Engenharia de Software Orientada a Serviços

Paulo Cesar Masiero

Roteiro

- Contexto
- Arquiteturas Orientadas a Serviços
- Serviços como componentes reusáveis
- Engenharia de Serviços
- Desenvolvimento de Software como Serviço.

Roteiro

- Contexto
- Arquiteturas Orientadas a Serviços
- Serviço como componente reusável
- Engenharia de Serviços
- Desenvolvimento de Software como Serviço.

Contexto

- Premissas do desenvolvimento de software tradicional (“mundo fechado”):
  - O mundo externo ao sistema desenvolvido muda muito devagar e o software pode permanecer estável por um longo período.
  - Todos os requisitos das interações do sistema com o mundo externo (fechado) podem ser previamente elicitados.
  - As mudanças são feitas por meio de solicitações que são organizadas por prioridades e em seguida implementadas, testadas e implantadas.
  - As mudanças são prejudiciais à qualidade do software e devem ser evitadas.

Contexto

- Cenário real e atual:
  - As premissas do desenvolvimento tradicional (mundo fechado) não são válidas para muitos casos atualmente.
  - Há muitas aplicações que funcionam em um “mundo aberto”, em que o ambiente muda constantemente e o software deve se adaptar e reagir dinamicamente a essas mudanças.
  - No mundo aberto os sistemas podem localizar e utilizar funcionalidades dinamicamente, mesmo as que não estavam disponíveis quando o software foi criado.
  - As mudanças devem ser “abraçadas”, pois são naturais e inevitáveis.

Roteiro

- Contexto
- Arquiteturas Orientadas a Serviços
- Serviço como componente reusável
- Engenharia de Serviços
- Desenvolvimento de Software como Serviço
- Teste de Serviços
- Tópicos de Pesquisa

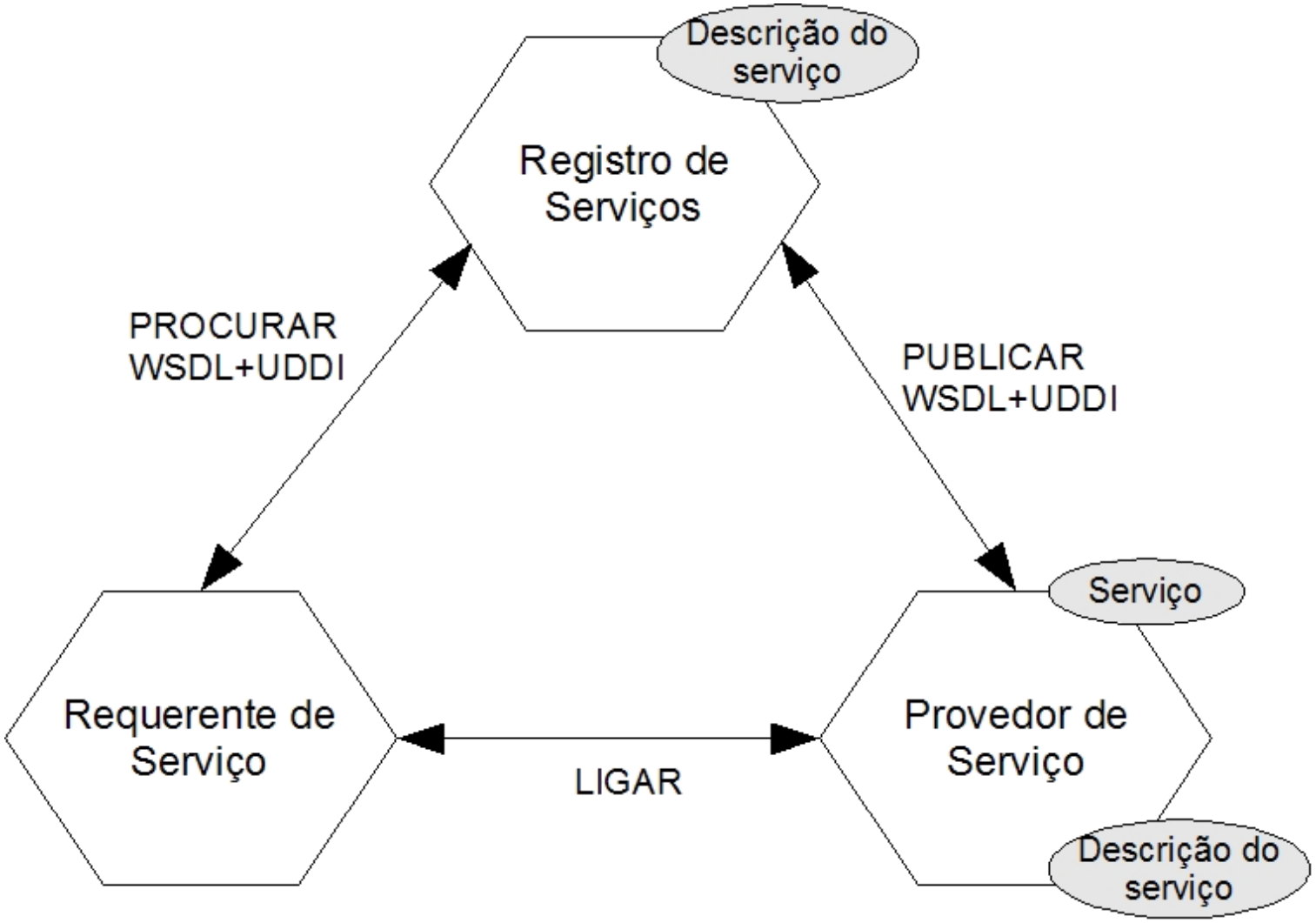
Arquiteturas Orientadas a Serviços

- O serviço é o bloco de construção principal de um software orientado a serviços
- Serviços são módulos independentes e autocontidos que oferecem funcionalidades de negócio específicas.
- O fornecimento do serviço (e sua manutenção) é independente da aplicação que o usa.
- Exemplo de aplicação: Sistemas de Gestão da Cadeia de Suprimentos → muitas trocas de informações entre empresas.

Arquiteturas Orientadas a Serviços

- Um serviço pode assumir dois papéis em uma arquitetura orientada a serviços:
  - Servidor/Provedor
    - Prestador de um serviço.
    - Possui interface publicada com a descrição dos serviços prestados.
  - Cliente/Consumidor
    - Solicita (consome) serviços de um provedor.

Arquiteturas Orientadas a Serviços



Serviços Web

- Um Web Service é um tipo de serviço, isto é, é uma instância de uma ideia mais geral de serviço.
- Os serviços são descritos de forma padronizada, possuem interfaces publicadas e se comunicam com outros serviços por meio de invocações remotas.
- Usam tecnologia XML para implementação dos padrões.
- A composição de serviços Web é feita por meio de linguagens como a BPEL (Business Process Execution Language), mas podem ser usadas outras linguagens.

Serviços Web

- As descrições dos serviços Web e suas interfaces são expressas por meio de um arquivo WSDL (Web Services Description Language).

- Foi proposta também uma forma para descobrir serviços, por meio do padrão UDDI (Universal Description, Discovery and Integration), mas ela não foi completamente implementada.
  - por isso, os serviços de registros não se transformaram em realidade.
- Atualmente, a busca de serviços é feito pelo padrão normal da Web, usando WSDL.

Arquiteturas Orientadas a Serviços

- Cada serviço Web deve ser identificado por uma URI (Uniform Resource Identifier).

