HENRIQUE SHOITI FUGITA

MAPOS: Método de Análise e Projeto Orientado a Serviços

**SÃO PAULO
2009**

HENRIQUE SHOITI FUGITA

MAPOS: Método de Análise e Projeto Orientado a Serviços

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
para obtenção do título de Mestre em
Engenharia

Área de Concentração: Sistemas Digitais

Orientador: Prof. Dr. Kechi Hirama

SÃO PAULO
2009

FICHA CATALOGRÁFICA

Fugita, Henrique Shoiti

**MAPOS: Método de Análise e Projeto Orientado a Serviços /
H.S. Fugita. -- São Paulo, 2009.
175 p.**

**Dissertação (Mestrado) - Escola Politécnica da Universidade
de São Paulo. Departamento de Engenharia de Computação e
Sistemas Digitais.**

**1. Métodos de desenvolvimento de software 2. Arquitetura de
software 3. Ciclo de vida de desenvolvimento de software I. Uni-
versidade de São Paulo. Escola Politécnica. Departamento de
Engenharia de Computação e Sistemas Digitais II. t.**

A Priscila Miyuki Miura, pelo apoio e motivação prestados ao longo de todo o caminho percorrido para a elaboração deste trabalho.

A meus pais Marina Mariko Nagata Fugita e Dirceu Kasuo Fugita, por serem a minha principal referência e por terem tornado possível a conquista deste objetivo.

AGRADECIMENTOS

Ao meu orientador Prof. Dr. Kechi Hiramã, por todo o conhecimento e experiência passados durante o processo de elaboração deste trabalho.

À Profa. Dra. Selma Melnikoff e ao Prof. Dr. Pedro Corrêa, por sua participação na banca de qualificação e pelos comentários que muito contribuíram para o resultado final desta pesquisa.

À Escola Politécnica da Universidade de São Paulo, por ter me dado a oportunidade de cursar tanto a graduação quanto o mestrado.

À IntVision Solutions, por acreditar no resultado deste trabalho e me permitir a realização dos estudos de caso em seus projetos.

Aos meus companheiros de projeto, pela colaboração na realização dos estudos de caso.

A todos os meus familiares, amigos e colegas de trabalho que contribuíram, mesmo que de forma indireta, para a realização deste trabalho.

Adoramos a perfeição porque não a podemos ter; repugná-la-íamos se a tivéssemos. O perfeito é desumano porque o humano é imperfeito.

Fernando Pessoa

RESUMO

FUGITA, H. S. *MAPOS: Método de Análise e Projeto Orientado a Serviços*. São Paulo: Escola Politécnica, Universidade de São Paulo, 2009. 175 p. Dissertação de Mestrado em Engenharia.

Com a crescente adoção do conceito de Arquitetura Orientada a Serviços (SOA) pelas organizações, torna-se necessário solucionar um dos principais desafios trazidos por este estilo arquitetural, que é a análise e projeto de serviços. Alguns métodos de análise e projeto de soluções orientadas a serviços vêm sendo propostos, mas ainda estão longe de convergir em direção a uma padronização.

Este trabalho realiza uma avaliação crítica dos métodos existentes e levanta um conjunto de requisitos de análise e projeto orientado a serviços. Baseado nesta análise, é proposto um método para unificar as boas práticas dos métodos existentes e atender aos requisitos levantados. Para verificar a aplicabilidade do método, dois estudos de caso foram conduzidos em projetos reais.

Palavras-chave: Engenharia de Software, Arquitetura Orientada a Serviços (SOA).

ABSTRACT

FUGITA, H. S. *Service-Oriented Analysis and Design Method*. São Paulo: Escola Politécnica, Universidade de São Paulo, 2009. 175 p. Dissertação de Mestrado em Engenharia.

With the growing adoption of Service-Oriented Architecture (SOA) by the organizations, it becomes necessary to address one of the main challenges imposed by this architectural style, which is services analysis and design. Some service-oriented applications analysis and design methods have been proposed, but have not yet converged towards standardization.

This work performs a critical assessment of existing methods and describes a set of requirements of service-oriented analysis and design. Based on that study, a new method is proposed to unify the best practices of existing methods and satisfy the gathered requirements. In order to verify the applicability of the method, two case studies were conducted in real projects.

Keywords: Software Engineering, Service-Oriented Architecture (SOA).

LISTA DE FIGURAS

Figura 2.1: Agentes de uma arquitetura SOA (PAPAZOGLU, 2003).....	20
Figura 2.2: Hierarquia de uma arquitetura SOA (PAPAZOGLU; VAN DEN HEUVEL, 2006).....	22
Figura 2.3: Relacionamentos entre os componentes de uma arquitetura SOA (ERL, 2005)	23
Figura 2.4: Elementos de uma arquitetura SOA (MARKS; BELL, 2006)	27
Figura 2.5: Elementos de uma Arquitetura Orientada a Serviços (KRAFZIG; BANKE; SLAMA, 2004)	30
Figura 2.6: Camadas de uma arquitetura SOA (ERL, 2005)	39
Figura 2.7: Camadas de serviços (ERL, 2005).....	40
Figura 3.1: Fases do método (PAPAZOGLU; VAN DEN HEUVEL, 2006)	52
Figura 3.2: Identificação de processo e escopo de processo (PAPAZOGLU, 2006)	55
Figura 3.3: Arquitetura do <i>Rational Unified Process</i> (SHUJA, KREBS, 2007).....	63
Figura 3.4: Abordagens de Identificação de Serviços (IBM, 2007).....	67
Figura 3.5: Exemplo de modelo de processo de negócio (IBM, 2007)	68
Figura 3.6: Mapeamento de casos de uso para serviços (IBM, 2007)	69
Figura 3.7: Disponibilização de funções legadas como serviços (IBM, 2007)	70
Figura 3.8: Ciclo de vida de desenvolvimento SOA (ERL, 2005)	76
Figura 3.9: Análise Orientada a Serviços (ERL, 2005)	78
Figura 3.10: Modelagem de serviços candidatos (ERL, 2005)	80
Figura 3.11: Notação para representar um serviço (ERL, 2007)	82
Figura 3.12: Projeto orientado a serviços (ERL, 2005).....	83
Figura 3.13: Ciclo de Vida de Serviços (MARKS; BELL, 2006).....	90
Figura 3.14: Processo de identificação e análise de serviços (MARKS; BELL, 2006)	93
Figura 3.15: Mapa de granularidade de serviços (MARKS; BELL, 2006).....	96
Figura 3.16: Processo de projeto de serviços (MARKS; BELL, 2006).....	98
Figura 3.17: Mapa de demarcação (MARKS; BELL, 2006)	99
Figura 3.18: Tipos de transformação lógico-físico (MARKS; BELL, 2006)	101
Figura 5.1: Método de Análise e Projeto Orientado a Serviços	116
Figura 5.2: Modelo de Serviços	129
Figura 6.1: Diagrama do processo "Incluir cliente pessoa física"	137
Figura 6.2: Operações candidatas identificadas.....	140
Figura 6.3: Serviços candidatos identificados	141

Figura 6.4: Exemplos de classes de mensagens	142
Figura 6.5: Realização do serviço "SvcClientePF"	145
Figura 6.6: Realização do serviço "SvcCorporativo"	146
Figura 6.7: Realização do serviço "InfraCliente"	147
Figura 6.8: Diagrama do processo Recepção de Lote de NF-e (ENCONTRO..., 2006)	156
Figura 6.9: Exemplos de serviços especificados.....	163

LISTA DE TABELAS

Tabela 4.1: Resumo da aderência dos métodos analisados aos requisitos	114
Tabela 6.1: Especificação da tarefa “Reagendar Visita”	138
Tabela 6.2: Operações candidatas do processo "Inclusão de cliente pessoa física"	139
Tabela 6.3: Cenários de reuso para operações candidatas	143
Tabela 6.4: Operações candidatas identificadas	157
Tabela 6.5: Operações candidatas agrupadas por serviço candidato	158
Tabela 6.6: Descrição do serviço candidata "Regularidade Fiscal"	159
Tabela 6.7: Descrição da operação candidata "Verificar Regularidade Fiscal"	159
Tabela 6.8: Cenários de reuso para operações candidatas	160
Tabela 6.9: Alocação dos serviços em camadas	163

LISTA DE SIGLAS E ABREVIATURAS

API	<i>Application Programming Interface</i>
BPM	<i>Business Process Management</i>
BPMN	<i>Business Process Modeling Notation</i>
CASE	<i>Computer-Aided Software Engineering</i>
CBD	<i>Component-Based Development</i>
CNPJ	Cadastro Nacional da Pessoa Jurídica
COTS	<i>Commercial Off-The-Shelf</i>
CPF	Cadastro de Pessoas Físicas
CRM	<i>Customer Relationship Management</i>
ENCAT	Encontro Nacional de Coordenadores e Administradores Tributários Estaduais
ERP	<i>Enterprise Resource Planning</i>
ESB	<i>Enterprise Service Bus</i>
GUI	<i>Graphical User Interface</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
IDE	<i>Integrated Development Environment</i>
IE	Inscrição Estadual
KPI	<i>Key Performance Indicator</i>
MAPOS	Método de Análise e Projeto Orientado a Serviços
NF-e	Nota Fiscal Eletrônica
OASIS	<i>Organization for the Advancement of Structured Information Standards</i>
QoS	<i>Quality of Service</i>
ROI	<i>Return On Investment</i>
RUP	<i>Rational Unified Process</i>
SLA	<i>Service Level Agreement</i>
SOA	<i>Service-Oriented Architecture</i>
SOAP	<i>Simple Object Access Protocol</i>
SOC	<i>Service-Oriented Computing</i>
SOMA	<i>Service-Oriented Modeling and Architecture</i>
TI	Tecnologia de Informação
UDDI	<i>Universal Description, Discovery and Integration</i>
UML	<i>Unified Modeling Language</i>
WS-BPEL	<i>Web Services Business Process Execution Language</i>

WSDL	<i>Web Services Description Language</i>
XML	<i>Extensible Markup Language</i>
XSD	<i>XML Schema Definition</i>

SUMÁRIO

RESUMO

ABSTRACT

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

1	INTRODUÇÃO	14
1.1	Motivação	15
1.2	Objetivo	16
1.3	Justificativa	16
1.4	Organização	18
2	ARQUITETURA ORIENTADA A SERVIÇOS	19
2.1	Definição de Arquitetura Orientada a Serviços	19
2.2	Elementos de uma Arquitetura Orientada a Serviços	29
2.3	Princípios de Orientação a Serviços	33
2.4	Benefícios e Desafios	41
2.5	Relação com outros paradigmas	47
2.6	Considerações Finais	49
3	MÉTODOS DE ANÁLISE E PROJETO ORIENTADOS A SERVIÇOS	50
3.1	Papazoglou e Van Den Heuvel	51
3.2	RUP <i>plugin for</i> SOMA	61
3.3	Erl	75
3.4	Marks e Bell	88
3.5	Considerações Finais	104
4	REQUISITOS DE UM MÉTODO DE ANÁLISE E PROJETO ORIENTADO A SERVIÇOS	105
4.1	Descrição dos Requisitos	105
4.2	Aderência dos métodos analisados aos requisitos	113
4.3	Considerações Finais	114
5	PROPOSTA DE MÉTODO	115
5.1	Ciclo de Vida	115
5.2	Atividades de Modelagem de Negócio	116
5.3	Atividades de Análise	117

5.4	Atividades de Projeto.....	122
5.5	Atividades de Construção.....	127
5.6	Papéis.....	127
5.7	Artefatos	128
5.8	Análise do Método	129
5.9	Considerações Finais	130
6	ESTUDOS DE CASO.....	132
6.1	Caso Banco BES.....	132
6.2	Caso NF-e	152
6.3	Análise dos Resultados	167
6.4	Considerações Finais	169
7	CONCLUSÃO	170
7.1	Contribuições do Trabalho.....	170
7.2	Trabalhos Futuros	171
	REFERÊNCIAS.....	173

1 INTRODUÇÃO

Atualmente, cada vez mais as empresas pertencentes às mais diversas áreas fazem uso da Tecnologia da Informação (TI) para conduzir seus negócios.

Devido ao avanço da Internet e das tecnologias de integração e comunicação, é cada vez mais comum as organizações fazerem uso do meio eletrônico para realizar suas interações com clientes, fornecedores e parceiros.

As empresas vêm empregando com cada vez mais intensidade sistemas de informação no suporte aos seus processos de negócio, com o intuito de automatizar atividades e aumentar a produtividade. As empresas utilizam sistemas de informação como ERPs para controlar sua operação, CRMs para gerenciar informações sobre o relacionamento com seus clientes, *Supply Chain Management* para gerenciar interações com parceiros, fornecedores e consumidores. Dentro desta infra-estrutura de TI, cada sistema possui um conjunto de dados e informações próprio. Devido ao fato de os processos de negócio envolverem diversas áreas da organização, eles passam a demandar funções de mais de um sistema de informação e tornam necessária a troca de informações entre sistemas.

Principalmente nas organizações de maior porte, estes sistemas de informação constituem ambientes extremamente complexos e heterogêneos, onde as aplicações geralmente fazem uso de diferentes tecnologias, linguagens e protocolos. É comum encontrar em um mesmo ambiente aplicações desenvolvidas com tecnologia Web, softwares *Commercial Off-The-Shelf* (COTS) proprietários e aplicações legadas em mainframe. Para suportar a comunicação entre elas, são utilizados adaptadores, conectores e *middlewares* de integração.

Os negócios têm se tornado mais dinâmicos, exigindo mais agilidade por parte das organizações e, conseqüentemente, de suas áreas de TI. É necessário que a arquitetura de sistemas de informação seja capaz de se adaptar rapidamente às necessidades de negócio, em constante mudança. No entanto, em um ambiente de TI heterogêneo este tipo de mudança é difícil de realizar.

Dentro deste cenário, vem se desenvolvendo e sendo aceito o conceito de Arquitetura Orientada a Serviços (SOA). O modelo de referência do OASIS (OASIS SOA REFERENCE MODEL TC, 2006) define SOA como um paradigma para

organizar e utilizar competências distribuídas entre vários domínios. Trata-se de um modo de estruturar as aplicações de software em elementos chamados serviços.

1.1 Motivação

Para se projetar aplicações que efetivamente sigam a orientação a serviços e para que elas realmente façam uso dos benefícios trazidos pelo paradigma, é necessário haver uma atenção especial na identificação e especificação dos serviços que as compõem.

Os serviços que apoiarão os processos de negócio e antederão aos requisitos devem ser definidos de forma adequada, para que eles apresentem atributos como reusabilidade e flexibilidade e estejam em alinhamento com as metas de negócio. Além disso, os serviços devem ser especificados de modo a fazer parte de um repositório de serviços reusáveis, que podem ser então orquestrados para apoiar os processos de negócio.

É necessário haver algum tipo de método para, a partir dos requisitos e metas de negócio, identificar e modelar processos consistentes que façam uso de serviços disponíveis na infra-estrutura de TI ou de novos serviços a serem desenvolvidos. No caso de novos serviços, o método deve abordar atividades para especificá-los de modo que eles possam ser implementados utilizando métodos já existentes e consolidados.

Tal método deve ser previsível e repetível, conduzindo de maneira consistente as atividades de análise e especificação. De preferência, deve fazer uso de ferramentas e notações amplamente conhecidas para a modelagem e representação das informações e para a geração de artefatos. Desta maneira, a adoção do método seria facilitada e traria menos impactos aos processos de desenvolvimento já existente na organização.

1.2 Objetivo

O objetivo deste trabalho consiste em propor um método de análise e projeto de aplicações orientadas a serviços baseado na análise crítica de métodos existentes. Para tal, um conjunto de requisitos de análise e projeto orientados a serviços é definido para guiar a elaboração do método proposto.

O método apresentado tem como propósito disciplinar o procedimento de identificação, análise e especificação de um conjunto de serviços alinhados aos requisitos de negócio e aderentes aos princípios de orientação a serviços. A aplicação do método tem como resultado um grupo de especificações de serviços, que pode ser então implementados utilizando-se técnicas de desenvolvimento convencionais.

A proposta de método tem como ponto de partida a modelagem de negócio, na forma de casos de uso e modelos de processo de negócio, passa pela identificação e especificação dos serviços que compõem a solução, e tem como produto final os artefatos que compõem a especificação de cada serviço, incluindo a interface do serviço, suas operações e mensagens, e seus requisitos funcionais e não-funcionais.

Considerando-se o ciclo de vida dos processos de negócio e o ciclo de vida dos sistemas de informação como dois processos distintos porém interligados, pode-se dizer que tal método de análise e projeto orientado a serviços serviria como ponte para chegar-se às especificações dos sistemas a partir dos processos de negócio modelados. Os serviços seriam os elementos responsáveis por realizar esta ligação, por possuírem a característica de serem elementos desenvolvidos pela TI para prover funções de negócio.

