SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Aplikačné architektúry softvérových systémov

Retailový e-shop telekomunikačného operátora

Semestrálne zadanie

**Autori:** Bc. Michal Farkaš

Bc. Matej Kováč

Bc. Jozef Zaťko

**Vypracované:** letný semester 2015/2016

# Úvod

Cieľom tejto práce je návrh architektúry informačného systému pre retailový e-shop telekomunikačného operátora. Práca sa venuje architektonickým pohľadom Rozanského a Woodsa. Modely sú vytvorené prevažne v jazyku UML. Práca obsahuje opisy týchto modelov a použitých architektonických princípov.

# Požiadavky na systém

Telekomunikačný operátor má svoj e-shop, cez ktorý predáva svoje mobilné služby, internetové služby a káblovú televíziu. Tieto zahŕňajú aj predaj zariadení k týmto službám: mobilných telefónov rôznych typov, notebookov, routerov, set-top boxov a iných zariadení.

Požiadavkou na retailový e-shop je, aby poskytoval webové rozhranie a používateľské obrazovky boli vysoko interaktívne. Z uvedeného je využitý v návrhu architektúry JavaScript MVC framework AngularJS.

## Funkcionálne požiadavky

* štandardná funkcionalita „nákupného košíka“
* objednanie tovaru vrátane potvrdenia
* platba za tovar vrátane potvrdenia
* predaj paušálov vrátane súčastí:
  + volania v rámci siete toho istého operátora
  + volania do iných sietí
  + SMS/MMS
  + internet v mobile
  + volanie blízkym osobám
  + paušál na zahraničné volania (do vybraných krajín)
  + poistenie telefónu
* predaj internetových balíkov
  + podľa objemu dát
  + podľa rýchlosti sťahovania/odosielania
* predaj rôznych typov balíkov televíznych programov
* poskytovanie zľavnených cien podľa balíkov služieb

## Nefunkcionálne požiadavky

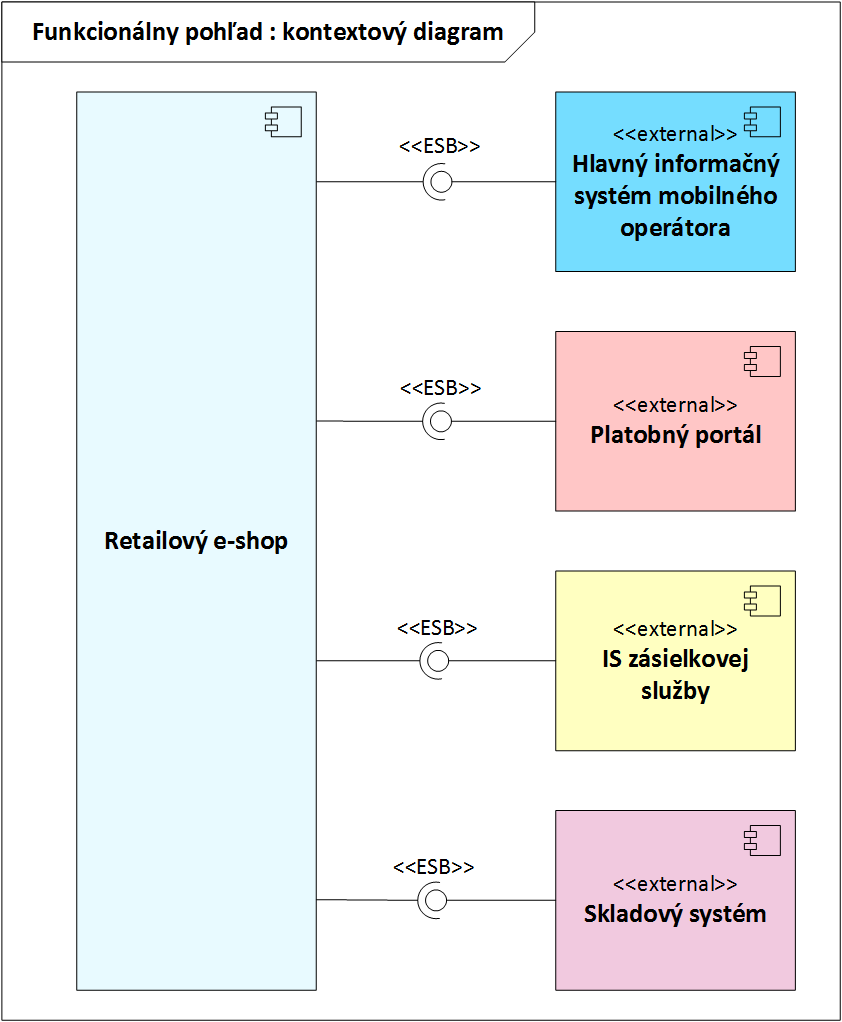
* Spoľahlivosť
* Bezpečnosť
  + identifikácia a autentifikácia klienta
  + bezpečný platobný portál
* Dostupnosť
  + zálohovanie
  + replikácia dát
  + vyrovnávanie záťaže
* Škálovateľnosť a výkonnosť
* Modifikovateľnosť
* Prepojiteľnosť s externými systémami
* Použiteľnosť webového rozhrania

# Funkcionálny pohľad

Opisuje najdôležitejšie elementy systému z pohľadu poskytovanej funkcionality - ich zodpovednosť, rozhrania a vzájomnú interakciu.

## 1. úroveň: Kontextový diagram

Opisuje systém ako celok a jeho spoluprácu s externými systémami.



