M U N I

Popis softwarového výsledku

ANALYZA – Datový sklad

vytvořeného v rámci řešení projektu
Komplexní analýza a vizualizace heterogenních dat velkého rozsahu („ANALYZA“)

Období řešení projektu: 1. ledna 2017 – 31. prosince 2020

Výzkumný program: Program bezpečnostního výzkumu České republiky 2015-2020

Hlavní řešitel: RNDr. Tomáš Rebok, Ph.D.

Řešitelský tým:

RNDr. Tomáš Rebok, Ph.D. (*2017-2020*)
RNDr. Michal Batko, Ph.D. (*2017-2020*)
RNDr. Milan Čermák (*2017-2020*)
RNDr. Martin Drašar, Ph.D. (*2017-2020*)
Mgr. Miroslava Jarešová (*2017*)
doc. RNDr. Barbora Kozlíková, Ph.D. (*2017-2020)*
RNDr. Vladimír Míč, Ph.D. (*2017-2018*)
RNDr. Filip Nálepa, Ph.D. (*2017-2018*)
RNDr. David Novák, Ph.D. (*2017-2018*)
RNDr. Daniel Tovarňák, Ph.D. (*2017-2018*)
Mgr. Kristína Zákopčanová (*2017-2020*)
prof. Ing. Pavel Zezula, CSc. (*2017-2020*)
Mgr. Martin Macák (*2018*)
Mgr. Matúš Guoth (*2018*)
RNDr. Jakub Peschel (*2019-2020*)
RNDr. Katarína Furmanová, Ph.D. (*2019*)
Bc. Denis Kasanič (*2019-2020*)
Bc. Marko Řeháček (*2019-2020*)
Bc. Denisa Šrámková (*2019-2020*)
Matej Babej (*2019-2020*)
Bc. Dávid Brilla (*2019-2020*)
Bc. Martin Kažimír (*2019-2020*)
Erik Hricišák (*2019-2020*)
Bc. Jozef Bátrna (*2019-2020*)
Daniel Plakinger (*2020*)
Vladimír Lazárik (*2020*)
Kristián Gutič (*2020*)
Ing. Helena Mojdlová (*2017*)
Ing. Jitka Ročková (*2018-2020*)

V Brně, dne 30. ledna 2021

Masarykova univerzita

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

Obsah

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

1 Kontext SW komponenty „Datový sklad“ ve vyvinutém systému

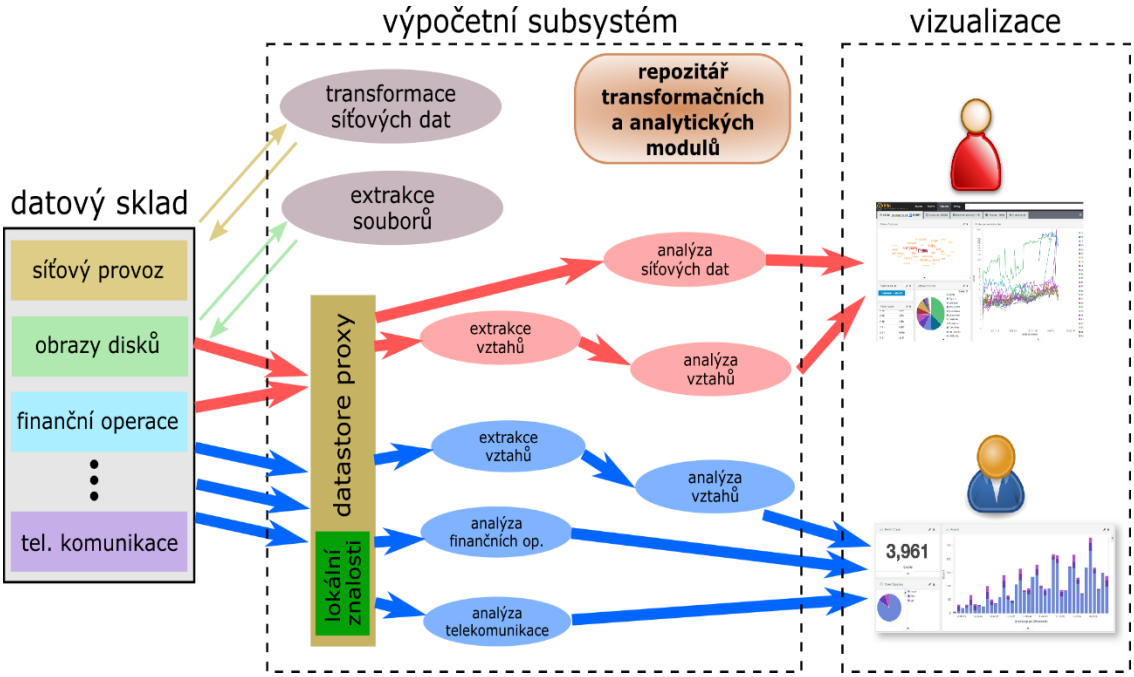
Hlavním cílem námi řešeného projektu byl návrh a realizace distribuovaného systému umožňujícího komplexní analýzu heterogenních dat velkého rozsahu. Navržený systém je koncipován jako nástroj umožňující jednotné uložení netriviálně velkých datových sad (např. záchytů síťového provozu, obrazů disků, komunikačních informací či znalostí z terénu), nad nimiž jsou vyvolávány nejrůznější mezidoménové analýzy (např. identifikace skupin, které jsou v kontaktu nezávisle na tom, jaký komunikační prostředek volí, případně identifikace po telekomunikační síti komunikujících stran, které si současně vyměňují netriviální finanční prostředky atp.). Takto navržená koncepce má za cíl prozkoumat možnosti a limity podobných mezidoménových analýz, jejichž ambicemi je omezit izolované analýzy různých datových domén a nezbytnost budování mezidoménového kontextu jen v režii analytika. Navržená koncepce vypomůže jak s budováním a udržováním tohoto kontextu přímo nad analyzovanými datovými sadami, tak i s odhalováním nových nebo složitě zaznamenatelných datových souvislostí a kontextů, které by jinak byly buď stěží nebo zcela vůbec odhalitelné.

Od tohoto cíle se odvíjely základní požadavky na vyvinutý systém, z nichž ty nejzásadnější lze v krátkosti sumarizovat následovně:

- *Škálovatelnost* – základní požadavek jakéhokoli systému, jehož ambicemi je práce s velkými daty či náročnými výpočty. Nejinak je tomu i v případě námi vyvinutého systému, kdy jak návrh celé architektury, tak i návrh a implementace dílčích komponent podporuje principy distribuovaného či paralelizovaného zpracování a práce s velkými daty.
- *Flexibilita* – neméně zásadní požadavek na systém, jehož ambicemi je práce s různými datovými doménami a různorodými analýzami. Námi vyvinutý systém je vyvinut s předpokladem, že nelze specifikovat konkrétní datové sady a informace, které má systém udržovat, ale naopak musí být připraven v budoucnu jednoduše integrovat různé, i nyní neznámé datové sady. Podobně i všechny analytické funkce systému nebylo možno specifikovat v době jeho vývoje, ale poskytnutím obecného frameworku s dostatečnou flexibilitou je umožněno jejich budoucí doplňování a adaptace novým potřebám.
- *Interaktivita* – požadavek úzce související se škálovatelností systému, jehož ambicí je poskytnout co nejinteraktivnější zpracování a reakce na požadavky uživatele – v závislosti na množství analyzovaných dat – v co nejkratším čase, což je nezbytně důležité zejména pro účely tzv. explorativních analýz (postupného odhalování znalostí).
- *Spolehlivost a důvěryhodnost* – neméně důležité požadavky, posilující důvěru uživatelů v odhalené skutečnosti a schopnosti systému.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

- *Bezpečnost* – jelikož tato netriviální oblast byla nad časový i personální rámec aktivit tohoto projektu, systém jsme vyvinuli tak, aby byl připraven pro dodatečnou implementaci rozsáhlých bezpečnostních mechanismů, včetně práce principů AAI (autentizace, autorizace, identity).