1.3 Justificativa

Existem alguns poucos estudos na área de metodologias de desenvolvimento voltadas para SOA. Papazoglou e Van Den Heuvel (2006) descrevem as atividades do ciclo de vida dos serviços, que consiste num processo cíclico e iterativo, que

passa pela modelagem de processos, análise, projeto, implementação e execução e monitoria. O método de Papazoglou e Van Den Heuvel utiliza a tecnologia de *Web Services* para especificar e implementar os serviços.

A empresa IBM (2007) possui uma contribuição, que é um *plugin* a ser aplicado ao *Rational Unified Process* (RUP) para o desenvolvimento de aplicações orientadas a serviços. Este *plugin* acrescenta uma série de tarefas, papéis e artefatos ao RUP de modo a adaptá-lo para projetos de aplicações orientadas a serviços.

Em seu livro sobre SOA, Erl (2005) descreve um método de análise e projeto de serviços baseado nos princípios da orientação a serviços. Marks e Bell (2006) também propõem um método de desenvolvimento de serviços, focando nas etapas de análise e projeto.

Ainda não existe uma convergência das diversas propostas em direção a uma padronização, o que deixa o terreno aberto para discussão e apresentação de novas propostas (RAMOLLARI; DRANIDIS; SIMONS, 2007). Torna-se necessária uma análise das propostas existentes e dos benefícios que cada uma delas traz. Com base nesta análise, pode-se propor e instanciar um método de modo que unifique as boas práticas existentes.

O trabalho realizado tem como resultado três contribuições acadêmicas principais. A primeira consiste em uma análise crítica e comparativa das propostas de método de desenvolvimento orientado a serviços disponíveis atualmente. Com a crescente adoção do conceito de SOA pelas organizações, torna-se necessária uma discussão sobre como desenvolver software segundo o paradigma da orientação a serviços. Apesar de existirem diversas propostas de método, ainda não há uma abordagem predominante ou um consenso sobre o assunto. Por esse motivo, este trabalho busca contribuir realizando uma análise desses métodos, criticando-os e identificando as boas práticas propostas.

A segunda é o levantamento de um conjunto de requisitos de análise e projeto orientados a serviços. Baseado na análise dos métodos e das boas práticas identificadas, este trabalho levanta requisitos que devem ser atendidos por um método orientado a serviços.

A terceira contribuição é o próprio método elaborado a partir do estudo realizado. Além de discutir sobre as alternativas existentes, este trabalho ainda procura

contribuir apresentando uma nova proposta de método com o intuito de atender aos requisitos levantados. O resultado é um método de análise e projeto orientado a serviços com suas atividades, papéis e artefatos definidos. Além da descrição do método, são apresentados estudos de caso que documentam sua aplicação.

1.4 Organização

O presente trabalho está organizado em sete capítulos, mais a seção de referências.

O primeiro capítulo, a Introdução, apresenta o trabalho descrevendo suas motivações, seus objetivos e justificativas.

No segundo capítulo, Arquitetura Orientada a Serviços, é feita uma caracterização do conceito de SOA.

O terceiro capítulo, Métodos de Análise e Projeto Orientados a Serviços, contém as descrições e análises das propostas de métodos existentes.

O quarto capítulo, Requisitos de um Método de Análise e Projeto Orientado a Serviços, enumera e descreve uma série de requisitos levantados a partir das análises do capítulo três.

O capítulo cinco, Proposta de Método, mostra o resultado das análises dos métodos e dos requisitos, na forma de um novo método de análise e projeto SOA.

O sexto capítulo, Estudos de Caso, descreve os casos de aplicação do método a análise dos resultados.

Por fim, o capítulo Conclusão discute as contribuições do trabalho e propões trabalhos futuros que poderão dar continuidade ao método proposto.

Na seção Referências, são apresentadas as referências usadas para a elaboração deste trabalho.

2 ARQUITETURA ORIENTADA A SERVIÇOS

Para se poder discutir métodos de análise e projeto orientados a serviços, é necessário um entendimento claro do paradigma de Arquitetura Orientada a Serviços (SOA) e do próprio conceito de serviço em si. Por ser um tema que tem ganhado relevância há não muito tempo, não existe ainda um consenso definitivo sobre tais conceitos. Por meio da análise das características do paradigma e das interpretações de alguns autores, este capítulo busca estabelecer um entendimento sobre sua definição.

2.1 Definição de Arquitetura Orientada a Serviços

Basicamente, Arquitetura Orientada a Serviços (SOA) é um paradigma de construção e integração de software que estrutura aplicações em elementos modulares chamados serviços. O serviço, a unidade fundamental de uma arquitetura SOA, é um elemento computacional que tem como propósito desempenhar uma função específica e que pode ser utilizado por um cliente.

Segundo os conceitos básicos de SOA, um serviço é composto por uma interface e uma implementação. Geralmente, um serviço consiste em uma função de negócio desempenhada por um módulo de software (a implementação) e encapsulada por uma interface bem definida e acessível àqueles que desejam utilizar o serviço, ou seja, os potenciais clientes. Os clientes do serviço não têm acesso aos detalhes de como ele foi construído, mas apenas aos detalhes expostos em sua interface. A interface define as funções desempenhadas pelo serviço e eventuais precondições para utilizá-lo, mas não revela como estas funções são realizadas. Esta forma de encapsulamento é conhecida como caixa-preta e é um princípio característico de paradigmas como orientação a objetos e componentes de software. Porém, diferentemente destes, serviços representam funções completas de negócio e são projetados de modo a serem usados não somente no âmbito de um programa ou sistema, mas no âmbito da organização ou até entre organizações (PAPAZOGLU, 2003).

Desta maneira, uma arquitetura SOA possibilita uma infra-estrutura para computação distribuída, por meio de serviços que podem ser fornecidos e consumidos dentro de uma organização e entre organizações, por meio de redes de comunicação como a Internet.

Uma arquitetura SOA básica é caracterizada pelas interações entre três tipos de agentes de software: os Provedores de Serviço, os Consumidores (ou Clientes) de Serviço e o Registro de Serviço (HUHNS; SINGH, 2005). As interações entre estes agentes podem ser visualizadas na Figura 2.1.

Figura 2.1: Agentes de uma arquitetura SOA (PAPAZOGLU, 2003)

Os serviços são oferecidos pelos Provedores de Serviço, organizações responsáveis por desenvolver suas implementações, fornecer suas descrições e prestar suporte técnico e de negócio. De modo geral, os Provedores disponibilizam módulos de software (as implementações dos serviços) que podem ser acessados através de uma rede e publicam suas descrições em um Registro de Serviços, agente que abriga informações sobre as funções oferecidas, os requisitos para se utilizar o serviço e orientações sobre como realizar a interação. É o Registro de Serviços que torna essas informações disponíveis para serem consultadas por clientes em potencial. Por sua vez, os Consumidores de Serviço são os agentes que necessitam solicitar a execução de um Serviço. Os Consumidores buscam nos Registros a descrição de um serviço que satisfaça às suas necessidades e, ao encontrá-la, utilizam esta descrição para ligar-se ao Provedor e realizar a invocação do serviço. Note que os papéis de Provedor e Consumidor são lógicos, de modo que um mesmo agente pode exibir características de ambos dependendo do contexto (PAPAZOGLU, 2003).

A palavra serviço pode ser definida como a execução de trabalho ou desempenho de funções, ordenados por um requisitante. No contexto específico de sistemas de software, serviço ainda está ligado a conceitos como a capacidade de realizar trabalho para outro, a especificação do trabalho oferecido e a oferta de realizar trabalho para outro. No contexto de SOA, um serviço tem significado semelhante, já que consiste em um software capaz de realizar uma função específica quando solicitado por seus consumidores.

Os diversos autores da literatura sobre o assunto citam e descrevem seus próprios conjuntos de características de um serviço, mas não há um consenso sobre o que define exatamente um serviço, pois cada um foca em aspectos diferentes do paradigma. Por isso, cada uma destas características deve ser analisada para chegar-se a um entendimento do que vem a ser orientação a serviços. A seguir são analisadas as diversas interpretações do conceito de SOA, em ordem cronológica de publicação.

2.1.1 Papazoglou

Papazoglou (2003) discute o assunto SOA sob uma visão mais ampla, fazendo uso do termo Computação Orientada a Serviços (SOC). Papazoglou define SOC como sendo o paradigma de computação que utiliza serviços como elementos fundamentais para desenvolver soluções e aplicações distribuídas. Segundo o autor, um serviço seria um elemento computacional auto-descritivo e independente de plataforma que possibilita a composição de aplicações distribuídas de maneira rápida e com baixo custo. Os serviços podem executar funções dos mais diversos níveis de granularidade, representando uma simples resposta a uma requisição ou até um processo de negócio inteiro. Ainda, segundo Papazoglou, serviços são baseados em padrões e protocolos abertos de interação e descrição de interfaces.

Conforme pode ser visto na Figura 2.2, uma arquitetura orientada a serviços pode ser interpretada por uma hierarquia com diversos níveis de abstração. Segundo esta hierarquia, a organização pode ser dividida em domínios de negócio, que agrupam processos com funções e competências em comum. Um domínio é formado por processos de negócio, que por sua vez, são composições de diversos serviços de

negócio. Estes serviços de negócio são construídos sobre uma camada de serviços de infra-estrutura, que são implementados por componentes de software. Dando suporte a estes componentes, podem existir diversos tipos de sistemas de *back-end*, como sistemas de ERP e CRM, bases de dados, aplicações-pacote e aplicações legadas.

Figura 2.2: Hierarquia de uma arquitetura SOA (PAPAZOGLU; VAN DEN HEUVEL, 2006)

2.1.2 Erl

Thomas Erl (2005) descreve o conceito de SOA discursando sobre suas características e possíveis benefícios. Erl apóia fortemente o uso de padrões como o *Extensible Markup Language* (XML) (BRAY et al., 2006) para a representação de dados e a tecnologia *Web Services* para implementar as interações entre serviços. Apesar de abordar a implementação de serviços através do uso de *Web Services*, Erl procura esclarecer que uma arquitetura SOA não se limita a um ambiente de computação distribuída utilizando *Web Services*, mas também deve seguir os princípios de orientação a serviços.

Ao descrever a estrutura de uma arquitetura SOA, Erl faz analogia com a arquitetura de *Web Services* e define os componentes lógicos da arquitetura como sendo mensagem, operação, serviço e processo, cujas interações são exibidas na Figura 2.3. As mensagens são as unidades de comunicação e representam os dados necessários para a realização de uma unidade de trabalho e os dados produzidos por estas unidades. Já as operações, que por sua vez são unidades de trabalho executadas, processam os dados recebidos através das mensagens. Um serviço é composto por um conjunto de operações relacionadas, constituindo uma unidade de lógica de processamento. Por fim, um processo é uma unidade de lógica de automação e é formado pela agregação de diversas unidades de trabalho (operações executadas) coordenadas de acordo com algumas regras de negócio. O processo representa uma parcela grande de trabalho que é executada por uma série de unidades menores de trabalho.

Figura 2.3: Relacionamentos entre os componentes de uma arquitetura SOA (ERL, 2005)

Segundo Erl (2005), SOA tem como origem a teoria de engenharia de software conhecida como separação de assuntos. Segundo esta teoria, deve-se dividir um problema grande em assuntos menores para que a solução possa ser decomposta em várias partes, sendo que cada parte deve solucionar um assunto. Assim como a orientação a objetos, a orientação a serviços é uma maneira de se conseguir essa separação de assuntos, através de princípios como reuso, baixo acoplamento e abstração.

Em publicações posteriores (ERL, 2007), o autor aprofunda a discussão do paradigma de orientação a serviços, descrevendo-o como uma composição de diversos princípios.

2.1.3 IBM

Já a IBM, empresa multinacional de origem norte-americana que atua nas áreas de software, hardware e consultoria, publicou um livro sobre SOA (BIEBERSTEIN et al., 2005) onde são discutidas várias definições do termo. Algumas definições de negócio que citam características como baixo acoplamento, reuso e interoperabilidade são mostradas com outras definições mais técnicas, que citam a existência de interfaces invocáveis e a utilização de padrões de *Web Services*. A partir destas definições, os autores chegam à seguinte:

Uma arquitetura orientada a serviços é um arcabouço para integrar processos de negócio e a infra-estrutura de TI de suporte na forma de componentes padronizados e seguros – serviços – que podem ser reusados e combinados para atender prioridades de negócio em mudança constante.

A IBM ainda destaca o alinhamento entre os processos de negócio e a infra-estrutura de TI, conceituando serviços como elementos que podem ser reusados e combinados para atender às necessidades de negócio.

2.1.4 Marks e Bell

Segundo a definição de Marks e Bell (2006), SOA é uma arquitetura de negócios conceitual onde as funções de negócio, ou lógica de aplicação, são disponibilizadas para os consumidores na forma de serviços reusáveis e compartilhados em uma rede de TI. Serviços são módulos de funcionalidade de negócio com interfaces expostas que são invocados por meio de troca de mensagens entre consumidores e provedores de serviços.

Marks e Bell afirmam que o conceito de serviço é adequado para representar transações que a área de TI oferece para seus clientes e usuários de negócio. Serviço é um conceito facilmente compreendido pela perspectiva de TI e pela de

negócio, além de possuir a granularidade apropriada para ser mapeado para atividades automatizadas de um processo de negócio.

Segundo os autores, em uma organização podem existir dois tipos de serviços: os serviços de negócio e os serviços de tecnologia. Os serviços de negócio refletem conceitos e eventos relativos aos processos de negócio. A execução de um serviço de negócio pode ser associada à realização de uma função ou atividade de um processo. Estes serviços possuem alta granularidade e representam conceitos reais de negócio como abrir nova conta ou obter informações de cliente. Estes são exemplos de eventos reais que podem ser executados por uma pessoa (um atendente, por exemplo), por um sistema ou por um serviço de negócio.

Já os serviços de tecnologia são independentes de áreas ou domínios de negócio, sendo reusáveis em diversos processos, podendo ser comuns à organização como um todo. Eles representam serviços de TI, como segurança, auditoria, transformação de dados, entre outros. Os serviços de tecnologia provêm a infraestrutura de apoio aos serviços de negócio.

Devido à sua natureza, os serviços de negócio estabelecem um termo comum às linguagens de negócio e de TI, podendo assim, facilitar a comunicação e o alinhamento entre a área de TI e os usuários/clientes de negócio durante a análise e o projeto de serviços. Por ser a unidade fundamental de análise do SOA e por possuir correspondência direta com o conceito de processo de negócio, o serviço de negócio possibilita uma melhoria na identificação de requisitos de negócio e promove o alinhamento entre as áreas de negócio e de TI.

Serviços podem servir como abstrações para componentes de tecnologia, processos de negócio, entidades de dados ou elementos conceituais, encapsulando negócio e tecnologia. Baseado nisso, Marks e Bell (2006) afirmam que serviços podem ser interpretados por três diferentes visões: uma visão conceitual, uma visão de processo de negócio e uma visão tecnológica. A visão conceitual define um serviço com base em suas características, seu escopo lógico e o valor que ele agrega ao negócio. A visão de processo de negócio interpreta serviços como uma seqüência orquestrada de atividades de um processo. Já segundo a visão tecnológica, um serviço é uma abstração de uma solução tecnológica, isto é, uma interface para uma

implementação específica, como aplicações comerciais, sistemas legados ou plataformas de integração.

A Figura 2.4 mostra os elementos que formam uma arquitetura SOA segundo Marks e Bell (2006). A **estratégia de SOA** é composta pelo planejamento e pelas decisões relativas à arquitetura orientada a serviços da organização. Esta estratégia direciona o modelo de **governança de SOA** (KENNEY; PLUMMER, 2007), isto é, o conjunto de padrões e políticas que definem a visão conceitual da arquitetura, ou seu **modelo de arquitetura**. A governança abrange processos, papéis e responsabilidades que devem ser seguidos para garantir a conformidade da arquitetura implementada com o modelo de arquitetura. De acordo com a governança estabelecida, é aplicada a **tecnologia de implementação** que torna realidade a visão conceitual da arquitetura SOA. É esta tecnologia que possibilita a construção e operação dos **serviços**, que por sua vez são o núcleo da arquitetura. Eles constituem o principal ativo de uma arquitetura SOA e, por isso, necessitam de um processo de análise e projeto bem definido que garanta que eles possuam as características de reusabilidade, interoperabilidade, entre outras. A governança de SOA também deve levar em consideração aspectos de **comportamento e cultura** da organização, que poderão sofrer mudanças para adequação ao modelo de arquitetura. Por fim, um conjunto de **métricas** deverá ser estabelecido e aferido para verificar os resultados obtidos pela arquitetura SOA, tanto em termos de desempenho de processos de negócio quanto de retorno sobre investimento (ROI).