Obrázok : Funkcionálny pohľad: kontextový diagram

### Opis externých komponentov

**Hlavný informačný systém mobilného operátora** - poskytuje informácie o klientoch, ich zriadených službách

**Platobný portál** - zabezpečuje elektronický spôsob zaplatenia za zakúpené produkty, okrem post-paid služieb (paušálov)

**IS zásielkovej služby** - poskytuje najmä informácie o aktuálnom stave zásielky v prípade, že si zákazník objedná niektoré zo zariadení, ktoré mu je doručované zásielkovou službou

**Skladový systém** - slúži pre evidenciu tovaru, najmä množstva a dostupnosti jednotlivých zariadení

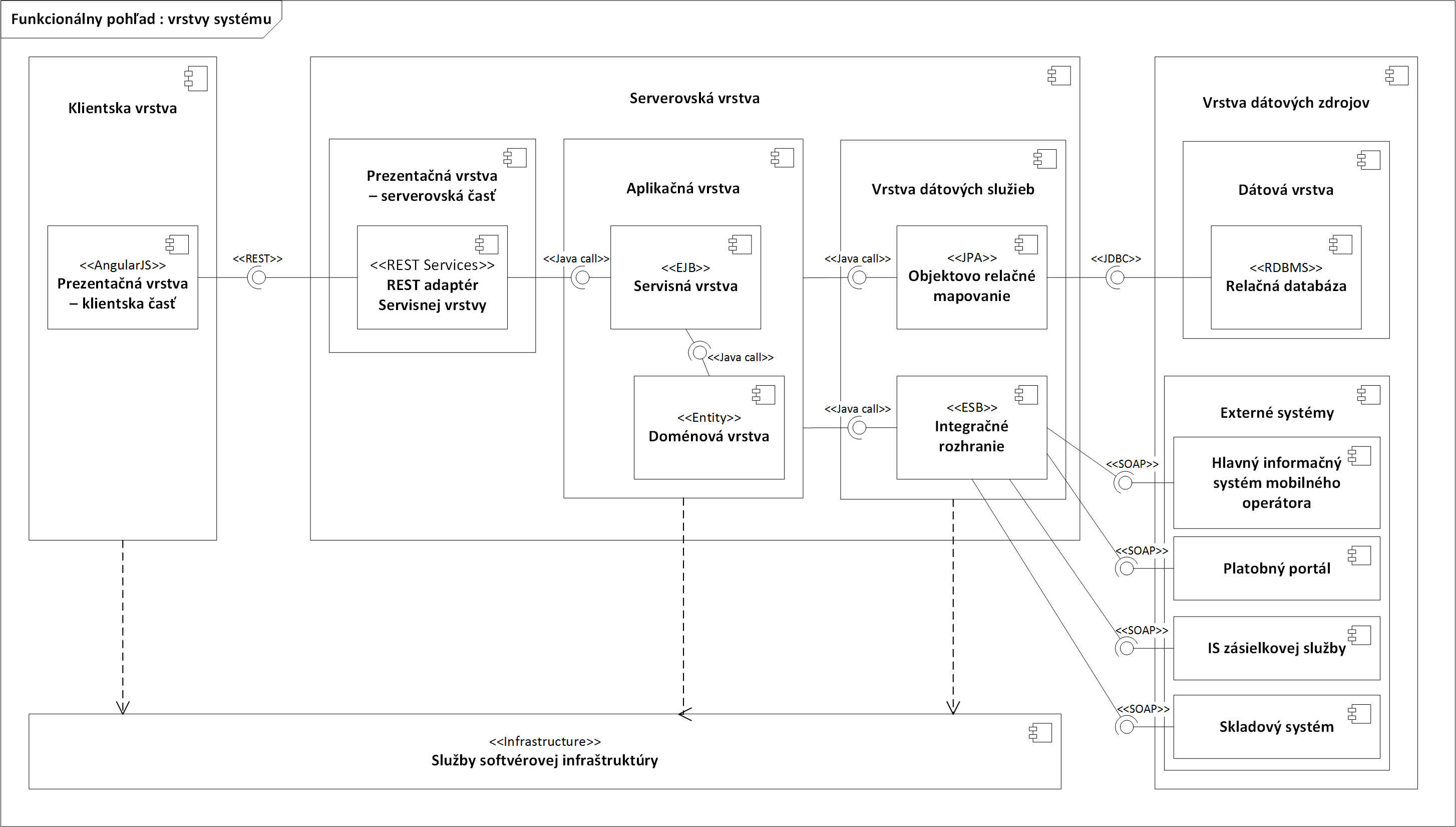
### Použité architektonické princípy

**ESB (Enterprise Service Bus)** - je model softvérovej architektúry využívaný pri navrhovaní a realizácii komunikácie medzi vzájomne sa ovplyvňujúcimi softvérovými aplikáciami v architektúre orientovanej na služby (SOA). Predstavuje zbernicu rôznych služieb, t.j. softvérový architektonický model pre distribuovanú komunikáciu medzi aplikáciami. Jeho primárnym použitím je jednotná integrácia viacerých podnikových aplikácií s odlišným prostredím.

