Rozšíření dispečinkového systému o modul zpoždění

Bc. Dalibor Dobeš

Diplomová práce
2018

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

*** Nascanované zadání, strana 1 ***

*** Nascanované zadání, strana 2 ***

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomové práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky. Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně .....

podpis autora

ABSTRAKT

Cílem diplomové práce je analyzovat, doplnit a rozšířit stavajici dispečinkový system o nové zdroje dat zpoždění a polohy spojů. To bude zahrnovat připojení na SOAP [1] API společnosti SŽDC odkud budem načítat aktualní zpoždění a další doplňkové informace. Pro účeli návrhu výpočtu zpoždění a polohy vozů se bude potřeba připojít k RESTovému api Polohavozu.cz . Jako poslední pak bude potřeba načítat strukturu vlakového spoje pomocí JavaMail api, to vše se bude ukládat do databaze stavajícího dispečinkovehé systému.

Klíčová slova: Java, REST, SOAP, API, EXCEL, CSV, WSDL, JSON

ABSTRACT

Text of the abstract

Keywords: Java, REST, SOAP, API, EXCEL, CSV, WSDL, JSON

Život je nepřetržitá řada průserů nepravidelně za sebou jdoucích.

OBSAH

ÚVOD

První odstavec pod nadpisem se neodsazuje, ostatní ano (pouze první řádek, odsazení vertikální mezy odstavci je typycké pro anglickou sazbu; czech babel toto respektuje, netřeba do textu přidávat jakékoliv explicitní formátování, viz ukázka sazby tohoto textu s následujícím odstavcem).

Formátování druhého odstavce. Text text text text text text text text text text text text.

I. TEORETICKÁ ČÁST

1 Vývojové nástroje, protokoly a aplikace

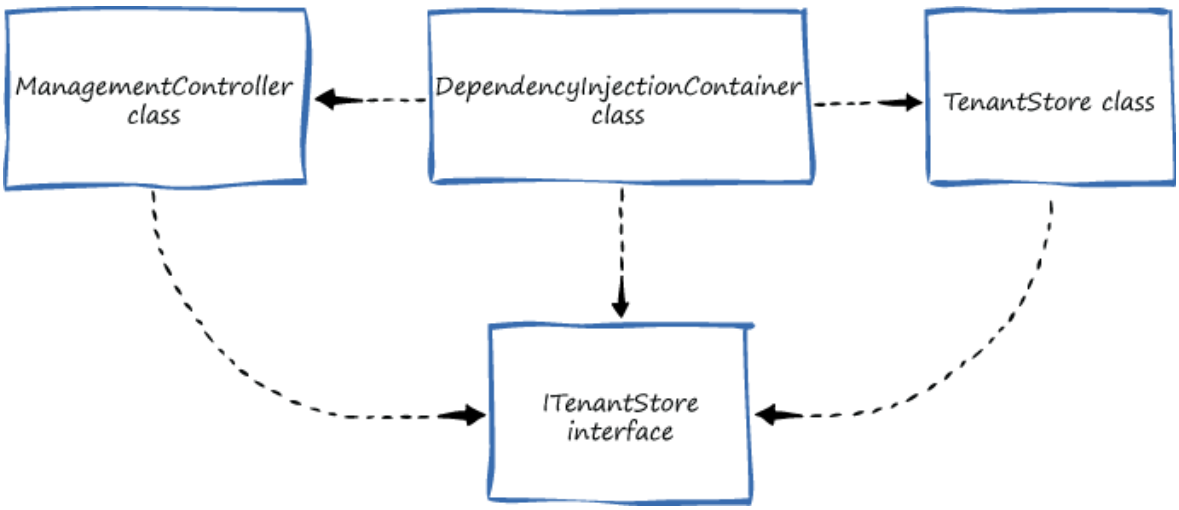
text

1.1 Spring

Spring Framework [2] (dále jen spring) je populární open-source [8] pro vývoj J2EE [3] aplikací. Jádro springu je využívá návrhový vzor IoC (Inversion of Control [4]) a je označován jako IoC kontejner. Tento návrhový vzor funguje na principu přesunutí zodpovědnosti za vytvoření a provázání objektů z aplikace na framework.

Objekty lze získat prostřednictvím Dependency Injection [5] neboli vsazování závislostí (Obr. 1.1). Jedná se o speciální případ IoC. Dependency Injection lze realizovat třemi způsoby

- Setter Injection,
- Constructor Injection,
- Interface Injection.



