

Razvoj REST servisa korištenjem Oracle REST Data Services

Developing REST service in Oracle REST Data Services

¹Dino Kalamari, ²Miran Kovačić, ³Marko Markulija
¹Međimurje IPC d.d., Mihovljanska 72, 40 000 Čakovec

²Media-Soft d.o.o., Braće Graner 14, 40 000 Čakovec

³Tehnička škola Čakovec, Športska ul. 5, 40 000 Čakovec
e-mail: ¹dino.kalamari@ipc.hr, ²miran.k@live.com, ³marko.markulija@mev.hr

Sažetak: *U suvremenom IT svijetu aplikacije su pisane u različitim programskim jezicima, izvršavaju se na različitim operacijskim sustavima, koriste razne oblike spremanja podataka te je stoga nužno osigurati način da one na jednostavan način mogu razmjenjivati podatke. Ovo posebno dolazi do izražaja u poslovnim okolinama u kojima se aplikativni sustav može sastojati od mnoštva mikroservisa koji svaki ima specifičnu namjenu te oni međusobno moraju razmjenjivati podatke. Jedan od najvećih ponuđača rješenja za spremanje podataka u bazu podataka je Oracle koji nudi mnoštvo alata za izradu aplikacija i sustava temeljenih na njihovim tehnologijama. Jedan od njihovih proizvoda je Oracle REST Data Services (ORDS) koji nudi jednostavan, lagan i brz način za izradu programskih sučelja prema podacima spremljenim u njihovim bazama podataka, koje je uz to i jako robusno, sigurno i skalabilno.*

Ključne riječi: *Oracle, REST, web servisi, Oracle REST Data Services*

Abstract: *In modern IT world applications are written in different programming languages, are ran on various operating systems, use various forms of data storage so it is important to provide a way to data exchange data between them. This is very crucial in business environments where system is made from a bunch of microservices which need to exchange data. One of the biggest database providers, Oracle, provides entire set of tools for developing applications based on their data ecosystem. One of their product is Oracle REST*

Data Services (ORDS) whose purpose is to make web service in a simple, easy and a quick way and that at the same time this solution will be robust, secure and scalable.

Key words: Oracle, REST, web services, Oracle REST Data Services

1. Uvod

Suvremene aplikacije pisane su na različitim platformama i različitim programskim jezicima, izvršavaju se na raznim arhitekturama računala i različitim operacijskim sustavima, imaju različite interne logike i strukture podataka. Da bi osigurali interoperabilnost između tih svih aplikacija potreban je zajednički “jezik” koji će biti neovisan o platformi, programskom jeziku i operacijskom sustavu, a koji će sve aplikacije razumjeti. Taj jezik su web-servisi, a njihov cilj je da stvore distribuiranu okolinu u kojem aplikacije mogu na jednostavan način razmjenjivati podatke, to jest da budu interoperabilne.

Kod web-servisa postoje 2 sudionika: server i klijent, te se komunikacija odvija na sljedeći način: klijent šalje poruku zahtjeva (engl. *request*) serveru, koju on obradi i korisniku pošalje status te obrade, tj. poruku odgovora (engl. *response*). Klijentu se pošalje specifikacija sučelja u obliku WSDL (engl. *Web service description language*) XML datoteke, ukoliko se radi o SOAP web servisu, ili OpenAPI specifikacija ukoliko se radi o RESTful servisu. Te specifikacije definiraju formate i strukturu poruka zahtjeva i odgovora, tipove podataka, URL metoda. Takve specifikacije pojednostavljuje doradu klijentske aplikacije jer razvojne okoline na temelju nje mogu generirati programski kôd potreban za komunikaciju s tim servisom te programsku dokumentaciju.

2. RESTful servisi

REST (engl. *Representational state transfer*) je koncept softverske arhitekture koji definira skup pravila za izradu web-servisa. Podaci i funkcionalnosti smatraju se resursima i pristupa im se kroz adresu, tj. URI (engl. *Uniform Resource Identifier*). Za svaki resurs se definiraju metode kojima se mu može pristupiti i raditi određene operacije nad njime. Sudionici komunikacije obično razmjenjuju te resurse koristeći standardizirana sučelja i protokol, a većinom je to HTTP (engl. *HyperText Transfer Protocol*). HTTP je je protokol za komunikaciju hipertekstualnim dokumentima između servera i klijenta putem lokalne mreže ili interneta. Svaki resurs može imati više reprezentacija, što znači da se može razmjenjivati u raznim formatima podataka: HTML (engl. *HyperText Markup Language*), XML (engl.

eXtensible Markup Language), PDF (engl. *Portable Document Format*), JSON (engl. *JavaScript Object Notation*) i drugima, sve ovisno o potrebama klijenta. REST servis je *stateless* jer nema čuvanja podataka o konekciji između klijenta i servera na serveru već su svi podatci za ispravno tumačenje sesije na klijentu [1].

Metapodatci u porukama zaglavlja mogu se koristiti za kontrolu memorije, otkrivanje grešaka u komunikaciji, dogovaranje formata podataka, autentifikaciju i ograničenje pristupa [2].

RESTful koristi 4 osnovne HTTP metode koje odgovaraju CRUD operacijama (engl. *Create, Read, Update, Delete*) [3]:

- HTTP GET - dohvaćanje reprezentacije resursa
- HTTP POST - kreiranje novoga resursa
- HTTP PUT - ažuriranje postojećega resursa
- HTTP DELETE - brisanje postojećega resursa

Prilikom komunikacije i korištenja REST servisa moguća su razna stanja i statusi uspješnosti izvršene operacije. Tako HTTP statusni kod 200 označava da je zahtjev uspješno izvršen, dok npr. statusni kod 404 daje informaciju da traženi URL ne postoji. Kod POST i PUT metoda podatci su spremljeni u tijelu HTTP poruke dok se kod GET i DELETE metoda podaci zahtjeva šalju kao atributi unutar URL-a. Osim tijela poruke i URL-a, podatci se mogu slati i unutar HTTP zaglavlja. Format podataka u kojem očekujemo odgovor od servera možemo definirati slanjem *Accept* retka unutar HTTP zaglavlja s formatom u kojem se očekuje odgovor, npr: *application/JSON* ili *application/XML* [2].

3. Oracle Rest Data Services (ORDS)

Oracle je godinama u samom vrhu proizvođača baza podataka. Kako bi korisnici izradili aplikacije temeljene na njihovim bazama podataka nude razvojne okoline u kojima se programira na njihovom vlastitom PL/SQL jeziku. 90tih godina su to bile Oracle Forms, sve većim razvojem interneta i prebacivanjem aplikacija u web nastale su Web Forms i Application server, dok je zadnjih par godina njihov najveći fokus na razvoju alata pod nazivom Oracle APEX (Application Express) i ORDS. Dok je APEX namijenjen za izradu interaktivnih aplikacija nad Oracle bazom podataka koji prati najnovije trendove, ORDS je zamišljen kao alat za izradu REST web servisa nad podacima u bazi podataka.

ORDS je elegantan, jednostavan i brz način za izradu REST sučelja do podataka u Oracle bazi podataka. Osim pristupa podacima, ORDS se može koristiti i za pozivanje procedura i

funkcija ili bilo kojega validnoga SQL ili PL/SQL bloka naredbi. To je idealan alat ako na Oracle bazi podataka trebamo razvijati web-servise koji će biti konzumirani od strane raznih klijenata, npr: od Android mobilnih aplikacija, React web aplikacija, do Microsoft WPF desktop aplikacija. Na ovaj način ostajemo u Oracle razvojnoj okolini i ne trebamo posezati za nekim drugim programskim jezicima tipa Java, C#, Python ili NodeJS. Ogromna prednost ORDS je što programeri koji imaju iskustva u PL/SQL mogu vrlo brzo naučiti izrađivati web-servise jer se programira u PL/SQL programskom jeziku. Na ovaj način možemo pomoću ORDS izraditi web-aplikaciju koristeći samo najjednostavniji JS framework kao što je jQuery i HTML, dok će sva logika vezana za obradu podataka biti u ORDS servisu.

ORDS je u suštini Java aplikacija bazirana na Java EE tehnologiji. Dostupna je uz bilo koju Oracle Database vrstu licence, uključujući i besplatnu Oracle XE bazu. Za hosting možemo odabrati Oracle WebLogic, Apache Tomcat te GlassFish servere ili samostalni način rada aplikacije koji nudi slabije mogućnosti konfiguriranja od aplikacijskih servera. ORDS je vezan za Oracle bazu podataka te stoga ne može biti namijenjen za izvršavanja nad drugim bazama podataka, kao što su Microsoft SQL ili MySQL [4].

