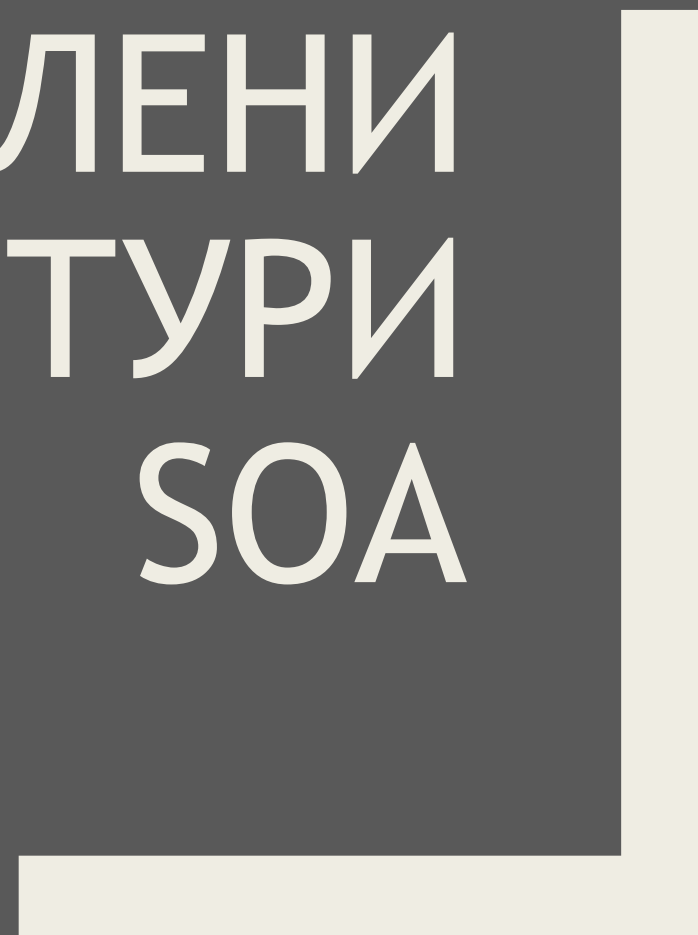


РАЗПРЕДЕЛЕНИ ПРИЛОЖЕНИЯ

Павел Кюркчиев
Ас. към ПУ „Паисий Хилендарски“
@pkyurkchiev

РАЗПРЕДЕЛЕНИ АРХИТЕКТУРИ SOA



Service-oriented architecture (SOA)

- SOA е стил на софтуерния дизайн, при който услугите се предоставят посредством софтуерни компоненти, чрез комуникационен протокол по мрежата

SOA основни компоненти

- Service provider
 - Той създава услугата и предоставя информацията за нея към съответния регистър (Предоставя достъп до нея).
- Service broker, service registry or service repository
 - Основната му функция е да предостави информацията за веб услугата на всеки потенциален клиент.
- Service requester/consumer
 - Това са всички хора който достъпват и използват информацията предоставена от услугите.