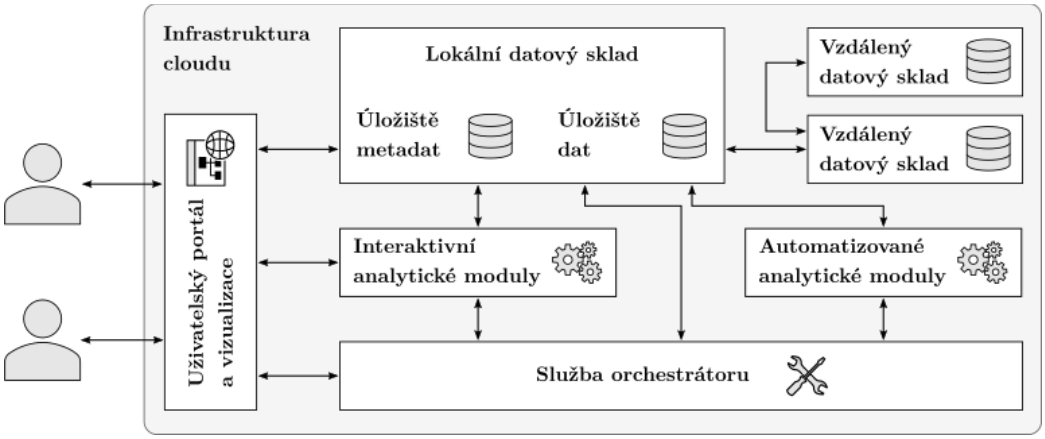


Obrázek 1: Ilustrace myšlenky sjednocených datových analýz s využitím analytických operací sestavených do datového workflow

Pro demonstraci této myšlenky sjednocených analýz (Obrázek 1) jsme navrhli a implementovali architekturu systému, jehož základní komponenty jsou ilustrovány na následujícím obrázku (Obrázek 2). Těmito základními komponentami jsou:

- ***Datový sklad*** v hierarchické sestavě distribuovaných datových úložišť udržujících veškerá analyzovaná data (z technického, programátorského a uživatelského pohledu popsaný v následujících kapitolách),
- ***Výpočetní subsystém***, poskytující sadu transformačních a analytických modulů a ***Orchestrační službu***, která tyto moduly vhodně skládá do datových workflow realizujících transformace dat v Datovém skladu či vlastní datové analýzy,
- ***Vizualizační komponentu*** poskytující analytické možnosti a prezentující výsledky jednotlivých analýz uživateli.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.



Obrázek 2: Architektura vyvinutého distribuovaného systému pro analýzu rozsáhlých heterogenních dat

Jádro celého implementovaného systému tvoří *analytické moduly*, které vhodně sestaveny do datového workflow (tzv. *analytické operace*) reprezentují analytické možnosti celého modulárního systému. Veškerá vstupní data – včetně poznatků, pozorování a výsledků provedených analýz – udržuje komponenta Datového skladu, ze které jsou potřebná data vyzvedávána analytickými moduly (a následně tamtéž ukládány zpracované mezivýsledky). Analytické moduly jsou ovládány Orchestrátorem, který zajišťuje jejich správné spuštění, včetně předání parametrů specifických pro daný modul. Současně zajišťuje řetězení jednotlivých modulů do analytických operací tak, aby se analytik mohl soustředit pouze na cílový analytický modul, který mu zpřístupní požadovaná data bez toho, aby je musel zpracovávat a připravovat odděleně. Různé analytické operace a moduly jsou spouštěny skrze Vizualizační komponentu, která umožnuje zachycení jednotlivých poznatků a případně výsledků analýz ve formě vztahového diagramu. Na základě výběru jednotlivých uzlů a hran daného grafu pak může analytik vybrat data, se kterými chce zacházet, a předat je Orchestrátoru a jednotlivým analytickým modulům. Jednotlivé komponenty analytického systému tvoří efektivně propojený celek tak, aby se plně využil potenciál pokročilých analýz a nových nástrojů pro zpracování dat spojených s policejním vyšetřováním.

Analytické operace jsou typicky složeny ze specializovaných automatizovaných a interaktivních analytických modulů. Automatizované moduly poskytují funkce transformace daných dat a různé formy detekce na základě předaných parametrů. Výsledky těchto modulů jsou uloženy v rámci Datového skladu, kde je možné s nimi dále pracovat ať už v navazujících analytických modulech nebo ve vztahové vizualizaci. Interaktivní analytické moduly zpřístupňují analytikovi daná data v rámci uživatelského rozhraní, kde je možné je analyzovat pomocí pokročilých funkcí daného analytického nástroje. S využitím pokročilých funkcí Orchestrátoru je možné analytické moduly spouštět samostatně nebo jako klastr distribuovaných výpočtů, pokud to daný modul podporuje.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

2 SW komponenta „Datový sklad“

Systémová komponenta Datový sklad slouží jako úložiště veškerých dat v systému, prakticky uložených ve formě souborů, objektů a vazeb (vztahů) mezi nimi. Kromě uložení dat určených k analýze představuje Datový sklad i prostor, do kterého analytické moduly ukládají mezivýsledky svých operací – například v podobě rozšíření stávajících objektů o nové informace, atributy nebo i vytvořením zcela nových objektů či souborů. Díky flexibilitě a komplexnosti zvoleného datového modelu je možné Datový sklad používat i pro ukládání uživatelem definovaných rozšiřitelných atributů vložených pro podchycení nově objevených, ne nutně potvrzených informací (např. pro potřeby značkování nebo agregace entit napříč běžícími analýzami).

Při návrhu této komponenty jsme kladli důraz především na:

- *škálovatelnost* – předpokládá se ukládání v řádech až jednotek petabyte;
- *spolehlivost* – odolnost proti výpadku hardware, využití replikace;
- *rychlost* – schopnost ukládat velkoobjemová data v reálném čase (například zachycené proudy dat datové komunikace), schopnost rychlého vybavování dat (v závislosti na náročnosti operací);
- *flexibilitu (možnost uložení libovolných typů dat)* – variabilita ukládání, možnost uložení strukturovaných i nestrukturovaných dat, binární formáty od jednotlivých souborů až po obrazy disků atp.

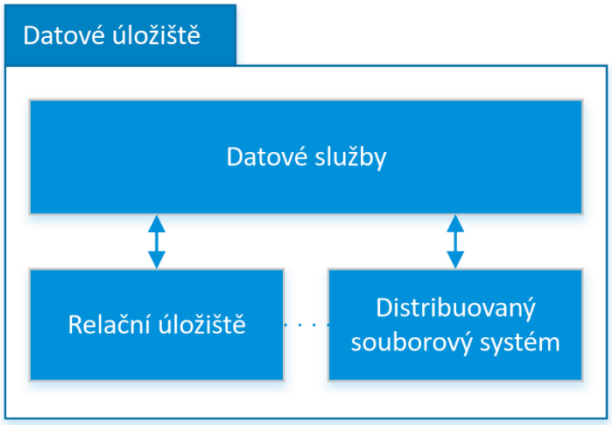
Nad datovým skladem lze provádět základní filtrovací operace – pro potřeby konkrétní analýzy je vždy vybrána podmnožina dat z datového skladu. Objekty uložené v datovém skladu pak lze propojovat do externích systémů – jako odkaz je využita URL adresa webové služby, která zajistí dostupnost příslušného objektu. Zabezpečení dat ve skladu je v budoucnu možné provádět pomocí externí autorizace zajišťující, že pro využití daných dat musí mít osoba příslušná oprávnění.

2.1 Architektura

Architektura Datového skladu, ilustrovaná na následujícím obrázku, sestává ze tří základních komponent: *úložiště dat* (vhodně zvolený distribuovaný souborový či objektový systém – pro potřeby prototypové implementace byl zvolen Hadoop Distributed Filesystem[1]), *relačního úložiště* (vhodně zvolená SQL databáze – pro potřeby prototypu je využíváno PostgreSQL[2]) a *API rozhraní* (námi v jazyce Java implementované komunikační rozhraní pro potřeby integrace, komunikace a celkové manipulace s uloženými daty a objekty).

[1] https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html
[2] https://www.postgresql.org/

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.



Obrázek 3: Schéma komponent Datového skladu

Komponenta distribuovaného souborového systému slouží k fyzickému uložení veškerých binárních dat skladu, a to i velkoobjemových souborů v řádech tera- až peta-bajtů. Pro potřeby projektu byla zvolena technologie Hadoop Distributed Filesystem (HDFS)[3], ale vzhledem k vyčlenění a zapouzdření je možné komponentu v budoucnu snadno nahradit jinou implementací. Distribuovaný souborový systém se skládá ze skupin komoditních serverů a je horizontálně škálovatelný, čímž zajišťuje požadavek na škálovatelnost. Data se v distribuovaných souborových systémech ukládají dělením souborů na bloky. Bloky jsou ukládány ve více kopiích, pro případ očekávaného selhání komoditních serverů. Replikací bloků distribuovaný souborový systém zajišťuje požadavek na spolehlivost a dostupnost. Datové bloky jsou replikovány, proto je k nim možné přistupovat z více zdrojů, což zvyšuje propustnost, a tedy rychlost čtení dat. Rychlost zápisu také profituje z dělení dat na bloky, neboť je možné zapisovat na více serverů zároveň. Distribuovaný souborový systém ukládá soubory, které mohou mít různou strukturu (strukturované – CSV soubory, polostrukturované – záchyt datové komunikace, nestrukturovaná – obrázky), čímž splňuje požadavek flexibility.