## 2. úroveň: Vrstvy systému

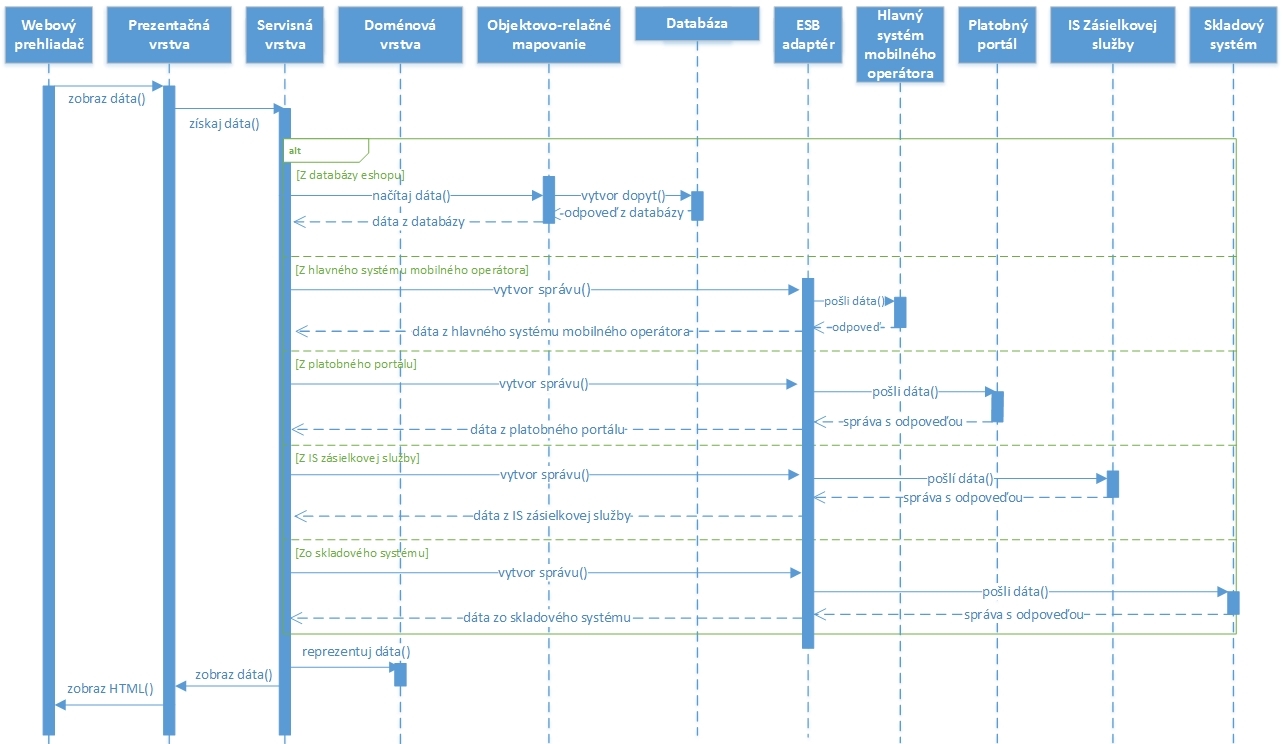
Opisuje systém z pohľadu architektonického štýlu vrstvy.

### Funkcionálny pohľad: vrstvy systému



Obrázok : Funkcionálny pohľad vrstvy systému

### Funkcionálny pohľad: komunikácia vrstiev



Obrázok : Funkcionálny pohľad: komunikácia vrstiev

### Zodpovednosti jednotlivých vrstiev

**Prezentačná vrstva** - Zodpovedá za zobrazovanie dát používateľovi, poskytuje komponenty používateľského rozhrania. Slúži ako vstupný bod pre komunikáciu používateľa so systémom. Delí sa na klientsku časť (JavaScript, AngularJS) vykonávanú vo webovom prehliadači a na serverovskú časť (Servlet).

**Aplikačná vrstva** - Zabezpečuje spracovávanie požiadaviek používateľa. Obsahuje aplikačnú logiku.

**Vrstva dátových služieb** - Zabezpečuje prepojenie s vrstvou dátových zdrojov.

**Vrstva dátových zdrojov** - Zabezpečuje prístup k relačnej databáze. Zodpovedá za vkladanie, výber, aktualizácie a odstraňovanie dát z databázy.

### Použité architektonické princípy

**Architektonický štýl vrstvy** - Tento architektonický štýl rozdeľuje systém na vrstvy, ktoré sú špecifikované na základe úrovní abstrakcie. Vrstva predstavuje zoskupenie elementov, kde elementy s príbuznou funkcionalitou sa nachádzajú v spoločnej vrstve. Používanie vrstiev podporuje oddeľovanie záležitostí a umožňuje tak lepšiu pružnosť a udržiavateľnosť systému.

**JavaScript MVC** - Model-Pohľad-Ovládač (angl. Model-View-Controller) je architektonický štýl, ktorý sa používa pri architektúre prezentačnej vrstvy systému. Pozostáva z troch častí: model, ktorý je zodpovedný za údržbu dát, pohľad zodpovedný za zobrazovanie dát používateľovi a ovládač, ktorý má na starosti komunikáciu medzi modelom a pohľadom.