Figura 2.4: Elementos de uma arquitetura SOA (MARKS; BELL, 2006)

2.1.5 OASIS

O OASIS (*Organization for the Advancement of Structured Information Standards*) é uma organização sem fins lucrativos que trabalha no desenvolvimento de padrões abertos de Tecnologia de Informação. Dentre os padrões desenvolvidos, encontram-se diversas especificações e modelos relacionados a *Web Services* e SOA.

O OASIS elaborou um modelo de referência de SOA (OASIS SOA REFERENCE MODEL TC, 2006) com o intuito de definir os conceitos envolvidos em uma arquitetura SOA e os relacionamentos entre eles. O propósito deste modelo de referência é estabelecer uma base conceitual que servirá de guia para a criação de padrões e arquiteturas de referência e implementações. Este modelo aborda a arquitetura orientada a serviços como um modelo abstrato, não focando em requisitos técnicos ou especificando métodos de implementação.

Segundo a definição do OASIS, arquitetura orientada a serviços é um paradigma para organizar e utilizar competências distribuídas que podem estar sob o controle

de diferentes domínios. Competência é definida como um elemento criado para solucionar ou dar suporte para um determinado problema de negócio. E, no cenário da computação distribuída, é comum os requisitos de uma entidade poderem ser satisfeitos por uma competência oferecida por outro domínio.

Neste contexto surgem algumas questões, como a não existência de uma correlação exata entre necessidades e competências, a possibilidade das competências possuírem diversas granularidades e o fato de certas necessidades exigirem a combinação de várias competências. A arquitetura orientada a serviços vem para servir como um arcabouço para casar necessidades e competências e combinar competências para atender a estas necessidades.

Assim, chega-se à definição de serviços segundo o OASIS, que é o mecanismo que possibilita que necessidades e competências sejam combinadas.

O modelo de referência baseia o paradigma de orientação a serviços em três conceitos fundamentais: visibilidade, interação e efeito no mundo real. **Visibilidade** corresponde à possibilidade de as entidades com necessidades e competências verem umas às outras, isto é, de contatar-se. É obtida pela disponibilização de descrições das funções, dos requisitos técnicos, restrições e políticas e pela existência de um meio de conexão entre as entidades. **Interação** consiste no ato de se utilizar uma competência. Geralmente, a interação é realizada por meio de troca de mensagens entre os consumidores e provedores de serviço. Já o propósito de se utilizar uma competência é o de obter um ou mais **Efeitos no mundo real**. Um efeito é o resultado de uma interação e pode ser um retorno de informação ou uma mudança no estado de uma entidade envolvida na interação. Os efeitos no mundo real esperados determinam se uma competência atende aos requisitos de uma dada necessidade.

2.1.6 Open Group

O Open Group é outra organização que desenvolve padrões da área de TI, que produziu também uma definição do que é SOA (THE OPEN GROUP, 2006).

Segundo esta definição, arquitetura orientada a serviços é um estilo arquitetural que suporta orientação a serviços. Estilo arquitetural é definido como a combinação de características distintivas sobre a qual uma arquitetura é realizada ou expressada.

Ainda segundo o Open Group, um serviço:

- É uma representação lógica de uma atividade de negócio repetível que possui um resultado específico.
- É auto-contido.
- Pode ser composto por outros serviços.
- É uma caixa preta para seus consumidores.

O estilo arquitetural SOA possui as seguintes características distintivas:

- É baseado no projeto de serviços – que representam atividades de negócio do mundo real – compreendendo os processos de negócio da organização e entre organizações.
- A representação de serviço utiliza descrições de negócio para prover contexto (por exemplo, processo de negócio, meta, regra, política, interface de serviço e componente de serviço) e implementa serviços usando orquestração de serviços.
- Exige requisitos únicos da infra-estrutura – é recomendado que implementações usem padrões abertos para realizar interoperabilidade e transparência de localização
- Implementações são específicas de um ambiente – elas estão restritas ou habilitadas por contexto e devem ser descritas dentro deste contexto.
- Requer forte padronização de representação e implementação de serviço.
- Requer um Teste Litmus¹, que determina um “bom serviço”.

2.2 Elementos de uma Arquitetura Orientada a Serviços

Segundo Krafzig, Banke e Slama (2004), uma arquitetura orientada a serviços é composta por diversos elementos que se relacionam entre si e que, por sua vez,

¹ Um teste cujo resultado baseia-se em um único fator. Neste caso, o Teste Litmus indica se um serviço é “bom” ou não.

podem ser compostos por outros. Sendo assim uma arquitetura SOA pode ser representada por uma hierarquia de elementos, como na Figura 2.5.

Figura 2.5: Elementos de uma Arquitetura Orientada a Serviços (KRAFZIG; BANKE; SLAMA, 2004)

2.2.1 Aplicações *Front-end*

Como parte da arquitetura SOA, estão as aplicações de *front-end*, que constituem a camada de apresentação para o usuário final. São aplicações que possuem algum tipo de Interface Gráfica com o Usuário (GUI), como páginas *Web* e portais. É por meio destas interfaces que os usuários de negócio executam atividades de um processo de negócio, monitoram a execução e o desempenho de serviços e processos.

2.2.2 Serviço

Em uma arquitetura SOA, a camada de negócio não é formada por aplicações e sistemas separados, cada um com seus próprios conjuntos de dados e funções. As

funções são disponibilizadas na forma de serviços, que podem ser compostos para suportar um processo de negócio.

Serviços são compostos por um contrato, uma implementação e uma interface.

2.2.3 Repositório de Serviços

Um repositório de serviços é uma base de dados que provê informações sobre os serviços em uma arquitetura SOA. Potenciais consumidores consultam um ou mais repositórios para obter informações sobre serviços que podem satisfazer suas necessidades e sobre como acessá-los (o contrato do serviço). Para isso, um repositório pode prover funções de busca de serviços.

Demais informações que podem ser oferecidas por um repositório são: localização física do serviço, dados do provedor, custos de utilização, restrições de ordem técnica, requisitos de segurança e de nível de serviço.

Em implementações de SOA com *Web Services*, repositórios são comumente construídos baseados no padrão *Universal Description, Discovery and Integration* (UDDI).

2.2.4 Barramento de Serviços

O barramento de serviços, ou *Enterprise Service Bus* (ESB), é um *middleware* que possibilita a comunicação entre vários componentes de uma arquitetura SOA. O barramento pode adicionalmente prover funções como segurança, tolerância a falhas, transações distribuídas, auditoria, acesso unificado a diferentes modos síncronos e assíncronos, escalabilidade, entre outros (KEEN et al., 2005).

O ESB permite que um ambiente heterogêneo com diversas tecnologias e protocolos de comunicação possam se conectar em uma arquitetura SOA.

2.2.5 Contrato

O contrato oferece uma descrição do serviço de maneira independente de tecnologia, definindo funcionalidade, propósito, modo de utilização e restrições. O contrato do serviço normalmente é representado com o uso de uma linguagem, de preferência baseada em padrões abertos. Os contratos de serviços são geralmente armazenados e disponibilizados nos repositórios de serviço.

2.2.6 Implementação

É o elemento que efetivamente executa a funcionalidade definida pelo serviço.

Pelo fato de não ser acessada diretamente pelo cliente (este tem conhecimento apenas de sua interface), a implementação pode sofrer modificações de manutenção e evolução, sem que o impacto seja percebido pelo cliente. Uma implementação pode ser inclusive substituída sem que ocorra tal impacto. Isto se dá devido ao desacoplamento proporcionado pela interface.

Para executar a funcionalidade do serviço, a implementação combina lógica de negócio mais dados.

2.2.7 Interface

A interface é a porção do serviço exposta pelo Provedor para os Consumidores. Seu propósito é abstrair para os Consumidores os detalhes da implementação, expondo apenas as informações necessárias para que o Consumidor seja capaz de utilizar o serviço.

2.2.8 Lógica de Negócio

Corresponde às regras de negócio envolvidas na funcionalidade dos serviços e que são executadas por sua implementação.

2.2.9 Dados

As regras de negócio do serviço manipulam dados, que podem ser fornecidos pelos consumidores ou serem provenientes de aplicações de *back-end*, como bases de dados, sistemas de informação corporativos e até sistemas legados em alta plataforma.

2.3 Princípios de Orientação a Serviços

Segundo Erl (2005), o paradigma de orientação a serviços pode ser descrito a partir de uma série de princípios, que devem ser seguidos na construção de uma arquitetura: reuso, contrato padronizado, baixo acoplamento, abstração, facilidade de composição, autonomia, independência de estado, localização e interoperabilidade.

2.3.1 Reuso

A orientação a serviços visa tornar cada serviço reusável, mesmo que não haja requisitos específicos para um dado serviço no momento em que ele é desenvolvido. (ERL, 2005).

Serviços reusáveis são aqueles que podem possuir valor em diversos contextos de processos de negócio e interações intra e inter-organizacionais e, por isso, devem suportar diversos padrões de consumo. Devido a essa natureza, novos clientes podem reusar um serviço de uma maneira que não foi prevista em seu projeto original. Nestes casos, deve haver uma infra-estrutura de gerenciamento e governança de serviços que garanta o desempenho e devido funcionamento de tal serviço (MARKS; BELL, 2006).

Os serviços devem ser projetados considerando-se as várias formas de reuso, como transações entre aplicações, composição de serviços e processos de negócio e uso como serviços utilitários. A reusabilidade de um serviço é relacionada à sua

granularidade, sendo que quanto mais alta a granularidade de um serviço, menores são as possibilidades de ele ser usado em mais de um processo de negócio.

2.3.2 Contrato padronizado

Um serviço deve possuir um contrato bem definido que descreve para o mundo externo a funcionalidade por ele exposta e ao mesmo tempo encapsula os detalhes específicos de sua implementação (MARKS; BELL, 2006). O contrato, ou a interface do serviço, disponibiliza as seguintes informações para os consumidores: a localização do serviço, as operações executadas, as mensagens de entrada e saída utilizadas pelo serviço e regras para sua utilização (ERL, 2005). Podem conter também detalhes semânticos das operações do serviço.

O contrato representa um acordo firmado entre o Provedor do serviço e seus Consumidores, criando uma relação de dependência. Por isso, deve haver uma atenção especial na manutenção e versionamento destes documentos.

Contratos de serviço devem ser representados em uma linguagem ou notação bem definida, seguindo padrões abertos que garantam a interoperabilidade do serviço. No contexto de *Web Services*, contratos de serviço são definidos por documentos em *Web Services Description Language* (WSDL) (CHISTENSEN et al., 2006), esquemas na linguagem *XML Schema Definition* (XSD) (FALLSIDE, 2004) e políticas. Em um documento WSDL, são especificados os tipos de dados envolvidos, as mensagens e as operações do serviço. Já os esquemas XSD são a especificação dos tipos de dados e das mensagens XML que serão utilizadas.

2.3.3 Baixo acoplamento

Uma característica importante de serviços em uma arquitetura orientada a serviços é a de que eles devem possuir baixo acoplamento entre si, ou devem ser fracamente acoplados. No entanto, na literatura, o termo baixo acoplamento pode possuir vários significados práticos.

Um serviço fracamente acoplado pode significar um serviço cuja implementação pode ser substituída, modificada ou evoluída ao longo do tempo sem causar impactos nos Consumidores do serviço (MARKS; BELL, 2006). Neste caso, está-se referindo ao acoplamento entre a implementação do serviço e seus Consumidores.

Nesta mesma linha, baixo acoplamento pode ser indicado pela transparência de localização. Um serviço pode ser consumido independentemente de onde seu Provedor está fisicamente localizado, criando um desacoplamento entre Consumidor e Provedor (PAPAZOGLU, 2003).

De maneira geral, o baixo acoplamento é uma condição em que um serviço está ciente da existência de outros serviços mas ainda é independente deles (ERL, 2005).

2.3.4 Abstração

Um princípio importante na orientação a serviços é a abstração ou encapsulamento da lógica interna de um serviço por meio de sua interface. Ao separar interface de implementação, o serviço age como uma caixa preta e expõe aos seus Consumidores somente informações sobre as operações disponíveis.

A granularidade de um serviço é diretamente relacionada com a quantidade de funcionalidade que é encapsulada pelas operações de um serviço (ERL, 2005).

2.3.5 Facilidade de composição

Serviços devem possuir facilidade de composição, isto é, devem ser projetados de forma que possam participar de composições com outros, formando serviços compostos, ou ser orquestrados no contexto de um processo de negócio. Para isso, devem ser desacoplados e suas operações devem ser as mais atômicas possíveis (MARKS; BELL, 2006) (ERL, 2005).

2.3.6 Autonomia

Autonomia exige que a lógica de processamento encapsulada por um serviço fique restrita dentro de uma certa fronteira estabelecida, o que evita a dependência com relação a outros serviços. Isto permite que um serviço tenha controle total sobre sua própria lógica e possa exercer um tipo de auto-governança.

A autonomia de um serviço pode-se dar em dois níveis diferentes (ERL, 2005): no nível do serviço e autonomia pura.

Na autonomia no nível do serviço, os limites entre os serviços são distintos, mas eles ainda podem compartilhar recursos encapsulados, como bases de dados e sistemas legados. Este cenário é comum no caso de diversos serviços desenvolvidos a partir de uma mesma aplicação já existente. Neste caso, os serviços compartilham a aplicação, mas fazem uso dela de maneira independente.

Na autonomia pura, a lógica e os dados encapsulados estão sob completo controle do serviço, não sendo compartilhada com outros.

2.3.7 Independência de Estado

Um serviço deve evitar armazenar informações de estado e contexto ou minimizar o período de tempo que estas informações são mantidas (ERL, 2005). Isto quer dizer que a resposta do serviço para uma mensagem deve ser independente das mensagens anteriores processadas pelo serviço. Dependência de contexto ou de estado diminui o potencial de reuso do serviço. Além disso, enquanto o serviço reter informações de estado de um Consumidor, ele se torna indisponível para atender chamadas de outros Consumidores.

Para tornar um serviço o mais independente possível de estado, as mensagens devem ser inteligentes, isto é, deve ser adicionado às mensagens o máximo de informação para que elas se tornem auto-suficientes e possam ser processadas sem a necessidade de dados de estado.

2.3.8 Localização

Os serviços devem ser localizáveis, o que significa que suas interfaces devem ser de alguma forma publicadas e tornadas visíveis para os potenciais Consumidores. Na arquitetura SOA básica, essas interfaces são publicadas em um Registro de Serviços, de onde elas podem ser buscadas pelos Consumidores (MARKS; BELL, 2006).

2.3.9 Interoperabilidade

Serviços devem ser interoperáveis, isto é, deve ser possível interagir com um serviço independentemente da tecnologia empregada em sua implementação. Quaisquer que sejam a linguagem de programação, a plataforma ou o sistema operacional utilizados na construção dos Provedores, Clientes e Registros de Serviços, as interações entre eles devem ser possíveis. Para isso, essas interações devem ser realizadas utilizando-se padrões e protocolos abertos compatíveis com qualquer tecnologia ou plataforma. Por isso, em uma arquitetura orientada a serviços, a provisão e o consumo de serviços ocorrem de maneira transparente às tecnologias e linguagens utilizadas em sua implementação, garantindo a interoperabilidade entre serviços.

Atualmente, os padrões relacionados à tecnologia *Web Services* (WSDL, SOAP, UDDI) permitem a descrição de interfaces, a troca de mensagens e a busca por serviços de maneira neutra às plataformas e linguagens utilizadas. Por basearem-se em padrões abertos como o XML e o protocolo HTTP, que são largamente aceitos e suportados, a tecnologia *Web Services* possibilita a obtenção desta interoperabilidade.

2.3.10 Granularidade de Serviço

Granularidade é um aspecto importante para se determinar o escopo de um serviço. Ao se identificar um serviço, deve-se determinar se ele possuirá baixa granularidade

(serviço técnico ou de baixo nível) ou alta granularidade (serviço de negócio ou de alto nível). A granularidade depende de quanta funcionalidade é encapsulada por um serviço, sendo que quanto mais funcionalidade, mais abrangente é o escopo do serviço e mais alta é sua granularidade. Serviços de granularidade mais alta são utilizados diretamente pelo negócio e podem ser formados pela composição de serviços de granularidade mais baixa.