Obr. 1.1 Náhled možností dependency injection

Vytvářené objekty se nazývají JavaBeans [6]. Ty jsou vytvářeny typicky na základě načtení konfiguračního souboru ve formátu XML [7], který obsahuje definice těchto Beans. Spring se nezabývá řešením již vyřešených problémů. Místo toho využívá prověřených a dobře fungujících existujících open-source nástrojů, které v sobě integruje. Tím se stává jejich použití často jednodušším. Spring je modulární framework. Umožňuje využít jen část, která se zrovna hodí k řešení daného problému. Účelem springu je:

- zjednodušení návrhu J2EE aplikací se zaměřením na architekturu aplikace (místo na technologie),

- jednoduchá testovatelnost,
- neinvazivní rozvoj a modularita.

1.2 Hibernate

Hibernate framework [9] je napsaný v jazyce Java [10], který umožňuje tzv. objektově-relační mapování (ORM [11]). Usnadňuje řešení otázky zachování dat objektů i po ukončení běhu aplikace.Hibernate poskytuje způsob, pomocí něhož je možné zachovat stav objektů mezi dvěma spuštěnými aplikacemi. Říkáme tedy, že udržuje data persistentní. Dosahuje toho pomocí ORM, což znamená, že mapuje Javovské objekty na entity v relační databázi. K tomu používá tzv. mapovací soubory, ve kterých je popsáno, jakým způsobem se mají data z objektu transformovat do databáze a naopak, a jakým způsobem se z databázových tabulek mají vytvořit objekty.

Druhý způsob jak mapovat objekty je použít anotace místo mapovacích souborů. V Hibernate se tedy pracuje s běžnými business objekty, přičemž mohou být sloupce tabulky spojeny přímo s atributy objektu, nebo mohou být připojeny skrze metody get/set a metody hashCode() a equals(). Nutno podotknout, že nelze použít EJB [12] (viz JavaBean), ale pouze klasické objekty - tzv. POJO (Plain Old Java Object) [13]. Poté, co jsou objekty uložené v databázi, se na ně lze dotazovat jazykem HQL (Hibernate Query Language), který je odvozen z SQL [14] a je mu tedy velice podobný.

1.3 Maven

Apache Maven je nástroj pro správu, řízení a automatizaci buildů aplikací. Ačkoliv je možné použít tento nástroj pro projekty psané v různých programovacích jazycích, podporován je převážně jazyk Java. Základním principem fungování Mavenu je popsání projektu pomocí Project Object Model. Tento model popisuje softwarový projekt nejen z pohledu jeho zdrojového kódu, ale včetně závislostí na externích knihovnách, popisu procesu buildování a různých funkcí s tím spojených (jako je spouštění testů, sbírání informací o zdrojových kódech a podobně). Maven sám je postaven na modulární architektuře a funguje na principu volání jednotlivých pluginů. Maven sám pouze obstarává dodání a spuštění nadefinovaných pluginů. Maven nemá žádné vlastní grafické uživatelské rozhraní a běží pouze na příkazové řádce a pluginy tak mohou využívat všechny nástroje, které dokáží komunikovat pomocí standardních vstupů. Project Object Model koncept popisu projektu jako objektu. Za tímto účelem je definovaná jednoduchá XML struktura, která definuje jednotlivé části projektu a jeho závislosti na externích knihovnách a nástrojích. Současně je možné definovat konstanty, které pak mohou využít jednotlivé pluginy. Tento XML dokument se nachází v kořenovém adresáři projektu a je pojmenován pom.xml. Pokud je projekt složen z více dílčích projektů nebo modulů,

každý z nich má pak svůj vlastní pom.xml soubor, který dědí vlastnosti od nadřazeného souboru a může přidávat další položky. Díky této struktuře je pak možné sestavit celý projekt jediným příkazem. V pom.xml je možné u každého projektu nadefinovat jeho závislosti na externích knihovnách. Jednotlivé prvky Artifacts jsou jednoznačně definovány podle atributů <groupId> a <artifactId>. Maven pak automaticky vyhledá a nainstaluje potřebné knihovny. Samotné vyhledávání probíhá v definovaných úložištích (repository). Kromě globální maven repository, která je veřejně přístupná, je možné založit i další soukromá nebo firemní úložiště.