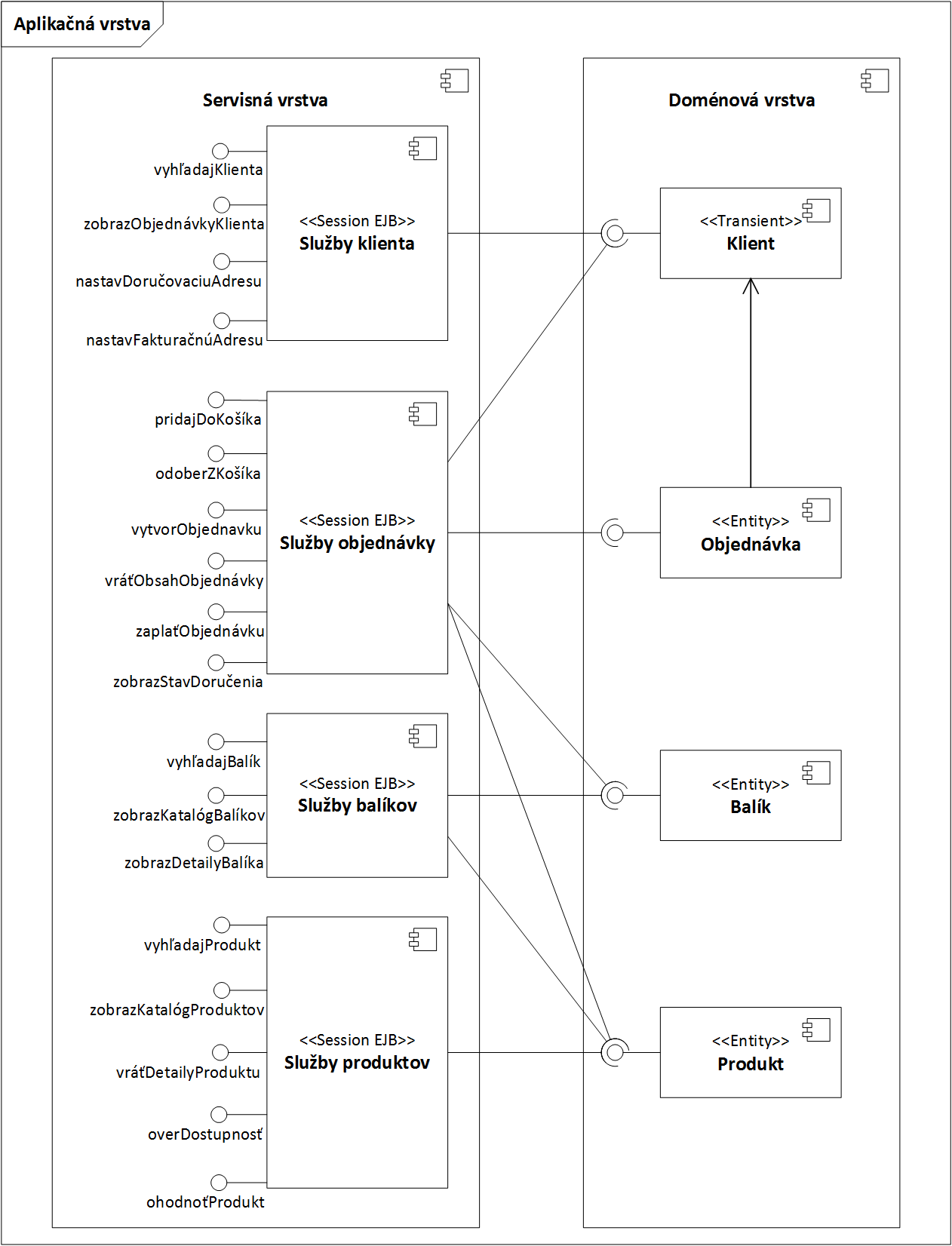
**REST (representational state transfer)** - REST je architektonický štýl založený na rozhraniach poskytujúcich služby, ktoré je možné zavolať jednoduchou HTTP požiadavkou.

**SOAP (Simple Object Access Protocol)** - SOAP je protokol slúžiaci pre výmenou štruktúrovaných informácií s využitím implementácie webových služieb. Pre výmenu dát používa formát XML a protokoly aplikačnej vrstvy ako HTTP, prípadne SMTP.

**JPA (Java Persistence API**) - Je špecifikácia rozhraní pre správu dát v relačných databázach pre platformu Java. Predstavuje iba špecifikáciu a tým pádom pre použitie vyžaduje použitie implementácie, napríklad Hibernate, EclipseLink, TopLink a iné.

## 3. úroveň: Komponenty aplikačnej vrstvy

Zobrazuje členenie aplikačnej vrstvy na servisnú a doménovú vrstvu a komponenty v nich obsiahnuté.



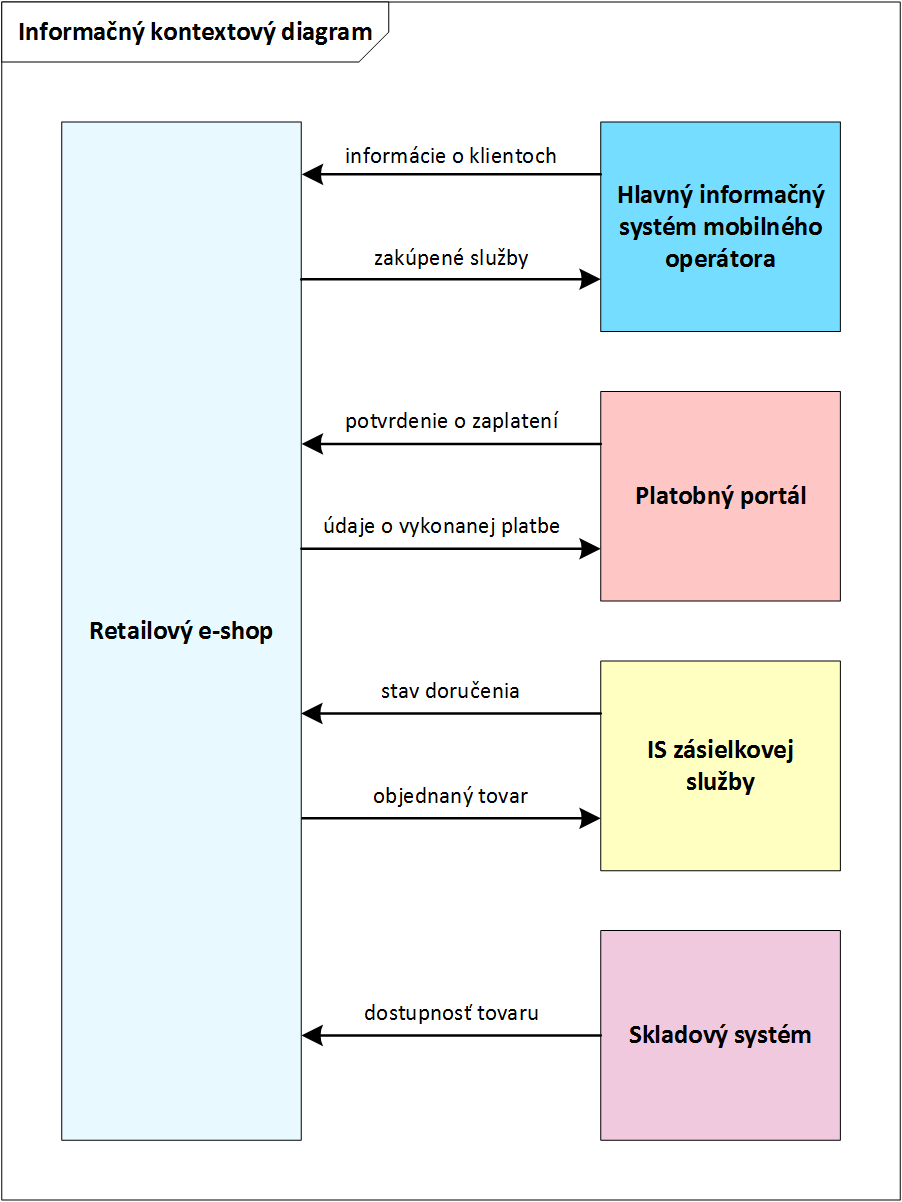
Obrázok : Funkcionálny pohľad: komponenty aplikačnej vrstvy

# Informačný pohľad

Špecifikuje štruktúru informácií systému, spôsob ich ukladania, manipulácie, správy a distribúcie.