Segundo a interpretação da empresa IBM (BIEBERSTEIN et al., 2005), os dois tipos de serviços são necessários. Uma arquitetura SOA deve conter serviços diretamente relacionados e com alto valor para o negócio e serviços técnicos que encapsulam lógica com alto potencial de reuso. Os serviços de granularidade mais alta terão menos consumidores, mas agregarão mais valor para o negócio. Eles terão também alta dependência em relação a serviços de granularidade baixa, que por sua vez, terão alto potencial de reuso e terão baixa dependência com outros serviços.

Já Marks e Bell afirmam que serviços devem possuir alta granularidade e representar funções, processos e transações de negócio, encapsulando demais componentes de baixa granularidade (MARKS; BELL, 2006). Segundo os autores, serviços de baixa granularidade, como componentes de software, teriam pouca utilidade direta para o negócio. Por isso, eles não compensariam o *overhead* de se transformar em um serviço e deveriam ser encapsulados por serviços maiores.

2.3.11 Camadas de Serviço

A orientação a serviços pode trazer uma abstração entre a lógica de negócio e a infra-estrutura de TI, gerando um desacoplamento entre os modelos de processos de negócios e a arquitetura de sistemas de informação. Em uma arquitetura orientada a serviços dando suporte tecnológico aos processos de negócio de uma empresa, a camada de serviços localiza-se exatamente entre a camada de processos e a infra-estrutura de aplicações de TI (ERL, 2005). A camada de serviços cria uma abstração entre os processos de negócio e as aplicações (BIEBERSTEIN et al., 2005), conforme mostra a Figura 2.6.

Figura 2.6: Camadas de uma arquitetura SOA (ERL, 2005)

Baseado no princípio de composição, é possível dividir a camada de serviços em mais três camadas de abstração que determinam o tipo e a granularidade dos serviços. Assim, temos: a camada de serviços de aplicação, a camada de serviços de negócio e a camada de orquestração de serviços. Na Figura 2.7, pode ser visualizada a hierarquia das camadas de serviços.

Figura 2.7: Camadas de serviços (ERL, 2005)

Os serviços de aplicação possuem menor granularidade e representam os serviços de infra-estrutura de uma arquitetura SOA, oferecendo funções específicas de tecnologia. O propósito dos serviços de aplicação é prover funções reusáveis e processar dados contidos em sistemas legados e novas aplicações desenvolvidas.

Os serviços de negócio são os elementos fundamentais da arquitetura SOA, pois são aqueles que representam a lógica de negócio da organização. São formados pela composição de diversos serviços de aplicação para implementar a lógica de negócio. Podem representar tarefas de processo ou entidades de negócio.

Por fim, a camada de orquestração de serviços realiza a ligação entre os modelos de processos de negócio e os serviços da arquitetura SOA. O conceito de orquestração baseia-se na construção de processos de negócio a partir da composição de diversos serviços, por exemplo, utilizando uma linguagem de orquestração como o *Web Services Business Process Execution Language* (WS-BPEL) (OASIS WSBPEL TC, 2007). Trata-se de uma linguagem para se especificar o fluxo de trabalho e lógica de negócio envolvidos nas chamadas de *Web Services*. Nesta camada estão os serviços de processo, que possuem alta granularidade e representam processos de negócio completos, encapsulando a lógica e regras de negócio envolvidas.

2.4 Benefícios e Desafios

Atingindo-se um entendimento sobre o conceito de SOA, pode-se pensar sobre quais resultados podem ser obtidos por uma organização ao adotar a orientação a serviços. Implementar e manter uma arquitetura orientada a serviços dentro de uma organização é uma tarefa custosa, que exige tempo, recursos e esforço, além de promover mudanças no comportamento e na cultura da organização. Os resultados do SOA devem ser tangíveis e devem compensar o investimento realizado.

Desta maneira, são discutidos os benefícios e possíveis desafios que podem ser trazidos pela adoção da orientação a serviços nas organizações.

2.4.1 Facilidade de integração

Uma arquitetura SOA é baseada em serviços intrinsecamente interoperáveis (ERL, 2005). Devido ao uso de padrões abertos de interação, aplicações estruturadas na forma de serviços inerentemente possuem a capacidade de se comunicarem de maneira interoperável. É como se as aplicações orientadas a serviços já nascessem pré-integradas, minimizando o esforço de se desenvolver integrações entre aplicações.

O modelo de serviços tende a facilitar a integração de aplicações, uma vez que uma arquitetura SOA é uma maneira de se reorganizar uma infra-estrutura formada de silos de software em um portfólio de serviços compartilhados. Em uma arquitetura SOA, as aplicações existentes e futuras podem acessar os serviços sem a necessidade de se desenvolver integrações ponto-a-ponto baseadas em protocolos proprietários. Assim, esta arquitetura pode ser utilizada em ambientes com múltiplas aplicações baseadas em variadas tecnologias e plataformas que necessitam comunicar-se entre si. Para efetuar transações entre elas, pode-se conectar e reusar serviços com esforço mínimo de programação e integração (PAPAZOGLU, 2003).

Marks e Bell chegam até a chamar este benefício de integração de custo zero (MARKS; BELL, 2006). Tal afirmação baseia-se no fato de que não é necessário nenhum esforço para realizar a conversão entre protocolos e formatos para se

estabelecer a comunicação entre aplicações em uma arquitetura SOA, uma vez que elas já estariam baseadas nos mesmos padrões abertos de interação.

2.4.2 Padronização das tecnologias

Como parte da implementação da arquitetura SOA em uma organização, é desenvolvida toda uma infra-estrutura de comunicações para viabilizar as interações inter e intra-aplicações seguindo os princípios de orientação a serviços. Esta infra-estrutura de comunicação de serviços passa a fazer parte da infra-estrutura de TI e torna-se o padrão de comunicação no âmbito da organização como um todo. Deste modo, a organização investe seus recursos focando-se em uma única plataforma tecnológica de comunicação (ERL, 2005).

Além disso, se forem utilizados a tecnologia de *Web Services* e os padrões baseados em XML para a implementação da arquitetura SOA, tem-se a oportunidade de estabelecer o XML como uma plataforma padronizada de representação de dados na organização (ERL, 2005). Tendo-se um formato padrão para representar dados pode reduzir a complexidade do ambiente de aplicações. Desta maneira, estabelecem-se os documentos XML como o padrão para troca de informações e os esquemas XML como o padrão para representação das entidades de dados.

A padronização traz seus benefícios para a organização, mas sua implementação torna-se um desafio importante para as organizações ao adotar o conceito de SOA (ERL, 2005). É necessário garantir que os esquemas definidos nos vários projetos sigam as mesmas diretrizes e padrões e formem um modelo de dados organizacional, ou corre-se o risco de obter uma arquitetura não flexível e acoplada a aplicações existentes.

2.4.3 Maior reuso de recursos

A orientação a serviços prega que os serviços sejam especificados de forma que possam suportar o reuso de maneira inerente. O projeto de aplicações voltadas para

o reuso permite um reaproveitamento imediato de lógica pré-existente, bem como cria oportunidades futuras de reuso por potenciais clientes (ERL, 2005). No longo prazo, o reuso de serviços tende a reduzir custos, prazos e esforço para implementação de soluções (BIEBERSTEIN et al., 2005).

Apesar de o projeto de serviços reusáveis exigir um esforço adicional (se comparado ao esforço de se desenvolver um serviço sem suporte a reuso), tal investimento é recompensado quando futuras aplicações desenvolvidas reaproveitam as funções existentes. Marks e Bell estimam como algo em torno de 50% o esforço adicional para tornar um serviço reusável. Portanto, o ROI é obtido no primeiro reuso do serviço, se considerarmos que o reuso representa uma economia do custo de se desenvolver o serviço a partir do zero (MARKS; BELL, 2006).

O uso de *Web Services* permite também encapsular funções de ambientes legados de forma que estes possam fazer parte de uma arquitetura SOA. Isso permite a interoperabilidade de sistemas legados sem a necessidade de integrações ponto-a-ponto, que são custosas e inflexíveis. Assim, reduz-se o custo de se integrar legados e novas aplicações e torna-se possível que as novas aplicações reusem funções existentes no legado dentro do contexto de orientação a serviços. Com a possibilidade de se reusar sistemas legados, a necessidade de se substituí-los passa a não ser tão imediata (ERL, 2005).

2.4.4 Agilidade no negócio

A orientação a serviços promove flexibilidade nas soluções construídas, uma vez que são constituídas por serviços fracamente acoplados, sendo, portanto, preparadas para mudanças. O projeto de serviços com baixo acoplamento e a possibilidade de realizar composições entre eles torna a arquitetura SOA um ambiente mais adaptativo. A estruturação dos serviços em camadas de negócio e de tecnologia também gera flexibilidade, pois permite que ambos os domínios evoluam e sofram alterações de maneira isolada (ERL, 2005).

A orientação a serviços pressupõe que as aplicações construídas tendem a evoluir com o tempo, conforme os requisitos de negócio se alteram ou novos surgem. Em uma arquitetura SOA bem estruturada, o impacto dessa evolução é minimizado, uma

vez que propriedades como reuso e interoperabilidade permitem que a área de TI responda mais rapidamente às solicitações da área de negócio. Isto promove a agilidade organizacional, que permite à organização responder prontamente aos eventos do ambiente, reduzir o *time-to-market* de seus produtos e serviços e obter vantagem competitiva. Não somente o tempo para se ter uma solução pronta é diminuído, como também o custo e o esforço (ERL, 2005).

Deste modo, a orientação a serviços busca responder aos requisitos de negócio em prazos mais curtos e oferecer soluções mais flexíveis. As mudanças, que costumam ser custosas e danosas em ambientes não flexíveis, passam a ser facilitadas (BIEBERSTEIN et al., 2005).

Para se atingir tal estado, é necessário que a infra-estrutura esteja padronizada e que os serviços efetivamente possuam características de baixo acoplamento, reusabilidade e interoperabilidade.

2.4.5 Alinhamento com o negócio

A orientação a serviços promove o alinhamento estratégico entre TI e negócio. Devido ao fato de os serviços representarem conceitos de negócio, o vínculo entre as soluções entregues pela área de TI (os serviços) e as metas estratégicas da organização (tanto de TI como de negócio) torna-se mais claro. Desta maneira, os investimentos destinados a TI são justificados devido a essa percepção do valor agregado (BIEBERSTEIN et al., 2005).

O serviço pode ser visto também como uma maneira de uniformizar os vocabulários das áreas de negócio e de TI dentro de uma organização. Em uma organização com uma arquitetura SOA, o conceito de serviço pode tornar-se um elemento de ligação entre os profissionais de TI e os usuários/clientes das áreas de negócio. Com um termo comum no vocabulário de ambas as áreas, torna-se possível estabelecer metas de negócio baseadas diretamente no conceito de serviço e tendo em mente os serviços existentes, além de melhorar a comunicação e o entendimento no relacionamento TI-negócio (MARKS; BELL, 2006).

Quando o paradigma de orientação a serviços é aplicado para dar suporte aos processos de negócio, a noção de atividade de processo pode ser mapeada diretamente à execução de um serviço de negócio orquestrado pelo processo.

Com o surgimento de linguagens para especificação e execução de processos de negócio como o WS-BPEL, reduziu-se o *gap* que existia entre os modelos de processos elaborados pelos analistas de negócio e a implementação do fluxo de trabalho desenvolvida pelos profissionais de TI. Quando um modelo de processo é elaborado seguindo a sintaxe do WS-BPEL, o próprio modelo torna-se o código executável que será utilizado na implementação do processo (ERL, 2005).

2.4.6 Requisitos de desempenho

Em uma arquitetura com um grande número de serviços interagindo uns com os outros, é esperado um aumento no volume de comunicações baseadas em mensagens. Caso o ambiente de execução destes serviços não esteja preparado para tal demanda, pode começar a haver latência de processamento e degradação do tempo de resposta das operações (ERL, 2005).

Como a arquitetura orientada a serviços pode ser composta de diversas camadas, com necessidade de processamento nas interações entre elas, é esperado um aumento nos requisitos de desempenho. Além, disso o processamento e transporte de dados nos formatos e protocolos da tecnologia *Web Services*, como XML, XSD e SOAP, tende a ser custoso para a infra-estrutura. Ao adotar a orientação a serviços, as organizações devem atentar para tais requisitos de desempenho no momento de dimensionar a infra-estrutura necessária para seu ambiente.

2.4.7 Requisitos de segurança

Em um ambiente de serviços autônomos disponíveis para quaisquer potenciais consumidores, surge a preocupação com a segurança das operações dos serviços. Alguns tipos de operações mais críticas necessitam meios de garantir que apenas consumidores autorizados possam utilizá-las. Além disso, como a comunicação

entre provedores e consumidores de serviços é baseada em mensagens, é preciso assegurar a integridade e confidencialidade de certos tipos de informação (ERL, 2005).

Deste modo, é necessário que a infra-estrutura e as ferramentas utilizadas na implantação de uma arquitetura SOA possuam suporte a mecanismos e tecnologias que atendam a tais requisitos de segurança para serviços. O padrão WS-Security busca solucionar esses requisitos em um ambiente de *Web Services*.

2.4.8 Complexidade de análise e projeto

O desenvolvimento de soluções compostas por diversos serviços independentes e desacoplados traz um aumento na complexidade da arquitetura e introduz um número maior de pontos de falha (BIEBERSTEIN et al., 2005).

Além disto, é preciso assegurar que os serviços desenvolvidos possuam características aderentes aos princípios de orientação a serviços. Sem essas características, pode-se acabar com uma arquitetura de serviços que não trazem os benefícios esperados, como reuso, facilidade de integração e alinhamento com o negócio.

Segundo Papazoglou e Van Den Heuvel (2006), os paradigmas de desenvolvimento existentes, como a orientação a objetos e o desenvolvimento baseado em componentes, não são diretamente aplicáveis a arquiteturas orientadas a serviços e baseadas em tecnologia *Web Services*. Deste modo, fazem-se necessários técnicas e métodos específicos para análise e projeto de soluções orientadas a serviços.

2.4.9 Necessidade de governança

Com a existência de um portfólio de serviços disponíveis para toda a organização, pode haver diversas equipes de projeto desenvolvendo e utilizando serviços deste portfólio ao mesmo tempo. Um serviço deixa de ser responsabilidade de uma única equipe de projeto e deixa também de pertencer a um único processo de negócio.

O paradigma de orientação a serviços exige que seja definida uma estrutura de governança na organização, de modo a estabelecer papéis, responsabilidades, processos para organizar os serviços desenvolvidos (ERL, 2007). A governança define mecanismos para desenvolver, manter e divulgar os serviços do portfólio e garantir a padronização da arquitetura.

2.5 Relação com outros paradigmas

O paradigma de orientação a serviços não significa uma ruptura com relação a outros paradigmas de desenvolvimento de software, pois representa uma evolução derivada da orientação a objetos e do desenvolvimento baseado em componentes e pode inclusive ser aplicado juntamente com eles. A seguir, é descrita a relação da orientação a serviços com esses outros paradigmas e suas diferenças.

2.5.1 Orientação a objetos

Conforme citado anteriormente, o paradigma de orientação a serviços tem como ponto em comum com o paradigma de orientação a objetos o fato de ambas serem maneiras de se construir software com base na separação de assuntos (ERL, 2005). Princípios como abstração, encapsulamento e composição de serviços foram formulados a partir de conceitos de orientação a objetos. Além disso, orientação a objetos é comumente utilizada para implementar a lógica de aplicação encapsulada em um serviço. Entretanto, os dois paradigmas possuem algumas diferenças que serão discutidas a seguir.

A orientação a serviços prega o baixo acoplamento entre suas unidades (os serviços). Apesar de objetos possuírem rotinas desacopladas e até reusáveis, as classes por definição possuem relacionamentos entre si (agregação, herança), o que gera certo grau de dependência entre os objetos.

Na orientação a serviços, muito das informações necessárias para o processamento está contido nas mensagens, de forma que os serviços guardem o mínimo possível de informação de estado. A orientação a serviços encoraja a amarração da lógica de

processamento com dados, mantendo mais informação contida nos objetos e criando uma dependência de estado.

2.5.2 Orientação a componentes

Apesar de serem relacionados, os conceitos de componente e de serviço guardam diferenças entre si e não devem ser confundidos. Alguns princípios da orientação a serviço, como a abstração de interface e implementação e a transparência de localização, são oriundos de boas práticas da orientação a componentes. Porém, o serviço difere dos componentes ao não se limitar a uma determinada plataforma, linguagem ou tecnologia de implementação.

Algumas fontes afirmam que serviços podem ser implementados utilizando-se componentes (ENDREI et al., 2005), o que pode levar à falsa idéia de que componentes e serviços podem ser mapeados de um para um. Existem dois pontos a serem considerados que desfazem esse tipo de conclusão.