1.4 SOAP

SOAP je protokolem pro výměnu zpráv založených na XML přes síť, hlavně pomocí HTTP. Formát SOAP tvoří základní vrstvu komunikace mezi webovými službami a poskytuje prostředí pro tvorbu složitější komunikace. Existuje několik různých druhů šablon pro komunikaci na protokolu SOAP. Nejznámější z nich je RPC šablona, kde jeden z účastníků komunikace je klient a na druhé straně je server. Server ihned odpovídá na požadavky klienta.

1.4.1 WSDL

WSDL je jazykem pro popis funkcí, jež nabízí tzv. webová služba, a dále pro popis vstupů a výstupů těchto funkcí (jinými slovy, co webová služba poskytuje a jak si o to říci). Jelikož webová služba v principu komunikuje protokolem SOAP, WSDL zpravidla popisuje SOAP komunikaci. WSDL vychází z formátu XML. Podporované operace a zprávy jsou popsány abstraktně, a potom se omezují na konkrétní síťový protokol a formát zprávy. Z toho plyne, že WSDL popisuje veřejné rozhraní webové služby.

1.4.2 SoapUI

SoapUI je open source aplikace pro testování webových služeb pro architektury orientované na služby (SOA) a reprezentace státních přenosů (REST). Její funkce zahrnují kontrolu webových služeb, vyvolání, vývoj, simulaci a výsměch, testování funkčnosti, zatížení a testování shody. Komerční verze SoapUI Pro, která se zaměřuje hlavně na funkce určené ke zvýšení produktivity, byla také vyvinuta softwarem Eviware. V roce 2011 společnost SmartBear Software získala produkt Eviware. SoapUI byl zpočátku propuštěn do společnosti SourceForge v září 2005. Je to svobodný software, licencovaný na základě podmínek veřejné licence Evropské unie. Je postavena výhradně na platformě Java a používá rozhraní Swing pro uživatelské rozhraní. To znamená, že SoapUI je multiplatformní. Dnes SoapUI podporuje i IDEA, Eclipse a NetBeans. SoapUI může testovat webové služby SOAP a REST, JMS, AMF, stejně jako volání HTTP

(S) a JDBC. vyhody - je jednoduše čitelnější pro člověka nevyhody - Velký zápis komunikace. Složitost. Pomalé zpracování jednotlivými systémy (složité na parsovaní a validaci).

1.5 REST

REST je architektura rozhraní, navržená pro distribuované prostředí. REST navrhl a popsal v roce 2000 Roy Fielding (jeden ze spoluautorů protokolu http) v rámci disertační práce Architectural Styles and the Design of Network-based Software Architectures. Rozhraní REST je použitelné pro jednotný a snadný přístup ke zdrojům (resources). Zdrojem mohou být data, stejně jako stavy aplikace (pokud je lze popsat konkrétními daty). REST je tedy na rozdíl od známějších XML-RPC či SOAP, orientován datově, nikoli procedurálně. Všechny zdroje mají vlastní identifikátor URI a REST definuje čtyři základní metody pro přístup k nim, které jsou známé pod označením CRUD, tedy vytvoření dat (Create), získání požadovaných dat (Retrieve), změnu (Update) a smazání (Delete). Tyto metody jsou implementovány pomocí odpovídajících metod HTTP protokolu.

1.5.1 JSON

JSON je způsob zápisu dat (datový formát) nezávislý na počítačové platformě, určený pro přenos dat, která mohou být organizována v polích nebo agregována v objektech. Vstupem je libovolná datová struktura (číslo, řetězec, boolean, objekt nebo z nich složené pole), výstupem je vždy řetězec. Složitost hierarchie vstupní proměnné není teoreticky nijak omezena. Kolekce párů název/hodnota. Ta bývá v rozličných jazycích realizována jako objekt, záznam (record), struktura (struct), slovník (dictionary), hash tabulka, klíčový seznam (keyed list) nebo asociativní pole. Seřazený seznam hodnot. Ten je ve většině jazyků realizován jako pole, vektor, seznam (list) nebo posloupnost (sequence). Jedná se o univerzální datové struktury a v podstatě všechny moderní programovací jazyky je v nějaké formě podporují. Je tedy logické, aby na nich byl založen i na jazyce nezávislý výměnný formát.