## Kontextový diagram toku dát

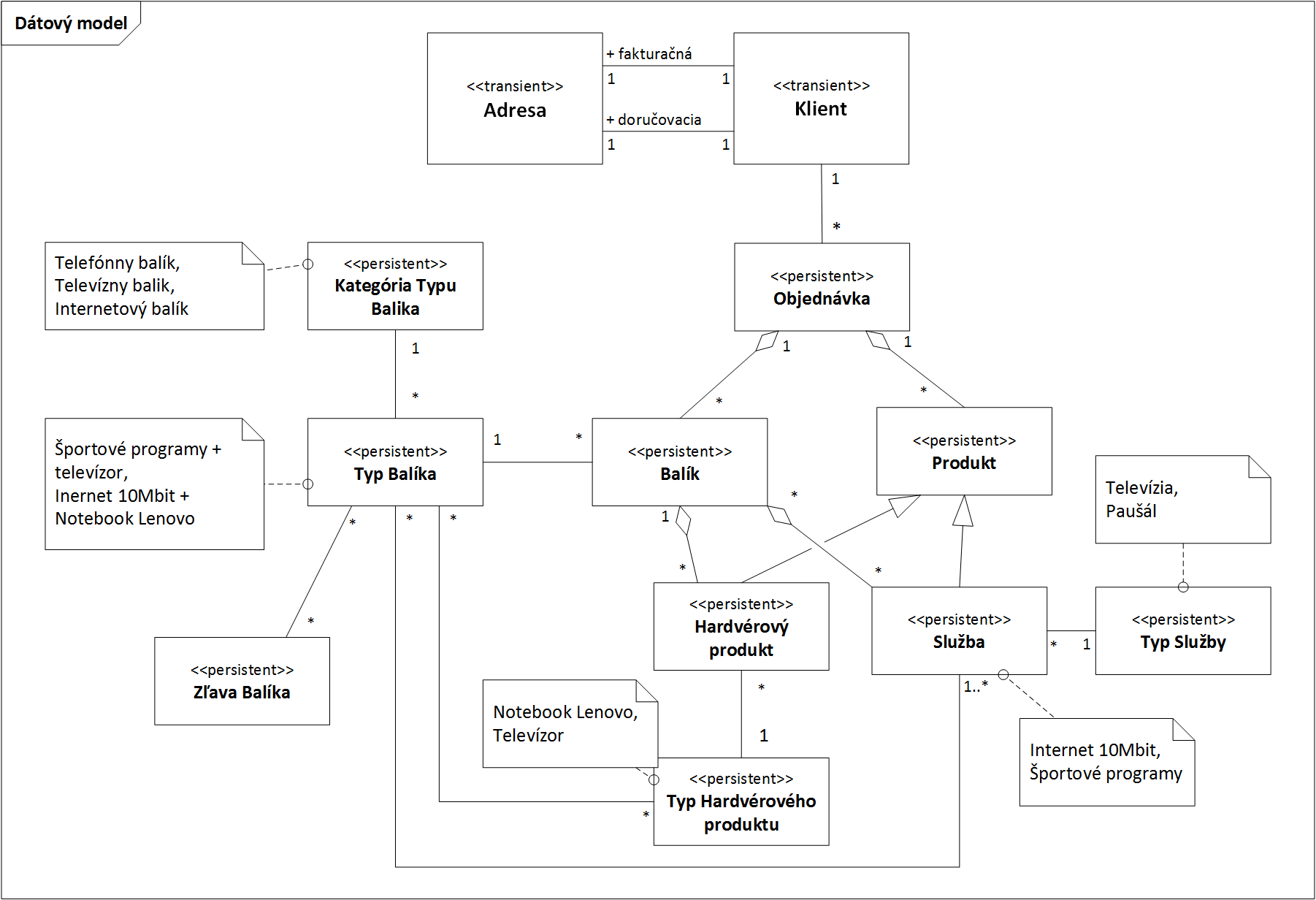
Ukazuje typy dát, ktoré prúdia medzi systémom a jeho externými systémami