O primeiro é a granularidade de componentes e serviços. As funções oferecidas por componentes em geral possuem granularidade baixa, não possuindo valor direto para os processos de negócio (MARKS; BELL, 2006). Serviços devem possuir granularidade um pouco mais alta, pois representam atividades de negócio e provêm funções de utilidade direta para o negócio. Portanto, a implementação de um serviço geralmente é construída pela composição de mais de um componente de baixa granularidade, resultando em algo de granularidade um pouco mais alta.

O segundo é o fato de, apesar de possuírem interface e implementação separados assim como os componentes, os serviços operam sob um tipo de contrato que estabelece um acordo e cria expectativas com base em suas características semânticas. Tais características, pelo fato de serem complexas e de natureza humana, não podem ser representadas por um simples conjunto de assinaturas de funções, como as descrições de interface de arcabouços de componentes atuais (PERREY; LYCETT, 2003). Mesmo a tecnologia de *Web Services*, que é frequentemente empregada na implementação de SOA, ainda não oferece uma maneira consolidada de representar estas características semânticas.

2.6 Considerações Finais

Neste Capítulo 2, foi apresentado o conceito de SOA e foram discutidas algumas de suas definições e características. Tais definições e características deverão ser mantidos em mente para se poder avaliar métodos de análise e projeto orientados a serviços e propor melhorias de um novo método.

O paradigma de orientação a serviços tende a ser cada vez mais adotado pelas organizações, tendo em vista benefícios que se espera que ele traga. Porém, esta transição impõe alguns desafios a serem considerados. Um novo método de análise e projeto orientado a serviços deve buscar resolver de modo direto a complexidade de análise e projeto, e indiretamente tratar requisitos de desempenho, segurança e governança.

Na elaboração deste método, é especialmente importante focar nos princípios de orientação a serviços e em conceitos como granularidade e camadas de serviços. São estes os conceitos-chave que definem as características desejadas nos serviços desenvolvidos como resultado da aplicação do método.

3 MÉTODOS DE ANÁLISE E PROJETO ORIENTADOS A SERVIÇOS

Como discutido no capítulo anterior, a orientação a serviços possui uma série de características e princípios peculiares e pode trazer benefícios que motivam sua adoção dentro das organizações. Mas para que as promessas de benefícios da orientação a serviços se tornem realidade é fundamental que seus elementos principais – os serviços – apresentem as características adequadas e sigam os princípios do paradigma.

Deste modo, um grande desafio à implementação de SOA é estabelecer um método para desenvolver os serviços que satisfazem as metas do negócio e agregam valor a ele. É necessário garantir que os serviços desenvolvidos na organização sejam eficientes e eficazes. Devem ser eficientes no sentido de ser bem projetados e operar de acordo com os requisitos funcionais, de desempenho e de qualidade. E também devem ser eficazes, atendendo às necessidades críticas para o negócio.

É necessário que a concepção do serviço – as atividades de análise e projeto – seja realizada de forma que o produto final seja uma especificação de serviço que possua características como reusabilidade, baixo acoplamento e granularidade adequada, além de oferecer valor direto para o negócio.

A seguir, são descritos alguns métodos de análise e projeto orientados a serviços presentes na literatura. A descrição de cada método contém a seguinte estrutura:

- **Ciclo de Vida:** Os métodos de análise e projeto analisados fazem parte de um ciclo de vida completo de desenvolvimento de software, que vai desde a concepção até a execução dos serviços. Esta seção descreve como as atividades de análise e projeto posicionam-se no contexto do ciclo de vida completo.
- **Atividades de Análise:** Descreve as atividades da fase de análise propostas pelo método.
- **Atividades de Projeto:** Descreve as atividades da fase de projeto propostas pelo método.
- **Papéis:** Descreve os papéis, ou seja, os perfis responsáveis por executar as atividades do método.

- Artefatos: Descreve os produtos de trabalho gerados, modificados ou utilizados durante as atividades do método.
- Análise do Método: Análise crítica do método destacando as boas práticas propostas e eventuais limitações.

3.1 Papazoglou e Van Den Heuvel

Os autores Papazoglou e Van Den Heuvel (2006) argumentam que uma arquitetura SOA consiste de sistemas em múltiplas camadas, com diversas tecnologias de implementação, o que resulta em um grande leque de requisitos de negócio e de desempenho, assim como em diferentes contextos de execução e reuso. Para realizar a especificação, construção e refinamento de processos de negócio a partir de serviços, é necessário um processo de desenvolvimento baseado em serviços.

Em artigo publicado, os autores descrevem o processo Ciclo de Vida de Desenvolvimento de *Web Services*, que é parcialmente baseado em outros processos já estabelecidos como o RUP, o desenvolvimento baseado em componentes e a Modelagem de Processos de Negócio.

Para Papazoglou e Van Den Heuvel (2006), um método de desenvolvimento orientado a serviços deve focar-se nos processos de negócio, que são os elementos reusáveis totalmente independentes de aplicações e plataformas. Para que os processos de negócio desenvolvidos a partir da base de serviços criados e existentes sejam realmente úteis e confiáveis, dois princípios devem ser levados em conta durante todo o ciclo: baixo acoplamento e alta coesão entre os serviços.

3.1.1 Ciclo de Vida

O método de análise e projeto apresentado faz parte de um processo cíclico, executado em iterações. Essa abordagem cíclica apregoa um processo contínuo de descoberta, criação e implementação com cada iteração incorporando mais aos artefatos da solução, de modo previsível e repetível. O processo é composto por uma fase preparatória e mais oito fases principais distintas, conforme a Figura 3.1.

Figura 3.1: Fases do método (PAPAZOGLOU; VAN DEN HEUVEL, 2006)

A fase preparatória, a de **Planejamento**, consiste na preparação do projeto da solução. Nesta etapa, é determinada a viabilidade, a natureza e o escopo da solução. Deve-se estabelecer uma clara compreensão sobre o ambiente de negócio em questão e inserir os controles necessários ao projeto. Algumas atividades desta fase seriam: análise das metas de negócio, análise do ambiente tecnológico, concepção em alto nível dos novos requisitos e o mapeamento destes requisitos para as alternativas de implementação. A fase de planejamento inclui também atividades gerenciais como análise de custos, análise de retorno de investimento e elaboração do plano de projeto, com tarefas, entregáveis e prazos.

O ciclo inicia-se na fase de **Análise**, durante a qual são definidos os requisitos da aplicação, que devem ser derivados das metas de negócio que direcionam o desenvolvimento dos processos de negócio. Nesta fase, deve ser identificado o modelo *as-is* dos processos, que sofrerá reengenharia, resultando no modelo *to-be*. É feita então uma análise de *gap*, para determinar quais dos serviços necessários já existem no portfólio e quais devem ser desenvolvidos. Finalmente, a análise de realização define qual será a estratégia de implementação dos novos serviços.

Em seguida vem a fase de **Projeto**, quando os processos e serviços conceituais definidos na Análise são transformados em especificações de interfaces

independentes de plataforma. Nesta fase, duas atividades paralelas ocorrem: a produção dos serviços reusáveis e a composição destes em processos. O projeto de aplicações orientadas a serviços exige que as interfaces sejam muito bem definidas e documentadas antes de serem implementadas. Além disso, deve-se garantir que os serviços projetados possuam as características adequadas de granularidade, reusabilidade e facilidade de composição.

Na fase de **Construção**, é desenvolvida a implementação dos serviços e a descrição de suas interfaces e implementações. Estes serviços podem ser implementados através da criação de novos serviços a partir do zero, da exposição de aplicações existentes ou da composição de serviços existentes reusáveis. Nesta fase, são seguidos os cenários de desenvolvimento definidos durante a Análise.

A fase seguinte, a de **Teste**, tem o intuito de validar se os requisitos foram satisfeitos e se os artefatos entregues estão de acordo com o que foi definido nas fases de Análise e Projeto. São realizados testes funcionais, de desempenho, de interface e de composição.

A fase de **Provisão** trata dos aspectos organizacionais dos serviços, tanto dentro de uma organização como nas relações entre mais de uma. Em caso de serviços que gerem receita, devem ser realizadas atividades para medição de seu uso e a devida cobrança. São consideradas também questões de governança, certificação de serviços e alinhamento estratégico.

Na **Implantação**, os novos processos são publicados e expostos para todos os participantes, incluindo organizações, aplicações e outros processos. Nesta fase a interface e a implementação dos serviços são publicadas nos repositórios.

A **Execução** corresponde à fase em que os serviços e processos estão em operação.

Por fim, na fase de **Monitoração**, é medido o nível de serviço dos processos, com o intuito de garantir qualidade na execução. Métricas e *Key Performance Indicators* (KPI) são aferidos para determinar e reportar o desempenho dos processos e detectar pontos em que pode haver melhora. Trata-se de um processo contínuo de verificação do atendimento das metas de negócio definidas na Análise. Os dados obtidos servirão como entrada para uma nova iteração do ciclo de vida, iniciando um novo projeto.

O método descrito consiste das fases de Análise e Projeto do ciclo de vida proposto por Papazoglou e Van Den Heuvel.

3.1.2 Atividades de Análise

O objetivo da fase de Análise é a definição dos requisitos das novas aplicações, incluindo a definição dos processos de negócio envolvidos e seus respectivos escopos. Para isto, a fase de análise abrange quatro atividades principais: identificação de processo, escopo de processo, análise de *gap* e análise de realização.

3.1.2.1 Identificação de processos

Nesta atividade é estabelecida a compreensão de como a organização funciona em termos de processos de negócio e quais os serviços demandados por estes processos.

A partir da análise da funcionalidade de aplicações, é elaborado um modelo *as-is* dos processos de negócio em questão. Em seguida, este modelo é analisado para se determinar quais alterações podem ser realizadas e quais funções (serviços) podem ser agregadas ao processo. Esta etapa de remodelagem dos processos pode ser repetida diversas vezes, com vários candidatos a modelos *to-be* de processo sendo gerados e analisados até se chegar a um modelo *to-be* definitivo, conforme a Figura 3.2. Este modelo *to-be* consiste nos processos remodelados com possíveis melhorias e a agregação de novos serviços ou serviços modificados.

Figura 3.2: Identificação de processo e escopo de processo (PAPAZOGLU, 2006)

Nesta análise, pode-se comparar o portfólio atual de processos de negócio com definições padrão de processos para a proposição de candidatos a modelo *to-be*.

3.1.2.2 Escopo de processos

Conforme pode ser visto na Figura 3.2, o Escopo de Processos não se trata de uma atividade separada da Identificação de Processos, mas uma parte da etapa de elaboração do modelo *as-is*. Esta atividade representa um controle que deve existir para que os processos de negócio identificados não se tornem abrangentes demais, realizando o papel de uma aplicação inteira.

Deve-se procurar identificar sub-processos que possam ser desacoplados e com pouca coesão com o processo principal e especificá-los como processos separados. Assim, obtém-se controle sobre o escopo dos processos. Escopo de um processo de negócio inclui a definição de onde o processo começa e termina, dos usuários do processo, de suas entradas e saídas, das entidades externas que interagem com o processo e dos eventos envolvidos.

3.1.2.3 Análise de *gap*

Nesta atividade, é feita uma comparação entre os candidatos a serviços descritos no modelo *to-be* e o portfólio atual de serviços e funções contidas nas aplicações da organização. O resultado da análise de *gap* é uma recomendação de quais serviços devem ser desenvolvidos, quais podem ser reusados e quais podem ser consumidos de um provedor externo.

3.1.2.4 Análise de realização

Na análise de realização, define-se qual abordagem será adotada para a implementação dos serviços demandados pelo novo modelo de processos de negócio proposto. Cada cenário de realização de processos de negócio é avaliado em termos de custos, riscos, benefícios e retorno de investimento de acordo com os requisitos de negócio. Os cenários considerados são os seguintes: *green-field*, *top-down*, *bottom-up* e *meet-in-the-middle*.

- *Green-field*: Neste tipo de realização, novas interfaces de serviços são especificadas a partir de uma nova implementação de serviço.
- *Top-down*: No desenvolvimento *top-down*, uma nova implementação de serviço é criada a partir de uma interface existente de serviço. Uma vantagem do *top-down* é a consistência entre os processos, os serviços e a integração de aplicações. Esta alternativa, entretanto, tem um alto custo de desenvolvimento e estabelecimento de uma arquitetura organizacional.
- *Bottom-up*: Nesta opção, novas interfaces de serviço são criadas a partir de aplicações já existentes. Sistemas legados e aplicações corporativas são expostos na forma de serviços, por exemplo, criando-se interfaces *Web Services* a partir de suas *Application Programming Interfaces* (APIs). Esta alternativa é recomendada em ambientes heterogêneos com diversas tecnologias.
- *Meet-in-the-middle*: Esta opção é utilizada quando uma interface de serviço existente é mapeada parcialmente para uma implementação que também já existe. Esta abordagem pode envolver a criação de um encapsulamento de

serviço para uma aplicação existente ou a combinação com interfaces de serviços. O *meet-in-the-middle* procura unir os benefícios dos outros cenários, reduzindo seus riscos.

A escolha entre estas abordagens dependerá das estimativas de custos de implementação, de integração e de ajustes para cada uma delas. Devem ser desenvolvidas métricas para avaliar a flexibilidade, facilidade de manutenção, capacidade de extensão, acoplamento e coesão das abordagens e poder tomar a decisão por uma delas.

3.1.3 Atividades de Projeto

Após a fase de Análise vem a fase de Projeto, durante a qual os processos e serviços conceituais são transformados em um conjunto de especificações de interfaces independentes de plataforma. Em um método de desenvolvimento orientado a serviços é necessário que as interfaces sejam muito bem definidas e documentadas antes da implementação (PAPAZOGLU; VAN DEN HEUVEL, 2006).

A fase de Projeto é composta por duas frentes de desenvolvimento: uma dedicada a produzir os serviços (novas implementações ou componentes pré-existent) e outra dedicada a compor novos serviços a partir de um portfólio de serviços reusáveis.

O Projeto orientado a serviços deve especificar serviços levando em conta os seguintes aspectos: granularidade, reuso e composição.

3.1.3.1 Especificação de Serviços

Uma especificação de serviço consiste de três elementos: estrutura, comportamento e política.

- Especificação estrutural: É composta por informações de tipos de dados envolvidos em um serviço, mensagens e operações.

- Especificação de comportamento: Define os efeitos esperados e colaterais da execução das operações de um serviço, sejam estes efeitos de caráter técnico ou semântico.
- Especificação de política: Consiste nas regras e restrições do serviço, como segurança e gerenciamento.

Esta atividade foca em especificar as interfaces identificadas na fase de Análise, seguindo os princípios de acoplamento e coesão e os aspectos discutidos anteriormente. Para tal, a interface de um determinado serviço deve somente conter operações logicamente relacionadas entre si, para que exista coesão. Além disso, as mensagens de uma mesma interface devem possuir coesão de representação (por exemplo, utilizando os mesmos tipos de dados de esquema XML) e coesão de comunicação (por exemplo, utilizando os mesmos protocolos, como SOAP e HTTP). Deve-se procurar minimizar o acoplamento entre diferentes serviços, eliminando interdependências.

Como os autores descrevem o Projeto baseado na tecnologia *Web Services*, a especificação estrutural e de comportamento é realizada desenvolvendo interfaces WSDL. Estes dois aspectos da especificação podem ser devidamente representados nos tipos de portas, operações, mensagens e tipos de dados de um WSDL. Assim, para se produzir um serviço, definem-se as operações que ele proverá e, em seguida, os parâmetros destas operações (mensagens). Por fim, também devem ser definidos no WSDL os protocolos de comunicação utilizados para disponibilizar o serviço.

Já a especificação das políticas de um serviço leva em conta requisitos não-funcionais, como segurança, qualidade de serviço (desempenho, escalabilidade, disponibilidade), transações ou gerenciamento de mudanças (notificações). Por exemplo, um serviço pode exigir que seus potenciais consumidores utilizem *tokens* de segurança, assinaturas digitais ou criptografia de acordo com os padrões WS-Security para invocar suas operações. Estas informações devem ser devidamente documentadas na descrição do serviço.

3.1.3.2 Especificação de Processos

A outra atividade de projeto é o detalhamento dos processos de negócio identificados e delimitados durante a Análise. Deve-se definir a estrutura do processo, associar os papéis envolvidos e especificar seus requisitos não-funcionais. Para a especificação da estrutura, deve-se escolher entre as abordagens de orquestração e coreografia de serviços. Devido ao fato de as tecnologias de orquestração estarem mais maduras e difundidas quando da publicação do artigo de Papazoglou e Van Den Heuvel (2006), os autores optaram por utilizar a linguagem de orquestração WS-BPEL para especificar a estrutura dos processos de negócio.

A especificação da estrutura de um processo de negócio consiste na definição das atividades que o compõem e das relações entre elas. Deve-se identificar e descrever cada uma das atividades para então decidir como cada uma delas será implementada e quais serão executadas por serviços. A partir daí, são descritas as dependências e condições das atividades. Por fim, todas estas informações são representadas de maneira formal por meio de uma definição em linguagem WS-BPEL, que descreve o fluxo das atividades e as mapeia para os serviços.

No segundo passo, são identificadas as responsabilidades associadas a cada atividade e os papéis atribuídos a cada uma delas são descritos. Cada atividade deve ser desempenhada por uma entidade (pessoa ou serviço) que faça parte de um papel específico e que tenha as responsabilidades necessárias. A definição destes papéis poderá servir de base também para a descrição das políticas, como de controle de acesso e autorização.

Por fim, para a especificação de requisitos não-funcionais, os autores recomendam a elaboração de *Service Level Agreements* (SLA). SLAs são acordos de nível de serviço que estabelecem quais são as responsabilidades e expectativas do provedor e do consumidor de serviços em relação um ao outro. Podem abordar questões de segurança, desempenho e disponibilidade, bem como estipular penalidades e planos de contingência caso uma das partes falhe em atender o acordo.

3.1.4 Papéis

O processo Ciclo de Vida de Desenvolvimento de *Web Services* não identifica os papéis associados a cada fase.

3.1.5 Artefatos

Os artefatos produzidos são os modelos de processo *as-is* e *to-be*, que representam respectivamente o estado atual do negócio e a solução proposta com serviços automatizando tarefas do processo.

A fase de Projeto produz como artefatos as interfaces de serviço em WSDL, os esquemas de dados em XSD, a orquestração do processo em WS-BPEL e as especificações de SLA.

3.1.6 Análise do Método

O método proposto por Papazoglou e Van Den Heuvel (2006) mostra-se bastante voltado para a aplicação de SOA junto da abordagem *Business Process Management* (BPM) (JESTON; NELIS, 2008), uma vez que toda a definição da solução baseia-se nas atividades de modelagem de processos *as-is* e *to-be*. No entanto, os autores não separam a análise de processo de negócio da análise orientada a serviços, sendo que a definição dos serviços que fazem parte da solução ocorre durante a definição do modelo de processo *to-be*. Como resultado, obtêm-se processos de negócio modelados em um nível menor de granularidade, uma vez que atividades dos processos devem ser mapeadas diretamente para operações de serviços. Conseqüentemente, nas atividades de modelagem são utilizados tanto conhecimentos de análise de processo de negócio quanto conhecimentos de orientação a serviços, devendo ser executadas por pessoas com este perfil.

Uma característica interessante do método é a existência de uma atividade inteira (Análise de *Gap*) dedicada a avaliar os recursos já existentes na organização e detectar oportunidades de reuso. Estas oportunidades, por sua vez, são

consideradas na Análise de Realização, quando são avaliadas alternativas de realização de serviços do tipo *top-down*, *bottom-up* e *meet-in-the-middle*. A Análise de Gap e a Análise de Realização orientam de forma bastante clara o procedimento de identificação de oportunidades de reuso e de tomada de decisões de realização.

Durante as atividades de especificação de serviços e de processo, os autores utilizam padrões relacionados à tecnologia *Web Services*, pois recomendam que a estrutura dos serviços seja representada nas linguagens XSD e WSDL.

Além de projetar a estrutura e os requisitos funcionais dos serviços, o método também aborda a definição de requisitos não-funcionais, ao serem elaboradas políticas de serviços e acordos de SLA.

Ainda na etapa de Projeto, ocorre uma validação dos serviços especificados, avaliando se as características dos serviços estão em aderência aos princípios de SOA coesão e baixo acoplamento.

Na atividade de especificação de processos, o método mostra-se novamente orientado à abordagem BPM, pois utiliza a técnica de orquestração de serviços e a linguagem WS-BPEL para representar a lógica de um processo de negócio e sua interação com os serviços.

3.2 RUP *plugin* for SOMA

O *Rational Unified Process* (RUP) é um processo de Engenharia de Software desenvolvido e comercializado pela IBM. Trata-se de um método de gerenciamento de atividades e papéis em uma organização de desenvolvimento de software. O RUP tem como objetivo principal disciplinar o desenvolvimento de modo a entregar software de qualidade que atenda aos requisitos do cliente dentro do prazo e orçamento previstos (GORNIK, 2003).

O RUP constitui uma maneira de se implementar uma série de boas práticas de projeto de software em uma organização. Essas boas práticas são resultado de experiência de muitos projetos de software que serviram de base para a criação do arcabouço formado pelo RUP (KRUTCHEN, 2000). Entre as boas práticas utilizadas no RUP, estão o desenvolvimento iterativo de software, o gerenciamento de

requisitos, a arquitetura baseada em componentes, o uso da *Unified Modeling Language* (UML) (RUMBAUGH; JACOBSON; BOOCH, 2004), a garantia da qualidade e o gerenciamento de mudanças.

O RUP não é para ser utilizado de maneira literal, pois se trata de um arcabouço a partir do qual a organização poderá construir seu próprio processo de desenvolvimento de modo que realmente atenda às suas particularidades de domínio e cultura. Para a construção deste processo personalizado, a IBM disponibiliza a ferramenta *Rational Method Composer* (IBM, 2007).

Outro aspecto relevante é alta dependência do RUP em relação ao uso de ferramentas para automação e controle do processo. A própria personalização do processo e sua publicação podem ser realizadas por meio de uma ferramenta específica, assim como a geração de artefatos UML, o controle de versão de documentos e códigos-fonte e o desenvolvimento em si.

Para a utilização do RUP em projetos de SOA, a IBM disponibiliza um *plugin*, o RUP *plugin for Service-Oriented Modeling and Architecture* (SOMA) (ZIMMERMANN, 2005) que contém atividades, papéis e artefatos relacionados ao desenvolvimento orientado a serviços. Estes elementos devem ser incorporados ao processo personalizado da organização que pretende utilizar o RUP para projetos de soluções orientadas a serviços.

3.2.1 Ciclo de Vida

O modelo do RUP pode ser descrito em duas dimensões, o conteúdo do método e o processo. O conteúdo do método consiste nas disciplinas, compostas por atividades, papéis e artefatos. Já o processo representa o fluxo de trabalho ao longo do tempo, com suas fases e iterações (SHUJA, KREBS, 2007). Esta estrutura pode ser visualizada na Figura 3.3.

Figura 3.3: Arquitetura do *Rational Unified Process* (SHUJA, KREBS, 2007)

De acordo com o RUP, o ciclo de vida de desenvolvimento de software é dividido em fases, sendo que ao final de cada uma delas, é desenvolvida uma nova geração do produto. Entre uma fase e outra, é definido um *milestone*, que representa um instante de tempo no qual as decisões devem ter sido tomadas e os objetivos atingidos. As quatro fases do RUP são: Iniciação, Elaboração, Construção e Transição.

- A Iniciação tem como objetivo acordar entre todas as partes interessadas os objetivos do projeto. Nesta fase é determinado o escopo do projeto, é realizada uma descrição do ambiente e de um esboço da arquitetura candidata do produto. Também são feitas estimativas de custo e prazo do projeto além dos riscos, isto é, as fontes de imprevisibilidade. Como resultados desta fase, temos um documento de visão, uma prévia dos casos de uso, uma avaliação inicial dos riscos do projeto e um plano de projeto.
- A Elaboração tem como objetivo analisar o domínio do problema e propor uma arquitetura. A visão é elaborada e são definidos o processo, a infraestrutura e o ambiente de desenvolvimento. A arquitetura do software é elaborada de modo a suportar a visão definida anteriormente. Os resultados

desta fase são: o modelo de casos de uso, os requisitos não-funcionais complementares, um plano detalhado do projeto e um protótipo executável da arquitetura.

- Na Construção, os componentes da aplicação são desenvolvidos e integrados e todas as funções são testadas de maneira detalhada. É nesta fase que os requisitos de qualidade da aplicação devem ser atingidos e verificados. Ao final da Construção, temos como resultado um produto de software devidamente integrado e pronto para ser entregue, além da documentação voltada para os usuários.
- A Transição tem o propósito de disponibilizar o produto de software para seus clientes. As atividades desta fase incluem treinamento dos usuários, tarefas de implantação em ambiente de produção e distribuição da aplicação desenvolvida e avaliação para verificar a aceitação do produto em relação à visão especificada.

No RUP, a definição do processo de desenvolvimento estabelece quem faz o quê, quando e como, por meio de papéis, atividades e artefatos. Entretanto, um processo não é constituído por uma simples enumeração de papéis, atividades e artefatos. O processo é constituído por um conjunto de disciplinas, em que cada uma representa uma seqüência de atividades que produz algum tipo de valor e mostra as interações entre os papéis. Pode-se descrever uma disciplina como um fluxo de trabalho, com vários papéis executando atividades, que possuem relações de dependência entre si. O RUP define nove disciplinas, que são descritas a seguir: Modelagem de Negócio, Requisitos, Análise e Projeto, Implementação, Teste, Implantação, Gerenciamento de Projetos, Gerenciamento de Configuração e Mudanças e Ambiente.

- A Modelagem de Negócio tem como objetivo modelar a estrutura e a dinâmica da organização onde será implantada a aplicação. Esta disciplina visa estabelecer um entendimento comum entre clientes, usuários e desenvolvedores acerca do problema a ser solucionado, eliminando eventuais *gaps* entre a linguagem de negócio e a linguagem de engenharia de software. Para isto, o RUP faz uso dos chamados casos de uso de negócio para representar os processos que serão suportados pela solução.

- O propósito da disciplina de Requisitos é estabelecer um acordo entre desenvolvedores e clientes sobre exatamente o que o sistema deverá fazer, isto é, o escopo da solução. As necessidades dos clientes são levantadas e documentadas com detalhes, servindo de base para a elaboração de uma proposta de solução. Os resultados obtidos são descritos na forma de casos de uso e no chamado documento de visão. O modelo de casos de uso elaborado aqui orientará todo o ciclo de desenvolvimento, passando por levantamento de requisitos, análise, projeto e testes.
- A disciplina de Análise e Projeto visa, a partir dos requisitos, projetar a solução de software, de modo que ela atenda a todos os requisitos especificados pelos casos de uso. Esta disciplina produz um modelo de projeto, que representa a arquitetura do software a ser desenvolvido. Tal modelo especifica subsistemas, componentes, classes, suas interfaces e como todos eles se relacionam para realizar os casos de uso. É uma representação de como o código-fonte será estruturado e escrito.
- Na Implementação, é definida a organização do código, em termos de subsistemas e camadas de abstração. Esta disciplina envolve também toda a implementação de classes e componentes, testes unitários destes componentes e sua integração constituindo um sistema executável.
- A disciplina de Teste tem como objetivo verificar e validar a interação entre objetos e a integração entre os componentes do software. Os testes realizados verificam se todos os requisitos foram implementados e atendidos de forma adequada e identificam e detectam falhas a serem solucionadas antes da implantação do software.
- A disciplina de Implantação é composta pelas atividades que visam disponibilizar o software para seus usuários finais. Entre as atividades desta disciplina estão: empacotamento, distribuição e instalação do software, suporte e assistência aos usuários, testes beta e migração de dados e de softwares existentes.
- A disciplina de Gerenciamento de Projetos envolve a condução do projeto, equilibrando objetivos conflitantes, gerenciando riscos e planejando e monitorando a execução das atividades. No RUP, esta disciplina apresenta

boas práticas e diretrizes para auxiliar a produção e entrega de software de qualidade.

- Na disciplina de Gerenciamento de Configuração e Mudanças, é descrito como controlar os diversos artefatos e itens de configuração produzidos pelas várias pessoas envolvidas num mesmo projeto. Esta disciplina contém atividades para lidar com questões como atualização simultânea de artefatos, notificação de mudanças e correções e controle de múltiplas versões.
- O propósito da disciplina de Ambiente é fornecer à organização desenvolvedora de software o ambiente de desenvolvimento, composto por processos e ferramentas que apoiarão a equipe de projeto. Esta disciplina envolve a configuração do processo de desenvolvimento para o contexto específico de um projeto.

O RUP *plugin for SOMA* (IBM, 2007) é uma extensão ao conteúdo original do RUP que traz modificações ao ciclo de vida, além de acrescentar novos conceitos, papéis, atividades e artefatos. A maior parte das adições trazidas pelo *plugin* é aplicada às primeiras fases (Iniciação e Elaboração) do ciclo e dizem respeito à disciplina de Análise e Projeto. Três macroatividades são adicionadas para tratar do desenvolvimento orientado a serviços: Identificação de Serviços, Especificação de Serviços e Realização de Serviços. Cada macroatividade pode ser composta por uma ou mais tarefas.

Para avaliar o RUP *plugin for SOMA* como método de análise e projeto, foram consideradas as macroatividades adicionadas, mais os papéis e artefatos relacionados. Apesar de não haver no RUP uma separação explícita de atividades de Análise de atividades de Projeto, são descritas a primeira como Análise e as duas últimas como Projeto, por analogia com outros métodos.

3.2.2 Atividades de Análise

A atividade de análise do RUP adaptado pelo *plugin* é a Identificação de Serviços. Ela tem como objetivo identificar e elaborar a especificação inicial dos serviços candidatos que comporão a solução a ser desenvolvida.

Dependendo do tipo de projeto e das informações disponíveis, podem-se utilizar diferentes fontes para identificar os serviços candidatos. Estas fontes não se excluem mutuamente, podendo-se utilizar mais de uma delas em um mesmo projeto. As fontes para identificação de serviços são exibidas na Figura 3.4.

Figura 3.4: Abordagens de Identificação de Serviços (IBM, 2007)

Os serviços candidatos identificados são documentados no Modelo de Serviços. A seguir são descritas as tarefas correspondentes à identificação de serviços a partir de cada tipo de fonte.

3.2.2.1 Análise de Processo de Negócio

Nesta tarefa, serviços são identificados a partir de modelos de processos de negócio, representados por diagramas em notações como a *Business Process Modeling Notation* (BPMN). Em um modelo de processo, temos um fluxo de tarefas que são desempenhadas por diferentes papéis, como exemplificado na Figura 3.5. Neste exemplo, cada raia do diagrama contém as tarefas desempenhadas por um determinado papel.

Figura 3.5: Exemplo de modelo de processo de negócio (IBM, 2007)

Neste caso, cada papel pode ser mapeado para um serviço e as tarefas associadas a este papel podem ser mapeadas para operações pertencentes ao serviço correspondente. As tarefas em que se pode realizar este tipo de mapeamento são as automatizadas e as que envolvem interação com algum sistema.

3.2.2.2 Análise Caso de Uso de Negócio

Quando for utilizado um modelo de casos de uso para representar o negócio, os casos de uso de negócio podem servir como insumo para a identificação de serviços. A partir de um determinado caso de uso, deve-se representar sua descrição na forma de diagrama de seqüência de análise de negócio, conforme exemplificado na Figura 3.6.

Figura 3.6: Mapeamento de casos de uso para serviços (IBM, 2007)

Neste caso, pode-se mapear o ator “Sistema de crédito” do caso de uso para um serviço e as interações executadas pelo ator podem ser mapeadas para chamadas de operações do serviço.

3.2.2.3 Análise de Modelo de Dados

Em muitas aplicações, os dados são modelados como Entidades de Negócio identificadas na Modelagem de Negócio. Assim, estas aplicações demandarão serviços que manipulem e acessem os dados das Entidades. Os modelos de dados existentes deverão ser analisados e os dados agrupados de forma a constituir um serviço. Este tipo de serviço é chamado de Serviço de Encapsulamento de Dados.

3.2.2.4 Análise de Regras de Negócio

Nesta tarefa, identificam-se a partir de um modelo de negócio (processo ou caso de uso) as regras de negócio que podem ser tornadas externas ao restante da solução,

sendo então encapsuladas por serviços. Geralmente, criam-se serviços a partir de regras de validação com potencial de reuso em diversos contextos.

3.2.2.5 Análise de Recursos Existentes

Na maior parte das vezes, as soluções orientadas a serviços deverão interagir com alguma aplicação legada da organização, reusando funções já existentes. Nesta abordagem, devem-se identificar as funções legadas que serão necessárias à solução sendo desenvolvida e disponibilizá-las na forma de serviço. Isto pode ser feito encapsulando a função com um serviço, por meio de adaptadores ou integrando a função a um serviço, conforme exemplificado na Figura 3.7.

Figura 3.7: Disponibilização de funções legadas como serviços (IBM, 2007)

3.2.3 Atividades de Projeto

Na fase de Projeto, o RUP *plugin for SOMA* descreve duas macroatividades: Especificação de Serviços e Realização de Serviços.

A Especificação de Serviços recebe como entrada o Modelo de Serviços candidatos identificados durante a Análise e define as especificações de interface dos serviços. É composta pelas tarefas: Teste Litmus de Serviço, Projeto de Mensagem e Especificação de Serviço.

Já na Realização de Serviços, é definido como os serviços serão implementados, passando pelas tarefas: Especificação de Componente e Decisões de Realização.

O RUP *plugin for* SOMA não determina uma seqüência exata para a execução destas tarefas, uma vez que pode haver diversas iterações de cada uma. Elas são descritas a seguir na ordem em que aparecem no *plugin*.

3.2.3.1 Teste Litmus de Serviço

Dos serviços candidatos identificados e documentados no Modelo de Serviços, nem todos serão efetivamente realizados e expostos no portfólio da organização. Nesta tarefa é executado um Teste Litmus para verificar algumas características dos serviços candidatos, tais como reusabilidade, alinhamento com os requisitos de negócio, possibilidade de composição e viabilidade técnica. O teste é realizado para decidir se um serviço candidato será realmente exposto como um serviço ou se deverá ser realizado como um componente não passível de exposição.

3.2.3.2 Projeto de Mensagem

As mensagens são os dados recebidos e retornados pelas operações do serviço. Nesta tarefa, devem-se especificar exatamente quais as informações envolvidas na execução de uma operação e definir a relação dessas informações com modelos de dados existentes. Devem ser tomadas decisões a respeito do mapeamento das mensagens, seu formato e protocolo de comunicação.

As mensagens podem ser representadas por um modelo de classes ou esquemas XSD.

3.2.3.3 Especificação de Serviço

Cada serviço deverá ser descrito e especificado levando-se em conta os seguintes aspectos: colaborações, políticas, dependências e composição.

As colaborações de um serviço são seus relacionamentos com outros serviços, possivelmente derivados do modelo de processo de negócio. Os serviços são vistos

assumindo um papel em uma determinada colaboração, sendo assim necessário especificar as responsabilidades associadas a este papel.

As políticas especificam os requisitos não-funcionais dos serviços, como segurança e QoS. Informações sobre políticas podem afetar qualquer elemento do serviço, seja uma operação, mensagem ou a colaboração entre serviços.

As dependências do serviço podem ser pré e pós-condições de processamento, relacionamentos com outros serviços, relacionamento entre o serviço e seu provedor ou o relacionamento com o canal que o conecta a outros serviços.

Por fim, devem ser especificadas as características de composição do serviço, caso ele seja um serviço composto, como um serviço de orquestração implementado em WS-BPEL. Devem-se indicar quais serviços são orquestrados e como são realizadas as interações.

Como resultado desta tarefa, obtêm-se as especificações de serviço no documento de Modelo de Serviços.

3.2.3.4 Especificação de Componente

Nesta tarefa, são especificados os componentes físicos que realizarão as Especificações de Serviço. São especificadas a interface do componente de serviço, seus atributos e estrutura interna.

A interface do componente deve conter as assinaturas das operações do serviço, sendo que os parâmetros dessas operações devem referenciar as mensagens definidas anteriormente. As interfaces podem ser representadas por modelos de classes ou documentos WSDL.

Atributos de um componente a serem descritos nesta tarefa são: propriedades, dependências de outros componentes, serviços fornecidos pelo componente e serviços utilizados.

Por fim, deve-se modelar a estrutura interna do componente, descrevendo-se as classes que o compõem e os relacionamentos entre elas. Neste ponto, podem ser aplicados padrões de projeto para realizar o comportamento especificado para o serviço.

3.2.3.5 Decisões de Realização

Quando se determinam os serviços que deverão ser expostos, deve-se determinar a abordagem de realização para cada um deles, decidindo por uma das opções a seguir:

- Construir internamente um novo componente de serviço
- Adquirir um serviço que possa ser hospedado internamente
- Transformar uma aplicação legada para expor um serviço
- Integrar uma aplicação legada, encapsulando-a em um serviço
- Terceirizar, consumindo um serviço externo

Para a tomada desta decisão, são analisados recursos existentes que podem ser reusados.

3.2.4 Papéis

O RUP *plugin for SOMA* traz três novos papéis voltados para atividades relacionadas ao desenvolvimento orientado a serviços: o Arquiteto de Serviços, o Projetista de Serviços e o Projetista de Dados de Serviços.

O Arquiteto de Serviços é o responsável por especificar a arquitetura da aplicação, tendo como entrada modelos de processos de negócio, modelos de casos de uso, modelos de dados e a partir daí identificando elementos e mecanismos de projeto a serem incorporados ao projeto. O Arquiteto de Serviços tem como principal atividade a Identificação de Serviços candidatos a partir dos modelos mencionados acima, resultando em um Modelo de Serviços parcialmente especificados.

O Projetista de Serviços é o responsável por refinar, revisar e detalhar os serviços candidatos descritos pelo Analista de Serviços, na execução das atividades de Especificação de Serviços e Realização de Serviços. O Projetista elabora ainda mais o Modelo de Serviços, especificando informações sobre mensagens, colaborações entre serviços e provedores. A partir daí o Projetista de Serviços pode também especificar Componentes de Serviço que implementarão as especificações contidas no Modelo de Serviço.

Já o Projetista de Dados de Serviços é uma extensão ao papel Projetista de Dados do RUP convencional, mas com a responsabilidade de também projetar bases de dados a partir da especificação de Componentes de Serviço.

3.2.5 Artefatos

Os novos artefatos introduzidos pelo RUP *plugin for SOMA* são o Modelo de Serviços e os Componentes de Serviço.

O Modelo de Serviços é uma abstração da especificação tecnológica de *Web Services* e é utilizado para elaborar e documentar o projeto de serviços em uma aplicação SOA. Trata-se de um artefato composto que inclui informações sobre especificações de serviços, provedores destes serviços, mensagens trocadas por eles e colaborações entre eles. Pode ser representado em linguagem UML.

O artefato Componente de Serviço descreve a realização de uma especificação de serviço. Trata-se da especificação do componente de software que implementará a interface de um ou mais serviços. Pode-se referir a um novo componente desenvolvido, um *Web Service*, uma aplicação existente, uma fonte de dados ou até uma composição de outros serviços. O artefato Componente de Serviço é utilizado como especificação por aqueles que irão implementá-los.

3.2.6 Análise do Método

O método caracterizado pelas atividades do RUP *plugin for SOMA* utiliza alguns conceitos de BPM, pois permite que serviços sejam identificados a partir de modelos de processo de negócio, porém não aborda de maneira explícita atividades de orquestração de serviços.

A Identificação de Serviços considera o reuso de recursos existentes, pois possibilita que serviços sejam identificados a partir de aplicações legadas, e oferece diversas fontes de identificação. Na tarefa de Decisões de Realização, as aplicações existentes são novamente avaliadas para se definir como cada serviço será realizado.

Por estas características, o RUP *plugin for SOMA* pode ser classificado como uma abordagem *meet-in-the-middle*, pois garante que os serviços estão alinhados tanto com os requisitos de negócio quanto com as restrições do ambiente existente.

O método utiliza o conceito de serviço candidato, que serve de ponte da Identificação para a Especificação de Serviços.

O RUP *plugin for SOMA* propõe o Teste Litmus de Serviço para determinar quais serviços candidatos serão efetivamente realizados. No entanto, esta validação leva mais em conta motivos práticos e econômicos para expor um serviço, mas não considera a adequação dele ao portfólio da organização e analisa somente a aderência dos serviços candidatos aos princípios de reusabilidade e facilidade de composição.

Na atividade de Especificação de Serviços, requisitos não-funcionais são tratados na definição de políticas de serviço e na análise de características de QoS.

O RUP *plugin for SOMA* integra-se ao ciclo tradicional de desenvolvimento, utilizando na implementação técnicas de desenvolvimento baseado em componentes e orientação a objetos.

O RUP *plugin for SOMA* não é um método de uso livre. Sua utilização está atrelada à aquisição de licenças de software da ferramenta *Rational Method Composer* da IBM.

3.3 Erl

Thomas Erl é um autor de publicações sobre SOA e membro de organizações de padronização nas áreas de XML e *Web Services*. Em suas publicações (ERL, 2005. ERL, 2007), o autor descreve um método para o desenvolvimento de uma arquitetura SOA e seus serviços.

Erl descreve o desenvolvimento de uma arquitetura SOA por meio de um ciclo de vida de desenvolvimento de serviços, focando principalmente nas atividades de análise e projeto, que são as etapas em que os serviços são identificados e especificados. O autor destaca o uso dos princípios de orientação a serviços durante estas atividades, para garantir que os serviços desenvolvidos sejam efetivos dentro

da arquitetura SOA. Em determinados pontos, tanto da análise quanto do projeto, a aderência aos princípios de orientação a serviços é verificada.

O autor assume a utilização da tecnologia *Web Services* na implementação dos serviços, fazendo uso de artefatos específicos como esquemas XSD e interfaces WSDL na descrição das atividades de projeto. A verificação dos princípios de orientação a serviços também é feita levando-se em conta os pontos de aderência da tecnologia *Web Services* aos princípios e os *gaps*.

3.3.1 Ciclo de Vida

Erl descreve o ciclo de vida de um projeto de soluções orientadas a serviços, composto por uma série de passos a serem seguidos para se construir serviços que compõem uma solução. De maneira geral, o processo para se desenvolver soluções é, segundo Erl, bastante semelhante ao desenvolvimento de aplicações distribuídas. Afinal, serviços devem ser projetados, implementados e postos em execução assim como outros tipos de componentes convencionais e tecnologias de *front-end* e *back-end*.

O ciclo de vida descrito pelo autor pode ser visualizado na Figura 3.8.

Figura 3.8: Ciclo de vida de desenvolvimento SOA (ERL, 2005)

A fase de Análise Orientada a Serviços é o estágio inicial em que é determinado o escopo da arquitetura SOA em desenvolvimento. As camadas de serviços (orquestração, negócio e aplicação) são mapeadas e serviços preliminares são modelados na forma de serviços candidatos.

Na fase seguinte, Projeto Orientado a Serviços, os serviços que farão parte da arquitetura SOA já estão definidos, sendo necessário então determinar como eles serão construídos. A atividade de projeto é altamente baseada em padrões e influenciada pelos princípios de orientação a serviços. Quatro diferentes estratégias podem ser utilizadas para se projetar um serviço: serviços baseados em entidades, serviços de aplicação, serviços baseados em tarefas e serviços de processo.

A fase de Desenvolvimento de Serviços é o momento em que os serviços são efetivamente construídos. Nesta fase são considerados os aspectos técnicos e especificidades da tecnologia de implementação, como linguagens de programação, ambiente de desenvolvimento e plataformas.

Pelo fato de poderem ser reusados e participar de composições de maneira não previstas, os serviços devem ser rigorosamente testados. Na fase de Teste de Serviços, deve ser verificada uma série de requisitos dos serviços, como aderência aos padrões, comunicação com diversos tipos de consumidores, aspectos de QoS e tratamento de exceções e compensação.

A fase de Implantação de Serviços trata de instalar e configurar os componentes distribuídos que implementam os serviços, as interfaces de serviços e produtos de *middleware* associados em ambiente de produção. Deve-se definir a alocação física de cada componente, avaliar a capacidade da infra-estrutura satisfazer aos requisitos de desempenho esperados, estabelecer as configurações de segurança e verificar as integrações com sistemas e aplicações.

Finalmente, a fase de Administração de Serviços acompanha a execução dos serviços, monitorando seu uso e desempenho, realizando o controle de versão dos artefatos relacionados e dando manutenção para que os serviços operem de modo satisfatório.

3.3.2 Atividades de Análise

As atividades de análise no método proposto por Erl correspondem à fase de Análise Orientada a Serviços do ciclo de vida. O objetivo principal desta fase é determinar quais os serviços que devem ser construídos e o escopo de cada um deles, ou seja, qual a lógica deve ser encapsulada por cada um.

Os passos para a realização da Análise Orientada a Serviços são exibidos na Figura 3.9 e são descritos a seguir.

Figura 3.9: Análise Orientada a Serviços (ERL, 2005)

3.3.2.1 Definição dos requisitos de negócio

Nesta atividade, os requisitos de negócio relacionados à solução orientada a serviços sendo construída são levantados e documentados. Esta documentação de requisitos servirá como ponto de partida para a Análise Orientada a Serviços.

Os requisitos de negócio devem estar definidos o suficiente para que um processo de automação possa ser especificado e utilizado no desenvolvimento da solução SOA.

3.3.2.2 Identificação de sistemas existentes

O objetivo específico desta atividade é identificar se existe alguma lógica de aplicação já existente que satisfaça total ou parcialmente a algum dos requisitos levantados na atividade anterior. Neste momento ainda não é necessário concentrar-

se em como os serviços interagirão com as aplicações legadas, mas apenas realizar um levantamento de alto nível para prever quais sistemas podem ser afetados.

Este tipo de informação será útil para a identificação dos serviços candidatos na etapa de modelagem.

3.3.2.3 Modelagem de serviços candidatos

A atividade de Modelagem de Serviços é o principal passo da fase de Análise Orientada a Serviços proposta por Erl. Nesta fase, são identificadas operações candidatas que devem ser agrupadas segundo seu contexto lógico, dando origem a serviços candidatos. Eventualmente, os serviços candidatos podem ser combinados em modelos de composição para formar a solução orientada a serviços.

A Modelagem de Serviços compreende uma série de sub-tarefas, apresentadas na Figura 3.10.

Figura 3.10: Modelagem de serviços candidatos (ERL, 2005)

- No primeiro passo, ocorre a **decomposição do processo de negócio** documentado em uma série de passos de baixa granularidade, sendo que este nível de granularidade obtido pode diferir do nível original.
- Em seguida, são **identificadas as operações candidatas dos serviços de negócio**. Basicamente, os candidatos a operações serão os passos do processo identificados na etapa anterior, exceto aquelas atividades que são tarefas humanas que não deverão ser automatizadas e passos que já são executados por aplicações legadas que não podem ser encapsuladas por serviços.
- Caso a arquitetura SOA em questão possua uma camada de orquestração de serviços, então se deve definir que partes da lógica de aplicação serão **abstraídas pela lógica de orquestração**. Isto é, deve-se definir qual a lógica estará representada no processo ao invés de ser executada por um serviço de negócio. Regras de negócio, lógica condicional, lógica de exceção e lógica

seqüencial são alguns exemplos de lógica que podem ser abstraídos pela camada de orquestração.

- As operações candidatas devem então ser analisadas para se **criar os candidatos a serviços de negócio**. As operações deverão assim ser agrupadas de acordo com o contexto lógico correspondente. Por exemplo, serviços baseados em tarefas seriam agrupamentos de acordo com processo ou serviços baseados em entidades seriam agrupamentos de operações segundo os relacionamentos entre entidades. Operações podem ser adicionadas aos serviços para aumentar sua reusabilidade.
- Definidos os serviços candidatos, **aplicam-se os princípios de orientação a serviços** para garantir que eles realmente atendam aos requisitos de uma arquitetura SOA. As operações candidatas devem ter seus escopos e lógica revisados para que sejam realmente reusáveis e autônomas (sem dependências entre si).
- Para se **identificar composições de serviços candidatos**, devem ser analisados diversos cenários de execução do processo de negócio. Destas análises, podem-se inferir possíveis composições de serviços e concluir se os agrupamentos de operações são adequados. Nos cenários, devem ser considerados os casos de falha ou exceção.
- Baseados nos resultados do passo anterior, **revisam-se os agrupamentos de operações candidatas em serviços**.
- Por último, opcionalmente, podem-se **revisar os requisitos de processamento** dos serviços candidatos. Tal revisão tem o propósito de determinar qual a lógica necessária para executar as operações candidatas e se tal lógica já existe ou deve ser desenvolvida.

Para documentar os serviços em seu método de análise de projeto, Erl propõe uma notação semelhante a uma notação de classe da UML, que exibe um serviço com suas operações. Esta notação é utilizada em diagramas que exibem composições de serviços. A notação utilizada por Erl pode ser vista na Figura 3.11.

Figura 3.11: Notação para representar um serviço (ERL, 2007)

3.3.2.4 Atividades de Projeto

O método descrito por Erl tem suas atividades de projeto especificadas na fase de Projeto Orientado a Serviços do ciclo de vida. Os principais objetivos desta fase são definir as interfaces dos serviços candidatos modelados na Análise, garantir a adequação aos princípios de orientação a serviços e definir quais padrões serão suportados e utilizados na implementação dos serviços.

Como resultados, temos as especificações dos serviços nas camadas de negócio, aplicação e orquestração.

Os passos do Projeto Orientado a Serviços podem ser visualizados na Figura 3.12 e são descritos nas seções seguintes.

Figura 3.12: Projeto orientado a serviços (ERL, 2005)

3.3.2.5 Composição da arquitetura orientada a serviços

Nesta primeira atividade, são determinadas quais camadas de serviços serão utilizadas na arquitetura sendo construída. Devem-se definir também os padrões que serão utilizados na especificação e implementação dos serviços. Por exemplo, se for utilizada a tecnologia *Web Services*, deve-se estabelecer quais especificações de XML e *Web Services* serão utilizadas e quais extensões (padrões WS-*) serão necessários.

3.3.2.6 Projeto de serviços baseados em entidades

Serviços de negócio baseados em entidades representam entidades de dados definidas no modelo de negócio da organização. Este tipo de serviço é completamente independente de processo de negócio, podendo ser reusado por qualquer solução que necessite acessar dados relacionados a uma entidade particular. Serviços baseados em entidades fazem parte da camada de serviços de negócio.

A atividade de projeto de serviços baseados em entidade tem como propósito especificar as interfaces e a lógica encapsulada por cada um deles.

Para a especificação das interfaces, Erl propõe o uso de documentos WSDL, que podem ser utilizados posteriormente para implementação com *Web Services*. Os esquemas de dados das entidades tratadas pelo serviço são definidos e representados na forma de esquemas XSD e comporão as mensagens trocadas pelas operações do serviço.

Princípios de orientação a serviços devem ser também aplicados aos serviços. Serviços baseados em entidades são intrinsecamente autônomos pelo fato de cada um ser responsável por manipular uma entidade ou grupo de entidades específico. Também não preservam informação de estado, uma vez que possuem pouca lógica de fluxo de trabalho encapsulada. Deve-se atentar para a reusabilidade do serviço, possivelmente adicionando operações que podem ser úteis em outros contextos além do considerado para uma solução em particular.

3.3.2.7 Projeto de serviços de aplicação

Os serviços da camada de aplicação são a principal força de trabalho da arquitetura SOA, executando funções demandadas pelas camadas de orquestração e de serviços de negócio. São abstrações orientadas a serviço do ambiente tecnológico da organização, não sendo por isso necessário considerar aspectos de negócio em seu projeto.

A atividade de projeto de serviços de aplicação inicia-se com a análise das operações candidatas propostas, para verificar se possuem o nível de granularidade e reusabilidade adequados. Devem então ser definidos os dados de entrada e saída para cada uma das operações, possivelmente na forma de esquemas XSD. Finalmente, a definição da interface WSDL deve ser completada com as operações correspondentes.

Como serviços de aplicação devem poder ser utilizados por diversos tipos de serviços de negócio, os dados de entrada e saída definidos para as operações devem ser definidos de modo simples e genérico.

Ao aplicarem-se os princípios de orientação a serviços, deve-se atentar para que eles se mantenham autônomos sem dependências entre sua lógica de negócio.

Por fim, devem ser identificadas as possíveis restrições de ordem técnica que se aplicam ao serviço de aplicação, como componentes, APIs e adaptadores necessários para a realização de conexões, restrições de segurança e possíveis requisitos de SLA.

3.3.2.8 Projeto de serviços baseados em tarefas

Serviços baseados em tarefas pertencem à camada de serviços de negócio e geralmente têm seu projeto simplificado por possuírem pouca necessidade de serem reusáveis. Tipicamente encapsulam lógica de fluxo de trabalho e realizam a orquestração de serviços de aplicação.

A lógica de fluxo de trabalho deve ser especificada, podendo ser representada por diagramas de atividades. Esta lógica especificada pode auxiliar na descoberta de novas operações necessárias. As operações candidatas têm suas entradas e saídas mapeadas e são representadas por meio de documentos XSD e WSDL.

3.3.2.9 Projeto de processo orientado a serviços

Os processos de negócio orientados a serviços correspondem à camada de orquestração da arquitetura SOA. O processo corresponde ao fluxo de execução de

serviços de negócio, com suas regras e lógica, descrevendo o que é chamado de orquestração de serviços. No método proposto por Erl, a linguagem WS-BPEL é utilizada para representar as definições dos processos de negócio.

Enquanto o projeto de serviços baseados em tarefas e em entidades concentrava-se em definir as interfaces dos serviços e as mensagens das operações, o projeto de processos de negócio busca criar um fluxo WS-BPEL que implemente a lógica de negócio relegada à camada de orquestração durante a Análise Orientada a Serviços. Tal lógica que foi deixada de fora dos serviços especificados é abstraída em um processo de negócio separado.

O primeiro passo do projeto de processos recebe como entrada a descrição da lógica de fluxo de trabalho a ser abstraída pela orquestração, o serviço de processo candidato e os serviços de negócio projetados nas atividades de projeto anteriores. Devem-se levantar quais os tipos de dados e informações serão necessários para executar a lógica e invocar os serviços de negócio. Com base neste levantamento, devem-se especificar os esquemas de dados das mensagens utilizadas no processo e a interface do serviço de processo. Neste ponto muitos esquemas definidos para os serviços de negócio poderão ser reaproveitados.

Definidas a interface e as mensagens, parte-se para o desenvolvimento da especificação WS-BPEL do processo. Neste passo, será especificado como os serviços serão orquestrados. Devem-se definir quais serviços serão invocados, criar as regras de negócio envolvidas nas invocações e o tratamento dos dados trocados com os serviços.

Com os processos WS-BPEL especificados, deve-se analisar a lógica de fluxo de trabalho criada frente aos cenários de execução discutidos anteriormente. Neste passo, pode ocorrer um refinamento do processo e dos serviços de negócio, dando origem a algumas alterações necessárias de acordo com os cenários.

3.3.3 Papéis

O método proposto por Erl não especifica papéis para a realização de suas atividades.

3.3.4 Artefatos

A documentação dos requisitos de negócio é um artefato utilizado na fase inicial de Análise Orientada a Serviços e especifica os requisitos relacionados à solução em desenvolvimento. Estes requisitos podem ser obtidos por meio de técnicas convencionais de levantamento utilizadas em métodos de desenvolvimento de software distribuído.

A documentação dos serviços candidatos é o resultado da Análise Orientada a Serviços e consiste em uma descrição em alto nível dos serviços criados a partir do agrupamento de operações candidatas identificadas. Este artefato descreve a lógica de negócio executada por cada operação candidata do serviço.

Durante o Projeto Orientado a Serviços, as mensagens trocadas pelas operações são definidas na forma de esquemas XSD e a descrição das interfaces é realizada por meio da linguagem de definição de interfaces WSDL. Os serviços de processo desenvolvidos para a camada de orquestração são especificados utilizando-se a linguagem de definição de processos WS-BPEL.

3.3.5 Análise do Método

O método proposto por Erl consiste em uma abordagem puramente *top-down* para identificação e especificação de serviços. Na atividade de modelagem de serviços candidatos, Erl descreve de forma detalhada os passos a serem executados para identificar operações candidatas e descrever os serviços candidatos resultantes.

O autor coloca a modelagem de processo como parte da atividade de definição de requisitos de negócio, separando-a das atividades de identificação e descrição dos serviços que ocorrem durante a modelagem de serviços candidatos. Apesar de não serem especificados os papéis associados às atividades, esta separação permite que cada atividade seja executada por pessoas com conhecimentos específicos.

Apesar de os recursos existentes serem identificados durante a Análise Orientada a Serviços, o reuso destes recursos não é considerado para compor os serviços descritos na modelagem de serviços candidatos. Tal identificação servirá para definir

se um passo do processo será executado por um serviço ou por uma aplicação legada, sendo que no segundo caso não é considerado como uma execução de serviço. Conseqüentemente, não há descrita no método nenhuma atividade para definição de como cada serviço será realizado.

O conceito de serviços candidatos é utilizado pelo método de Erl como transição da Análise para o Projeto. Os serviços são considerados candidatos desde sua modelagem até o momento em que têm sua especificação definida.

Trata-se de uma proposta bastante alinhada ao conceito de BPM, que é aplicado tanto durante a decomposição de processos de negócio em serviços quanto no projeto de processo de negócio orientado a serviço, atividade em que é aplicada a orquestração de processos em WS-BPEL.

Na fase de Projeto, o método leva em consideração as camadas de serviços, ao descrever atividades de projeto voltadas para cada tipo de serviço: entidade, tarefa e aplicação. Cada uma destas atividades tem como objetivo especificar serviços focando nas particularidades de um determinado tipo de serviço.

Erl dá bastante ênfase aos princípios de orientação a serviços, incluindo atividades específicas para validação tanto na Análise quanto no Projeto (ERL, 2007). Na atividade de modelagem de serviços candidatos, realiza-se uma avaliação para verificar se os serviços candidatos modelados possuem características de reusabilidade e autonomia. Já durante as atividades de projeto de serviços, são verificados os princípios de reusabilidade, autonomia, independência de estado e baixo acoplamento, dependendo do tipo de serviço sendo projetado. Como o método adota a tecnologia *Web Services* para a realização dos serviços, são utilizados os padrões XSD e WSDL para especificar os contratos de serviços.

3.4 Marks e Bell

Marks e Bell argumentam que a unidade fundamental de uma arquitetura SOA é o serviço. Sem que existam os serviços e os consumidores destes serviços, não se tem uma arquitetura SOA (MARKS; BELL, 2006). Estes serviços devem ser reusáveis, interoperáveis e devem possuir o nível adequado de granularidade.

Deste modo, um grande desafio à implementação de SOA é estabelecer um processo para identificar e construir os serviços que satisfaçam as metas do negócio e agreguem valor a ele. É necessário garantir que os serviços desenvolvidos na organização sejam eficientes e eficazes. Devem ser eficientes no sentido de ser bem projetados e operar de acordo com os requisitos funcionais, de desempenho e de qualidade. E também devem ser eficazes, atendendo a necessidades críticas para o negócio.

Marks e Bell (2006) apresentam uma abordagem para a adoção da orientação a serviços em uma organização, que parte das necessidades de negócio e metas estratégicas até chegar aos projetos de implementação da arquitetura e ao desenvolvimento dos serviços em si. Os autores descrevem o ciclo de vida dos serviços em uma arquitetura SOA e detalham um método de identificação, análise e projeto de serviços.

3.4.1 Ciclo de Vida

O ciclo de vida de serviços reflete o processo que vai desde a identificação e descoberta, análise e projeto de serviços até sua implementação e gerenciamento. Este processo compreende a evolução dos serviços desde sua concepção até sua maturidade, passando pela execução.

As fases do ciclo de vida dividem-se em dois domínios: o domínio do problema e o domínio da solução. O domínio do problema é formado pelas atividades do ciclo de vida que resultam na identificação e análise dos serviços da organização, tendo como produto um conjunto de serviços candidatos. Já o domínio da solução envolve o projeto dos serviços candidatos, a implementação e integração, resultando nos serviços de solução construídos.

As fases do ciclo de vida são: Motivação, Conceituação, Modelagem, Realização e Gerenciamento. As fases do ciclo de vida e suas atividades são exibidas na Figura 3.13 e descritas nas seções seguintes.

Figura 3.13: Ciclo de Vida de Serviços (MARKS; BELL, 2006)

3.4.1.1 Motivação

A fase de Motivação consiste nas forças motivadoras, como eventos, necessidades de negócio e de TI e problemas, influenciando a criação de novos serviços.

Eventos podem ser tendências de mercado, ameaças competitivas, regulamentações ou ocorrências tecnológicas que criam oportunidades e demandam algum tipo de resposta por parte da organização.

Necessidades de negócio são questões como baixa satisfação dos clientes, aumento na competição ou um longo *time-to-market*, isto é, questões relativas ao modelo de negócio da organização. Já as necessidades de TI são as questões do domínio da tecnologia, como custos excessivos de integração, dificuldade de suportar mudanças no negócio ou atrasos crônicos na entrega de projetos. São necessidades que devem ser resolvidas, ou a organização terá consequências negativas. Geralmente as necessidades de TI e de negócio estão relacionadas, de

modo que, solucionando-se uma necessidade de TI, contribui-se para resolver uma necessidade de negócio.

Por fim, podem ocorrer problemas quando não se possui um conjunto de medidas preventivas para tratar os impactos causados por eventos. Problemas ocorridos devem ser identificados e estudados, de modo a determinar sua severidade, conseqüências e possíveis causas.

Todas estas informações servirão como motivação para a criação de novos serviços, que terão o objetivo de tratar eventos e resolver necessidades e problemas da organização.

3.4.1.2 Conceituação

Na fase de Conceituação, são levantadas idéias e são definidos conceitos que serão posteriormente utilizadas na modelagem dos serviços.

As idéias consistem em alternativas de solução para as necessidades e problemas identificados na fase anterior, enquanto os conceitos são os termos de negócio, como processos e entidades, envolvidos na solução a ser proposta. Essas idéias e conceitos devem ser bem definidos neste momento, pois darão origem aos serviços de negócio candidatos.

3.4.1.3 Modelagem

A fase de Modelagem recebe como entrada os serviços candidatos identificados nas fases anteriores e é formada pelas atividades de análise e projeto.

Para os autores, esta fase é fundamental para identificar, projetar e implementar os serviços críticos para a organização. Os serviços desenvolvidos devem ser mapeados para as necessidades de TI e de negócio da organização e suportar suas metas estratégicas.

Na análise, são realizadas operações lógicas de modelagem sobre os serviços candidatos para compreender melhor os candidatos e seus relacionamentos. Estes

candidatos podem ser decompostos ou agregados e são também priorizados e detalhados, o que resulta em um conjunto de serviços de negócio.

Já a atividade de projeto define o escopo dos serviços de solução, que serão criados a partir dos serviços de negócio. Os serviços de negócio recebidos têm sua granularidade mapeada e são então agrupados de modo a suportar domínios de negócio específicos.

Esta é a fase analisada como método de análise e projeto orientado a serviços de Marks e Bell.

3.4.1.4 Realização

A fase de Realização recebe como entrada as especificações dos serviços de negócio finais e compreende as atividades de realização, implementação e integração.

Na realização, as especificações dos serviços de negócio finais são traduzidas em especificações técnicas de como as funções dos serviços serão executadas, seja por componentes de software a serem desenvolvidos ou por adaptação de funções já existentes no legado.

Na atividade de implementação, ocorrem o desenvolvimento e os testes dos componentes de acordo com a especificação do serviço, enquanto que na integração, são realizadas as interações com aplicações e serviços já existentes na arquitetura.

3.4.1.5 Gerenciamento

Por fim, na fase de Gerenciamento são executadas diversas atividades para garantir que os serviços operem adequadamente e atendam aos seus requisitos funcionais e não-funcionais (gerenciamento da operação, monitoração de QoS).

Esta fase engloba também atividades para manter a arquitetura SOA, como o gerenciamento do portfólio de serviços e do seu ciclo de vida, além do

estabelecimento de políticas e processos de governança, que garantirão o controle sobre os serviços.

3.4.2 Atividades de Análise

O método proposto por Marks e Bell aborda a análise pelos modos *top-down* e *bottom-up* em ciclos iterativos. Por um lado, a perspectiva *bottom-up* pode deixar os serviços presos a seu ambiente tecnológico de origem, gerando alto acoplamento e potencial limitado de reuso. Por outro lado, a perspectiva *top-down* pode facilitar o reuso e o alinhamento, mas os serviços resultantes podem ser difíceis de implementar devido à falta de técnicas de projeto mais concretas do ponto de vista da tecnologia. Por isso, os autores propõem o uso dos dois modos de maneira iterativa. A análise *top-down* é usada para identificar conceitualmente os serviços candidatos e analisá-los antes de se prosseguir com o projeto. Então, o método segue com o projeto sendo realizado por uma perspectiva *bottom-up*, quando os serviços candidatos são traduzidos em serviços de solução físicos.

Os passos do processo de identificação e análise de serviços segundo Marks e Bell podem ser resumidos na Figura 3.14.

Figura 3.14: Processo de identificação e análise de serviços (MARKS; BELL, 2006)

3.4.2.1 Identificação de Serviços Candidatos

O processo de identificação de serviços deve ser conduzido de maneira *top-down*, com foco em obter serviços de negócio candidatos. Isto é, deve-se preocupar em identificar serviços potenciais que suportem o modelo de negócio da organização,

sem considerar neste momento o atual ambiente físico e tecnológico. Tais aspectos devem ser levados em conta em uma etapa posterior.

Os autores propõem alguns modos de se identificar serviços de negócio candidatos: análise de processo de negócio, análise de entidades, experiência de domínio de negócio, serviços pré-existentes e aplicações existentes. Na maior parte dos casos, todos os modos podem e devem ser utilizados para identificar potenciais serviços de negócio da organização.

A **análise de processo de negócio** é indicada para organizações que já façam uso de modelagem de processos, já possuindo uma base de documentação com detalhes dos principais processos de seu modelo de negócio. