdy hledáme nejkratší cestu mezi dvěma stanicemi. Tabulka se skládá z následujících sloupců:

- ID_STATION – identifikační číslo stanice,
- STATION – název železniční stanice,
- TUDU – traťově definiční úsek, na kterém se železniční stanice nachází,
- KM – číslo určující kilometr, na kterém se uzel nachází,
- GPS – bod středu železniční sítě, který je reprezentován SDO_GEOMETRY,
- V_SEGMENTU – pokud obsahuje hodnotu „A“, tak daná stanice je v námi vymezeném segmentu železniční sítě. Pokud obsahuje hodnotu „N“, tak stanice není v námi vymezeném segmentu železniční sítě.

6.5 Práce s železniční sítí

U modelu železniční sítě neuvažujeme, že do budoucna budeme přidávat nové uzly a vazby mezi nimi. Dále také neuvažujeme, že si jednotlivé trasy budeme tvořit ručně. V testovací aplikaci nás bude zajímat hlavně přepočet souřadnic na model železniční sítě a používání síťových analýz, jako zjištění nejkratší cesty. Síť pro nás má statický charakter a funkcionality budeme používat převážně na paměťovém objektu ve virtuální paměti.

6.5.1 Vytvoření a rušení paměťového objektu RAILWAY sítě

Pro práci s paměťovým objektem potřebujeme mít vytvořeny dvě procedury. První procedura bude sloužit k vytvoření paměťového objektu pro naši RAILWAY síť a bude otevřena pouze pro čtení. Paměťový objekt potřebujeme pro provádění síťových analýz (hledání nejkratších cest mezi železničními stanicemi). Vytvoření paměťového objektu a funkce síťové analýzy jsou obsaženy v balíčku SDO_NET_MEM.NETWORK_MANAGER.

```
PROCEDURE RUN_NETWORK AS
BEGIN
  SDO_NET_MEM.NETWORK_MANAGER.READ_NETWORK('RAILWAY', 'FALSE');
END;
```

Druhá procedura má na starost paměťový objekt s načtenou RAILWAY sítí odstranit.

```
PROCEDURE STOP_NETWORK AS
BEGIN
  SDO_NET_MEM.NETWORK_MANAGER.DROP_NETWORK('RAILWAY');
END;
```

6.5.2 Identifikace polohy na železniční síti

GPS souřadnice, které jsou posílány z vlaků, nebudou nikdy přesně odpovídat skutečnému místu na železniční sítí. Pro identifikaci polohy vlaku na železniční síti, je potřeba vytvořit funkci, která přepočítává obdržené souřadnice na souřadnice, které odpovídají poloze na železniční síti. Pro zjištění bodu na železniční síti potřebujeme dvě metody, protože využíváme dvě sítě, Detailní a Abstraktní. První verze se týká detailní železniční sítě a použijeme pro ni následující příkaz:

```
SELECT N.GEOMETRY.SDO_POINT.X, N.GEOMETRY.SDO_POINT.Y
FROM RAILWAY_NODE$ N
WHERE SDO_NN( N.GEOMETRY,
              MDSYS.SDO_GEOMETRY(2001, 8307,
              MDSYS.SDO_POINT_TYPE(X,Y, NULL), NULL, NULL),
              'SDO_NUM_RES = 1', 1) = 'TRUE'
ORDER BY SDO_NN_DISTANCE(1);
```

Příkaz využívá operátor `SDO_NN`, který hledá nejbližší sousedy k dotazované geometrii. Proměnné X a Y jsou souřadnice zeměpisné délky a šířky. Parametry `SDO_NN` jsou: první parametr je tabulkový sloupec `SDO_GEOMETRY`, ve kterém hledáme, druhý parametr je dotazovaná geometrie, třetí parametr je ve formě řetězce a obsahuje parametr `SDO_NUM_RES = 1`, který znamená, že nás zajímá pouze první řádek výsledku neboli nejbližší geometrie, a poslední parametr slouží k funkci `SDO_NN_DISTANCE(1)`, která tento parametr používá pro zjištění distance mezi každou vyhledanou geometrií a dotazovanou geometrií. Pokud operátor nalezne potřebné informace, vrací hodnotu *`true`*. Pokud ji nenalezne, tak vrací *`false`*.

Druhá verze zjištění bodu na síti je zaměřena na abstraktní železniční síť. Při vyhledávání se nemůžeme odkazovat na body železniční sítě, ale musíme se zaměřit na umístění hledaného bodu na čáru.

```
SELECT P.POINT.SDO_POINT.X, P.POINT.SDO_POINT.Y
FROM (SELECT POINT_ON_LINE(L.GEOMETRY, MDSYS.SDO_GEOMETRY(2001,8307,
             MDSYS.SDO_POINT_TYPE(X, Y,NULL),NULL, NULL)) AS POINT
      FROM RAILWAY_ABS_LINK$ L
      WHERE SDO_NN(L.GEOMETRY,
                   MDSYS.SDO_GEOMETRY(2001,8307,
                   MDSYS.SDO_POINT_TYPE(X, Y,NULL),NULL, NULL),
                   'SDO_NUM_RES = 1', 1) = 'TRUE'
      ORDER BY SDO_NN_DISTANCE(1)) P;
```

Tato verze využívá jako první verze také operátor SDO_NN s tím, že se nezabývá hledáním nejbližší geometrie v tabulce uzlů ale v tabulce vazeb. Po získání nejbližší vazby je zavolána vlastní funkci POINT_ON_LINE s parametry: nalezená vazba, dotazovaná geometrie reprezentovaná bodem se souřadnicemi X a Y. Tato funkce využívá Spatial funkci SDO_GEOM.SDO_CLOSEST_POINTS, která hledá nejbližší bod k dotazované geometrii. Při hledání na čáře se nám vrátí bod, který je nejblíže k hledané geometrii a přitom leží na čáře. Funkce využívající tuto funkci:

```
FUNCTION POINT_ON_LINE(
   GEOMETRY1  IN SDO_GEOMETRY,
   GEOMETRY2  IN SDO_GEOMETRY
) RETURN SDO_GEOMETRY
AS
   PT1    SDO_GEOMETRY;
   PT2    SDO_GEOMETRY;
   DIST NUMBER;
BEGIN
   SDO_GEOM.SDO_CLOSEST_POINTS(
      GEOMETRY1,
      GEOMETRY2,
      0.05,
      NULL,
      DIST,
      PT1,
      PT2
   );
   RETURN PT1;
END;
```

Parametry funkce SDO_GEOM.SDO_CLOSEST_POINTS odpovídají: GEOMETRY1 je geometrie vazby, GEOMETRY2 je dotazovaná geometrie, tolerance je 0.05, volitelné parametry ve formě textového řetězce nejsou zadány, DIST vrací nejkratší vzdálenost mezi těmito geometriemi a poslední dva parametry jsou výstupní parametry. PT1 vrací nejbližší bod na vazbě k dotazované geometrii a PT2 je nejbližší bod v dotazované geometrii k vazbě. Parametr PT2 je tedy pro nás zbytečný, tak vracíme pouze PT1.

6.5.3 Vytvoření nejkratší cesty

Pro vytvoření nejkratší cesty použijeme paměťový objekt. Pro nalezení cesty máme dvě funkce, které nalezneme v balíčku SDO_NET_MEM.NETWORK_MANAGER. Jsou to funkce SHORTEST_PATH a SHORTEST_PATH_DIJKSTRA. První funkce hledá nejkratší cestu pomocí *A* vyhledávání* algoritmu a druhá funkce za pomocí *Dijkstra* algoritmu. Obě dvě metody využívají pro ohodnocení hran a uzlů sloupce nákladů (v našem příkladu sloupec COST v tabulce RAILWAY_LINK$). Do funkce jde dále přiřadit další omezení.

Následující vytvořená funkce slouží k nalezení nejkratší cesty mezi stanicemi. Parametry funkce jsou ID začáteční a koncové stanice.

```
FUNCTION SHORTEST_PATH(FROM_ID NUMBER, TO_ID NUMBER)
RETURN NUMBER
AS
  PATH_ID NUMBER;
  RES_GEOM SDO_GEOMETRY;
  NET_MEM VARCHAR2(25);
BEGIN
  NET_MEM := 'RAILWAY';
  PATH_ID := SDO_NET_MEM.NETWORK_MANAGER.SHORTEST_PATH(NET_MEM, FROM_ID,
TO_ID, null);
  SDO_NET_MEM.NETWORK.ADD_PATH(NET_MEM, PATH_ID);
  SDO_NET_MEM.PATH.SET_NAME('RAILWAY', PATH_ID,
    'Path from ' || FROM_ID || ' to ' || TO_ID);
  DBMS_OUTPUT.PUT_LINE(PATH_ID);
  -- COMPUTE_GEOMETRY
  SDO_NET_MEM.PATH.COMPUTE_GEOMETRY(NET_MEM, PATH_ID, 0.0000005);
  -- GET_GEOMETRY
  RES_GEOM := SDO_NET_MEM.PATH.GET_GEOMETRY(NET_MEM, PATH_ID);
  RETURN PATH_ID;
END;
```

Naše funkce přijímá parametry ID počátečního a ID koncového bodu cesty a vrací ID vytvořené cesty. V těle funkce je vytvoření nejkratší cesty pomocí funkce `SHORTEST_PATH`, přidání cesty do sítě, pojmenování sítě a výpočet geometrie sítě. U vypočítávání geometrie pomocí funkce `COMPUTE_GEOMETRY` nastává problém. Pokud je cesta moc dlouhá, tak tato funkce nedokáže vygenerovat geometrii této cesty a vrací pouze *`null`*.