- Object (Objekt)je uvozen znakem (levá složená závorka) a zakončen znakem (pravá složená závorka). Každý název je následován znakem : (dvojtečka) a páry název/hodnota jsou pak odděleny znakem , (čárka).
- Array (Pole) je seřazenou kolekcí hodnot. Začíná znakem [(levá hranatá závorka) and končí znakem] (pravá hranatá závorka). Hodnoty jsou odděleny znakem , (čárka).

- Value (Hodnotou) rozumíme řetezec uzavřený do dvojitých uvozovek, číslo, true, false, null, objekt nebo pole. Tyto struktury mohou být vnořovány.
- String (Řetězcem) je nula nebo více znaků kódování Unicode, uzavřených do dvojitých uvozovek a využívající únikových sekvencí (escape sequence) s použitím zpětného lomítka. Znak je reprezentován jako řetězec s jediným znakem. Řetězec je velmi podobný řetězcům z jazyků C nebo Java.
- Number (Číslo) je podobné číslům z jazyků C a Java. Jedinou výjimkou je, že není používán oktalový ani hexadecimální zápis.

1.5.2 Postman

Postman je open source aplikace pro testování webových služeb pro architektury orentované na REST.

1.6 JavaMail

Knihovna pro prijimani a odesilani mailu. Javamail je rozhraní API jazyka Java, které slouží k odesílání a přijímání e-mailů prostřednictvím protokolů SMTP, POP3 a IMAP. JavaMail je integrován do platformy Java EE, ale nabízí i volitelný balíček pro použití v Java SE.

1.7 CSV

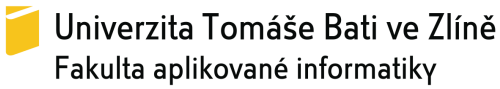
CSV je jednoduchy souborový formát pro výměnu tabulkových dat. Soubor ve formátu CSV sestává z řádků, ve kterých jsou jednotlivé položky odděleny znakem čárka (,). Hodnoty položek mohou být uzavřeny do uvozovek ("), což umožňuje, aby text položky obsahoval čárku. Pokud text položky obsahuje uvozovky, jsou tyto zdvojeny.

1.8 PostGIS

PostGIS je open source software. Jedná se o nadstavbu pro objektově-relační databázový systém PostgreSQL, která přidává podporu pro geografické objekty (tzv. geoprvky). PostGIS implementuje specifikaci „Simple Features for SQL“ konsorcia Open Geospatial Consortium.

2 Další nadpis

Tato sekce obsahuje ukázku vložení obrázku (Obr. 2.1).



Obr. 2.1 Popisek obrázku

2.1 Podnadpis

Tato sekce obsahuje ukázku vložení tabulky (Tab. 2.1).

Tab. 2.1 Popisek tabulky

	1	2	3	4	5	Cena [Kč]
F	(jedna)	(dva)	(tři)	(čtyři)	(pět)	300

2.1.1 Podpodnadpis

2.1.2 Podpodnadpis

Citace knihy. [?]

II. ANALYTICKÁ ČÁST

3 Analýza stávajícího dispečinkového systému

Stavajici system zalozeny na komunikaci pomoci SMS mezi dispecerem a obsluhou vozu (ridic , steward). SMS se parsuji a ukladaji do databaze.

3.1 Podnadpis

text

4 Navrhněte technické řešení pro výpočet zpoždění

- Vytvorime si zakladni trasu spoje pomoci GPS souradnic
- Tuto trasu rozdelime na trasy mezi zastavkami (aby nebyl prilis velky pocet souradnic)
- Pomoci Trasovani spoje (napojeni na polohavozu.cz) ziskame souradnice GPS a cas kdy byli tyto data porizeny
- Vytvorime modelovou trasu spoje s casem
- Porovanáním modelove trasy spoje a aktualni pozice vozu, muzem vypocist zpozdeni
- Vyuzijem PostGIS k vraceni nejblizsi souradnice, u teto souradnice zname i cas prujezdu
- Zpozdeni ulozime do databaze