Minimalni preduvjeti za instalaciju ORDS su [4]:

- Oracle Database 10g Release 2 ili novija
- Oracle APEX (Application Express) za kreiranje i spremanje ORDS servisa
- Java 6 Update 20 ili novija

Instalacija ORDS odvija se na način da se s službenih Oracle stranica preuzme ZIP datoteka, otpakira njen sadržaj i instalira koristeći naredbeni redak prema priloženim detaljnim uputama. Osim instalacije, moguće je preuzeti i već unaprijed konfiguriran virtualni *image* koji se pokreće putem Oracle VM Virtual Box aplikacije. Ovdje je bitno napomenuti da je ovo namijenjeno samo za testiranje aplikacija i servisa, nikako za produkciju.

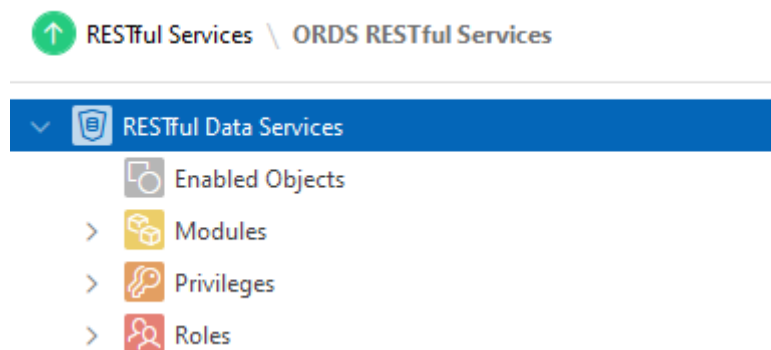
Razvoj ORDS servisa moguć je kroz APEX web-sučelje ili kroz Oracle SQL Developer. U ovom radu za prikaz razvoja biti će korišten APEX.

Prvi korak je registracija sheme, tj. davanje dozvola da neka shema iz baze podataka bude dostupna kroz REST. Na slici 1 je prikazana struktura ORDS aplikacije koja se sastoji od 4 područja:

- *Enabled objects* – Pregled entiteta baze koji su dostupni kroz REST putem AutoREST funkcije. AutoREST je funkcionalnost koja omogućuje da u par klikova bazne tablice, poglede (engl. *viewe*), pakete, procedure i funkcije učinimo dostupni prema van. Iako

je ovaj postupak najbrži on ima ograničenja u pogledu fleksibilnosti i daljnjih prilagodbi [5].

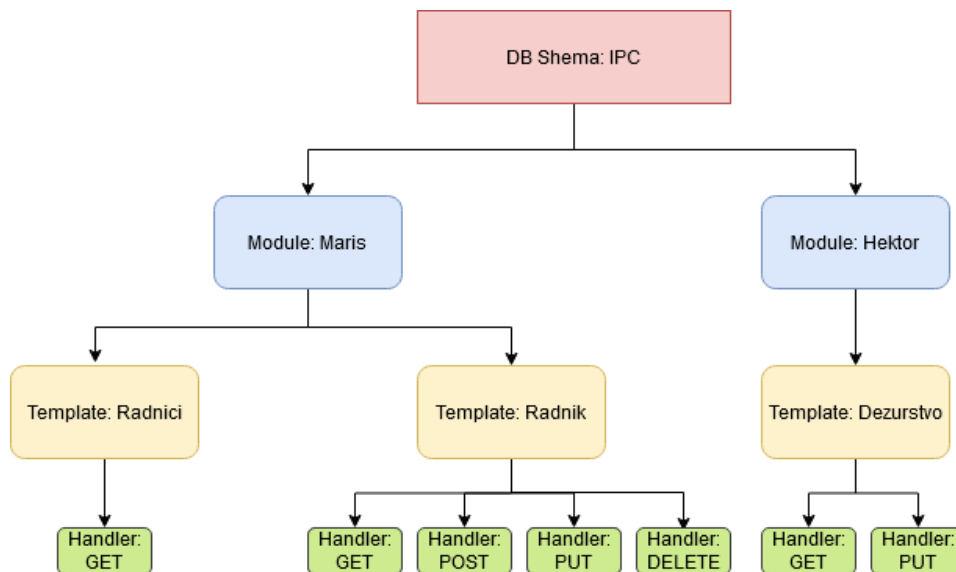
- *Modules* - Servis nad nekom shemom dijeli se u manje cjeline nazvane moduli. Moduli nam omogućavaju da grupiramo metode servisa prema njihovoj namjeni, pa tako možemo imati modul Zaposlenici, Vozila, Radni nalozi, dok neki drugi predlažu da se naziv definira prema verziji servisa, npr v1, v2 itd [6].
- *Privileges* – Privilegije omogućuju definiranje restrikcija do pojedinih modula ili metoda preko URL uzoraka. URL uzorak je dio punog URL-a servisa, npr. ako želimo zaštititi sve metode unutar modula maris, onda to definiramo URL uzorkom /maris/*. Privilegija važi za korisničke role koje definiramo [7].
- *Roles* – Pregled dostupnih rola. Rola služi za definiranje tipa korisnika, a neke od predefiniраниh rola su: SQL developer, OAuth2 Client Developer, Schema Administrator... [7]. Ukoliko je odabir popularan OAuth2 tip autentifikacije, onda se prije prvog slanja poziva mora odraditi postupak autentifikacije pri čemu ORDS izdaje autorizacijski token kojeg klijent šalje u svakom pozivu metoda modula unutar zaglavlja HTTP poruke. Token se izdaje na trajanje od 1 sat nakon čega je ponovno potrebno provesti postupak autentifikacije [8].