S takto vygenerovanou cestou v paměťovém objektu můžeme pracovat pomocí identifikačního čísla cesty. Jako příklad můžeme použít funkci, která zjistí všechny ID uzlů, které náleží dané cestě:

```
FUNCTION GET_PATH_NODE_IDS(PATH_ID NUMBER)
RETURN SDO_NUMBER_ARRAY
AS
  RES_ARRAY SDO_NUMBER_ARRAY;
BEGIN
  RES_ARRAY := SDO_NET_MEM.PATH.GET_NODE_IDS('RAILWAY', PATH_ID);
  RETURN RES_ARRAY;
END;
```

7 Mapa

Po předchozí části, kdy jsme si vytvořili železniční síť Railway, je potřeba vizualizace této sítě jako mapy. Vizualizaci prostorových dat nám poskytuje komponenta MapViewer. Při vizualizaci mapy potřebujeme vytvořit datový zdroj, mapu za použití nástroje Map Builder a následně vytvořit pomocí webové stránky a JavaScriptu jednotlivé funkcionality mapy.

7.1 Webové rozhraní MapVieweru

Webové rozhraní MapVieweru slouží jako centrum MapVieweru[10]. Stránka se skládá z pěti záložek, z toho jedna se zobrazí pouze při administrátorském přihlášení (viz Obrázek 21).

První záložka **Home** obsahuje odkazy na návody a na Oracle Map API. Druhá záložka **Requests** slouží k provádění dotazů na MapVieweru testování témat, map atd. Třetí záložka **Demos** obsahuje ukázky map a některé zdrojové kódy k nim. Čtvrtá záložka slouží jako náhled na metadata MapVieweru, kde jsou uloženy existující datové zdroje, předdefinovaná témata atd. Pátá záložka **APIs** odkazuje na příručku Java API pracující s MapViewerem. Tyto API se využívají, pokud tvoříme mapy za pomocí aplikace v Javě. Poslední záložka **Management** se zobrazuje pouze, pokud jsme přihlášení jako administrátor.

Obrázek 21: Webové rozhraní MapVieweru. Zdroj: [autor]

[10]Webové rozhraní MapVieweru naleznete na adrese http://localhost:8888/mapviewer/faces/home.jspx, kde port máte nastavený podle nainstalování MapVieweru.

7.1.1 Záložka Management

Tato záložka slouží k práci s konfigurací MapVieweru, datovými zdroji, mezipamětí načtených témat a spravuje deskové vrstvy.

Konfigurace MapVieweru je uložena v souboru `mapViewerConfig.xml`. Pracovat s tímto souborem můžeme pomocí webového rozhraní nebo pomocí textových procesorů. Najdeme zde spoustu zajímavých nastavení jako způsob vykreslování obrázku, globální nastavení map atd. Zde je pro nás ale nejdůležitější část týkající se předdefinovaných datových zdrojů. Datový zdroj nám umožňuje pracovat s Oracle Databází pomocí uživatelského jména a hesla. Pro práci se síťovým datovým modelem `RAILWAY` použijeme následující datový zdroj.

```
<map_data_source name="Railway"
                   jdbc_host="localhost"
                   jdbc_sid="orcl"
                   jdbc_port="1521"
                   jdbc_user="SpatialUser"
                   jdbc_password="v5ZXLu175mvqrLVB4P+LCWyZI+fbO+zS"
                   jdbc_mode="thin"
                   number_of_mappers="5"
                   allow_jdbc_theme_based_foi="true"
   />
```

Databáze *orcl*, kterou používáme jako datový zdroj, je uložena na místním počítači na portu 1521. K databázi se připojujeme jako uživatel *SpatialUser* a jeho hesla (pokud zadáme při prvním uložení našeho hesla, prefix „!“, tak se nám do souboru heslo zakóduje) v *thin* módu. Počet povolených mapování najednou je stanoveno na 5 a JDBC téma založená na FOI jsou povolena.

Záložka Management nám umožňuje vytvářet dynamický datový zdroj, které se drží v paměti do zavření MapVieweru. S datovými zdroji jsou přímo spojená možnost geometrické mezipaměti, kde můžeme uvolňovat témata zdrojů z mezipaměti, pokud je nepotřebujeme.

Poslední možnost záložky je vytváření a správa deskových vrstev, které budou popsány v jedné z dalších kapitol.

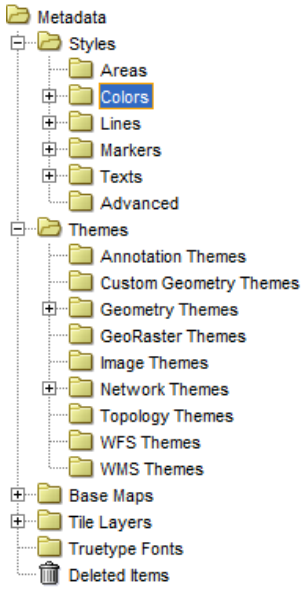
7.2 Vytvoření mapy

V této části se zaměříme na vytvoření mapy. Pro vytvoření mapy můžeme použít vytváření XML definic do příslušných pohledů (`USER_SDO_STYLES`, `USER_SDO_THEMES` atd.) nebo k tomu použít grafické prostřední nástroje Map Builder, které XML definice vygeneruje za nás. Pro vytvoření mapy použijeme Map, protože nám poskytuje náhled na vytvářené objekty a umožňuje jejich jednodušší úpravu. Vytváření mapy si můžeme rozložit do čtyř základních kroků, které jsou:

- vytvoření stylů,
- vytvoření témat,

- vytvoření mapy,
- vytvoření deskových vrstev.

Při použití nástroje Map Builder si musíme vytvořit připojení k databázi, které následně budeme používat pro generování našich stylů, témat, map a deskových vrstev. Jednotlivé generování probíhá přes položky nástroje Map Builder, která je zachycena na Obrázku 22.



Obrázek 22: Náhled na možnosti nástroje Map Builder. Zdroj: [autor]

7.2.1 Vytvoření stylů

Pro správné zobrazování map je třeba mít specifikované styly, které slouží k zobrazování jednotlivých částí mapy. Tato část obsahuje styly pro barvy, plochy, čáry, značky, texty a rozšířené, které se používají pro styly obsahující nějakou podmínku (např. barva podle hustoty zalidnění v krajích).

Naše vytvářené styly jsou uloženy v pohledu `USER_SDO_STYLES`, která se skládá z názvu, typu, popisu, XML definice, obrázku a geometrie. Název stylu by mělo označovat daný styl s tím, že by měl obsahovat prefix označující, jakému typu stylu náleží (C – barva, L – čára, M – značka, T – text, A – rozšířený). Typ je typ stylu. Popis slouží k popisu daného stylu. XML definice je určena k definování stylu, jak daný tvar vypadá. Sloupec obrázek používáme tehdy, pokud načítáme vlastní obrázek pro definici, jinak je nastavený na *null*. Jako poslední je sloupec geometry, který je využíván jako sloupec obrázek s tím, že místo obrázku zde máme objekt typu `SDO_GEOMETRY`.

Pro naši železniční síť potřebujeme styly pro znázornění bodů hektometrovníků, úseků mezi nimi, železničních stanic, kolejových vozidel a jejich textových popisů, které se budou znázorňovat při přiblížení mapy.

Vytvoření nových stylů provádíme pomocí nástroje Map Builder v sekci STYLES, kde si zvolíme příslušný podadresář pro každý styl (viz Obrázek 22) a pravým kliknutím myši vybereme vytvoření nového stylu. Zobrazí se nám průvodce pro vytvoření stylu.

7.2.2 Vytvoření témat

Další krok při vytváření mapy je vytvoření témat. Téma je zobrazení mapy s určitými styly. Například v našem případě využíváme téma pro zobrazení mapy s viditelnými body hektometrovníků a jejich popisy a pak další téma, kde jsou viditelné pouze vazby.

Námi vytvořená témata jsou uložena v databázi v pohledu USER_SDO_THEMES. Pohled se skládá z názvu, popisu, tabulky základu, sloupce geometrie a stylových pravidel. Název je označení daného tématu v pohledu. Popis slouží k popisu daného tématu. Tabulka základu určuje jaká tabulka se sloupcem SDO_GEOMETRY je základem pro téma. Sloupec geometrie je název sloupce typu SDO_GEOMETRY a tyto údaje chceme vykreslit. Stylové pravidla je UML definice tématu.

Témata můžeme zakládat více typů. Jako možnosti se nám nabízejí geometrická, síťová, topologická témata ale i další. Geometrické téma je založeno na prostorových datech, kdy je zapotřebí, aby tabulka obsahovala sloupec SDO_GEOMETRY. Síťové téma se využívá pro síťový datový model a topologické téma se využívá pro topologický síťový model.

Pro vytvoření tématu stačí opět k příslušnému podadresáři adresáře THEMES (viz Obrázek 22) kliknout pravým tlačítkem a postupovat podle zobrazeného průvodce. Průvodce se skládá ze čtyř kroků. V prvním kroku nastavíme tématu název, popis a tabulku základ s jeho sloupcem obsahujícím údaje typu SDO_GEOMETRY. V dalších krocích nastavíme styly zobrazování pro geometrii a popis. V poslední části je nám umožněno přidat podmínku pro filtrování dat v tabulce k zobrazení. Pokud vytváříme síťové téma, musíme zvolit také styly pro zobrazení uzlů, vazeb a cest.

Pro naši potřebu byla vytvořena síťová témata pro zobrazení železniční sítě a geometrické téma pro zobrazení železničních stanic. Témata železniční sítě jsou ještě rozděleny podle zobrazení pro jednotlivé přiblížení mapy.

Na následujícím příkladě můžete vidět téma STATION_03.

```
<?xml version="1.0" standalone="yes"?>
<styling_rules>
  <rule>
    <features style="M.SMALL_CIRCLE_RED"> (V_SEGMENTU = 'A') </features>
    <label column="STATION" style="T.HIGH_BLACK_WHITE"> 1 </label>
  </rule>
</styling_rules>
```

Téma má nastavený styl M.SMALL_CIRCLE_RED (značka malý červený kruh) pro zobrazování geometrie, styl T.HIGH_BLACK_WHITE (text velké černé písmo

s bílým okrajem) pro zobrazení popisu geometrie. Pro popis je určen sloupec `STATION` (obsahuje název). Dále si můžeme všimnout podmínky (`V_SEGMENTU = 'A'`). Tato podmínka říká, že budou použity pouze ty řádky, které mají ve sloupci `V_SEGMENTU` nastavený `'A'`. Náhled na dané téma je na Obrázek 23.