Obrázok : Informačný pohľad: kontextový diagram toku dát

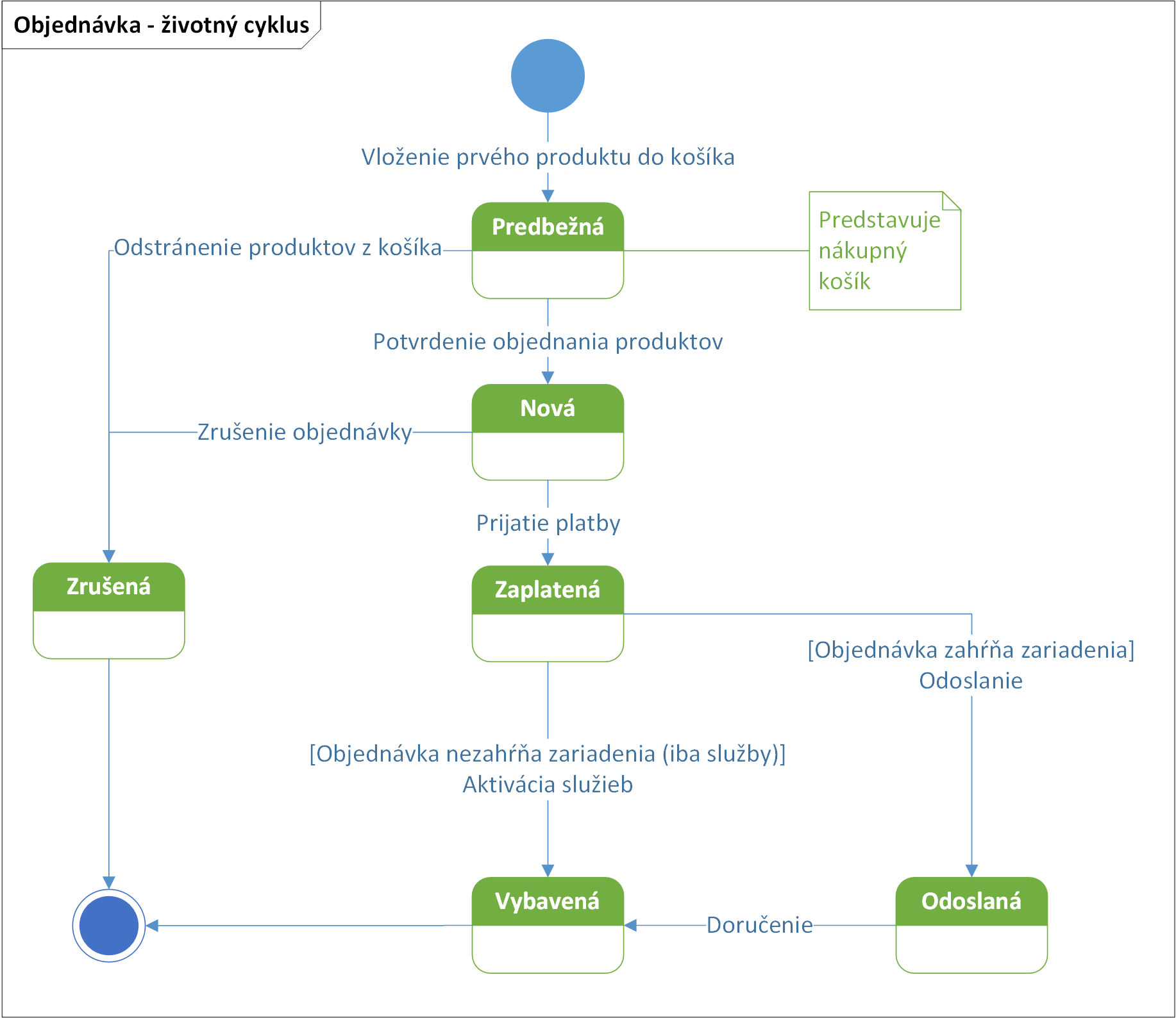
## Dátový model

Modeluje štruktúru dát v e-shope mobilného operátora.