Relační úložiště udržuje metadata o uložených souborech ve strukturované formě, a je přizpůsobené na práci s takovými daty. Umožňuje filtrování dat s vyjadřovací silou jazyka SQL a v rámci metadat má uloženy i unikátní identifikátory k jednotlivým souborům, díky čemuž zprostředkovává jejich dostupnost z distribuovaného datového skladu. Dostupnost práce s daty při výpadku relační databáze lze zajistit vícenásobným nasazením relační databáze v režimu active-standby nebo plnohodnotně distribuovanou technologií jako je Hive[4]. Pro potřeby projektu byla v této

[3] http://hadoop.apache.org
[4] https://hive.apache.org

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

fázi zvolena varianta používající relační databázi PostgreSQL. Pro implementaci je využíván princip objektově-relačního mapování dat, snadno lze tedy pouhou záměnou ovladače použít jinou podkladovou databázi.

Datové služby zastřešují subkomponenty datového skladu a zajišťují komunikaci se zbývajícími komponentami systému. Programové rozhraní obsahuje především sady metod pro výpis (*list*), čtení (*get*) a zápis (*put*) objektů do skladu (detaily k navrženému API jsou podrobněji rozepsány v ročních zprávách a popisu výsledku). Metody výpisu dat umožňují specifikaci filtru, který je definovaný jako komplexní podmínka na metadatové atributy, které jsou uloženy v relačním úložišti. Výstupem těchto metod je seznam objektů se specifikovanými metadaty, kde každý objekt obsahuje globálně unikátní identifikátor, pomocí kterého je dále možné přečíst binární data metodou *get*. Dále jsou poskytovány servisní metody především pro podporu tzv. *Uploaderu dat*, kterým je možné nahrávat do distribuovaného úložiště nové objekty. Každé volání metody rozhraní bude možné rozšířit o autorizaci s využitím *Security frameworku*, který ověřuje, zda je každý jednotlivý objekt možné poskytnout aktuálnímu uživateli.

2.2 Hierarchie Datových skladů

Idea v projektu původně plánovaného centrálního datové úložiště, kde budou udržována veškerá validní data, byla v průběhu projektu nahrazena variantou, kde stále existuje centrální Datový sklad, ale je možné vytvářet i lokální Datové sklady. Takový lokální Datový sklad pak přebírá vazbu na nadřazený Datový sklad (například centrální) a umožňuje transparentně přístup k datům v nadřazeném skladu, jako by byly uloženy lokálně. Veškerá ukládaná data (tedy změny nebo nové objekty) jsou ale zapisována pouze do lokálního skladu, ovšem s globálně unikátní identifikací objektu, která zůstává zachována při modifikaci objektu. Tedy z hlediska uživatele lokálního Datového skladu jsou data v nezměněné podobě buď poskytována z původního nadřazeného skladu nebo jsou nahrazena upravenou variantou z lokálního skladu. Tato funkcionalita pak může být poskytována zcela transparentně prostřednictvím API datových služeb daného lokálního skladu (s využitím tzv. Proxy komponenty).

V rámci projektu jsme implementovali pouze jednoduchý ilustrativní příklad této Proxy komponenty, jejíž funkcionalita demonstruje popsanou myšlenku. Její komplexní řešení a plnohodnotná implementace totiž musí podchytit i řadu produkčně potřebných případů, jako například konflikty změny stejného objektu ze dvou různých Datových skladů a problém duplicitních objektů, které reprezentujících stejnou reálnou entitu, ale vzniklých nezávisle na sobě ve dvou různých skladech.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

3 Použité technologie

V této kapitole popisujeme technologie, které byly využity pro implementaci komponenty Datového skladu.

Maven

Správa aplikace Datového skladu je řešena s využitím nástroje Maven, který umožňuje výrazně zjednodušit celou implementační správu tohoto aplikačního balíku. Hlavní funkce tohoto nástroje jsou například řízení závislostí, sestavování aplikace a vytváření dokumentace. Všechny potřebné informace pro Maven jsou v souboru POM.xml (např. informace o závislostech na externích knihovnách, o specifikaci buildování/sestavování skladu a podobně).

Spring

Jednou z komponent datového skladu má být aplikační rozhraní – API, skrze které aplikace využívající Datový sklad s tímto komunikují a ten následně transparentně pracuje s databází a souborovým systémem. Proto jsme se rozhodli využít aplikační rámec Spring, pomocí kterého vytvoříme aplikační rozhraní Datového skladu.

Konfigurace Spring aplikace je zajištěna pomocí tzv. Spring-bootu, který zajišťuje jeho minimální autokonfiguraci. Spring jako aplikační rámec přináší mnoho výhod. Například na implementaci webového aplikačního rozhraní REST API využíváme zpřístupněné REST kontroléry, které výrazně zjednodušují implementaci jednotlivých kontrolérů a zajišťují tak metody aplikačního rozhraní. Dále jsme se rozhodli využít například rozhraní *CrudRepository* a *Hibernate* – jednu z implementací Spring data JPA, které popisujeme v následujících sekcích.

CrudRepository

Rozhraní, které přináší tzv. CRUD – *Create, Read, Update* a *Delete* funkcionalitu pro entitu třídy, kterou spravuje. Po implementaci tohoto rozhraní je objekt schopen základních, ale i pokročilých CRUD operací s danými objekty entity, pro kterou je repositářem. Zajišťuje tedy základní operace a je spojením aplikačního rozhraní Datového skladu a databáze.

Hibernate

Hibernate reprezentuje tzv. ORM – objektově-relační mapovací nástroj, tedy nástroj, díky kterému jsme schopni převést relace (tabulky) z databáze na objekty v aplikaci Datového skladu. Tento nástroj zajišťuje mimo jiné zavedení datového modelu do databáze – úvodní definici tabulek a vztahů podle definovaných entitních tříd. Po vhodné přípravě entitních tříd je již zavedení objektů (tabulek) do databáze plně automatické a pohodlné. Hibernate poskytuje anotace, díky kterým dokážeme objektově nadefinovat datový model v POJO (*Plain Old Java Object*) entitních třídách.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

HDFS

Datový sklad musí být schopen uložit objemné soubory v reálném čase – příkladem těchto souborů budiž záchyty síťové komunikace, výpisy z logů, obrazy disků a podobně. Pro splnění požadavků spolehlivosti, flexibility, škálovatelnosti a rychlosti je vhodné použít distribuovaný souborový systém.

Distribuovaný souborový systém je souborový systém, který využívá metodu založenou na klient-server architektuře pro uložení a přístup k souborům. V distribuovaném souborovém systému jeden nebo více centrálních serverů ukládají data, ke kterým přistupuje několik autorizovaných vzdálených klientů. Samotný server může být distribuován napříč několika fyzickými počítačovými uzly, které poskytují celkově dostupnou úložnou kapacitu.

Jako souborový systém jsme zvolili distribuovaný souborový systém Hadoop HDFS, který vznikl pro potřeby dávkového zpracování dat technikou *MapReduce*, a který umožňuje uložení objemných datových souborů. HDFS nabízí stabilní rozhraní pro konfiguraci připojení souborového systému – *org.apache.hadoop.conf* a rozhraní souborového systému – *org.apache.hadoop.fs*. Rozhraní souborového systému obsahuje metody obdobné příkazům z prostředí HDFS aplikace v příkazovém řádku.

PostgreSQL

Databáze, jako součást Datového skladu zajišťuje několik funkcí: zajišťuje jak uchování metadat k souborům uloženým v souborovém systému, tak také uložení generických objektů, vztahů a ostatních potřebných údajů specifikovaných datovým modelem. Podobně pak zajišťuje možnosti následného zpřístupnění dat nebo souborů filtrováním uložených dat v databázi.

Pro potřeby implementace Datového skladu jsme se rozhodli zvolit relační databázi PostgreSQL[5], která poskytuje moderní objektově-relační přístup, a ve které je možné jednoduše implementovat všechny entity z datového modelu spolu s relacemi mezi těmito entitami.

Swagger

Nástroj, který z popisu jednotlivých metod API Datového skladu, obsahujícího specifikace parametrů a možných výstupních stavových kódů, umožňuje vygenerovat jeho komplexní dokumentaci. Tato je součástí přiloženého archivu jako YAML soubor. Dokumentaci lze vložit do Swagger Editoru, a tím zobrazit vygenerovanou a přehlednou Swagger dokumentaci.

[5] https://www.postgresql.org/

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

4 Návod pro nasazení Datového skladu

Pro nasazení Datového skladu jsou nezbytné 3 základní kroky:

1. Nasazení PostgreSQL serveru
2. Nasazení HDFS clusteru
3. Nasazení API Datového skladu

Každý z těchto kroků je součástí přílohy a v ni obsažených samostatných skriptů – *PostgreSQL.sh* pro nasazení databáze, *HDFS.sh* pro nasazení souborového systému, stejně jako skript *Deploy-DataWarehouse.sh*, který nasazuje API datového skladu, a ve kterém jsou obsaženy i předchozí dva skripty. V této sekci blíže popíšeme operace, které skripty provádějí. Je vhodné zmínit, že skripty jsou vytvořeny pro uživatele *ubuntu* a testovány v čisté instanci s operačním systémem Ubuntu 18.04.