Obrázek 23: Náhled na téma `STATION_03`. Zdroj: [autor]

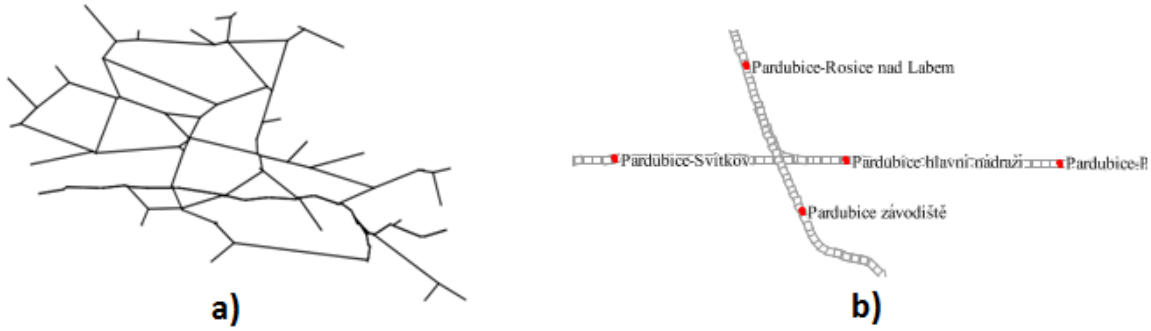
7.2.3 Vytvoření mapy

Nyní se dostáváme přímo k vytvoření mapy. K tomuto kroku potřebujeme mít vytvořený předchozí témata, z kterých mapu sestavíme. U jednotlivých témat nastavujeme od jakého a do jakého měřítka se dané téma bude na mapě zobrazovat.

Vytvořená mapa se ukládá do pohledu `USER_SDO_MAPS` a skládá se ze sloupců název, popis a definice. Název je označení mapy. Popis slouží k popisu dané mapy a definice je XML zápis mapy. Definici můžeme vidět na následujícím příkladu, kdy máme vytvořenou mapovou definice železniční sítě.

```
<?xml version="1.0" standalone="yes"?>
  <map_definition>
    <theme name="RAILWAY_ABS02" min_scale="1300000.0"
max_scale="910000.0" scale_mode="RATIO"/>
    <theme name="RAILWAY02" min_scale="910000.0" max_scale="160000.0"
scale_mode="RATIO"/>
    <theme name="RAILWAY03" min_scale="160000.0" max_scale="0.0"
scale_mode="RATIO"/>
    <theme name="STATION_01" min_scale="910000.0" max_scale="650000.0"
scale_mode="RATIO"/>
    <theme name="STATION_02" min_scale="650000.0" max_scale="0.0"
scale_mode="RATIO"/>
  </map_definition>
```

Definice se skládá z pěti témat: `RAILWAY_ABS02`, `RAILWAY02`, `RAILWAY03`, `STATION_01` a `STATION_02`. Největší oddálení je stanoveno na 1 300 000[11], kdy je na mapě zobrazeno pouze téma `RAILWAY_ABS02`, která obsahuje zobrazení abstraktní železniční sítě. Při přiblížení na 910 000 se téma `RAILWAY_ABS02` vypne a zobrazí se detailnější zobrazení železniční sítě téma `RAILWAY02` a k tomu ještě se zobrazí železniční stanice z tématu `STATION_01`. Když se přiblíží na 650 000, tak se předchozí dvě témata vypnou a zobrazí se témata `RAILWAY03` a `STATION_02`. Náhled na mapu je znázorněný na Obrázek 24, kde a) odpovídá přiblížení 1 200 000 a b) odpovídá přiblížení 120 000.



Obrázek 24: Náhledy na mapu `RAILWAY`. Zdroj: [autor]

Mapa může obsahovat kombinace dalších prvků a atributů, kromě témat. Mezi tyto prvky a atributy patří záhlaví mapy, legenda, obrázek na pozadí, copyright poznámka, dočasné dynamické styly, předdefinovaná témata (viz předchozí příklad) a JDBC témata.

JDBC témata jsou témata, které jsou volány dynamickými dotazy. Nevýhoda dynamických témat je značně omezená podpora. Bohužel MapViewer v současné verzi nedokáže vykreslovat JDBC síťová a topologická témata, která jdou jako předdefinovaná.

7.2.4 Vytvoření deskových vrstev

Když už máme vytvořenou základní mapu, potřebujeme vytvořit jednotlivé zoom úrovně, které jsou zde reprezentovány jako deskové vrstvy. V této části si pro zvolenou mapu vytvoříme hranici všech vrstev a následně vytvoříme zoom úrovně.

Při vytváření hranice vrstev můžeme použít vypočítání z mapy, kdy se nám vezme nejmenší možný MBR, do kterého se mapa vejde. Pokud chceme nastavovat ručně hranice, tak musíme zadat SRID a jednotlivé souřadnice pro minimální a maximální bod obdélníku, který bude tvořit hranici.

Následuje vytvoření vrstev pro jednotlivý zoom úrovně. Při vytváření těchto vrstev musíme nejdříve zadat největší a nejmenší měřítko zobrazení mapy. Tyto údaje lze zase vygenerovat z mapy. Zvolíme počet zoom úrovní a následně pro každou úroveň nastavíme měřítko. Tyto úrovně jdou opět vygenerovat nástrojem Map Builder.

[11] Měřítka použitá v příkladech odpovídají metrům k jednomu centimetru. Příklad 1 cm odpovídá 1 300 000 metrů ve skutečnosti.

Pro další zprávu deskových vrstev můžeme použít webové rozhraní MapVieweru pod administrátorským nastavením.

7.3 Funkcionalita mapy

V předchozí části jsme si vytvořili mapu železniční sítě Railway, která kromě zoomu a posouvání nám nic jiného neumožňuje. Tato kapitola je zaměřena na vytvoření funkčnosti mapy, jako je vytváření vlaků, zachycení pohybu vlaků a zobrazení ujetých tras. Funkcionality mapy se implementovány do webové stránky pomocí JavaScriptu, kdy voláme metody Oracle Map API. Metody pracují s MapViewerem pomocí AJAXu.

Naše webová stránka zobrazující mapu je rozdělena na tři soubory pro lepší přehlednost. První soubor tvoří kostru webové stránky. Tento soubor obsahuje odkaz na JavaScriptový soubor `oraclemaps.js`, v které jsou uloženy potřebné třídy a metody pro práci s MapViewerem. Další dva odkazy, které soubor `mapa.html` obsahuje, jsou zbylé dva JavaScriptové soubory (`MapControl.js` a `TrainControl.js`), které obsahují funkcionalitu naší mapy. Poslední důležité části souboru jsou blok z id „map“, do kterého se načítá mapa a nastavení volání JavaScriptové funkce `showMap` při načtení.

```
<body onload=javascript:showMap();>
  <div id="map" style="left:0px; top:0px; width:100%; height:100%;
background-color: #ffffff;"></div>
```

7.3.1 Soubor `MapControl`

Hlavním úkolem `MapControlu` je načtení mapy do bloku. Ostatní funkce jsou pomocné a převážně slouží k zobrazení jednotlivých nástrojů a FOI objektů. Struktura souboru je zachycena na Obrázek 25.

Obrázek 25: Struktura souboru `MapControl.js`. Zdroj: [autor]

`MapControl` obsahuje proměnnou `mapviewer`, která je inicializována při volání funkce *showMap*. Tato komponenta odráží naše nastavení MapVieweru mapy, FOI a další komponenty, které zobrazujeme na mapě (měřítko, náhled, …). Pro inicializaci proměnné `mapviewer` je zapotřebí blok s identifikátorem „map“ a adresou našeho MapVieweru.

```
var baseURL  = "http://localhost:8888/mapviewer";
mapview = new MVMapView(document.getElementById("map"), baseURL);
```

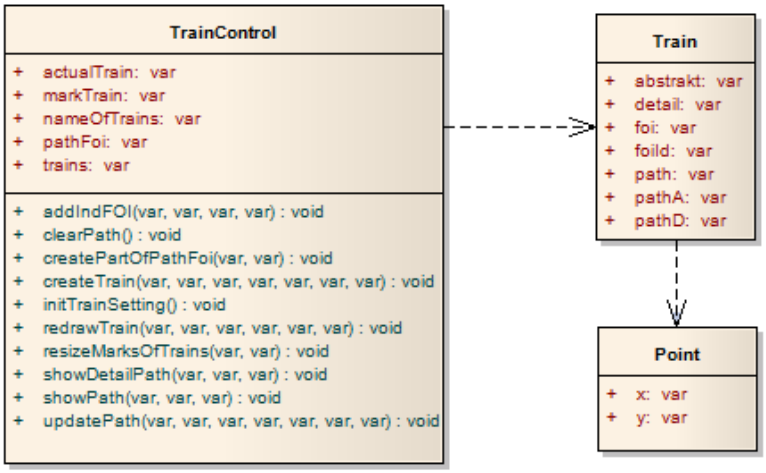
Druhá proměnná `toolbar` slouží k zobrazení nástrojů pro měření vzdálenosti a nástroje přiblížení pomocí obdélníku. K proměnné je možno přistupovat pomocí proměnné `mapviewer`, ale pro aplikaci byla vytvořena jako samostatná proměnná, aby se panel nástrojů dal také skrývat.

Soubor dále obsahuje metody, které slouží pro inicializaci jednotlivých komponent mapy, které jsou volány inicializační metodou `showMap`. Metoda `showMap` inicializuje proměnnou `mapviewer`, nastavuje mapu, střed mapy a volá metody pro načtení komponent měřítka (`showScaleBar`), náhledu (`showOverview`), panelu nástrojů (`showMarqueeZoomtool`) a zobrazení železničních stanic (`showStation`).

Dalším úkolem `MapControlu` je zobrazení nových vlastností mapy uživatelem. Uživatel využívá metody, které jsou volány pro vytvoření nového vlaku (`showTrain`), zobrazení cesty vlaku (`showPathAndTrain`), zachycení změny zoom úrovně (`visibilityFOIStanice`) a provedení změny zoom úrovně (`zoomSetting`).