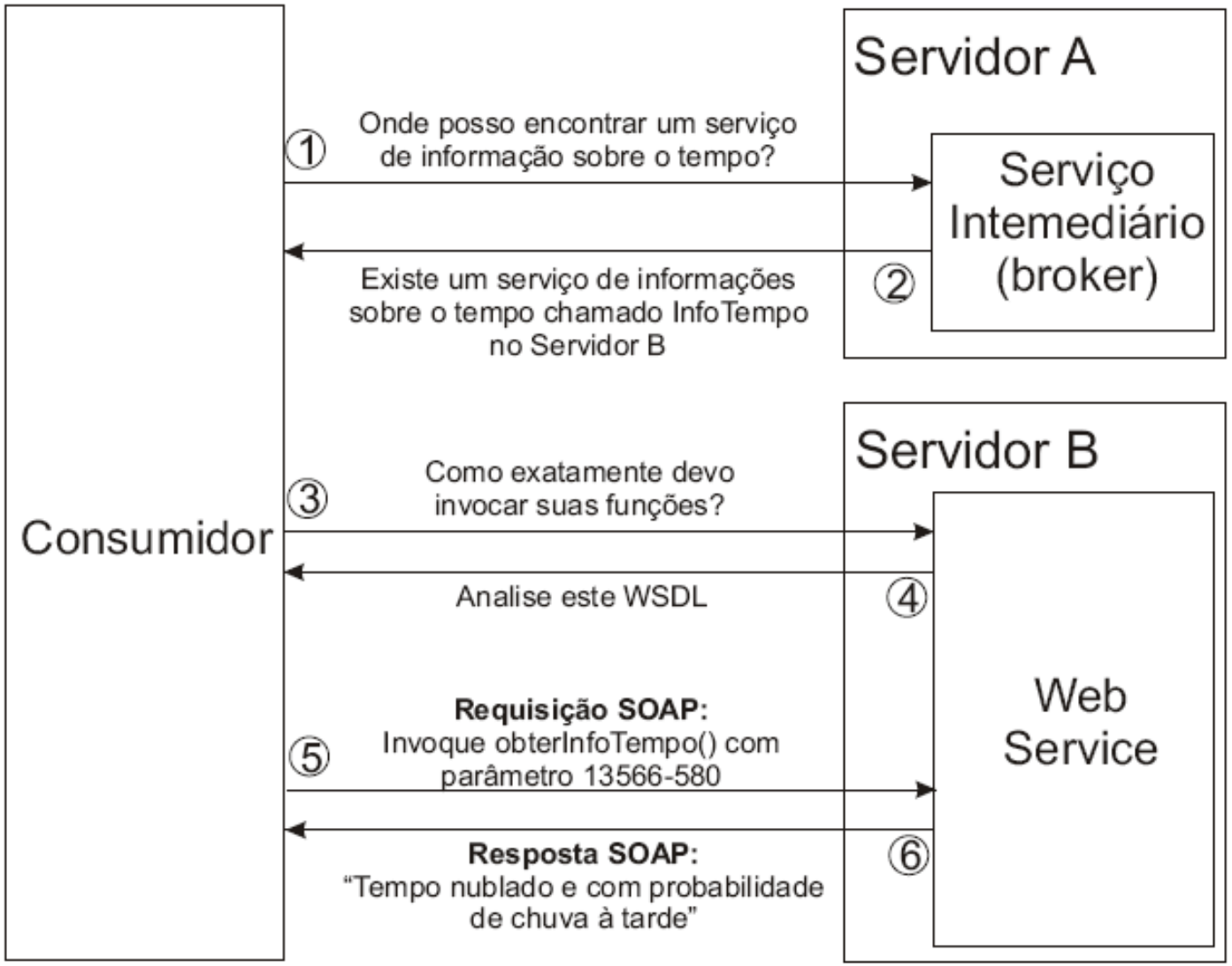
- A comunicação entre os serviços Web ocorre via protocolos como o SOAP (Simple Object Access Protocol).

- Os serviços se beneficiam com o desenvolvimento da computação “em nuvem”.

Serviços Web

- As implementações dos padrões Web são “pesadas”.
- Algumas empresas, como a Amazon, usam padrões mais simples, como “RESTful”
- RESTful é incompleto em relação ao apoio a segurança e transações.

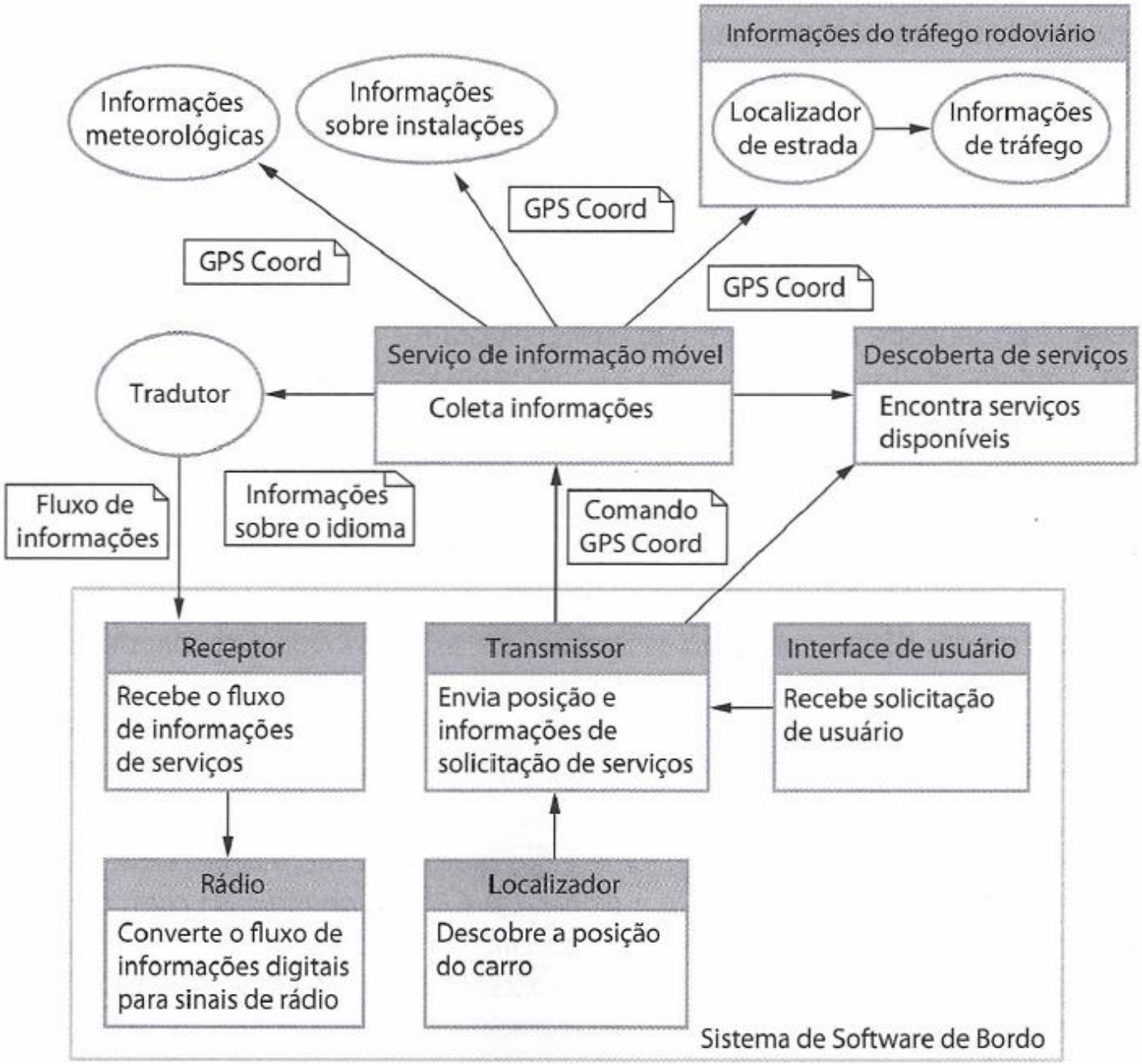
Exemplo


Consumidor
Servidor A
Serviço Intemediário (broker)
Servidor B
Web Service
① Onde posso encontrar um serviço de informação sobre o tempo?
② Existe um serviço de informações sobre o tempo chamado InfoTempo no Servidor B
③ Como exatamente devo invocar suas funções?
④ Analise este WSDL
⑤ Requisição SOAP: Invoque obterInfoTempo() com parâmetro 13566-580
⑥ Resposta SOAP: "Tempo nublado e com probabilidade de chuva à tarde"

Exemplo

Um sistema de informação em um carro fornece aos motoristas informações sobre o clima, condições de tráfego da estrada, informações locais etc. Ele é ligado ao rádio do carro para entregar informações como um sinal em um canal específico. O carro tem receptor GPS para descobrir sua posição e com base nessa posição recebe informação de serviços como postos e restaurantes. Em seguida a informação é entregue ao motorista na linguagem escolhida.