4.1 PostgreSQL

Skript pro nasazení PostgreSQL serveru jednoduše nainstaluje PostgreSQL pomocí *apt install postgresql postgresql-contrib*. Následně uživateli *postgres* (který je používán v Datovém skladu) upraví heslo. Dále vytvoří databázi s názvem *test_database*, kde přiřadí práva uživateli *postgres*. Nakonec pomocí příkazu *service postgresql start* spustí PostgreSQL server, což je možné ověřit běžícími procesy, resp. ověřením že server poslouchá na implicitním portu 5432. Tedy na URL *localhost:5432/test_database* by měl být schopen se připojit uživatel *postgres* s vytvořeným heslem.

4.2 HDFS

Pomocí skriptu pro nasazení HDFS clusteru a přiložených konfiguračních souborů ve složce *site-files* vytvoříme, nakonfigurujeme a spustíme lokální HDFS klastr. V úvodu nainstalujeme software Java verze 8, následně stáhneme Hadoop-2.7.3 archív s potřebnými soubory pro HDFS, který rozbalíme. Vygenerujeme SSH klíč a vytvoříme připojení na *localhost* skrze SSH pro námi vytvořený klíč. Následuje definování potřebných proměnných *HADOOP_HOME, HADOOP_CONF* apod., po kterém nakopírujeme zmíněné *site-files* konfigurační soubory, které jsou vhodně připraveny – obsahují jen minimální potřebné konfigurace pro běh HDFS. Nakonec naformátujeme *namenode* a spustíme HDFS, což je možné ověřit na URL *localhost:50070*, kde by měly být viditelné servisní informace o běžícím HDFS.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

4.3 API Datového skladu

Posledním skriptem *DeployDataWarehouse.sh* nasadíme a spustíme API datového skladu. Skript obsahuje i spuštění předchozích dvou skriptů, po jejichž vykonání vytvoří na HDFS souborovém systému složky *data*, která obsahuje soubory nahrané do HDFS, a *tmp_files*, kde se dočasně ukládají soubory před nahráním do HDFS. Následně nainstaluje Maven a po přesunu do složky monolith (která obsahuje zdrojové soubory pro API datového skladu) spustí Datový sklad příkazem *mvn spring-boot:run*. API datového skladu je pak přístupné na URL *localhost:8080*.

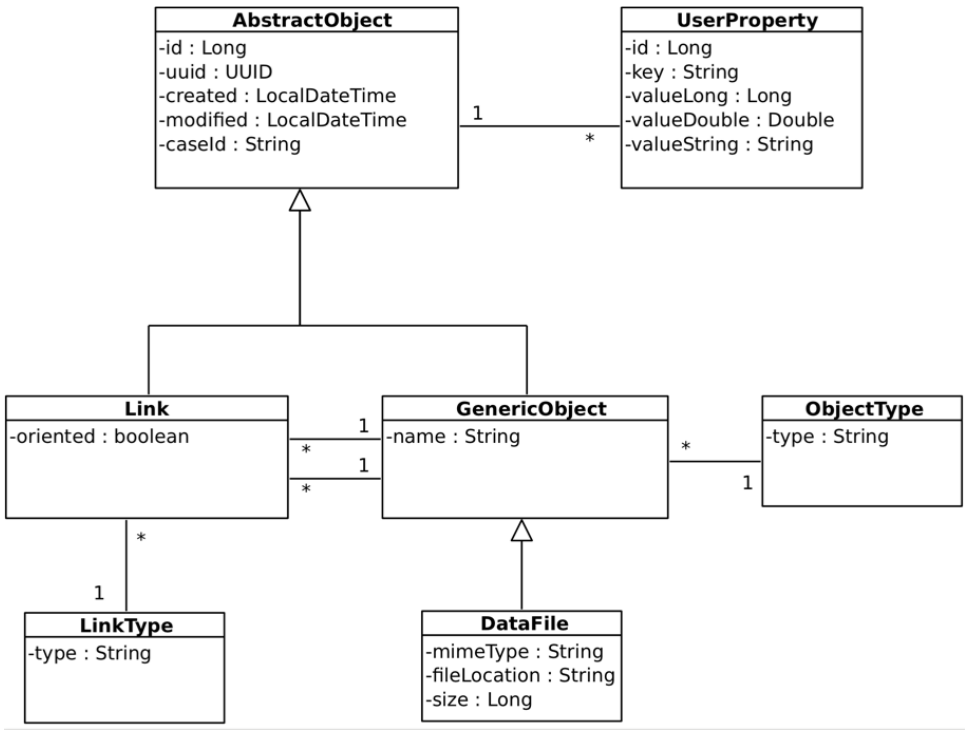
Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

5 Uživatelská dokumentace

V této uživatelské dokumentaci se věnujeme způsobu práce s Datovým skladem. Předpokladem pro vysvětlení jednotlivých operací, které Datový sklad nabízí, je pochopení množiny objektů, se kterými Datový sklad pracuje – tedy pochopení aplikovaného datového modelu. Dále tato sekce obsahuje obecný popis API metod Datového skladu. V sekci nazvané *Filtrování* pak popisujeme způsob použití filtrace pro získání objektů z Datového skladu – sestavení dotazů pro filtrování. V poslední sekci se pak věnujeme specifickému případu uložení derivátů.

5.1 Popis datového modelu

Každá operace, kterou datový sklad poskytuje, je spojena s jeho datovým modelem (Obrázek 4). Datový model definuje množinu různých objektů a jejich propojení, které je možné v Datovém skladu uložit, a se kterými je možné pracovat. Proto v následujících odstavcích vysvětlíme podstatu objektů, s nimiž pracujeme v Datovém skladu, protože obdobně metody REST API Datového skladu se vztahují na jednotlivé objekty – *GenericObject, Link, DataFile*.



Obrázek 4: Datový model využitý pro popis různorodých objektů a souvisejících informací (metadat)

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

Základ jakéhokoliv údaje uloženého v Datovém skladu je ***AbstractObject*** – objekt obsahující nutné minimum údajů, jako například databázový identifikátor objektu, UUID, datum a čas vytvoření objektu, datum a čas poslední úpravy objektu a identifikátor případu – *caseId* (identifikátor specifický pro policejní vyšetřovatele). Tento objekt zároveň obsahuje libovolné množství objektů *UserProperty*.

UserProperty představuje uživatelem definované atributy formátu *klíč-hodnota*, kde *klíč* tvoří libovolný řetězec a *hodnota* je specifikována buď jako celé číslo, desetinné číslo nebo řetězec.

GenericObject pak představuje jakýkoliv generický objekt, který má dva povinné atributy – jméno a *ObjectType*, tedy název typu generického objektu. Zároveň *GenericObject* rozšiřuje *AbstractObject*, což znamená, že obsahuje všechny jeho atributy, jakož i objekty *UserProperty*. Příklad objektu *GenericObject* může být například osoba, auto a podobně.

Speciální případ objektu *GenericObject* je ***DataFile***, tedy soubor, který oproti obecnému objektu obsahuje navíc atributy *typ, umístění* a *velikost* souboru.

Objekt ***Link*** pak reprezentuje vztah mezi dvěma libovolnými objekty typu *GenericObject* nebo *DataFile*. Každý vztah má atribut *LinkType* popisující typ vztahu a příznak orientace vztahu. Jelikož *Link* rozšiřuje *AbstractObject,* obsahuje také všechny jeho atributy včetně možnosti uložení objektů *UserProperty*. Je možné vytvořit i reflexivní vztah, tedy vztah objektu se sebou samým. Díky tomu, že *DataFile* rozšiřuje *GenericObject,* je možné vytvořit vztah nejen mezi objekty typu *GenericObject*, ale i mezi objekty typu *DataFile*.