Slika 1: Segmenti ORDS aplikacije

Generiranje API dokumentacije za slanje korisnicima moguće je kroz generiranje Swagger dokumenta u JSON obliku koji sadrži opis dostupnih metoda i format podataka koji se razmjenjuju kroz servis na temelju kojeg se mogu generirati klijentske aplikacije.

4. Izrada modula i metoda



Slika 2: Struktura ORDS servisa

Na slici 2 je prikazan primjer strukture ORDS servisa. Sastoji se od 4 razina [6]:

- Shema nad čijim podacima se izvršava servis
- Moduli kao struktura kojom grupiramo metode nad resursima
- *Template* ili predložak je resurs za koji nudimo servis
- *Resource Handler* je najniža razina u kojoj HTTP metodu uparujemo s određenom poslovnom logikom nad resursom

Iz strukture servisa dolazimo do URL servisa u obliku `/module/shema/template`.

Moduli nisu REST koncept već su korisna struktura za grupiranje metoda nad resursima. Kod kreiranja definiše se njegov naziv, je li dostupan prema van, veličina straničenja, tj. broj redaka u rezultatima poziva te komentar. Uz naziv modula, u URL ulazi i shema nad kojom je kreiran modul pa tako za modul *maris* nad shemom *ipc* dobivamo URL: `http://10.10.1.7:8080/maris/ipc/`. Modul se može sastojati od više predložaka [6].

ORDS Module Definition

* Module Name ?

* Base Path ?

Full URL ?

* Is Published Yes No ?

* Pagination Size ?

Origins Allowed ?

Comments

Slika 3: Definiranje modula

Nakon definiranja modula prelazimo na nižu razinu, definiranje predloška (engl. *template*). Pod pojmom predložak možemo smatrati resurs, npr. zaposlenik, poslovni partner, račun. Predložak je zadnji dio u URL-u te je preporuka da to bude naziv resursa, tj. imenica a ne glagol kao što je primjerice *AzurirajRadnika*. Primjer URL-a za dohvat radnika: `http://10.10.1.7:8080/maris/ipc/radnici`.

Preko *Resource handlera* definiramo operaciju nad resursom prema određenoj HTTP metodi. Na slici 4 prikazano je da poziv HTTP GET naredbe izvršava SQL upit koji vraća šifru, prezime i ime radnika. Kao što je već prije spomenuto, GET naredbu koristimo za izvršavanje upita i vraćanje rezultata klijentu, POST naredbu za kreiranje novoga resursa (npr. radnika), PUT za ažuriranje postojećega resursa dok DELETE koristimo za brisanje određenoga resursa. Pod *Source Type* možemo definirati u kojem obliku nam metoda vraća rezultat, da li listu objekata, listu objekata gdje svaki element ima URL na detalje toga elementa, samo jedan objekt, PL/SQL kôd ili medijski resurs. Od formata zapisa možemo birati JSON ili CSV format. Ukoliko je polje *Pagination Size* ostavljeno prazno onda tu postavku uzima s postavki modula.