7.3.2 Soubor `TrainControl`

`TrainControl` je používán pro práci s vlaky, které jsou zobrazovány na mapě. `TrainControl` kromě vlastních funkcionalit obsahuje třídy `Train`, která reprezentuje vlak, a třídu `Point`, která reprezentuje bod souřadnicového systému zeměpisné délky a šířky (viz Obrázek 26).



Obrázek 26: Struktura souboru `TrainControl.js`. Zdroj: [autor]

Třída `Point` je využívána vlakem pro nastavení polohy jednotlivých vlaků.

Třída `Train` je vlaková reprezentace, která slouží k zobrazování vlaku na mapě. Železniční síť je rozdělena na dvě úrovně: abstraktní a detailní, proto potřebujeme pro každou úroveň jiné souřadnice vlaku. Tyto souřadnice jsou uloženy v atributech `abstrakt` a `detail`. Každý vlak má název v atributu `foiId` a způsob zobrazení a nastavení FOI v atributu `foi`. Poslední tři atributy jsou využívány k uchování ujeté trasy vlaku rozdělený na část pro vykreslení na abstraktní úrovni, část pro vykreslení na detailní úrovni a trasa, která ukládá kromě polohy vlaku čas a TUDU.

`TrainControl` obsahuje proměnné `trains`, `nameOfTrains`, `markTrains`, `pathFoi` a `actualTrain`. Proměnná `trains` a `nameOfTrain` jsou pole, kam se ukládají jednotlivé vlaky. Pole `trains` obsahuje vlaky jako instance třídy `Train` a v poli `nameOfTrains` jsou uloženy všechny názvy vlaků. `MarkTrain` obsahuje nastavení velikosti značky vlaků, která se mění při změně zoom úrovně mapy. Proměnné `actualTrain` a `pathFoi` využíváme pro zachycení aktuálního vlaku a jeho cesty.

Při inicializaci stránky je volána metoda `iniTrainSetting`, která inicializuje proměnná `markTrain` určující velikost značek vlaků. Změna velikosti značek vlaku na mapě probíhá při každé změně zoom úrovně, kdy je volána metoda `resizeMarksOfTrains` s atributy velikosti značky a zoom úrovně. Metoda postupně prochází pole `trains` a pro každý vlak překreslí jeho značku na mapě pomocí metody `redrawTrain` a vymaže zobrazenou cestu metodou `clearPath`.

Vytváření nového vlaku se provádí pomocí metody `createTrain`, která vytvoří FOI vlaku, vloží nový vlak do pole `trains` a název do pole `nameOfTrains`. Pro vytvoření FOI se používá metoda `addIndFOI`, v této metodě se vytvoří značka zobrazení, nastaví se metoda volaná na událost kliknutí tlačítka na objekt a nakonec se FOI vloží do proměnné `mapviewer`.

Poslední funkce slouží k práci s trasou. Funkce `showPath` zobrazuje trasu aktuálního vlaku. Pro vykreslení trasy používá Oracle Map API funkci `MVSdoGeometry.createLineString`, která vytvoří z pole souřadnic (x1, y1, x2, y2, …) objekt typu `SDO_GEOMETRY`, který pak je vložen do nového FOI jako tvar. Tato funkce je bohužel omezená velikostí poslaného pole přes AJAX komponentě MapViewer. Proto v aplikaci vytváříme jednotlivé úseky trasy zvlášť pomocí funkce `createPartOfPathFoi`. Metoda `showDetailPath` se volá po kliknutí na část trasy a zobrazí detailní tabulku s projetými místy. Poslední metodou `updatePath` zasílám vlaku změnu lokace vlaku.

8 Návrh a implementace aplikace

8.1 Úvod

Tato kapitola obsahuje návrh a implementaci Railway – test aplikace. Aplikace je postavena nad železniční sítí Railway, která je vytvořená v Oracle Database 11g Enterprise, a využívá předpřipravené funkce a procedury pracující nad touto databází.

Railway – test aplikace je vybudován na platformě Microsoft .NET Framework 4.0, která byla v době psaní této práce aktuální. Aplikace využívá webovou stránku pro práci s MapViewerem a databázi Oracle. Programovací jazyky použité při budování aplikace jsou C##, JavaScript a PHP.

Aplikace využívá databázi z minulé kapitoly. Pro propojení mezi aplikací a Oracle databází se využívá knihovna ODP.NET a je dostupná ke stažení na domovských stránkách společnosti Oracle.

8.2 Požadavky na aplikaci

Aplikace Railway pracuje s železniční sítí Railway a měla by být schopná umožňovat následující funkcionality:

- vizualizace železniční sítě a kolejových vozidel na mapě,
- manipulace s mapou (posouvání, zoom, …),
- import a export vlaků,
- simulování pohybu vlaků na železniční síti,
- zobrazení ujetých tras vlaků,
- generování nejkratších tras mezi železničními stanicemi.

Požadavky, které jsou uvedeny výše, byly vzaty ze zadání diplomové práce a některé byly vytvořeny při konzultacích s vedoucím práce.

8.3 Využité návrhové vzory

Aplikace využívá ke své implementaci některé z návrhových vzorů a softwarovou architekturu Model-view-controller (MVC). Návrhové vzory jsou standardizované postupy, jak provádět určité problémy efektivně.

Program využívá pro připojení k databázi návrhový vzor Singleton. Tento vzor zajišťuje, aby při práci s databází bylo používáno pouze jedno připojení.

8.3.1 Architekturu Model-view-controller

Architektura Model-view-controller je architektura rozdělující aplikaci do třech vrstev datový model (model), uživatelské rozhraní (view) a řídící logiku (controller). MVC architektura je výhodná, když potřebujeme modifikovat některé komponenty. Změny mají minimální vliv na ostatní. MVC může být chápána jako návrhový vzor.

Model

Vrstva Model je brána jako nejnižší vrstva architektury, která slouží k uchovávání dat. Datový model umožňuje vybírat a manipulovat s těmito daty, tak aby byla zachována jejich konzistence.

View

Vrstva View (pohled) je využívána k reprezentaci datového modelu uživateli ve vhodné podobě. Vrstva obsahuje uživatelské rozhraní aplikace, se kterým uživatel pracuje. View vrstva by neměla obsahovat části pro manipulaci s daty.

Controller

Vrstva Controller (dispečer) reaguje na události, které nastaly při interakci uživatele s uživatelským rozhraním nebo jiným systémem. Dispečer zajišťuje zachycení změn spojených s takovýmito událostmi a rozhoduje, jak tyto změny budou reprezentovány v pohledech.

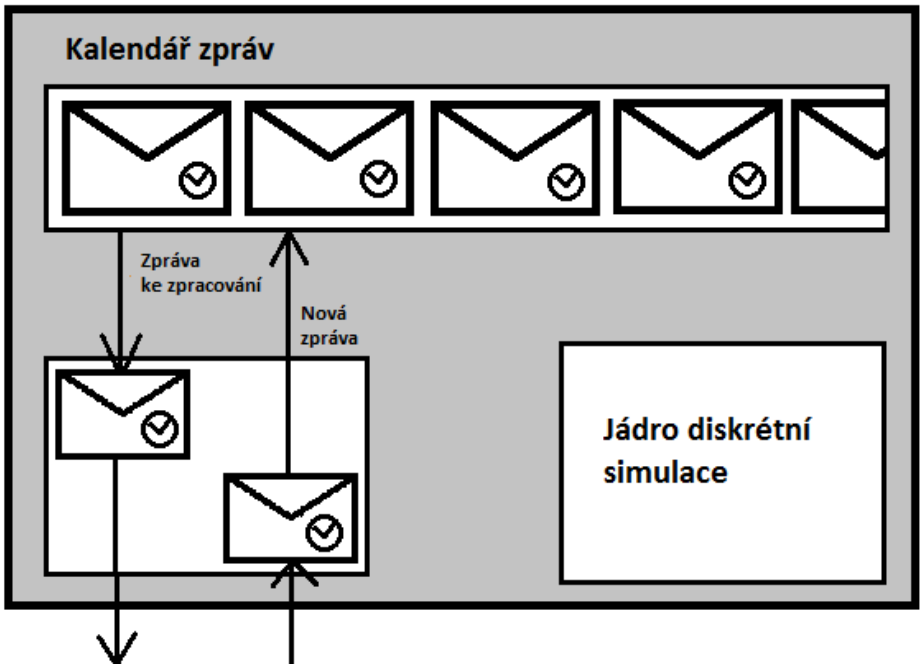
8.4 Simulace pohybu železničních vozidel

Jeden z požadavků na testovací aplikaci Railway bylo simulování pohybu vlaků na železniční síti. Simulace se dají rozdělit na (i)spojité, (ii)diskrétní nebo (iii)kombinované. Pro spojitou simulaci bychom mohli použít zrychlený čas s určitou periodou, v které by se vybírali zprávy s dřívějším časem než je perioda a aktualizovala tak jednotlivé vlaky.

Diskrétní simulace je pro naše potřeby výhodnější. Hlavním důvodem je, že jednotlivé události nastávají v nespojitém čase, resp. ve skocích. Další výhodou, kterou diskrétní simulace poskytuje, je lepší práce s rychlostí simulace, kdy rychlost si stanovíme pomocí časových prodlev mezi jednotlivými zpracováváními zpráv.

Diskrétní simulace znázorněná na Obrázku 27 se skládá z kalendáře zpráv, které čekají na zpracování, a jádra simulace, která řídí chod simulace. Jednotlivé kroky procesu diskrétní simulace jsou:

1. inicializace simulačního času,
2. ukončení simulace pokud je kalendář zpráv prázdný nebo byl vyčerpán čas pro běh simulace,
3. výběr zprávy s nejmenším časovým razítkem z kalendáře,
4. aktualizace simulačního času,
5. zpracování zprávy,
6. vrácení se na krok 1.



Obrázek 27: Modul diskrétní simulace. Zdroj: [autor]

Simulace se skládá ze dvou tříd modulu simulace a zprávy procesu, který se v simulaci provádí.