5.2 API metody

Datový sklad je implementován formou služby bez vlastního vizuálního grafického rozhraní, tedy komunikuje výhradně skrze navržené API funkce (podrobněji viz roční zprávy projektu nebo popis výsledku). Pro účely jednoduššího přidávání nových rozšíření API Datového skladu jsme jeho API funkce rozdělili do tří od sebe izolovaných úrovní, konkrétně:

- **nízkoúrovňové API**, které představuje množinu základních (nízkoúrovňových) operací s entitami v Datovém skladu,
- **obecné rozšiřující API** obsahující typicky kombinaci více metod nízkoúrovňového API, které reprezentuje a zjednodušuje nějaký obecný princip, jako je například uložení derivátů objektu,
- **doménově-specifické API** obsahující metody specifické pro konkrétní doménu použití Datového skladu – v našem případě metody specifické pro analytické moduly a Vizualizační komponentu využité pro potřeby policejního vyšetřování.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

Požadovaná metoda	URL	Popis volání
POST	/generic-objects	Nahrání datového objektu do Datového skladu (DS)
GET	/generic-objects/{uuid}	Získání objektu (specifikovaného UUID) z DS
PUT	/generic-objects/{uuid}	Aktualizace metadat objektu (dle UUID)
DELETE	/generic-objects/{uuid}	Výmaz objektu z DS, případně i včetně datových souborů
GET	/generic-objects/filter/{query}	Vyhledání datových objektů dle zadaného dotazu
POST	/generic-objects/data-files	Nahrání datového souboru a datového objektu do DS
GET	/generic-objects/data-files/{uuid}	Získání datového souboru z DS
PUT	/generic-objects/data-files/{uuid}	Aktualizace datového souboru (dle UUID)
DELETE	/generic-objects/data-files/{uuid}	Výmaz datového souboru z DS
POST	/links	Vytvoření vztahu mezi dvěma datovými objekty
GET	/links/{uuid}	Získání vztahu mezi objekty
PUT	/links/{uuid}	Aktualizace popisných informací vztahu
DELETE	/links/{uuid}	Výmaz existujícího vztahu mezi dat. objekty
GET	/links/filter/{generic-object-uuid}	Získání všech vztahů konkrétního datového objektu

Tabulka 1: Metody nízkoúrovňového API Datového skladu

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

Požadovaná metoda	URL	Popis volání
POST	/generic-objects/derivation	Nahrání derivátů do Datového skladu (DS)
GET	/generic-objects/derivation/{input-uuid}	Získání derivátů (dle UUID vstupního objektu) z DS
POST	/generic-objects/filter	Vytvoření nového objektu jako uloženého filtru z generických objektů v DS
GET	/generic-objects/filter/{filter-uuid}	Získání objektů nalezených aplikací uloženého dotazu v DS
DELETE	/generic-objects	Vymazání všech generických objektů podle zadaného dotazu v DS
GET	/generic-objects/light/filter	Vyhledání „lehkých“ generických objektů, obsahujících omezené informace dle zadaného dotazu
GET	/links/light/filter	Vyhledání „lehkých“ linek, obsahujících omezené informace podle zadaného dotazu z DS

Tabulka 2: Metody obecného rozšiřujícího API Datového skladu

Požadovaná metoda	URL	Popis volání
POST	/visualisation	Nahrání/aktualizace generických objektů a jejich linek odeslaných ve speciálním objektu vytvořeném pro potřeby Vizualizační komponenty
GET	/visualisation	Vyhledání generických objektů a jejich linek podle zadaného dotazu a jejich vrácení ve speciálním objektu vytvořeném pro potřeby Vizualizační komponenty

Tabulka 3: Metody doménově-specifického API Datového skladu

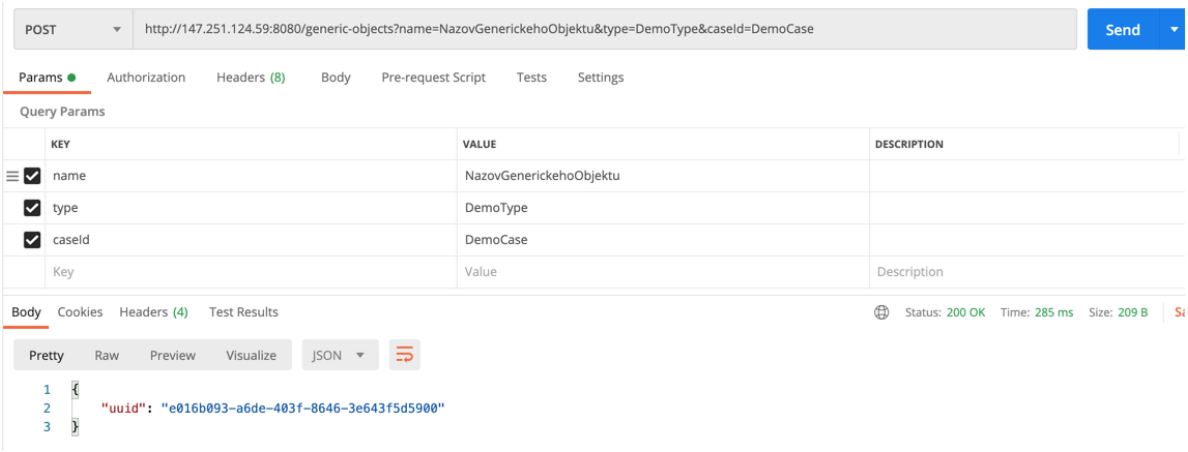
5.3 Ukázkové příklady použití Datového skladu

V této sekci ukážeme jednoduché ilustrativní příklady práce s Datovým skladem, pro které využijeme klienta *Postman*[6]. Postupně vytvoříme generický objekt, soubor, necháme si vrátit námi vytvořen generický objekt, necháme si stáhnout námi vytvořený soubor. Dále přidáme do generického objektu *userProperty*. Následně použijeme filtr při definici dotazu na základě *caseId*. Na konci vymažeme oba vytvořené objekty a ověříme, že se ve skladu již nenacházejí.

[6] https://www.postman.com

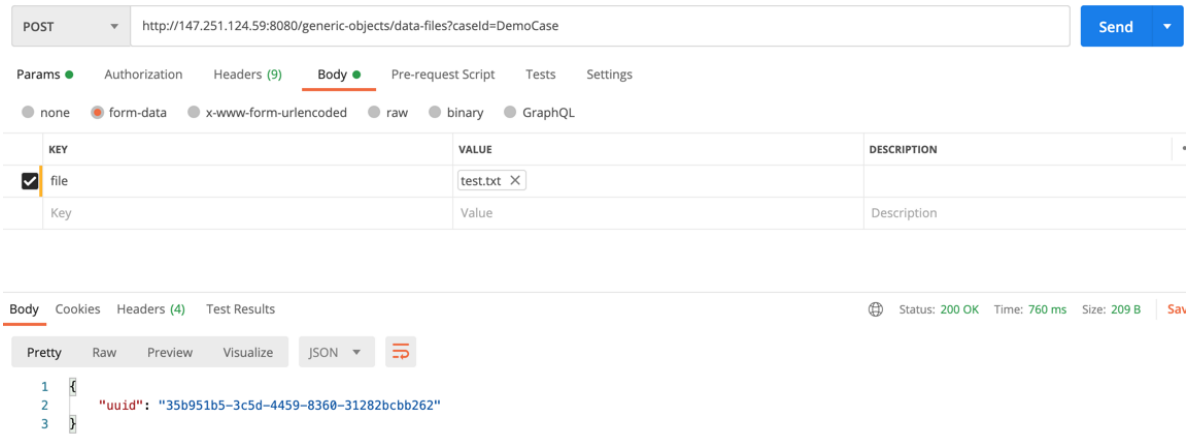
Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

Vytvoříme generický objekt s názvem *NazovGenerickehoObjektu*, typem *DemoType* a identifikátorem případu *DemoCase* požadavkem POST na dané URL. V odpovědi Datového skladu dostaneme UUID vytvořeného objektu – "e016b093-a6de-403f-8646-3e643f5d5900".



Obrázek 5: Vytvoření generického objektu příkazem /generic-objects?name=NazovGenerickehoObjektu&type=DemoType&caseId=DemoCase

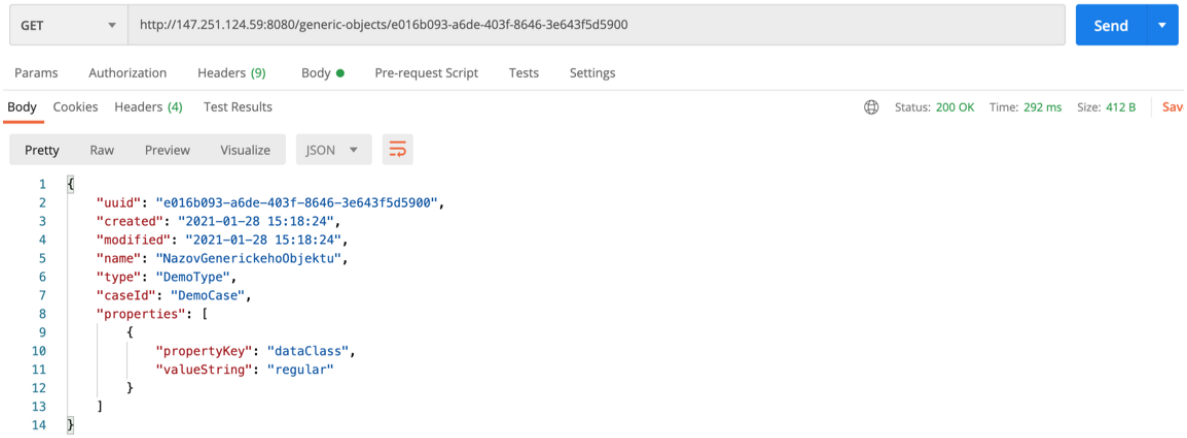
Následně uložíme soubor *test.txt* (obsahující řetězec "test test test"), kterému přiřadíme identifikátor případu *DemoCase* požadavkem POST na dané URL. V odpovědi datového skladu dostaneme UUID uloženého objektu, resp. v tomhle případě souboru – "35b951b5-3c5d-4459-8360-31282bcbb262".