Obrázok : Informačný pohľad: dátový model

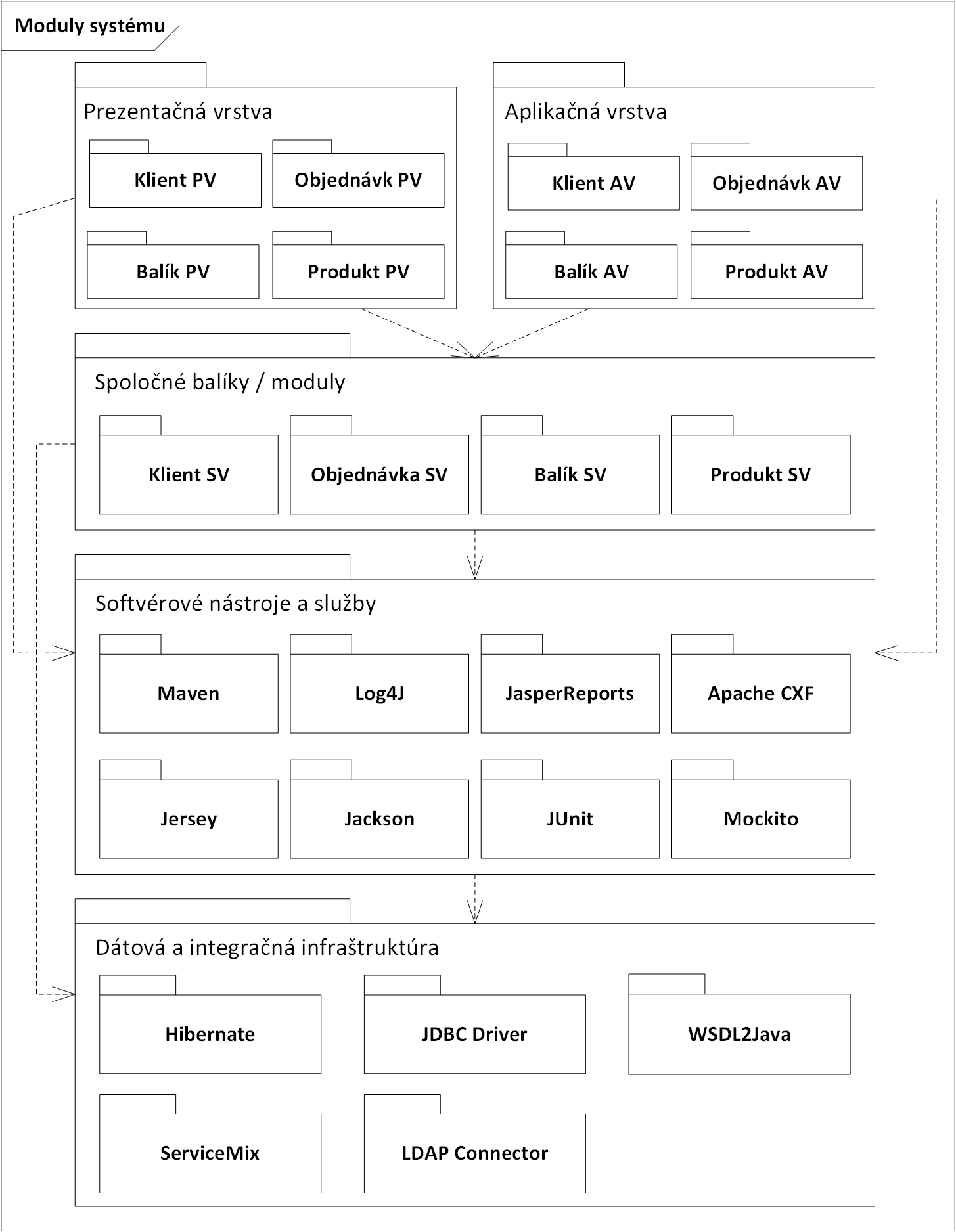
## Model životného cyklu: Objednávka



Obrázok : Model životného cyklu: Objednávka

# Vývojový pohľad

Zobrazuje základné moduly systému - programový kód, ich štruktúru, vzájomné vzťahy a závislosti. Zahŕňa potrebné softvérové nástroje a softvérové súčasti základnej infraštruktúry.

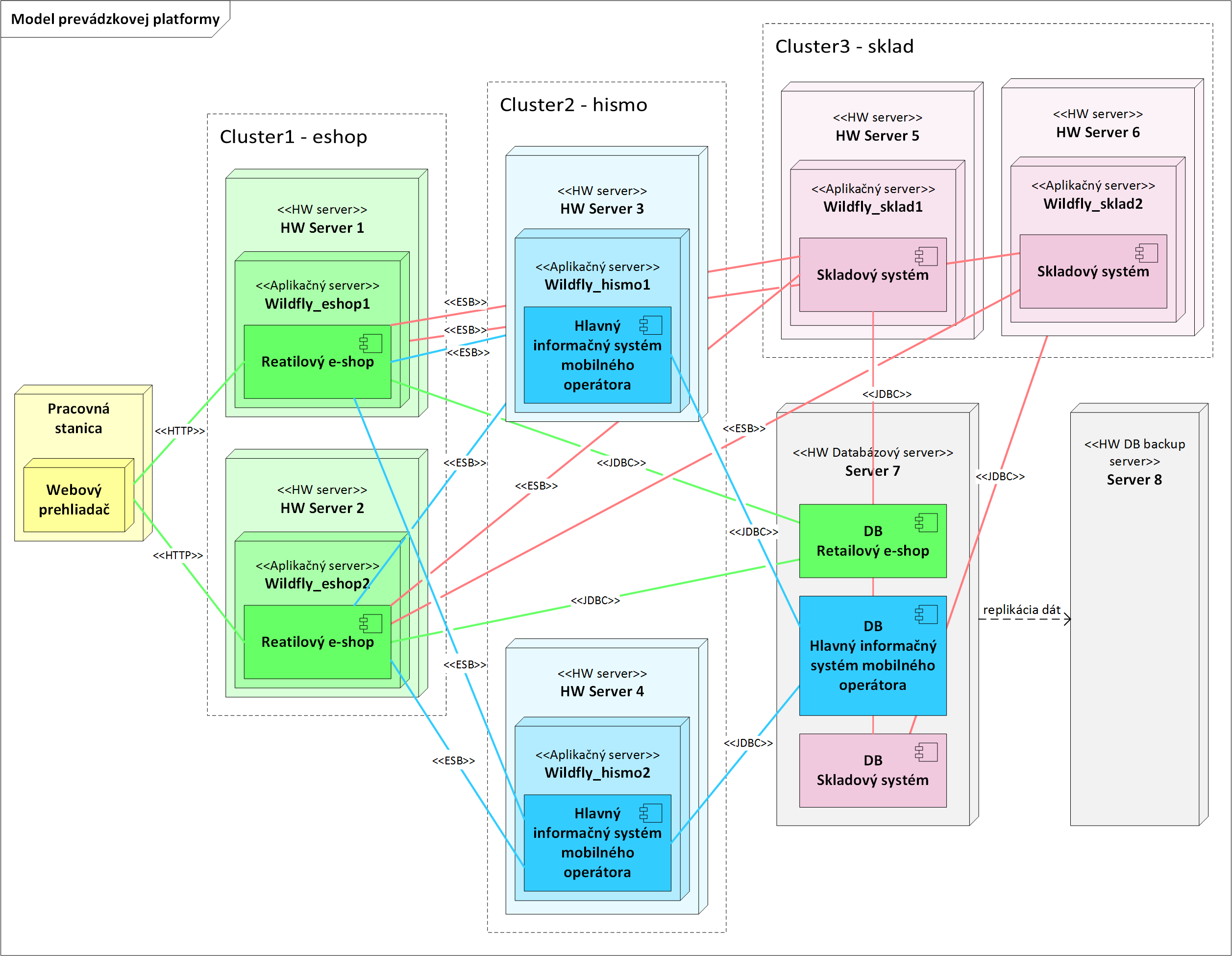


Obrázok : Vývojový pohľad: moduly systému

# Pohľad rozmiestnenia elementov

Špecifikuje na aké hardvérové, softvérové a sieťové elementy bude systém v prevádzke umiestnený.