.3 Um sistema de informações de bordo baseado em seviços



Roteiro

- Contexto
- Arquiteturas Orientadas a Serviços
- Serviços como componentes reusáveis
- Engenharia de Serviços
- Desenvolvimento de Software como Serviço
- Teste de Serviços

Serviços como componentes reusáveis

- Um serviço é um componente de software com baixo acoplamento, reusável, que encapsula funcionalidade discreta, que pode ser distribuída e acessada por meio de programas.
- Os padrões de serviços-web são, essencialmente, um modelo de componentes, como Corba, EJB etc.

Serviço como componente reusável

- São componentes reusáveis, com baixo acoplamento, encapsulam funcionalidade discreta (que pode ser distribuída) e são acessados por meio de programas.
- Têm apenas interfaces “provides” e não possuem interfaces “requires”.
- Geralmente não armazenam estado.
- São usados em composições (ou aplicações).

Componentes vs Serviços

- Serviços compartilham muitas características com os componentes de software:
  - São auto-contidos
  - São altamente reusáveis
  - São unidades de composição com interfaces bem definidas
  - São fornecidos como caixa-preta

Componentes vs Serviços

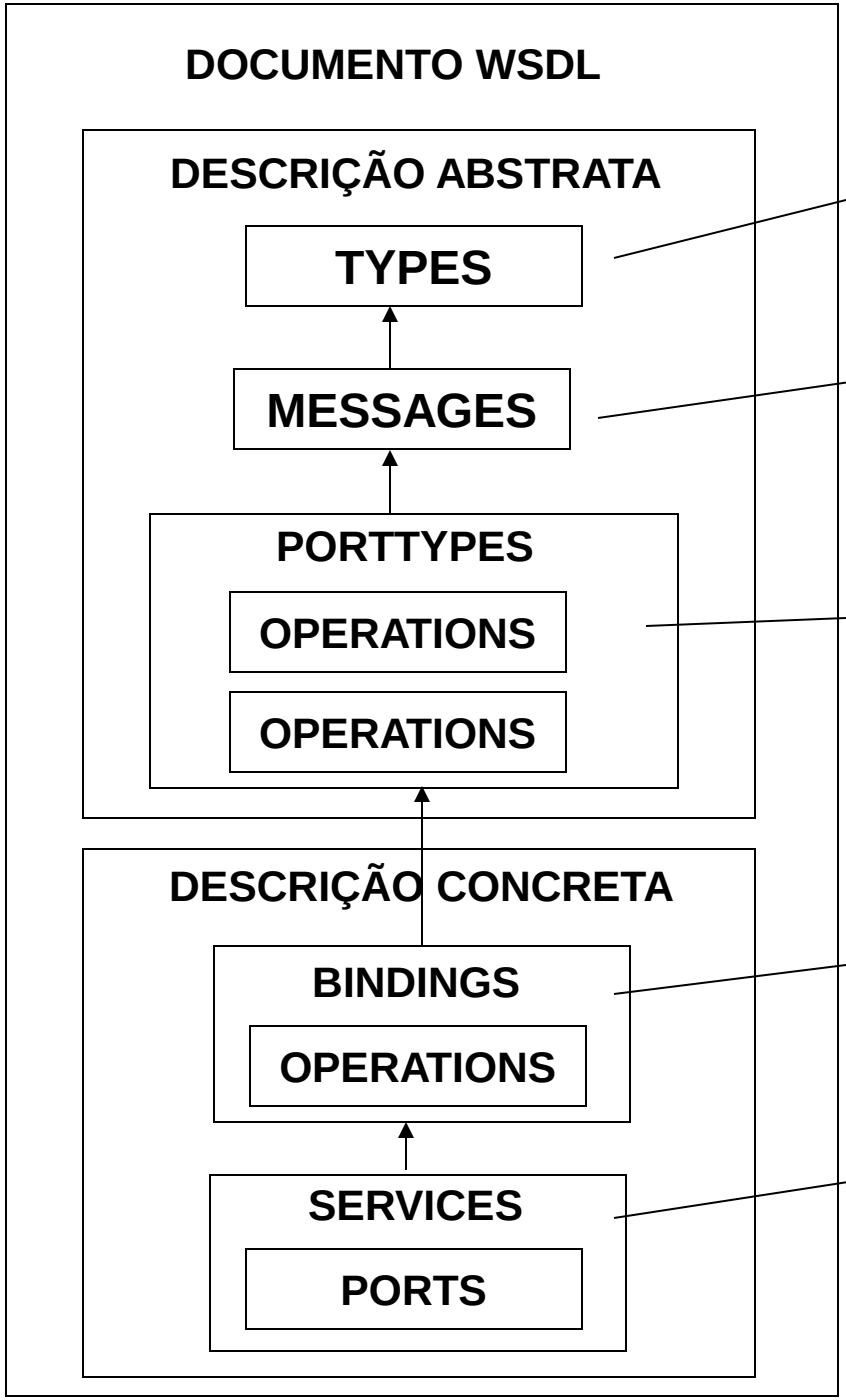
- Existem algumas diferenças entre componentes e serviços:
  - Serviços devem ser mais fracamente acoplados do que os componentes
  - Os componentes podem ser instalados fisicamente na máquina de seus clientes/consumidores enquanto os serviços são acessados apenas remotamente
  - Componentes mantém estado

Como usar um serviço

- É preciso saber a onde ele se encontra (URI) e os detalhes de sua interface.
- Esses detalhes são codificados na linguagem WSDL, que define:
  - O que o serviço faz : operações,
  - O mapeamento da interface abstrata para um conjunto de protocolos,
  - Descreve o local onde o serviço está implantado.