4.1 Podnadpis

text

III. PROJEKTOVÁ ČÁST

5 Připojení nových zdrojů dat

Soap api, rest api, javaMail, dao vrstva (hibernate), excel, csv , WSDL, JSON

5.1 Připojení SŽDC api

- Tvorba Provideru (inteface, implementace)
- Uprava pom.xml souboru pro pripojeni na soap api (dva endpointy)
- Vytvoreni wsdl factory trid, abstraktnich trid
- Tvorba dao vrstvy pro nacitani zastvaek a spoju podle typu spojeni a doby odjezdu (hibernate)
- Vytvoreni DO trid, datova struktura odpovedi
- Konvertor na tyto DO třídy
- Tvorba service (inteface, implementace)
- Nacitani zpozdeni jednotlivych spoju
- Nacitani nazvu kolej/nastupiste podle doby prijezdu do zastvaky
- Tvorba cron jobu pro tyto servicy

5.1.1 Podnadpis

text

5.2 Připojení Polohavozu.cz

- Tvorba Provideru (inteface, implementace)
- Pomoci Postman ziskame JSON fily ze kterych vytvorime POJO objekty
- dva endpointy
  - seznam vozu a jejich identifikace
  - GPS souradnice vozu obsahujici identifikator
- Konverze na DO objekty
- tvorba cron jobu pro pravidelý update
- Dao ukladani namere polohy do DB

5.2.1 Podnadpis

text

5.3 Načítaní struktury vlaku do dispečinkového systému

- Tvorba Provideru (inteface, implementace)
- Vytvoreni mailu pro odebírání excel souboru se strukturou vlaku rozepsanou na den
- Pripojeni se na tuto mailovou adresu pomocí javamail knihovny
- Nacteni prijatých mailu a odfiltrovani prebytecnych mailu, ktere jsou bud stare nebo neobsahuji prilohu
- Otevreni prilohy a nacteni exceloveho souboru
- Otevreni pracovniho sesitu a nacteni dat z excelových bunek do CSV souboru
- Parsovani CSV souboru a ulozeni dat (konverze dat) do datových prepravek
- Dao funkce (hibenate dotaz do DB)
- Tvorba service (inteface, implementace)
- Pomocí stavajicích funkci ulozíme data do DB
- Finalni uprava a kontrolni mechanismy na sparvnost cislovani vozu
- tvorba cron jobu pro pravidelý update

5.3.1 Podnandpis

text

6 Distribuce zpoždění

Plazmy, GTFS-real time, SMS, mail

6.1 Zobrazení na zpozdeni spoje

- Vyuzit stavajici funkci pro zobrazovani na Plazmach
- Zasilani SMS a mailu o zpozdeni pro jednotlive prodejce a dispecery podle skupin

6.1.1 Podnandpis

text

6.2 Zobrazení údajů o zpoždění

- Tvorba noveho endpontu pro real-time GTFS
- GTFS-RT musi obsahovat stejne identifikatory jako staticke GTFS
- Google Protokolbuffer
- Pripojeni nacitani zpozdeni a poloha GPS

6.2.1 Podnandpis

text

ZÁVĚR

Text závěru

SEZNAM POUŽITÉ LITERATURY

[1] SOAP. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/SOAP`.

[2] *Spring Framework* [online]. Pivotal Software, 2018 [cit. 2018-05-18]. Dostupné z: `https://projects.spring.io/spring-framework/`.

[3] Java EE. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://cs.wikipedia.org/wiki/Java_EE`.

[4] Inversion of control. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Inversion_of_control`.

[5] Dependency injection. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Dependency_injection`.

[6] JavaBeans. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://cs.wikipedia.org/wiki/JavaBeans`.

[7] XML. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/XML`.

[8] Open-source software. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Open-source_software`.

[9] Hibernate (framework). In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Hibernate_(framework)`.

[10] Java (programming language). In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Java_(programming_language)`.

[11] Object-relational mapping. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Object-relational_mapping`.

[12] Enterprise JavaBeans. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Enterprise_JavaBeans`.

[13] Plain old Java object. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/Plain_old_Java_object`.

[14] SQL. In: *Wikipedia: the free encyclopedi*a [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-18]. Dostupné z: `https://en.wikipedia.org/wiki/SQL`.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

API	Application Programming Interface
CSV	Comma Separated Values
EJB	Enterprise JavaBeans
HQL	Hibernate Query Language
IoC	Inversion of Control
J2EE	Java 2 Enterprice Edition
JSON	JavaScript Object Notation
ORM	Object-Relational Mapping
POJO	Plain Old Java Object
REST	Representational State Transfer
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
WSDL	Web Services Description Language
XML	eXtensible Markup Language

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I. NÁZEV PŘÍLOHY

Obsah přílohy