ORDS Handler Definition

RESTful Service Module **maris**

Module Base Path **/**

URI Template **radnici**

Full URL **http://10.10.1.7:8080/maris/tpc/radnici**

* Method **GET**

* Source Type **Collection Query**

Format **JSON**

Pagination Size

Comments

Source

```
1 SELECT sifra, prezime, ime FROM radnici ORDER BY prezime, ime
```

Slika 4: GET HTTP metoda

Pri dnu forme za kreiranje *Resource handlera* (slika 5) nalazi se tablica za definiranje ulaznih i izlaznih parametara.

Parameters

Search: All Text Columns Go Actions Edit Add Row Reset

<input type="checkbox"/>	☰	Name	Bind Variable	Access Method	Source Type	Data Type	Comments
<input checked="" type="checkbox"/>	☰	StatusMsg	statusMessage	OUT	RESPONSE	STRING	-
<input type="checkbox"/>	☰	RadnSifra	RadnSifra	IN	URI	STRING	Šifra radnika
<input type="checkbox"/>	☰	Lozinka	Lozinka	IN	URI	STRING	Lozinka
<input type="checkbox"/>	☰	X-APEX-STATUS-CODE	HttpStatus	OUT	HTTP HEADER	INTEGER	-

Slika 5: Resource handler parametri

Svaki parametar ima svoj naziv, naziv varijable u koji se preslikava za korištenje u SQL ili PL/SQL bloku naredbi, vrstu je li ulazni ili izlazni parametar. U koloni *Source type* definiramo gdje će se nalaziti vrijednost parametra. Za ulazne parametre ona može biti u URL-u ili zaglavlju HTTP poruke, dok za izlazne parametre ona može biti u tijelu ili zaglavlju HTTP poruke. Ovdje je bitno napomenuti da se parametri za POST metodu mogu slati i unutar tijela HTTP poruke, ali se onda oni dohvaćaju ručno unutar PL/SQL bloka. Svaki parametar može biti u jednom od sljedećih primitivnih tipova podataka: *string*, *number*,

long, double, boolean, timestamp i resultset. Ukoliko radimo običan SQL blok-naredbi onda ne moramo specificirati izlazne parametre, npr. šifra, prezime, ime već će ORDS sam pretvoriti rezultate SQL upita u JSON listu objekata. Izlazne parametre definiramo prije svega ako radimo PL/SQL blok-naredbi koje sadrže više manipulacija podacima pa onda putem tih parametara korisnika obavještavamo o statusu obrade. Za slanje HTTP statusnoga koda koristimo predefimirani izlazni parametar X-APEX-STATUS-CODE.

5. Zaključak

Gotovo sve aplikacije u svojem radu rade s podacima iz nekog izvora podataka, a obično su ti podatci spremljeni u bazama podataka. Ima više načina na koji te podatke možemo učiniti dostupnima izvana a jedan od popularnijih je putem RESTful web-servisa. Ukoliko koristimo Oracle bazu podataka ORDS je idealan alat za to jer programeri upoznati s PL/SQL jezikom će na jednostavan način moći kreirati REST servise putem APEX sučelja, a uz to će to rješenje biti optimizirano, pouzdano, robusno i sigurno.

Literatura

1. Oracle SQL Workshop Guide, <https://docs.oracle.com/en/database/oracle/application-express/19.1/aeutl/what-is-REST.html> (19.08.2020)
2. Dino Kalamari, (2012) Web servisi primjena i implementacija - diplomski rad
3. About Oracle RESTful services,
<https://docs.oracle.com/en/database/oracle/application-express/19.1/aeutl/about-Oracle-RESTful-services.html> (26.08.2020)
4. Installing Oracle REST Data Services,
https://docs.oracle.com/cd/E37099_01/doc.20/e25066/install.htm (26.08.2020)
5. How to enable REST objects from object browser,
<https://docs.oracle.com/en/database/oracle/application-express/19.1/aeutl/how-to-enable-REST-objects-from-object-browser.html> (29.08.2020)
6. How Oracle ORDS RESTful Services are Structured,
<https://www.jmjcloud.com/blog/how-oracle-ords-restful-services-are-structured> (26.08.2020)
7. Using Roles and Privileges to fine tune access to ORDS Web Services,
<https://www.jmjcloud.com/blog/using-roles-and-privileges-to-fine-tune-access-to-ords-web-services> (29.08.2020)

8. ORDS - Securing Services Using OAuth2 (Client Credentials),
<https://www.jmjcloud.com/blog/ords-securing-services-using-oauth2-2-legged>
(29.08.2020)