WSDL

- Tem uma parte introdutória que define *namespaces* de XML e outras descrições.
- Descrição (opcional) dos tipos de mensagens
- Descrição das interfaces
- Descrição das entradas e saídas
- Descrição da ligação usada pelo serviço. O *default* é o SOAP.
- Descrição do local físico do serviço (URI)



Tipos de dados usados e definidos no documento WSDL

Mensagens enviadas e recebidas Juntamente com suas partes (Parâmetros de entrada/saída)

Operações disponibilizadas pelo serviço

Como o serviço será acessado na rede, através de qual protocolo

Onde o serviço está localizado p/ acesso – endereço do serviço na rede

Quadro 19.1 Parte de uma descrição WSDL para um *web service*

Defina alguns dos tipos usados. Suponha que o prefixo de *namespace* 'ws' se refira ao *namespace* URI para esquemas XML e o prefixo de *namespace* associado com essa definição seja *weathns*.

```
<types>
  <xs: schema targetNameSpace = "http://.../weathns"
        xmlns: weathns = "http://.../weathns" >
        <xs:element name = "PlaceAndDate" type = "pdrec" />
        <xs:element name = "MaxMinTemp" type = "mmtrec" />
        <xs:element name = "InDataFault" type = "errmess" />
        <xs:complexType name = "pdrec"
        <xs:sequence>
              <xs:element name = "cidade" type = "xs:string"/>
              <xs:element name = "país" type = "xs:string"/>
              <xs:element name = "dia" type = "xs:date"/>
         </xs:complexType>
         Definições de MaxMinType e InDataFault aqui
  </schema>
</types>
```

Agora, defina a interface e suas operações. Nesse caso, existe apenas uma operação para retornar as temperaturas máximas e mínimas.

```
<interface name = "weatherInfo" >
  <operation name = "getMaxMinTemps" pattern = "wsdlns: in-out">
  <input messageLabel = "In" element = "weathns: PlaceAndDate" />
  <output messageLabel = "Out" element = "weathns:MaxMinTemp" />
  <outfault messageLabel = "Out" element = "weathns:InDataFault" />
</operation>
</interface>
```

Problemas com o WSDL

- A interface não inclui informação sobre a semântica do serviço ou suas características não funcionais, como desempenho e confiança.
- As funções precisam ser “descobertas” pelos atributos das mensagens de entrada e saída
- As características não funcionais precisam ser descobertas por meio de experimentos.
- Pesquisas: ontologias

Roteiro

- Contexto
- Arquiteturas Orientadas a Serviços
- Serviços como componentes reusáveis
- Engenharia de Serviços
- Desenvolvimento de Software como Serviço
- Teste de Serviços

Engenharia de serviços

- É o processo de desenvolvimento de serviços para reuso nas aplicações orientadas a serviços.
- Os engenheiros devem assegurar que o serviço desenvolvido representa uma abstração reusável que poderia ser útil em diversos sistemas
- Esse processo inclui a documentação e registro do serviços

Engenharia de serviços

- É o processo de desenvolvimento de serviços para reúso em aplicações orientadas a serviços.
- Existem três atividades principais
  - Identificação de serviço candidato
  - Projeto de interface do serviço
  - Implementação e implantação de serviço

Identificação de serviço candidato

- Envolve a compreensão e análise dos processos de negócio de uma organização para decidir quais serviços reusáveis são necessários para apoiar os processos
- Não existe uma fórmula pronta para definir o que é e o que não é um serviço
- O resultado do processo de identificação é uma lista de serviços candidatos e seus requisitos funcionais e não funcionais

Identificação de serviço candidato

- Três tipos fundamentais de serviços:
  - Utilitário: funcionalidades gerais que podem ser usadas por diferentes serviços.
  - De negócios: serviços associados a uma função (funcionalidade) específica de negócio.
  - De coordenação ou de processo: apoiam processos amplos que geralmente envolvem diferentes atores e atividades.

Identificação de serviço candidato

- Serviços podem ser orientados a entidades ou a tarefas
  - Serviços orientados a tarefas são os associados com alguma atividade, enquanto os orientados a entidade estão associados a alguma entidade do negócio (Earl).
- É preciso identificar os requisitos funcionais e não funcionais de cada serviço candidato
- Objetivo geral: identificar serviços que são logicamente coerentes, independentes e reusáveis

Tabela 19.1 A classificação de serviços

	Utilitário	**Negócios**	**Coordenação**
Tarefa	Conversor de moeda Localizador de funcionário	Validar formulário de reclamação Avaliar classificação de crédito	Processar despesas de reclamações Pagar fornecedor externo
Entidade	Verificador de estilo de documento Conversor de formulário Web para XML	Formulário de despesas Formulário de solicitação de estudante	

Identificação de serviço candidato

- A identificação de serviços não é simples, assim como não é tão simples decompor um sistema em objetos ou componentes.
- Existem métodos de identificação de componentes a partir dos requisitos (Ex. UML Components) e modelos de análise que podem ser utilizados na identificação de serviços

Identificação de serviço candidato

- Ao pensar em possíveis candidatos a serem fornecidos como serviços, algumas perguntas devem ser respondidas:

1. Para um serviço orientado a entidade, o serviço é associado a uma única entidade lógica usada em diferentes processos de negócios? Quais operações normalmente executadas sobre essa entidade devem ser fornecidas?

Identificação de serviço candidato

2. Para um serviço orientado a tarefas, a tarefa é cumprida por diferentes pessoas na organização? Elas estarão prontas a aceitar a inevitável padronização que ocorre quando um único serviço é fornecido?

3. O serviço é independente? Em que extensão ele depende de outros serviços?

Identificação de serviço candidato

4 – O serviço precisa manter estado? Existe um banco de dados para manter este estado? Em geral serviços que dependem de estado interno são menos reusáveis.

5 – O serviço poderia ser usado por clientes fora da organização? Ele pode ser acessado interna e externamente?

Identificação de serviço candidato

6 – Espera-se que diferentes consumidores do serviço tenham diferentes requisitos não funcionais? Se sim, isso sugere que, talvez, mais de uma versão do serviço deve ser implementada.

EXEMPLO

Uma grande empresa, que vende equipamentos de informática, organizou preços especiais para configurações aprovadas para alguns clientes. Para facilitar a encomenda automatizada, a empresa pretende produzir um serviço de catálogo que permitirá aos clientes selecionar o equipamento de que precisam. Ao contrário de um catálogo de consumidor, os pedidos não serão feitos diretamente por meio de uma interface de catálogo. Em vez disso, os produtos serão ordenados por meio do sistema de aquisição, baseado na Web, de cada empresa que acessa o catálogo como um web service. A maioria das empresas tem seus próprios procedimentos de orçamentação e aprovação de pedidos e seu próprios processos de pedidos devem ser seguidos quando um pedido é feito.

Os requisitos do serviço de catálogo são:

1. Uma versão específica de catálogo é fornecida para cada empresa do usuário. Isso inclui as configurações e os equipamentos que podem ser ordenados por funcionários da empresa do cliente e os preços combinados para cada item.
2. O catálogo deve permitir a um funcionário do cliente fazer o download de uma versão do catálogo para navegação *off-line*.
3. O catálogo deve permitir que os usuários comparem as especificações e os preços de até seis itens do catálogo.
4. O catálogo deve fornecer aos usuários facilidades de navegação e pesquisa.
5. Os usuários do catálogo devem ser capazes de descobrir a data prevista para a entrega de um dado número de itens específicos do catálogo.
6. Os usuários do catálogo serão capazes de colocar 'ordens virtuais', em que os itens necessários serão reservados para eles por 48 horas. As ordens virtuais devem ser confirmadas por uma ordem real colocada por um sistema de aquisição. Ela deve ser recebida dentro de 48 horas a partir da ordem virtual.

Além desses requisitos funcionais, o catálogo tem alguns requisitos não funcionais:

1. O acesso ao serviço de catálogo deve ser restrito apenas a funcionários de organizações autorizadas.
2. Os preços e as configurações oferecidas para um cliente serão confidenciais e não estarão disponíveis para os funcionários de qualquer outro cliente.
3. O catálogo estará disponível sem interrupção do serviço das 7h às 11h.
4. O serviço de catálogo deve ser capaz de processar até dez solicitações por segundo em momentos de pico.

Projeto de interface de serviço

- Depois de selecionar os serviços candidatos deve-se projetar as interfaces dos serviços.
- Deve-se projetar as operações e mensagens
  - Minimizar o número de trocas de mensagens para completar uma solicitação. O maior número de informações possíveis deve ser passada de uma vez (Exemplo: Garçom)
  - Sempre que possível usar uma mensagem, ao invés de interações síncronas de serviços.
- A tarefa de gerenciar estado é de responsabilidade do usuário do serviço. Por isso pode ser necessário passar essas informações nas mensagens.