Obrázek 6: Uložení souboru test.txt příkazem /generic-objects/data-files?caseId=DemoCase

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

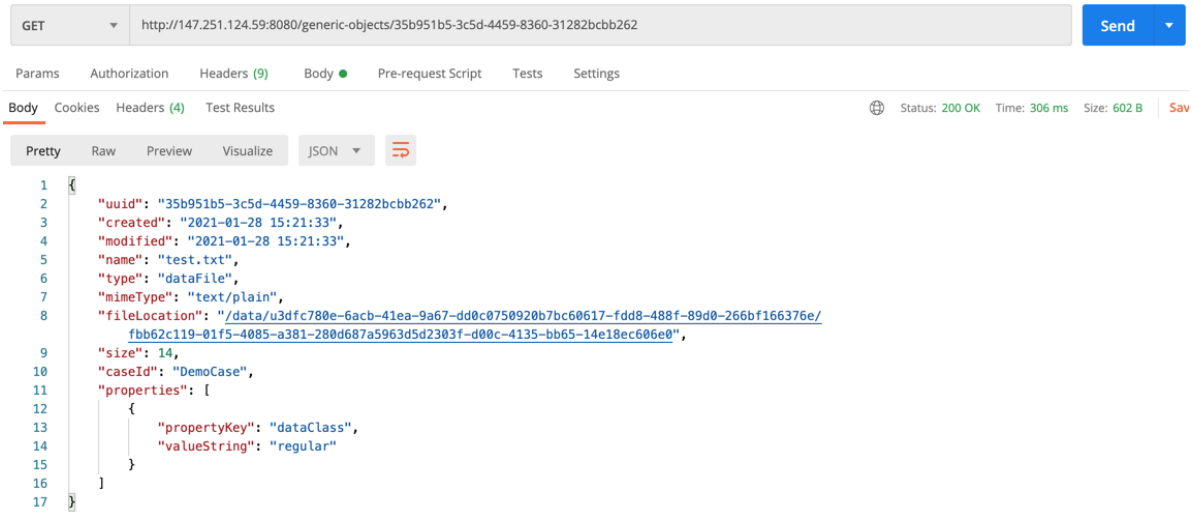
Vytvořený generický objekt s UUID "e016b093-a6de-403f-8646-3e643f5d5900" si necháme vrátit požadavkem GET na dané URL. Na následujícím obrázku (Obrázek 7) je možné vidět JSON reprezentaci objektu *GenericObject*.



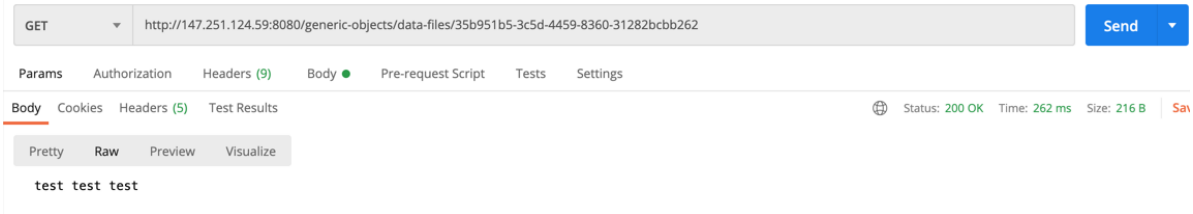
Obrázek 7: Vrácení námi vytvořeného generického objektu /generic-objects/e016b093-a6de-403f-8646-3e643f5d5900

Vytvořený soubor s UUID *"35b951b5-3c5d-4459-8360-31282bcbb262"* si necháme vrátit příkazem GET na dané URL. Na následujícím obrázku (Obrázek 8) je možné vidět JSON reprezentaci objektu *DataFile*. Na vrácení samotného souboru je však nezbytné využít příkaz ilustrovaný dále (Obrázek 9), kde použijeme příponu */data-files*. Obrázek tak ilustruje získání obsahu uloženého souboru, tedy řetězce „*test test test*".

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.



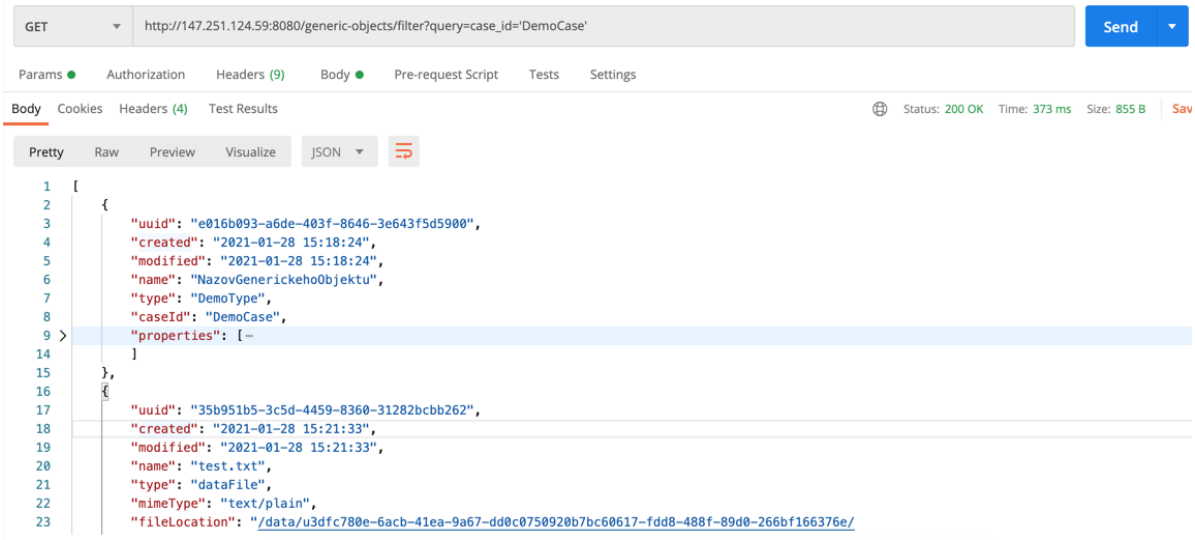
Obrázek 8: Vrácení objektu námi uloženého souboru /generic-objects/35b951b5-3c5d-4459-8360-31282bcbb262



Obrázek 9: Vrácení námi uloženého souboru /generic-objects/data-files/35b951b5-3c5d-4459-8360-31282bcbb262

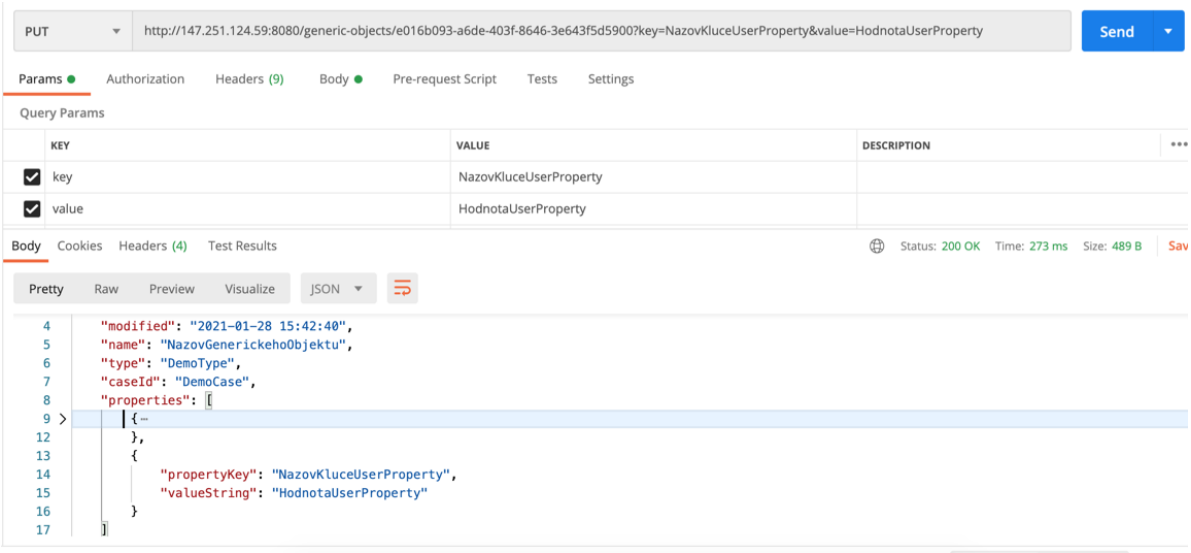
Použitím filtru s dotazem na *case_id='DemoCase'* ve skutečnosti vyfiltrujeme všechny objekty, které jsou uloženy ve skladu pod tímto identifikátorem případu (*caseId*). Použijeme GET příkaz na dané URL. Obdobně dokážeme objekty filtrovat s využitím ostatních atributů. Ve výpisu můžeme také vidět záznamy objektů, v našem případě námi vytvořený generický objekt a uložený soubor ve formátu JSON.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.



Obrázek 10: Filtrování na základě case_id
/generic-objects/filter?query=case_id='DemoCase'

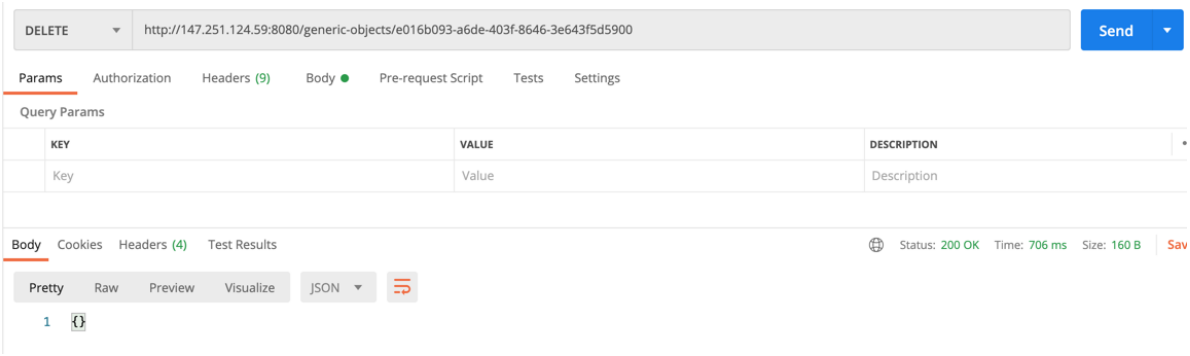
Nyní vytvoříme novou *userProperty* s klíčem *NazovKluceUserProperty* a s hodnotou *HodnotaUserProperty* příkazem PUT na dané URL. Na obrázku můžeme vidět změnu v JSON výpisu objektu, kde se v jeho vlastnostech nachází nový, uživatelem definovaný atribut.



Obrázek 11: Vložení uživatelem definovaného atributu
/generic-objects/e016b093-a6de-403f-8646-
3e643f5d5900?key=NazovKluceUserProperty&value=HodnotaUserProperty

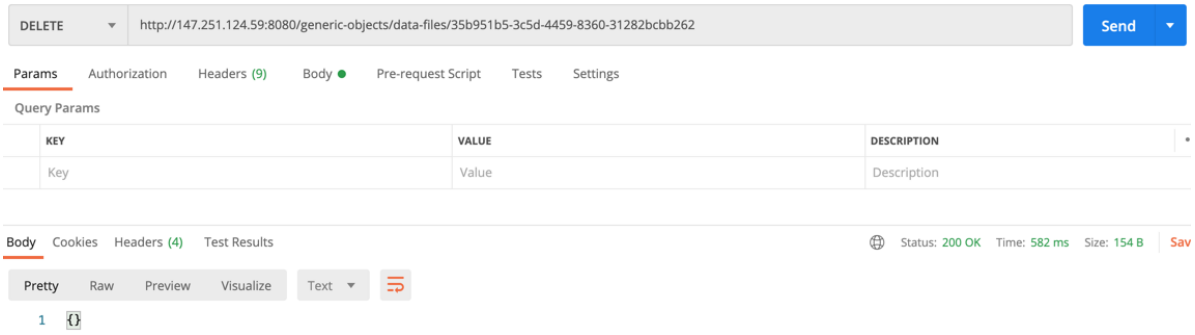
Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

Nyní vymažeme námi vytvořený a uživatelským atributem rozšířený generický objekt s danou UUID příkazem DELETE na dané URL. Ve výpise odpovědi Datového skladu je vrácen prázdný JSON.



Obrázek 12: Výmaz generického objektu /generic-objects/e016b093-a6de-403f-8646-3e643f5d5900

Obdobně můžeme vymazat i námi uložený soubor *test.txt* s daným UUID, a to příkazem DELETE zaslaným na dané URL. Ve výpisu odpovědi Datového skladu je vrácen prázdný JSON. Po takto realizovaném výmazu obou objektů je pak Datovým skladem vrácen prázdný seznam (datový sklad je prázdný) a požadavek na vrácení konkrétního objektu pak vrací chybovou hlášku *404 Not Found.*



Obrázek 13: Výmaz souboru /generic-objects/data-files/35b951b5-3c5d-4459-8360-31282bcbb262

5.4 Uložení derivátů

Jedním z důležitých požadavků na Datový sklad je uložení a správa derivátů. Jako *deriváty* označujeme soubory/objekty, které jsou nějakou definovanou funkcí odvozeny od originálních

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

(vstupních) souborů/objektů. Proces tvorby derivátů z originálních souborů nazýváme *transformační metodou*, která je zpravidla součástí analytických procesů.

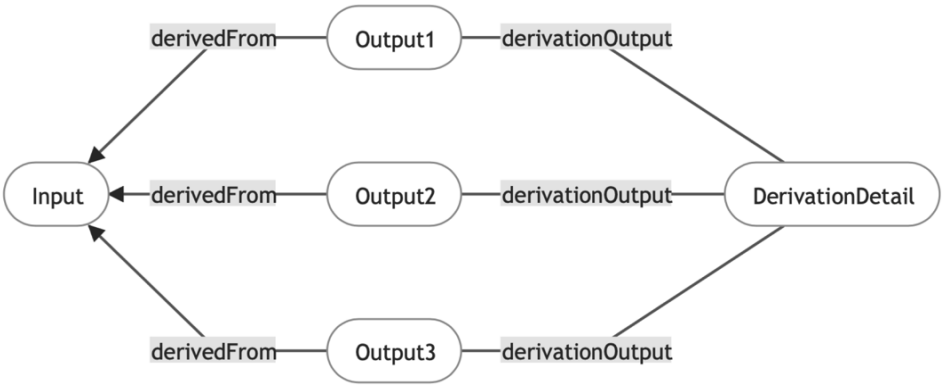
Primárním účelem takto tvořených a ukládaných derivátů je zefektivnění využití zdrojů – před požadavkem na zpracování vstupních objektů nějakou funkcí nejprve ověřeno, zda již na těchto vstupních objektech neexistuje derivát, který je vytvořen identickou transformační funkcí s identickými vstupními parametry. S výsledky takovýchto transformačních metod, deriváty, pak po procesu uložení pracujeme jako s obyčejnými soubory uloženými v Datovém skladu.

Proces uložení derivátů sestává z vytvoření vztahu typu *derivedFrom* mezi originálními (vstupními) soubory a jejich deriváty. Také je nutné uložit informace o použité transformační metodě, jako jsou její název, vstupní a výstupní parametry a další důležité informace. Tyto informace uložíme pomocí vytvoření dalšího generického objektu, jehož typ bude *DerivationDetail,* a jednotlivé parametry uložíme pomocí uživatelem zadaných atributů. Následně generický objekt propojíme s výstupními soubory transformační metody vztahem typu *derivationOutput.*

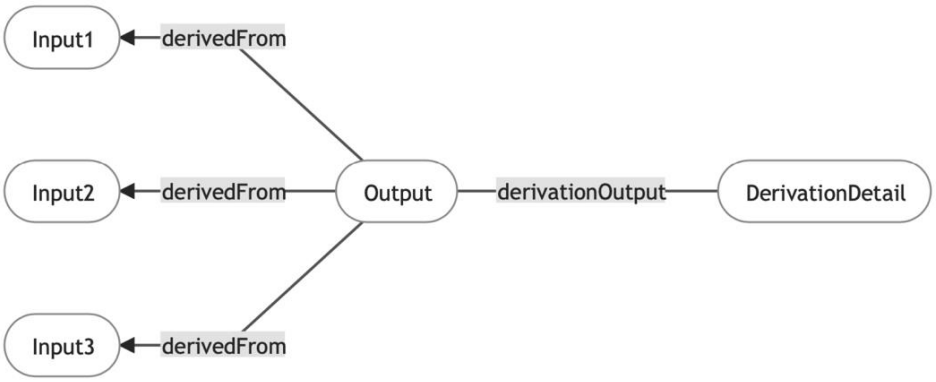
Existují čtyři možnosti vazby derivovaných objektů v závislosti na množství vstupních a výstupních souborů: jsou to případy 1-1, 1-m, m-1 a m-n. Všechny možnosti jsou ilustrovány na následujících obrázcích. Možnost m-n realizujeme jako kombinaci m-1 a 1-m, přičemž výstup m-1 je jeden archív, který je použit jako vstup pro 1-m.

Obrázek 14: Uložení derivátů typu 1-1

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.