## Model prevádzkovej platformy



Obrázok : Pohľad rozmiestnenia elementov: model prevádzkovej platformy

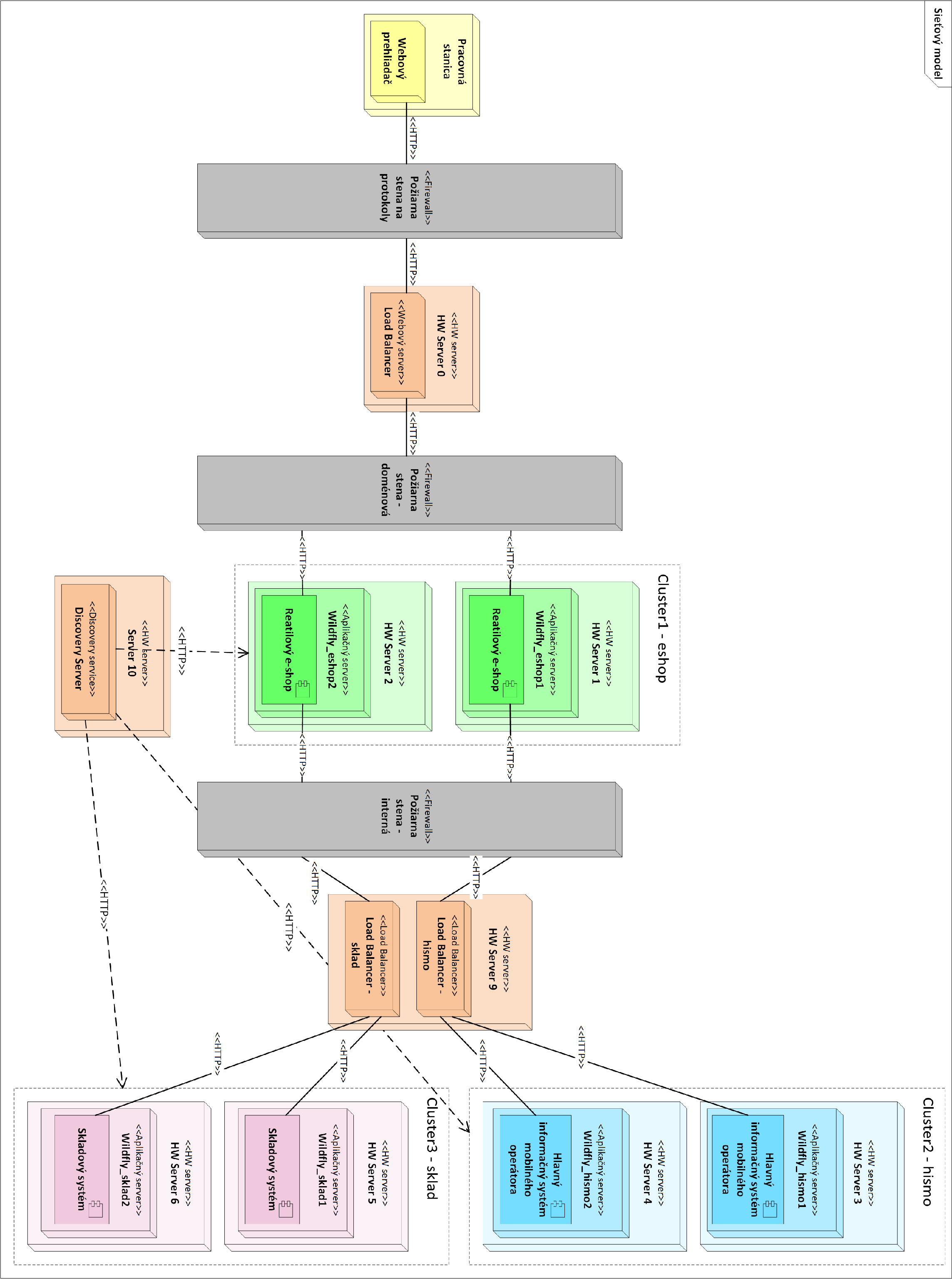
### Použité architektonické princípy

**Distribuovanie systémov** - architektonický princíp, pri ktorom sú komponenty rozmiestnené na viacerých strojoch, ktoré sú vzájomné prepojené sieťou a komunikujú výmenou správ. Jednotlivé komponenty vzájomne interagujú za účelom dosiahnutia spoločného cieľa. Účelom distribúcie je horizontálne škálovanie systému a vyrovnanie sa s výpadkom jedného systému.

**Vyrovnávanie záťaže (Load Balancing)** - technika pre rozloženie záťaže medzi dva alebo viacero strojov (prípadne iných zariadení) s cieľom optimálneho využitia priepustnosti so zlepšením doby odozvy. Použitie viacerých strojov pri vyrovnávaní záťaže môže tiež poskytovať lepšiu odolnosť proti výpadkom.

**Zálohovanie (replikácia) databázových dát** - spočíva vo vytvorení kópie dát z databázy. Vytvorená kópia sa uloží na iný stroj ako pôvodné dáta, vďaka čomu sa môže použiť pre (čiastočnú) obnovu stratených dát v prípade neočakávanej udalosti ako výpadok, odcudzenie a podobne.

## Sieťový model



Obrázok : pohľad rozmiestnenia elementov: sieťový model

### Použité architektonické princípy

**Firewall (požiarna stena)** - Predstavuje sieťové zariadenie alebo softvér, ktorého úlohou je oddeliť siete s rôznymi prístupovými právami (typicky napr. Internet a Intranet) a kontrolovať tok dát medzi týmito sieťami.

**Load Balancer (vyrovnávač záťaže)** - Zariadenie slúžiace pre rozloženie záťaže medzi dva alebo viacero strojov (prípadne iných zariadení) s cieľom optimálneho využitia priepustnosti so zlepšením doby odozvy. Použitie viacerých strojov poskytuje lepšiu proti výpadkom.

**Service Discovery** – Ide o službu, ktorá zisťuje dostupnosť a správne fungovanie služieb v systéme. Pri výpadku, alebo inom probléme upozorní na tento fakt vyrovnávače záťaže. Tie potom prestane smerovať požiadavky na túto nefunkčnú službu. Toto je umožnené vďaka použitiu distribuovaných systémov.

# Záver a zhodnotenie

Na základe špecifikácie zadania sme navrhli architektúru systému. Tá predstavuje abstraktný návrh, ktorý je ešte potrebné špecifikovať. Pri návrhu sme použili architektonické pohľady Rozanského a Woodsa. Modely sme vytvorili prevažne v jazyku UML s využitím nástroja Microsoft Visio 2013.

V predstavených modeloch sme sa snažili zachytiť požiadavky zo zadania a uplatniť súčasné architektonické princípy.