Customers

Cloud
service
consumers

Human
users

Enterprise Service Bus

Providers

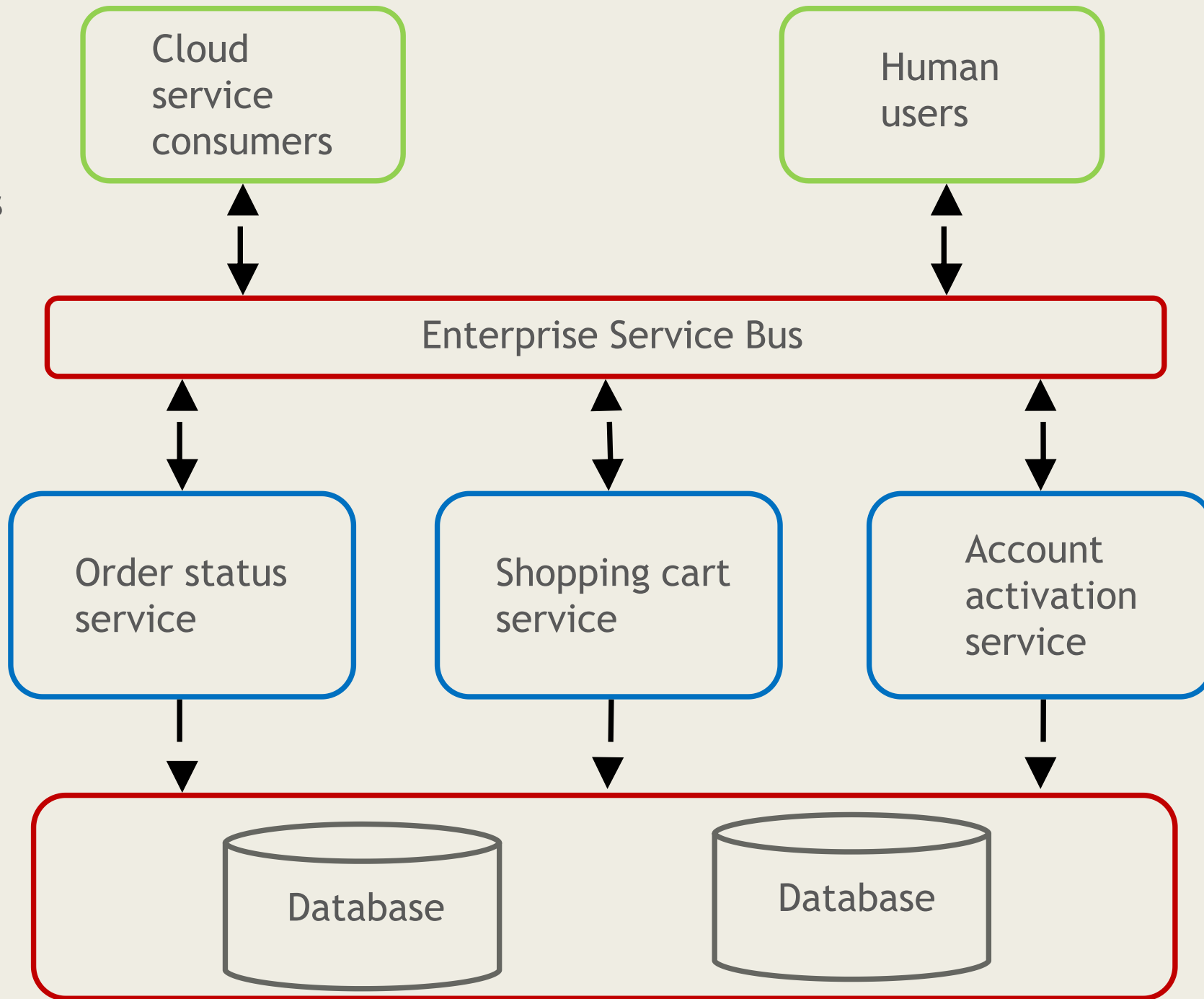
Order status
service

Shopping cart
service

Account
activation
service

Database

Database



SOA и Web 2.0

Web 2.0

- User generated content
- Services Model
 - *Сървис ориентира архитектура*
- Поява на RSS, Web API ...

SOA услуги

- Услуги са градивния блок на SOA архитектурата. Те могат да бъдат изградени от други услуги. Представяват черна кутия за потребителя(той не знае как те работят). Услуги представят бизнес логиката на приложението.

Характеристики на услугите

- Автономни - независими са от останалата част на системата
- Stateless - не знаят за предишни изпратени от вас заявки (не пазят състоянието на заявката)
- Приемат заявка и връщат отговор
- Добре описани са
- Стъпват на стандартизирани интерфейси

Характеристики на услугите

- Комуникацията се осъществява през стандартни протоколи:
 - *HTTP, FTP, SMTP, RPC, MSMQ....*
 - *JSON, XML, SOAP, RSS, WS-....*
- Платформено независими са

Свойства на услугите

- Услугата представлява бизнес активност със специален изход/резултат.
- Услугата е самостоятелна, самосъдържаща се.
- Услугата е черна кутия за ползвателите.
- Услугата може да се състои от други скрити отдолу услуги.

Принципи на SOA

- Стандартизирани условия на услугите
 - *Услугите се придържат към стандартни комуникационни условия, като се дефинират от един или повече документи, описващи услугите за дадения набор от услуги.*
- Автономност на връзката между услугите (част от loose coupling)
 - *Връзката между услугите е минимализирана до ниво, в което те знаят само за тяхното съществуване.*
- Прозрачност на местоположението на услугите (част от loose coupling)
 - *Услугите могат да бъдат извикани от всякъде в мрежата, без значение къде се намират.*

SOA принципи

■ Дълголетие на услугата

- *Услугите могат да се проектират така, че да имат дълъг живот. Където е възможно услугите трябва да избягват налагането над ползвателите да се променят ако те не желаят да се добавят нови функционалности. В сила трябва да е следното правило "ако извикаш една услуга днес, трябва да можеш да извикаш същата услуга и утре".*

■ Абстрактни услуги

- *Услугите трябва да работят като черна кутия, по този начин тяхната вътрешна логика е скрита от ползвателите.*

■ Автономни услуги

- *Услугите са независими и контролират функционалностите които скриват, от страна на Design-time(времето за разработка) и run-time(времето за действие).*

SOA принципи

- Услуги без състояние
 - *Услугите нямат състояние, те или връщат пожелания резултат, или връщат изключение(exception). Едно извикване на услугата не трябва да влияе на друго такова.*
- Принцип на детайлност на услугата
 - *Принцип целящ да подсигуре, че услугата ще има адекватен размер и обхват. Функционалностите предоставени на ползвателя от услугата трябва да бъдат уместни.*
- Нормализиране на услуга
 - *Услугите се декомпонизират и консолидират(нормализират), за да се намали излишното. За някои услуги това не може да се направи, това са случаите в които бързодействието, достъпа и агрегацията са нужни.*

SOA принципи

- Композитност на услуги
 - *Услугите могат да се използват за да се композират други услуги.*
- Откриване на услуги
 - *Услугите имат специални комуникационни метаданни чрез които услугите лесно могат да се откриват и достъпват.*
- Преизползване на услуги
 - *Логиката е разделена на малки парчета, което позволява отделни малки услуги и тяхното лесно преизползване.*
- Енкапсулация на услуги
 - *Много услуги, които не в началото не са планирани като SOA, могат да се енкапсулират(обвият) и да станат част от SOA.*

Подходи на имплементация и технологии

- Web services based on WSDL and SOAP
- Messaging, e.g., with ActiveMQ, JMS, RabbitMQ
- RESTful HTTP, with Representational state transfer (REST) constituting its own constraints-based architectural style
- OPC-UA
- WCF (Microsoft's implementation of Web services, forming a part of WCF)
- Apache Thrift
- SORCER