Obrázek 15: Uložení derivátů typu 1-m



Obrázek 16: Uložení derivátů typu m-1

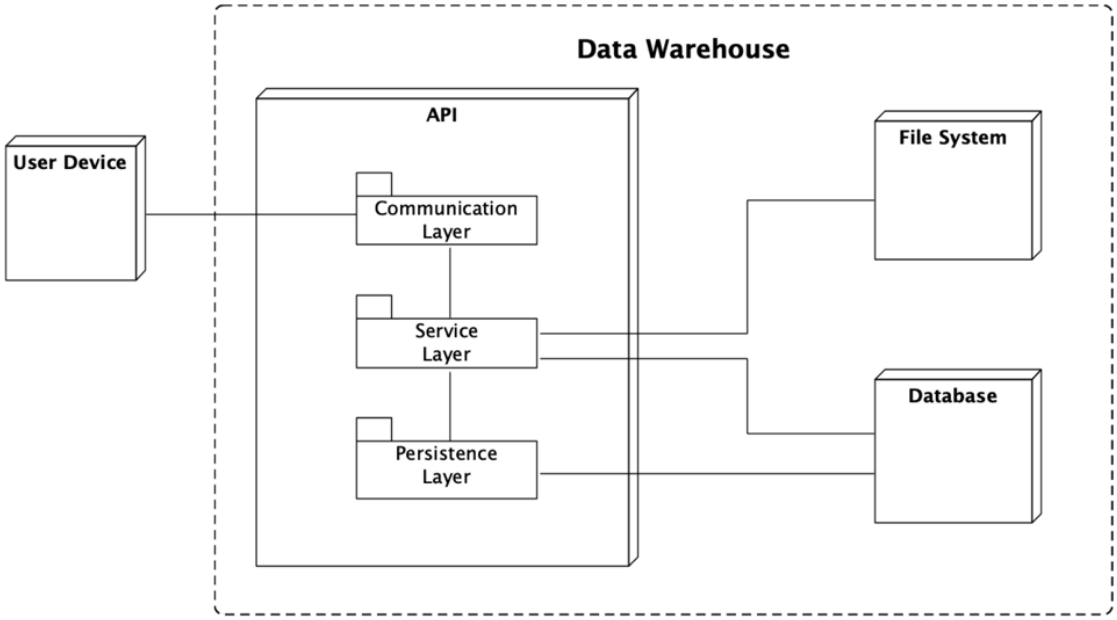
Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

6 Programátorská dokumentace

V této sekci postupně popisujeme návrh architektury, odpovědnost jednotlivých balíků tříd, implementační závislosti servisních balíků a implementační informace ohledně jednotlivých tříd.

6.1 Architektura

Diagram nasazení je znázorněn na následujícím obrázku (Obrázek 17) a velmi dobře podchycuje celou architekturu Datového skladu jako celku tří komponent. Sestává ze souborového systému HDFS, relační databáze PostgreSQL a Spring aplikace – API datového skladu. Dále hierarchicky dělí aplikaci API na 3 vrstvy – komunikační, servisní a perzistentní, které jsou detailně popsány v následující sekci Implementace.



Obrázek 17: Diagram nasazení Datového skladu

6.2 Implementace,

V sekci implementace hierarchicky postupně procházíme perzistentní, servisní, a nakonec komunikační vrstvou Datového skladu. Pro jednotlivé vrstvy popisujeme implementační závislosti balíků a implementační informace ohledně jednotlivých tříd a vrstev.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

6.2.1 Perzistentní vrstva

Perzistentní vrstva se podle svého návrhu stará o definici datového modelu do databáze PostgreSQL a o komunikaci a výměnu dat mezi aplikací a databází. Implementace této vrstvy obsahuje balíky *entity* a *repository*.

Pro práci s databází využíváme techniku objektově-relačního mapování (ORM). Jelikož API je implementováno v rámci aplikačního rámce Spring, použili jsme *Hibernate*. Pomocí *Hibernate* implementujeme balík *entity*, který obsahuje entitní třídy, které jsou využívány napříč celou aplikací. Entitní třídy jsou definovány podle datového modelu, jako POJO objekty, které jsou za použití JPA anotací spolu s nástrojem *Hibernate* transformovány pomocí automaticky generovaného databázového skriptu na tabulky.

Přístup k datům v databázi je řešen skrze balík *repository*, který obsahuje třídy definované jako úložiště jednotlivých entit. Třídy balíku *repository* jsou definovány jako rozhraní, které rozšiřují objekt *CrudRepository*, díky čemuž není nutná další modifikace tříd balíku *repository*, protože rovnou bez úprav nabízí dostatečné množství nebytných metod.

6.2.2 Servisní vrstva

Implementace servisní vrstvy odráží návrh, který sestává ze základních funkcí servisní vrstvy – interakce s rozhraními souborového systému, databáze a přidání aplikační logiky nad převzatými daty od komunikační vrstvy. Všechny servisní třídy se nacházejí v balíku *service*.

Komunikaci se souborovým systémem HDFS realizujeme díky nástrojům na konfiguraci připojení souborového systému – *org.apache.hadoop.conf*, a samotnému rozhraní souborového systému – *org.apache.hadoop.fs*, které obsahuje metody obdobné příkazům z příkazového řádku prostředí HDFS aplikace.

Pro interakci s databází Datového skladu používáme perzistentní vrstvu, která nabízí *repository* objekty. Pro případ nedostatku těchto tříd poskytuje Spring Data JPA možnost použití manažera entit. Manažer entit je schopen provést jakýkoli dotaz nad připojenou databází, včetně jednoduchých operací, které jsou obsluhované skrze *repository* objekty nebo skrze náročnější JOIN výběry a podobně. Pro lepší čitelnost a přehlednost kódu preferujeme použití *repository* objektů tam, kde je to možné.

6.2.3 Komunikační vrstva

Komunikační vrstva je vstupně-výstupní komponentou uživatele (resp. uživatelské aplikace) Datového skladu. Její implementace sestává z implementace REST rozhraní, které slouží pro specifikaci povolených a očekávaných vstupních parametrů. Jako výstupní formát API Datového

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

skladu je použit JSON. Rovněž je třeba specifikovat povolený rozsah různých stavových kódů (*status codes*), které jsou důležitou součástí výstupu API.

Implementace REST API je kompletně zabalená v balíku *controller*. REST kontroléry jsou komponenty, ke kterým se konkrétní HTTP žádost přiřadí na základě typu a URL žádosti. Tvorba kontrolérů je v našem případě řešena pomocí anotace jednoduchých tříd, v nichž každá metoda s vhodnou anotací představuje samostatnou HTTP žádost. V těle metod jsou následně provedeny operace, které konkrétní žádost požaduje.

V kontrolérech jsou také specifikovány výstupní stavové kódy jednotlivých metod API. Zvolili jsme následující množinu stavových kódů:

- *200 (OK)* – žádost je korektně zpracovaná, výstup je přípraven;
- *400 (Bad Request)* – HTTP žádost je nekorektní (důvodem jsou například chybějící povinné parametry, chybné URL apod.);
- *404 (Not Found)* – nelze najít požadovaný objekt (typicky nesprávné UUID v žádosti);
- *405 (Method Not Allowed)* – nesprávný typ HTTP žádosti (tedy neexistuje metoda s URL, která by měla zadaný typ žádosti);
- *406 (Not Acceptable)* – žádost obsahuje nesprávný formát vstupu nebo požaduje nepodporovaný formát výstupu;
- *500 (Internal Server Error)* – interní chyba Dátového skladu, bližší popis je přiložen do chybové zprávy, kde je výpis výjimky, kterou chyba vytvořila.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

7 Obsah přiloženého archívu

Přiložený archív obsahuje následující přílohy:

- *monolith* – adresář zdrojového kódu API aplikace
- *DeployDataWarehouse.sh* – skript pro nasazení Datového skladu, včetně zavedení HDFS clusteru a PostgreSQL serveru
- *HDFS.sh* – skript pro zavedení jednoho lokálního HDFS clusteru
- *site-files* – adresář s konfiguračními soubory pro zavedení HDFS
- *PostgreSQL.sh* – skript pro zavedení lokálního PostgreSQL serveru
- *swagger.yaml* – dokumentace API vytvořená nástrojem Swagger (možno zobrazit ve Swagger Editoru[7])

[7] https://editor.swagger.io

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.

8 Poděkování

Vytvořeno v rámci projektu *„Komplexní analýza a vizualizace heterogenních dat velkého rozsahu (ANALYZA)“* (kód projektu VI20172020096, období řešení 1.1.2017 – 31.12.2020) v rámci Programu bezpečnostního výzkumu České republiky v letech 2015-2020 Ministerstva vnitra ČR.

Žerotínovo nám. 617/9, 601 77 Brno, Česká republika
T: +420 549 49 1111, E: info@muni.cz, www.muni.cz
Bankovní spojení: KB Brno-město, ČÚ: 85636621/0100, IČ: 00216224, DIČ: CZ00216224
V odpovědi prosím uvádějte naše číslo jednací.