8.4.1 Modul simulace

Modul simulace obsahuje hlavní metodu provádějící simulaci a kalendář zpráv procesů, které se v simulaci provádějí podle jejich časového razítka. Další metody, které simulace musí obsahovat, jsou metody na příjem nových zpráv s procesy, metoda pozastavení simulace a změna rychlosti provádění simulace.

Simulace se provádí do té doby, dokud nedojdou zprávy v kalendáři zpráv nebo nevyprší čas simulace (aplikace je ukončena). V každém cyklu simulace je vybrána zpráva s nejmenším časovým razítkem z kalendáře zpráv, aktualizace času simulace, provedení procesu zprávy a nakonec časová prodleva mezi dalším cyklem.

Metoda simulace je prováděna v samostatném vlákně, abychom mohli mezitím pracovat s aplikací, posílat nové zprávy do simulace, měnit rychlost simulace nebo abychom měli možnost pozastavit simulaci.

8.4.2 Proces simulace

Procesy simulace mohou být rozdílné podle jejich účelu, ke kterému slouží. Každý proces simulace obsahuje časové razítko, které obsahuje čas, kdy se má daný proces provést, a metodu, která daný proces provádí.

Pro naši simulaci existují pouze procesy, které aktualizují polohu železničních vozidel, resp. aktualizují aktuální pozici vlaku, přidávají souřadnice do projeté trasy a aktualizují umístění vlaku na mapě.

Procesy musí implementovat rozhraní IComparable, aby mohli být řazeny v kalendáři zpráv v modulu simulace. Porovnávání procesů mezi sebou probíhá pomocí časových razítek. Čím menší má proces časové razítko, tím dříve daný proces musí nastat.

8.5 Popis hlavních tříd

V této části se zaměříme na strukturu aplikace, kde si postupně popíšeme jednotlivé třídy aplikace. Aplikace se skládá ze dvou formulářů a čtyř dispečer (viz. Obrázek 28). Zbytek tříd slouží jako modely aplikace nebo specificitou činnost.

8.5.1 Hlavní formulář a dispečery

Základním kamenem aplikace je hlavní formulář, který tvoří uživatelské rozhraní testovací aplikace. Pomocí hlavního formuláře můžeme pracovat s mapou a volat všechny požadované metody. Akce, které provádíme na formuláři, jsou posílány dispečerovi, který je zpracovává a vrací jejich výsledky. Znázornění diagramu tříd hlavního formuláře a dispečerů je zobrazeno na Obrázku 28.

Hlavní formulář *`FormMain`*

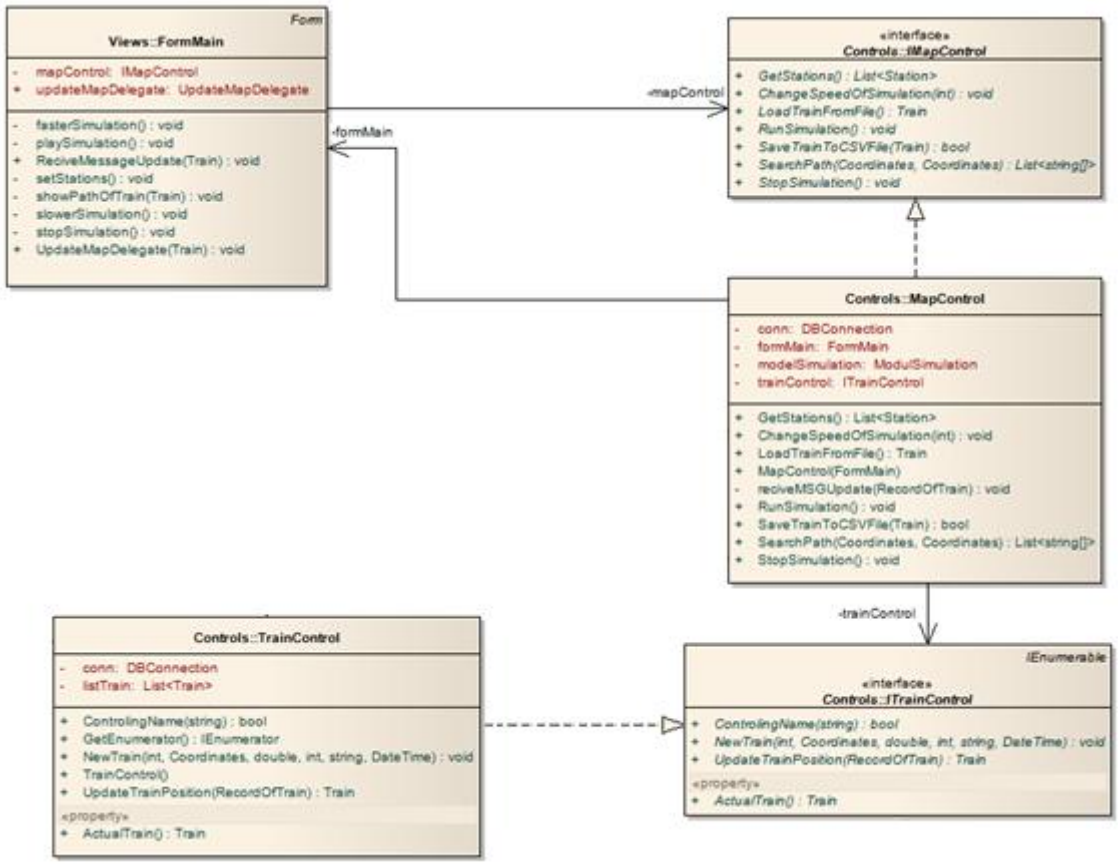
Hlavní formulář *`FormMain`*, který je inicializován při spuštění aplikace. Formulář slouží jako grafické rozhraní aplikace a skládá se z nástrojů, kterými se volají funkce dispečera implementující rozhraní `IMapControl`, a komponenty `WebBrowserMapa`. Komponenta `WebBrowserMapa` slouží k zobrazování mapy, kterou jsme si vytvořili v kapitole 0.

Dispečer *`MapControl`*

Tento dispečer zodpovídá za všechny události, které nastanou ve formulářovém oknu *`FormMain`*. Dispečer obsahuje kromě zpáteční vazby na formulář ještě odkaz na dispečera vlaků implementující rozhraní `ITrainControl` a modelu simulace. Dispečer *`MapControl`* zodpovídá za načtení železničních stanic při startu aplikace, import a export tras vlaků, zprostředkovává komunikaci se simulací a vlakovým dispečerem a také posílání požadavku na překreslení mapy.

Dispečer *`TrainControl`*

Třída se stará o správu vlaků, s kterými aplikace pracuje. Mezi hlavní úkoly dispečera patří vytváření nových vlaků a změny umístění vlaku.



Obrázek 28: Diagram tříd hlavního formuláře a dispečerů. Zdroj: [autor]

8.5.2 Entitní třídy

Entitní třídy jsou třídy, které slouží převážně k uchovávání dat a jejich prezentaci. Třídy neobsahují řízení jako třídy dispečera ani práci s externími zdroji, které mohou být prezentovány jako databázové systémy nebo soubory.

Třída `Coordinates`

Třída `Coordinates` slouží jako záznam souřadnic zeměpisné šířky a délky, které poskytuje pomocí svých vlastností `Latitude` (zeměpisná šířka) a `Longtitude` (zeměpisná délka).

Třída `RecordOfTrain`

Třída `RecordOfTrain` odpovídá záznamu vlaku, který přichází do aplikace z externích zdrojů (souborů). Jeden záznam odpovídá jednomu vlaku, jeho identifikačnímu číslu, souřadnici, času, rychlosti a azimutu. Třída `RecordOfTrain` se využívá převážně k importu a exportu vlaků a simulaci, kdy jsou záznamy posílány na zpracování. Pro potřeby simulace třída realizuje rozhraní `IComparable`, aby jednotlivé záznamy mohly být řazeny v kalendáři zpráv v simulaci pomocí časového razítka.

Třída `Position`

Třída `Position` slouží k uchovávání pozice vlaku. Když je třída `RecordOfTrain` zpracována, tak je vytvořena nová pozice vlaku z tohoto záznamu. Třída `Position` obsahuje souřadnice, které obsahoval daný záznam, a souřadnice, které byli přepočítány na znázornění na mapě (jak pro detailní úroveň tak pro abstraktní).

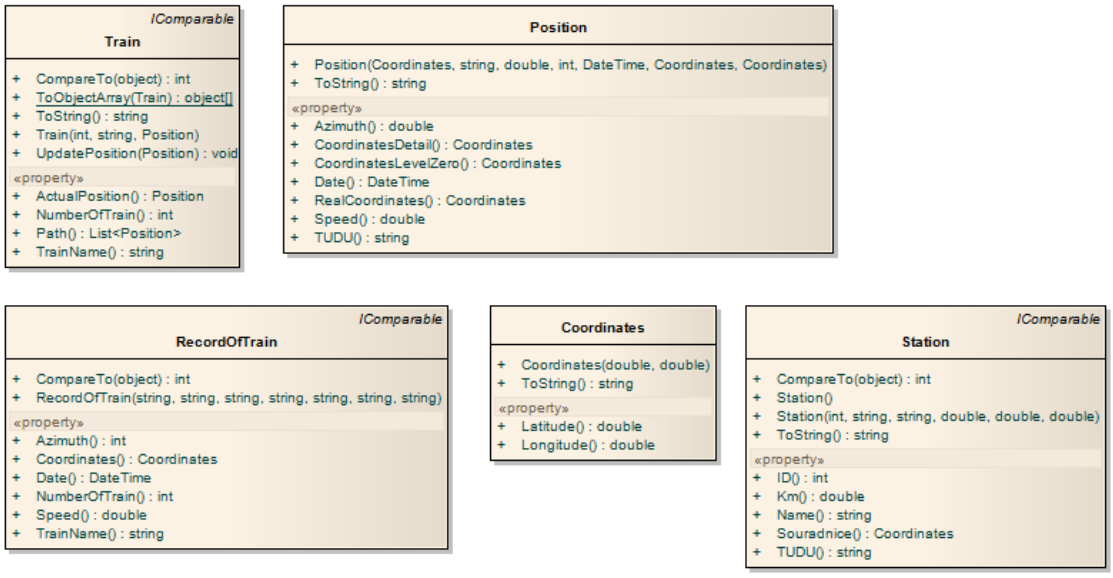
Třída `Train`

Třída `Train` reprezentuje vlak a jeho aktuální ujetou trasu. Třída obsahuje vlastnosti, které odráží aktuální pozici vlaku a jeho ujetou cestu v čase pomocí lineárního pole záznamů `RecordOfTrain`. Metody na změnu aktuální pozice vlaku a metodu na převedení vlaku na pole objektu, které se využívá na posílání vlaku do mapy. Třída realizuje rozhraní `IComperable`, aby mohli být vlaky porovnávány mezi sebou pomocí čísla vlaku.