Projeto de interface de serviço

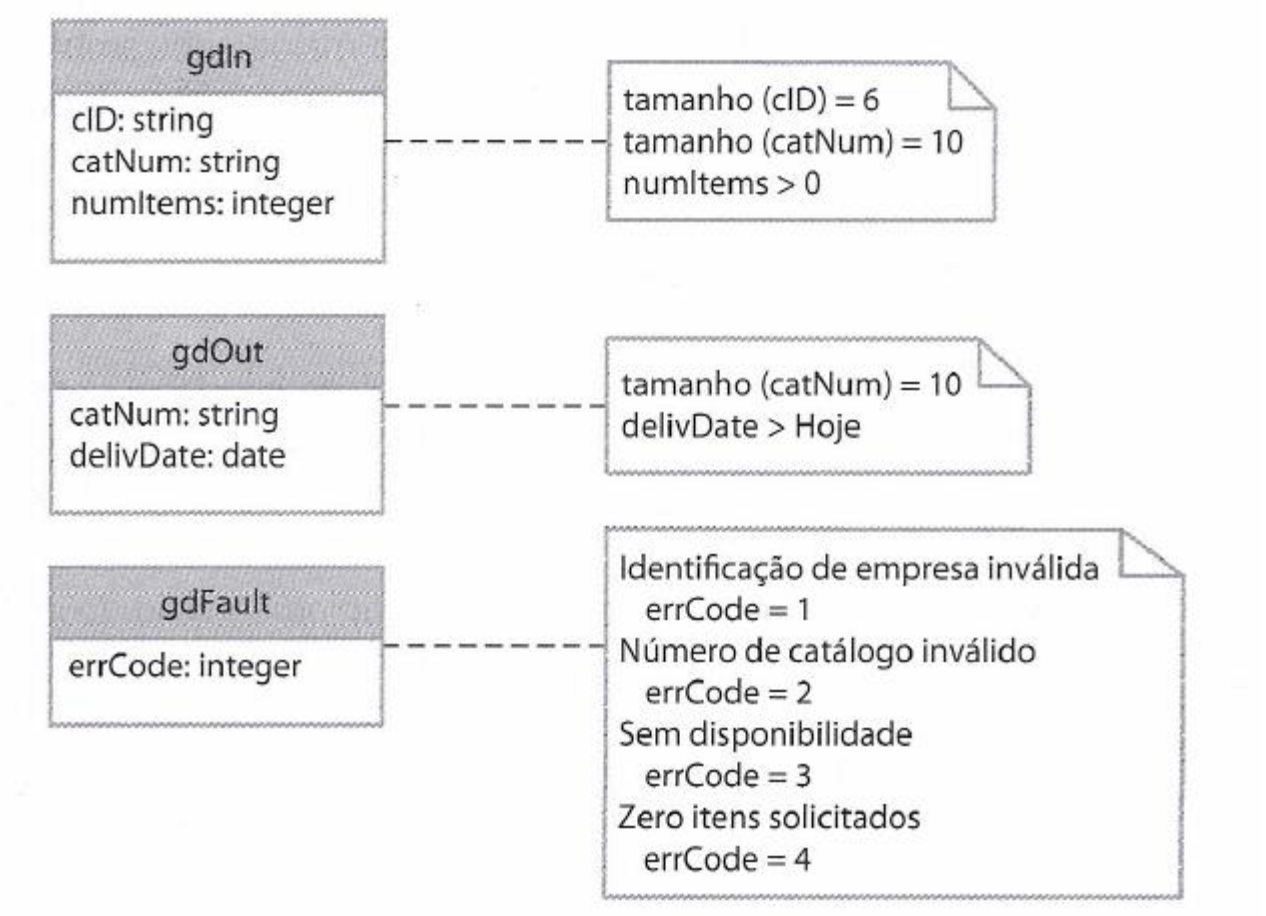
- Há três estágios na especificação da interface:
  - Projeto de interface lógica: identificação das operações associadas ao serviço, as entradas e as saídas, e as exceções
  - Projeto de mensagem: estruturação das mensagens enviadas e recebidas pelo serviço
  - Criação da interface: tradução do projeto lógico e das mensagens para uma descrição abstrata escrita em WSDL.

Tabela 19.2 Descrições funcionais de operações de serviços de catálogo

Operação	Descrição
MakeCatalog	Cria uma versão do catálogo sob medida para um cliente específico. Inclui um parâmetro opcional para criar uma versão em PDF para download do catálogo.
Compare	Fornece uma comparação de até seis características (por exemplo, preço, dimensões, velocidade do processador etc.) de até quatro itens do catálogo.
Lookup	Exibe todos os dados associados a um item especificado do catálogo.
Search	Essa operação usa uma expressão lógica e procura o catálogo de acordo com essa expressão. Ela exibe uma lista de todos os itens que correspondam à expressão da busca.
CheckDelivery	Retorna a data de entrega prevista para um item ordenado naquele dia.
MakeVirtualOrder	Reserva o número de itens encomendado por um cliente e fornece informações sobre o item para o sistema de aquisição do cliente.

Tabela 19.3 Definição em UML de mensagens de entrada e saída

Operação	Entradas	Saídas	Exceções
MakeCatalog	*mcIn* Id de empresa PDF-bandeira	*mcOut* URL do catálogo para essa empresa	*mcFault* Id de empresa inválida
Compare	*compIn* Id de empresa Atributo de entrada (até 6) Número de catálogo (até 4)	*compOut* URL da página apresentando tabela de comparação	*compFault* Id de empresa inválida Atributo desconhecido Número de catálogo inválido
Lookup	*lookIn* Id de empresa Número de catálogo	*lookOut* URL da página com as informações de item	*lookFault* Id de empresa inválida Número de catálogo inválido
Search	*searchIn* Id de empresa *String* de busca	*searchOut* URL da página Web com resultados de busca	*searchFault* Id de empresa inválida *String* de busca mal formado
CheckDelivery	*gdIn* Id de empresa Número de catálogo Número de itens necessários	*gdOut* Número de catálogo Data esperada de espera	*gdFault* Id de empresa inválida Número de catálogo inválido Sem disponibilidade Zero item solicitado
PlaceOrder	*poIn* Id de empresa Número de itens necessários Número de catálogo	*poOut* Número de catálogo Número de itens requisitados Data prevista de entrega Estimativa de preço unitário Estimativa de preço total	*poFault* Id de empresa inválido Número de catálogo inválido Zero item requisitado


gdIn
clD: string
catNum: string
numItems: integer
tamanho (clD) = 6
tamanho (catNum) = 10
numItems > 0
gdOut
catNum: string
delivDate: date
tamanho (catNum) = 10
delivDate > Hoje
gdFault
errCode: integer
Identificação de empresa inválida
errCode = 1
Número de catálogo inválido
errCode = 2
Sem disponibilidade
errCode = 3
Zero itens solicitados
errCode = 4

Projeto de interface de serviço

- O estágio final do processo de projeto de serviço é traduzir o projeto de interface de serviço em WSDL
- A tradução da especificação da interface para o WSDL pode ser feita manualmente, mas em geral os ambientes de desenvolvimento fazem isto automaticamente

Implementação e Implantação

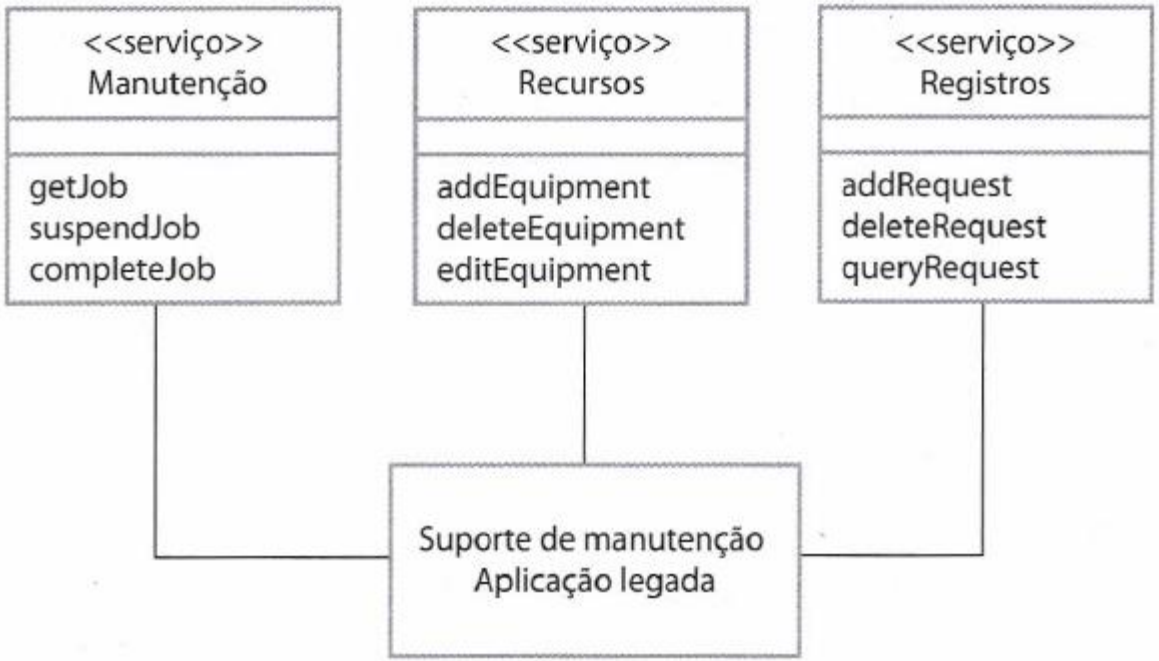
- A implementação de um serviço pode ser feita em diversas linguagens.
- A linguagem em que o serviço é desenvolvido geralmente não importa para o cliente final, pois a comunicação é feita via rede e com protocolos padronizados
- Testar:
  - Conformidade das interfaces com o WSDL
  - Comportamento funcional?

Implementação e Implantação

- Após a implementação o serviço deve ser publicado para acesso de seus consumidores
- Se o objetivo é tornar o serviço público para outras organizações deve-se registrá-lo em um (ou mais) registro de serviços ou usar outro meio, fornecendo informações apropriadas.

Serviços de sistemas legados

- Um dos usos mais importantes de serviços é a criação de “wrappers” em sistemas legados.
- Esses sistemas podem então ser acessados por meio da Web e integrados com outras aplicações, mesmo de forma heterogênea


<<serviço>>
Manutenção
getJob
suspendJob
completeJob
<<serviço>>
Recursos
addEquipment
deleteEquipment
editEquipment
<<serviço>>
Registros
addRequest
deleteRequest
queryRequest
Suporte de manutenção
Aplicação legada

Roteiro

- Contexto
- Arquiteturas Orientadas a Serviços
- Serviços como componentes reusáveis
- Engenharia de Serviços
- Desenvolvimento de Software como Serviço

Desenvolvimento de software como serviço

- Baseia-se na idéia de compor e configurar serviços para criar um serviço composto (composição).
- Os serviços envolvidos em uma composição podem ser:
  - criados especificamente para a aplicação
  - desenvolvidos por uma empresa de serviços de negócios
  - reusados de um provedor externo

Desenvolvimento de software como serviço

- Muitas empresas dedicam-se à conversão de aplicações em serviços. Com isso abre-se a possibilidade para ampliar o reuso dentro da empresa.
- Os serviços também são utilizados de forma interorganizacional entre fornecedores confiáveis.
- Em um cenário mais amplo as empresas podem acessar “mercados de serviços” para adquirir serviços para suas composições.

Desenvolvimento de software como serviço

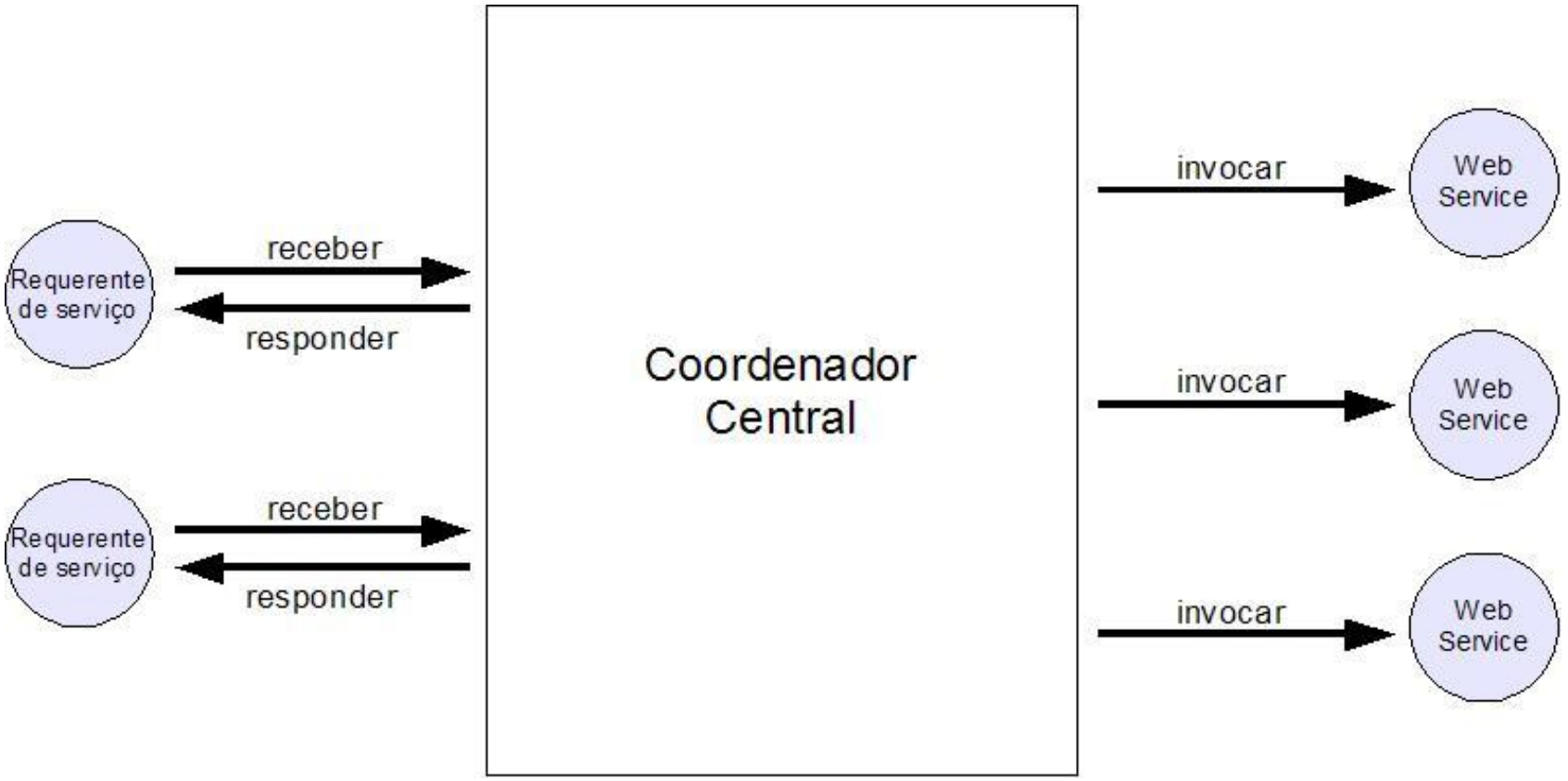
- A composição de serviços pode ser usada para integrar processos de negócios separados a fim de fornecer um processo integrado que ofereça funcionalidades extensas e mais completas.
- Obs. Não confundir com o termo “software como serviço” usado para um tipo de modelo de negócio.