Třída `Station`

Třída `Station` uchovává v sobě záznamy železničních stanic, které jsou načítány z databáze. Třída také realizuje rozhraní `IComparable`, aby jednotlivé stanice mohli být řazeny podle názvu.

Třídy entit jsou zobrazeny na Obrázku 29.



Obrázek 29: Diagram tříd entit. Zdroj: [autor]

8.5.3 Modelové třídy

Modelové třídy jsou takové třídy, které slouží ke komunikaci s externími zdroji. Třídy manipulují s daty v databázi a soubory *xls*, *xlsx* a *csv*.

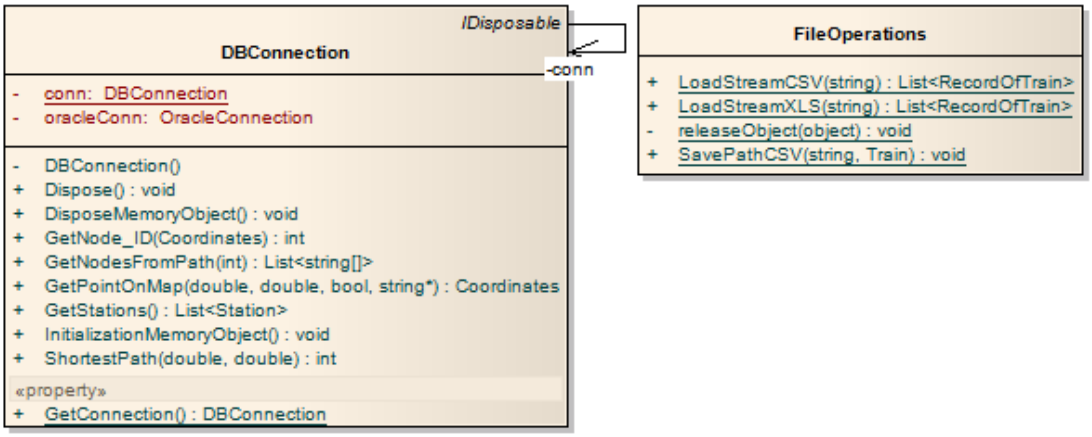
Třída `DBConnection`

Třída `DBConnection` pracuje s databází. Třída využívá návrhový vzor Singleton, aby v aplikaci nebylo použito více než jedno připojení k databázi, ale pracovalo se pouze s jedním připojením. Tato třída používá funkce, které jsme si vytvořili v kapitole 6.5.

Třída `FileOperations`

Třída `FileOperations` slouží k práci se soubory. Třída umožňuje importovat vlaky ze souborů *xls*, *xlsx* a *csv* do seznamů `RecordOfTrain` a exportovat vlaky s jejich cestou z aplikace do souborů *csv*.

Metody obou tříd jsou zachyceny na Obrázku 30.



Obrázek 30: Diagram tříd modelové části aplikace. Zdroj: [autor]

8.5.4 Třídy simulace

Třídy simulace reprezentují diskrétní simulaci pohybu železničních vozidel na železniční síti. Simulace se skládá z modulu simulace, která implementuje rozhraní `IModulSimulation`, a simulačních procesů, které implementují rozhraní `IProcess` (viz Obrázek 31).

Třída `UpdatePositionProcess`

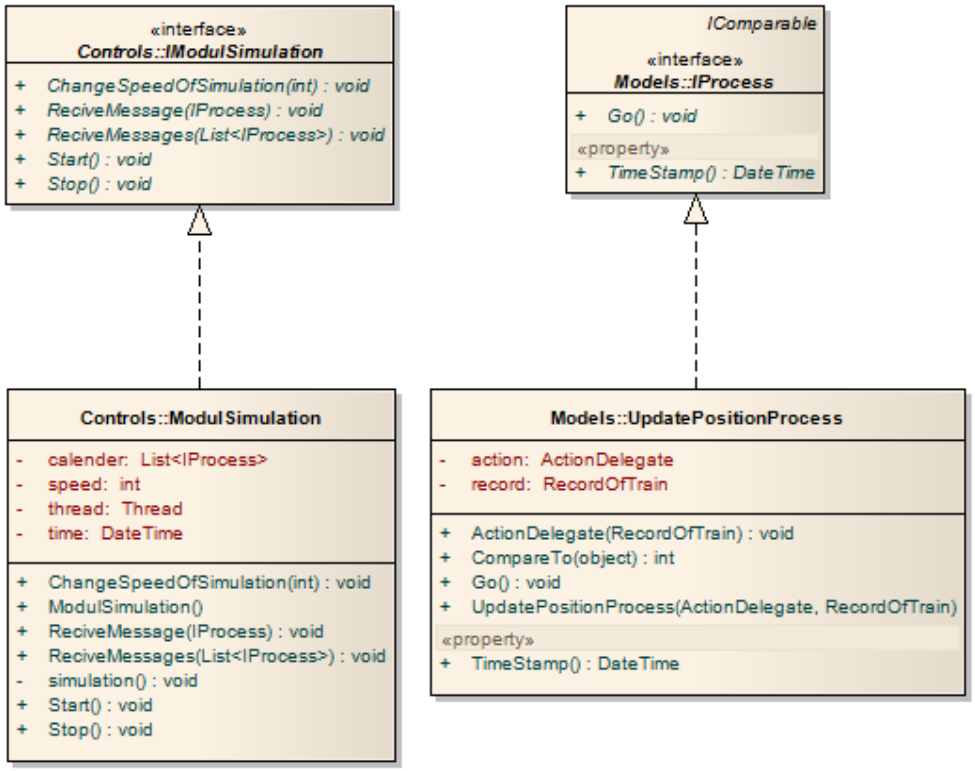
Třída `UpdatePositionProcess` reprezentuje událost změny polohy vlaku. Třída obsahuje delegát `ActionDelegate`, který pracuje s parametrem `RecordOfTrain`. Proměnná delegátu `ActionDelegate` obsahuje metodu `reciveMSGUpdate` z dispečera `mapControl`, která aktualizuje pozici vlaku a zavolá překreslení vlaku na mapě. Provádění procesů implementující rozhraní `IProcess` probíhá pomocí metody `Go ()`. Každý proces má svoje časové razítko, které značí, kdy má být daný proces proveden.

Třída `ModulSimulation`

Třída `ModulSimulation` slouží jako jádro diskrétní simulace. Skládá se z kalendáře zpráv, který obsahuje zprávy o změně pozic vlaků `UpdatePositionProcess` řazených podle časových razítek procesů.

Rychlost simulace odpovídá pozastavení simulace mezi jednotlivými procesy. Standardně je nastavený čas 3 vteřiny. Nejrychlejší simulace má mezi jednotlivými procesy 100 milisekund zastavení a nejpomalejší simulace má mezi jednotlivými procesy 10 vteřin zastavení.

Třída `ModulSimulation` přijímá zprávy z okolí buď po jedné zprávě, nebo jako seznam zpráv.



Obrázek 31: Diagram tříd simulace. Zdroj: [autor]

8.5.5 Ostatní důležité třídy

V této části jsou třídy, které tvoří nové komponenty pro uživatelské grafické rozhraní, a třída informačního dialogu o aplikaci `AboutBox`.

Třídy komponent jsou `TextBox2` a `TrackBarToolStripMenuItem`. Obě dvě třídy slouží k přidání nástrojů do komponenty `MenuStrip`. Tyto nástroje jsou `TextBox` a `TrackBar`.

9 Závěr

Všechny cíle diplomové práce byly splněny v celém rozsahu. Nejdříve jsem čtenáře seznámil s technologiemi pro uchování multidimenzionálních dat a s datovými strukturami vhodnými k reprezentaci segmentu železniční sítě. Technologií pro uchovávání multidimenzionálních dat jsem analyzoval uložení prostorových dat v souborech, v databázích a datových strukturách (2D strom, Range strom, Prioritní vyhledávací strom). Jako možnost uchování prostorových dat zde byly popsány prostorové nástroje databází od firmy Oracle a Microsoft.

Další část diplomové práce byla podrobněji zaměřena na technologii Oracle Spatial. Byly zde rozebrány výhody a jednotlivé částí této technologie se zaměřením na datový typ `SDO_GEOMETRY`.

Dále byly popsány technologie Oracle Topology a Network Data models. Model Topology přinesl možnost modelování územních celků za pomocí datového typu `SDO_TOPO_GEOMETRY`. U tohoto modelu jsem dále demonstroval, jak rozdělit jednotlivé prvky jako uzly, hrany a plochy do jednotlivých vrstev topologie.

Druhým datovým modelem, který je zde popsán, je datový model Network. V rámci tohoto modelu můžeme v Oracle Spatial modelovat síťovou reprezentaci prostorových dat pomocí uzlů a hran. Network data model byl použit v praktické části modelování železniční sítě.

Pro možnost vizualizace prostorových dat v Oracle Spatial byl v práci popsán Oracle Aplication Server se zaměřením na komponentu MapViewer.

Praktická část diplomové práce se prvotně zaměřuje na vybudování sítové reprezentace, jež odráží vybranou část infrastruktury železniční sítě. K vytvoření sítě byla zvolena technologie Oracle Spatial Network Data Model. Tato technologie byla vybrána z důvodu podrobnějšího zobrazení infrastruktury železniční sítě pomocí uzlů a vazeb mezi nimi a také díky možnosti síťové analýzy. V této části byl popsán postup pro vytvoření, naplnění a nastavení sítě Railway.

Dalším krokem praktické části byla vizualizace železniční sítě Railway. Tento krok byl rozdělen na dvě části a to vytvoření samotné mapy za použití nástroje Map Builder a vytvoření funkcionalit mapy. Funkcionality mapy byly zachyceny na webové stránce `mapa.html`, která byla použita na testovací aplikaci. Funkcionalita mapy na webové stránce byla napsána programovacím jazykem JavaScript.

Poslední částí diplomové práce byl návrh a implementace desktopová testovací aplikace, která vizualizuje segment železniční sítě, pokrývající Pardubický a Kralohradecký kraj, včetně pohybu kolejových vozidel. Aplikace byla naprogramována v jazyce C# na platformě Microsoft .NET Framework ve vývojovém prostředí Microsoft

Visual Studio 2010. Pro potřeby vizualizace mapy byla použita komponenta `WebBrowser`, která obsahovala vytvořenou mapu.

Pro simulaci provozu na železniční síti byla realizována pomocí diskrétní simulace, která pracuje s importovanými soubory tras jednotlivých vlaků. Při tvorbě aplikace byl brán zřetel na maximální využití návrhových vzorů a co nejrychlejší vykreslování mapy.

Technologie Oracle Spatial je velice zajímavá technologie pro práci s prostorovými daty, kde díky funkcím a operátorům pracujícím nad prostorovými indexy dokážeme provádět velmi efektivní analýzy těchto dat. Bohužel při vizualizaci prostorových dat pomocí komponenty MapViewer jsme zatím omezeni, protože nyní nejsou tolik podporovány datové modely Network a Topology v Oracle Database 11g. Například dynamické znázornění síťového tématu a hierarchie síťového modelu není zatím podporováno.

Přínosem diplomové práce bylo vytvoření a ověření použitelného návrhu modelu železniční sítě za pomocí technologie Oracle Spatial Topology and Network Data Models a ověření využitelnosti těchto technologií při práci s prostorovými daty. Problém, který nastal ohledně vizualizace hierarchie železniční sítě, byl vyřešen pomocí vytvoření dvou sítí: abstraktní a detailní.

Další možné rozšíření diplomové práce je detekce kolizí kolejových vozidel na jednokolejných tratí a operativní řízení vlaků při výlukách a zpoždění vlaků.

Literatura

[1] JEDLINSKÝ, Jan. *Způsoby uložení prostorových dat v databázi pro účely pozemkového datového modelu*. Plzeň, 2006. Diplomová práce. Západočeksá univerzita v Plzni.

[2] ČINČURA, Jiří. *MS SQL 2008 - prostorová data poprvé* [online]. 12.10.2009 [cit. 2012-04-22]. Dostupné z: http://www.dbsvet.cz/view.php?cisloclanku=2009101201

[3] *Microsoft TechNet*, [online]. c2011 [cit. 2012-04-22]. *Spatial Indexing Overview* Dostupné z: http://msdn.microsoft.com/cs-cz/library/bb964712%28v=sql.100%29.aspx

[4] ŘEHÁK, Tomáš. *Ukládání obrazových dat v Oracle Spatial* [online]. 12.10.2007 [cit. 2012-04-22]. Dostupné z: http://gis.zcu.cz/studium/pdb/referaty/2007/Rehak_GeoRaster/index.html

[5] HELJULA, Anthony. *Oracle Business Intelligence 11g Spatial Integration* [online]. 2010 [cit. 2012-04-22]. Dostupné z: http://www.rittmanmead.com/files/biforum2011/Heljula_Spatial.pdf

[6] MURRAY, Chuck, et al. *Oracle® Spatial : User's Guide and Reference 10g Release 2 (10.2)* [online]. 2005 [cit. 2012-04-22]. Dostupné z : http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14255.pdf

[7] SAMET, Hanan. *Foundations of multidimensional and metric data structures*. Boston: Elsevier/Morgan Kaufmann, c2006, 993 s. ISBN 978-012-3694-461.

[8] ČD M12. *Předpis pro jednotné označování tratí a kolejišť v informačním systému ČD*. Brno : České dráhy, 1996. 56 s. Dostupné z: http://www.szdc.cz/documentpublisher/download?documentId=1%3B%23a60c838b-a464-4a11-8355-eca1a6fbfc91&contentId=0.

[9] KAVIČKA, Antoním, *Graf*. Elektronické sylaby přednášek k předmětu Datové struktury a algoritmy. Pardubice, 2010.

[10] PROKEŠ, Jan, *Generování železniční* sítě. Univerzita Pardubice, 2012, 77 s. Diplomová práce. Univerzita Pardubice

[11] KOTHURI, Ravi, Albert GODFRIND a Euro BEINAT. *Pro Oracle Spatial for Oracle database 11g*. New York, NY: Distributed to the book trade worldwideby Springer-Verlag New York, c2007, xxxiv, 787 p. ISBN 15-905-9899-7.

[12] MURRAY, Chuck, et al. *Oracle® Fusion Middleware : User's Guide for Oracle MapViewer 11g Release 1 (11.1.1)* [online]. 2010 [cit. 2012-07-09]. Dostupné z : http://docs.oracle.com/cd/E14571_01/web.1111/e10145.pdf

[13] MURRAY, Chuck, et al. *Oracle® Spatial : Topology and Network Data Models 10g Release 2 (10.2)* [online]. 2006 [cit. 2012-07-09]. Dostupné z : http://docs.oracle.com/cd/B19306_01/appdev.102/b14256.pdf

Příloha A: Uživatelská příručka k Railway aplikaci

Aplikace Railway je testovací program sítě Railway vymodelovaný v Oracle Database 11g. Aplikace slouží vizualizaci mapy a používání prostorových analýz na síti. Aplikace komunikuje s uživatelem pomocí grafického rozhraní.

Požadavky

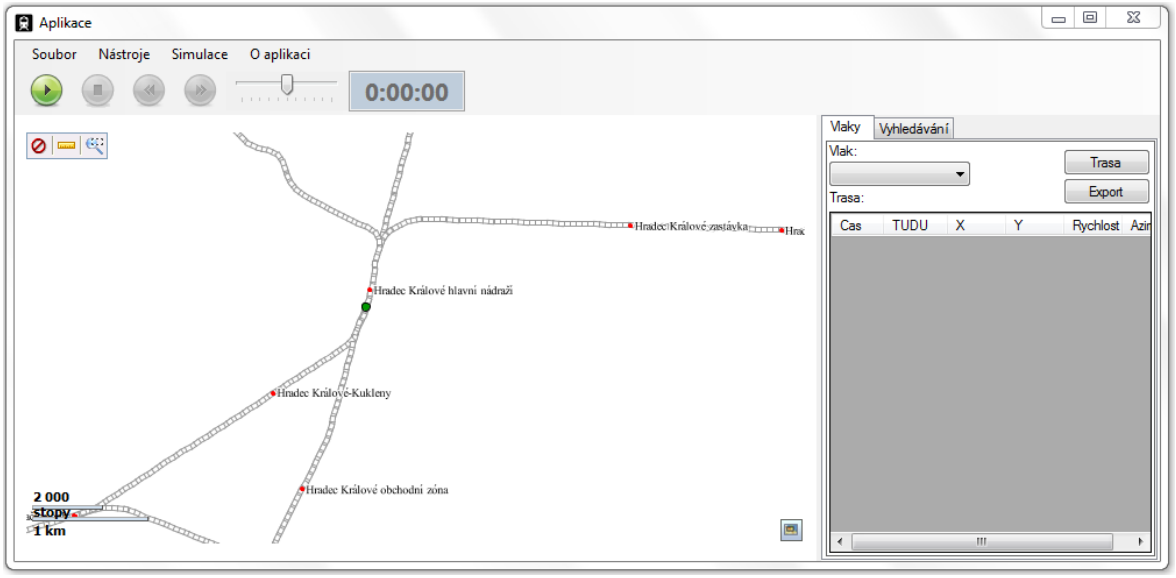
Aplikace pro správnou funkčnost potřebuje mít nainstalovanou databázi Oracle Database 11g Enterprise, knihovnu Microsoft .NET Framework 4.0 a knihovnu ODP.NET.

Uživatelské rozhraní

Grafické uživatelské rozhraní je rozděleno na pět částí. Každá část má specifickou funkci v aplikaci. Jednotlivé části, které si dále popíšeme v následujících částech uživatelské příručky, jsou:

- mapa,
- hlavní menu,
- menu simulace,
- záložka Vlaky,
- záložka Vyhledávání.

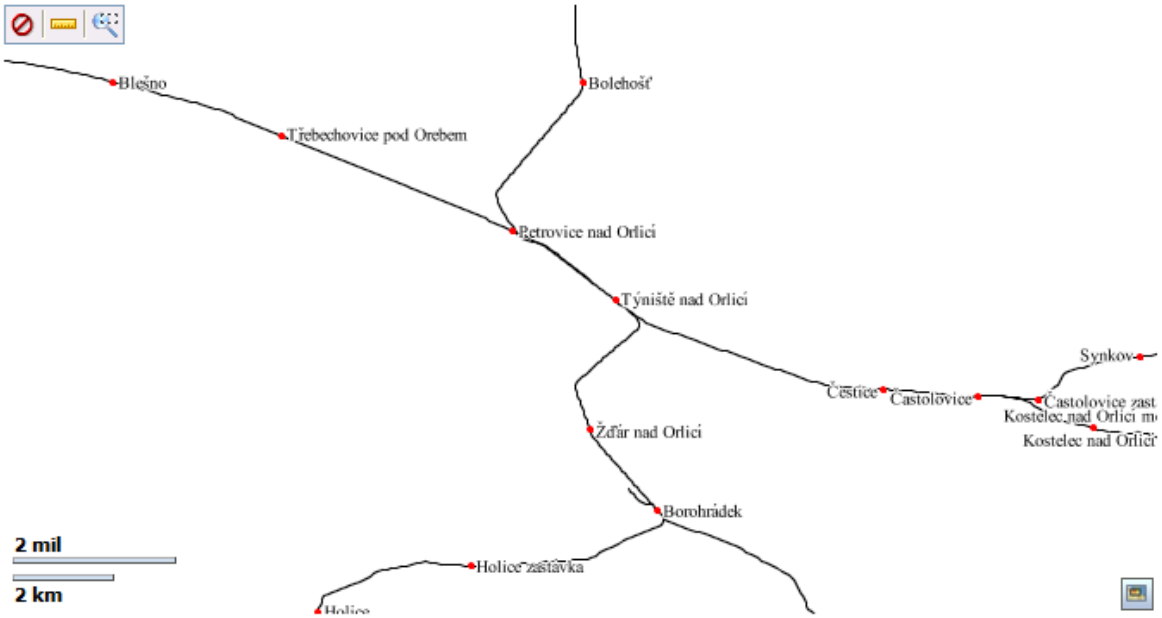
Grafické uživatelské rozhraní je znázorněno na Obrázek 322.



Obrázek 32: Grafické uživatelské rozhraní aplikace. Zdroj: [autor]

Mapa

Mapa slouží v aplikaci na vykreslování železniční sítě a jednotlivých vlaků, které jsou na ní importovány. Tato část je znázorněna na Obrázek 333. Mapa kromě tohoto zobrazení obsahuje nástroje pro měření vzdáleností a obdélníkové přibližování. Tyto nástroje jsou situovány v horním levém rohu mapy. Na mapě dále nalezneme aktuální měřítko přiblížení mapy, které je znázorněno v levém dolním rohu.



Obrázek 33: Náhled na Mapu. Zdroj: [autor]

Poslední část mapy je přehled mapy, který je schovaný pod ikonou v pravém dolním rohu. Po otevření ikony se nám rozevře oddálená mapa s obdélníkovým posouváním (viz Obrázek 344).

Obrázek 34: Náhled na Přehled mapy. Zdroj: [autor]

Hlavní menu

Hlavní menu se skládá ze čtyř položek (viz Obrázek 35). První položka je *Soubor*, který obsahuje ukončení aplikace a možnost načtení vlaku ze souborů *.xls*, *.xlsx* a *.csv*. Druhá položka *Nástroje* umožňuje zapínat a vypínat panel nástrojů na mapě. Třetí možnost *Simulace* slouží jako náhradní možnost *Menu simulace*. Poslední položka *O aplikaci* spouští dialogové okno se základními informacemi o aplikaci.

Obrázek 35: Náhled na Hlavní menu. Zdroj: [autor]

Menu simulace

Menu simulace obsahuje zleva doprava tlačítko pro spuštění simulace, tlačítko pro zastavení simulace, dvě tlačítka a track panel pro zrychlení a zpomalení simulace, box s aktuálním časem simulace (viz Obrázek 36).

Jednotlivá tlačítka jsou propojeny s možnostmi v položce Hlavního menu Simulace, které slouží ke stejným funkcím.

Při načtení aplikace je povoleno pouze tlačítko a možnost spuštění simulace (Play). Po spuštění simulace se nám povolí další tlačítka (resp. možnosti) pro zastavení simulace a změna rychlosti simulace, ale zakáže se nám použití tlačítka pro spuštění simulace (resp. možnost).

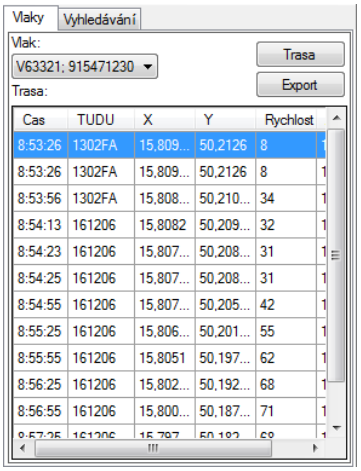
Box s aktuálním časem simulace je při načtení aplikace nastaven na 0:00:00. Při běhu simulace se čas nastaví na aktuální čas simulace.

Obrázek 36: Náhled na Menu simulace. Zdroj: [autor]

Záložka Vlaky

Záložka Vlaky (viz Obrázek 37) slouží k zjištění ujeté trasy vlaků. Z nabídky vlak si vybereme určený vlak a klikneme na tlačítko *Trasa*. Po zmáčknutí tlačítka *Trasa* se načte seznam projetých bodů ve tvaru čas, TUDU, x souřadnice, y souřadnice, rychlost a azimut, uloží se do výpisu pod výběrem vlaku a na mapě se zobrazí ujetá trasa.

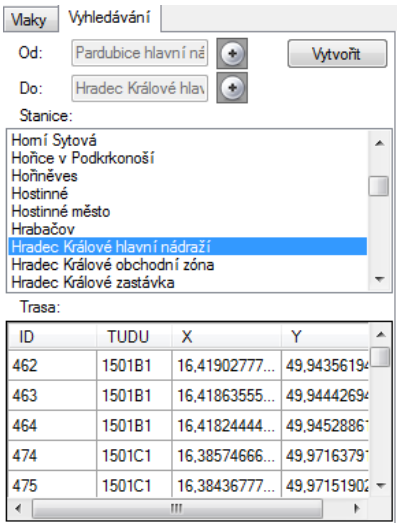
Trasa může být dále exportována do souboru *.csv* pomocí tlačítka Export. Tento soubor je stejného typu jako soubory používané pro načítání vlaků.



Obrázek 37: Náhled na záložku Vlaky. Zdroj: [autor]

Záložka Vyhledávání

Záložka Vyhledávání (viz Obrázek 38) slouží k vyhledávání trasy mezi dvěma železničními stanicemi, které jsou načteny v seznamu *Stanice*. Ze seznamu se postupně vyberou dvě stanice a přidávají se do kolonek *Od* a *Do* pomocí tlačítek přidat, které jsou za příslušnými kolonkami tlačítka přidat (značený plusem v kruhu). Následně pro vytvoření nejkratší cesty se použije tlačítko *Vytvořit*, které vygeneruje trasu do seznamu *Trasa*.



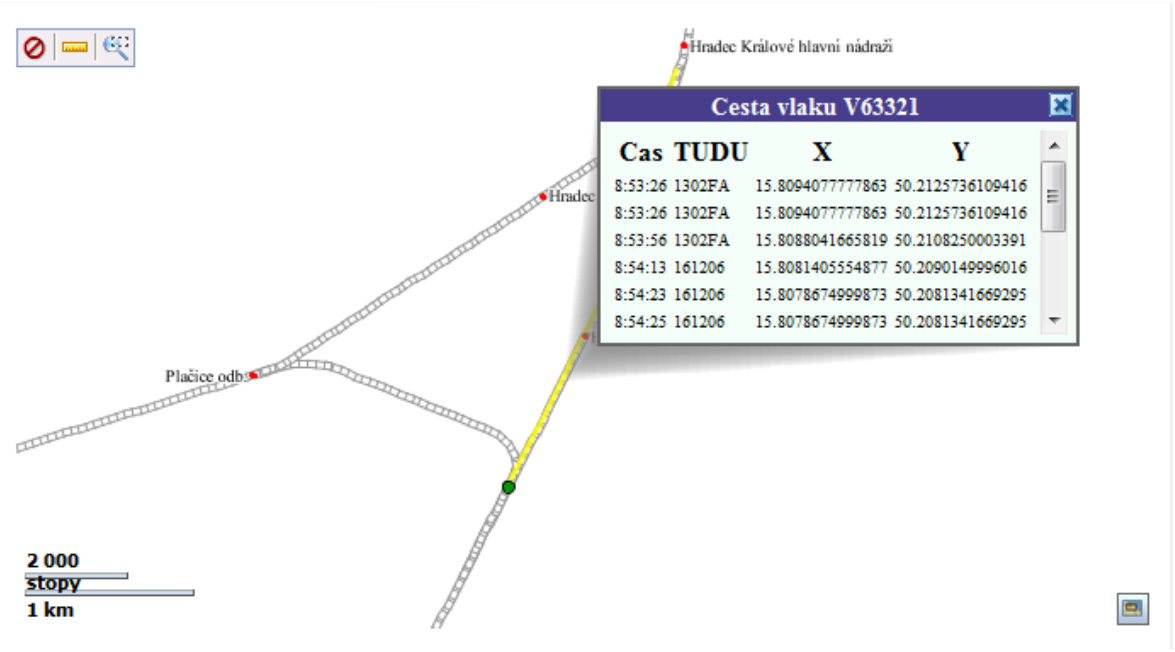
Obrázek 38: Náhled na záložku Vyhledávání. Zdroj: [autor]

Ujetá trasa

Ujetá trasa slouží k znázornění ujeté trasy vlaku na mapě. Při kliknutí na vlak nebo pomocí tlačítka *Trasa* ze záložky Vlaky se nám zobrazí u příslušného vlaku žlutá křivka reprezentující jeho trasu.

Po kliknutí na žlutou čáru se nám v mapě zobrazí tabulka obsahující seznam projetých bodů.

Zobrazení obou případů je znázorněno na Obrázek 399.



Obrázek 39: Zobrazení trasy a tabulky projetých bodů. Zdroj: [autor]

Příloha B: Struktura souborů a adresářů na zdrojovém CD

Na přiloženém CD naleznete:

- adresář *Aplikace* – obsahuje spustitelnou aplikaci (demonstrační aplikace potřebuje mít nainstalované a nastavené knihovny .NET Framework 4.0, ODP.NET, databázový server Oracle Databae 11g Enterprise a nástroj MapViewer), před spuštěním musí být také zapnutý MapViewer,
- adresář *Data* – obsahující soubory pro import tras vlaků,
- adresář *Instalace* – obsahuje návod instalací a instalační balíčky, které byly použity při vývoji aplikace,
- adresář *ZdrojoveKody* – obsahující zdrojové kódy mapy a testovacího projektu vytvořeného v nástroji Visual Studio 2010,
- soubor *RezaninaE_NavrhModeluZeleznicniSite_JF_2012.pdf* – soubor obsahující text diplomové práce v elektronickém formátu.