Desenvolvimento de software como serviço

- Estágios no processo de construção de um serviço composto (ou uma aplicação):
  1. Formular esboço de workflow
  2. Descobrir serviços
  3. Selecionar serviços
  4. Refinar workflow
  5. Criar programa de workflow
  6. Testar serviço

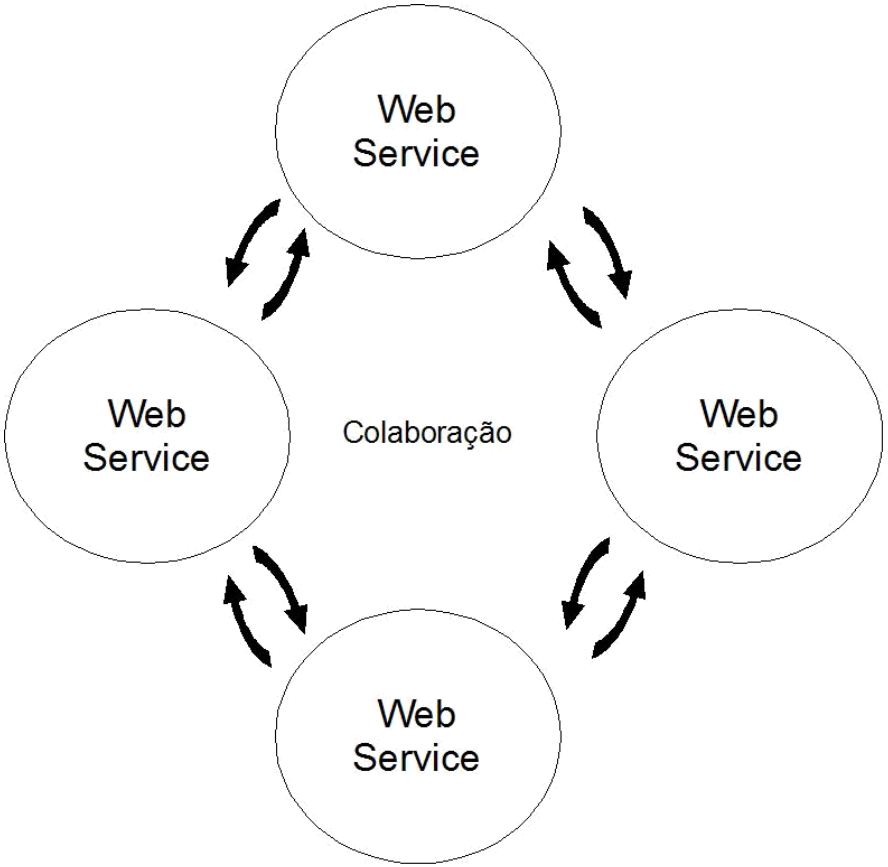
Composição de serviços

- Orquestração



Composição de serviços

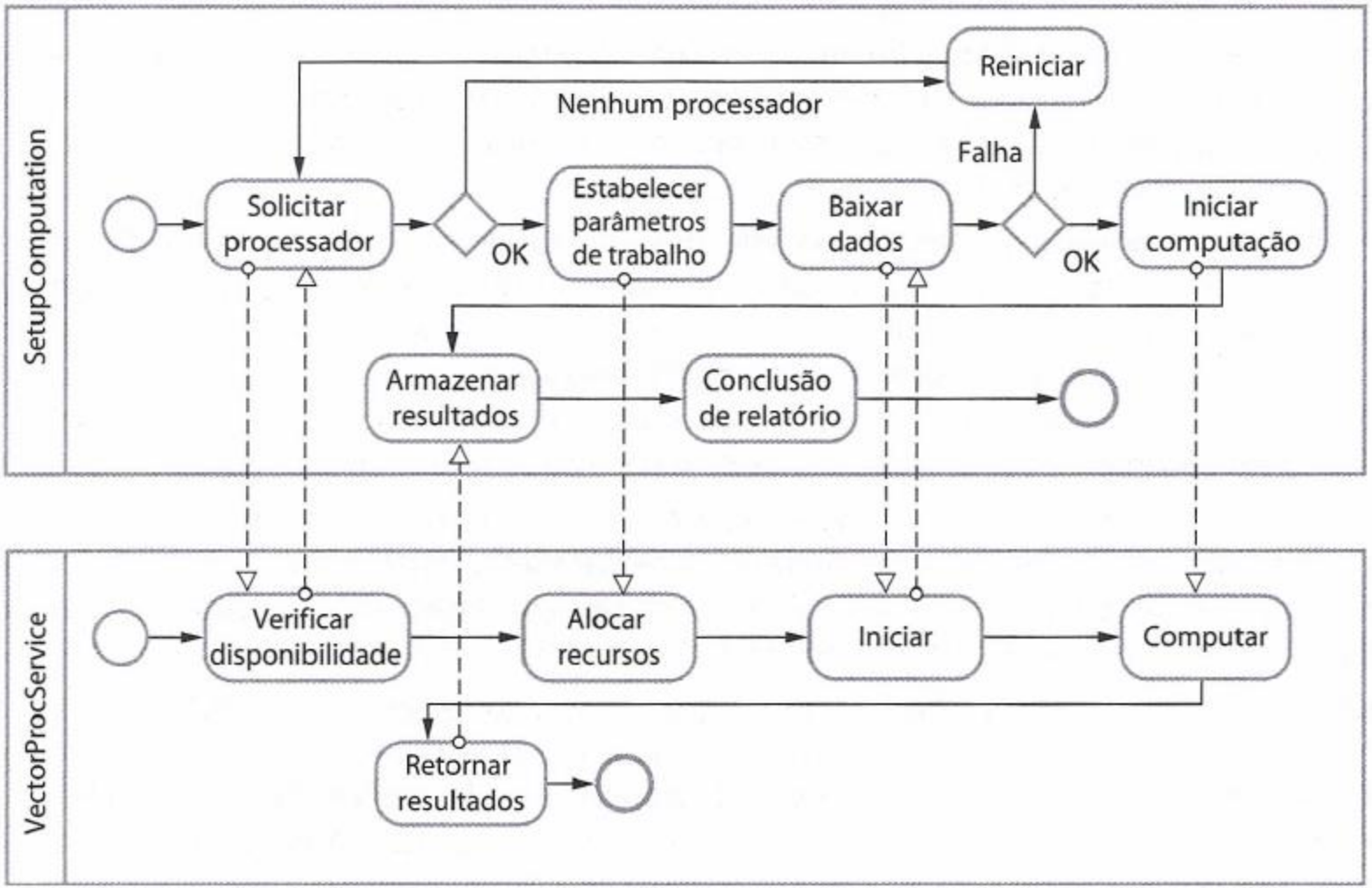
- Coreografia (Colaboração)



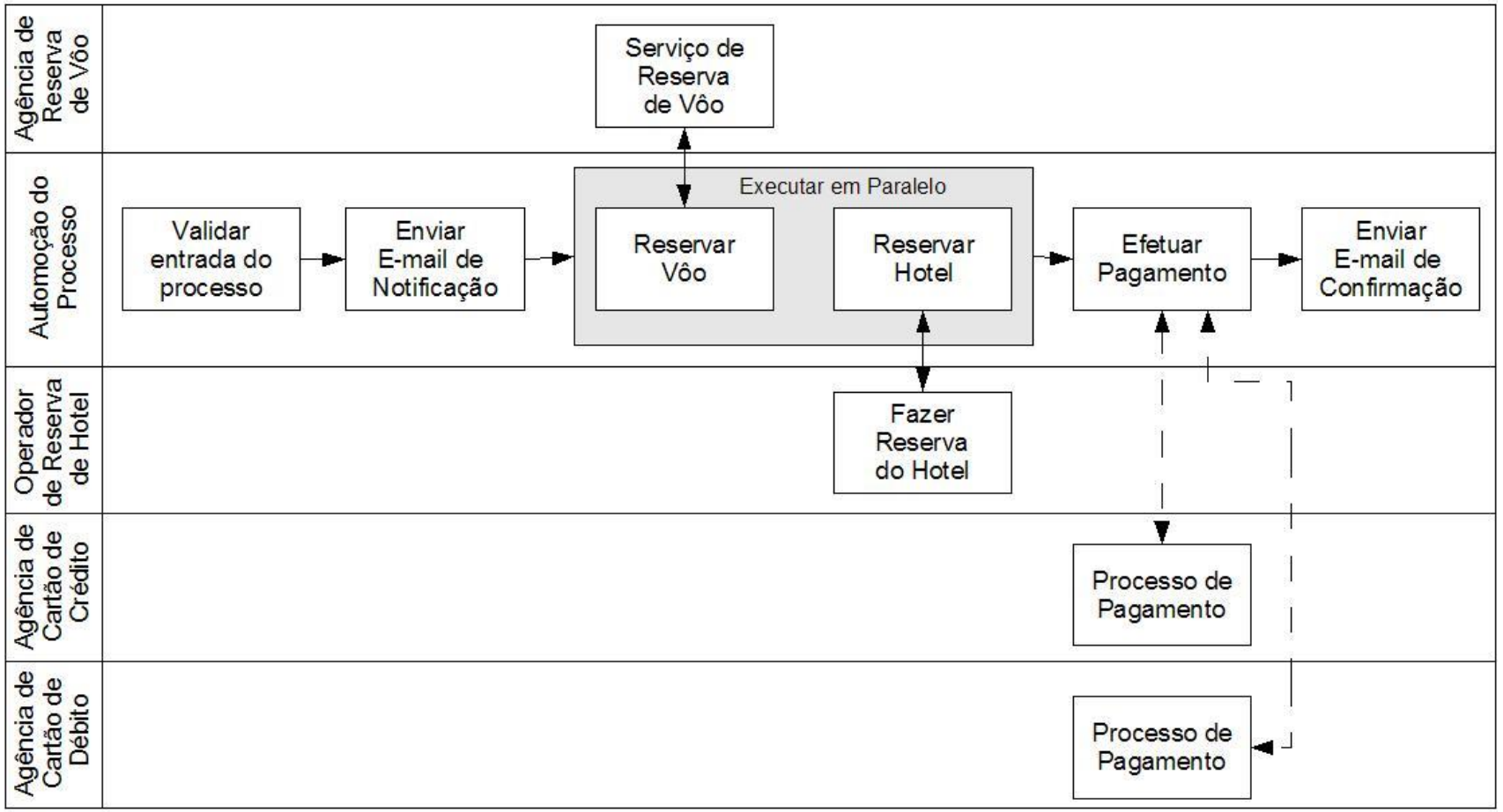
Exemplo de composição

- Compartilhamento de recursos de alto desempenho.
- Duas organizações com workflows separados que colaboram entre si.
- No exemplo usamos a linguagem gráfica BPMN

11 Interação de *workflows*

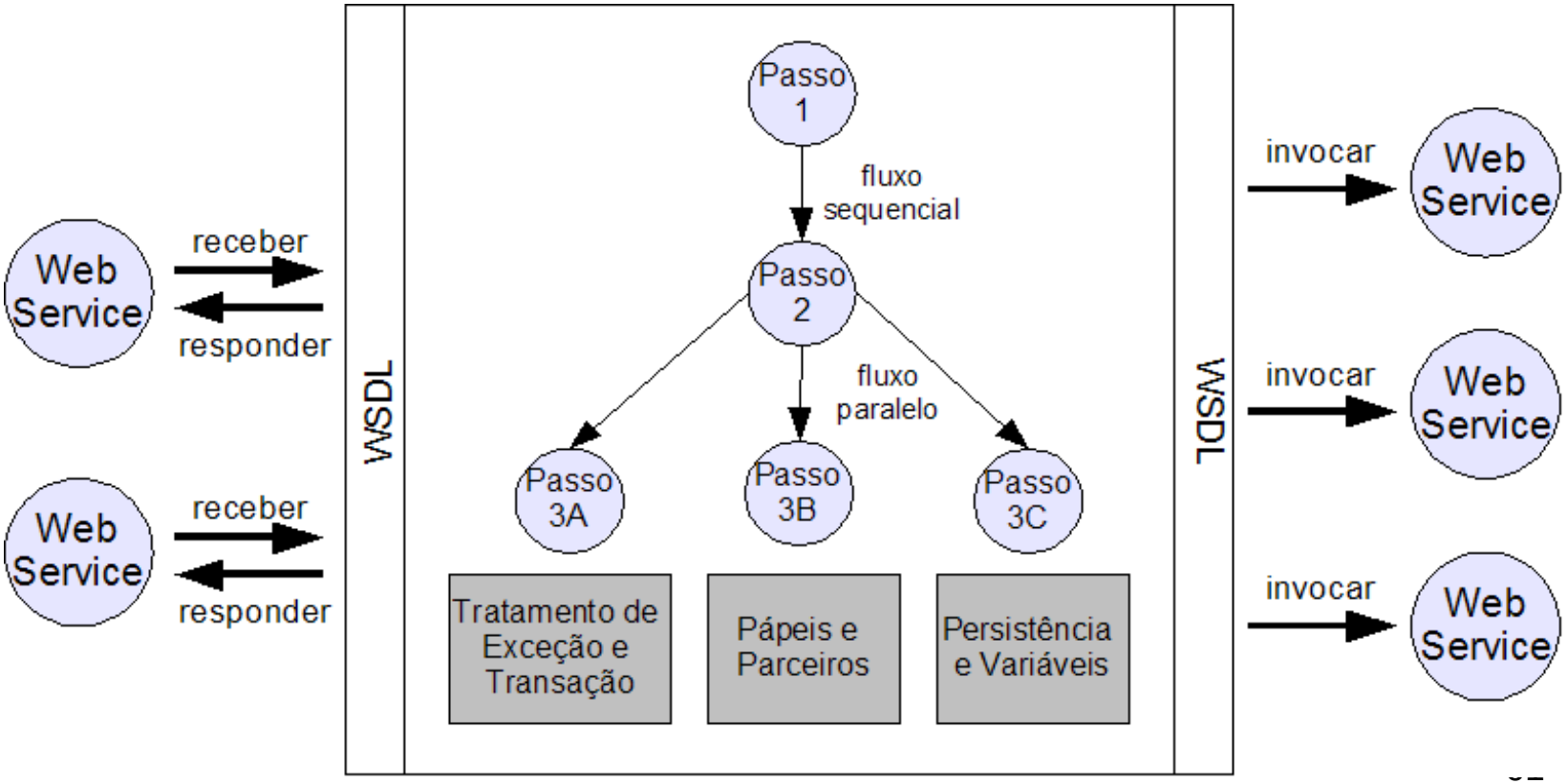


Exemplo de processo



Composição de serviços

- processo BPEL



BPEL

- Uma linguagem de especificação de composição de serviços Web
- É usada para compor um conjunto de serviços Web em um fluxo de negócios e funciona basicamente como um código de ligação (glue code).
- Possui mecanismos para especificar parceiros, variáveis, atividades básicas, estruturadas e de tratamento de dados.
- A linguagem também trata eventos e exceções.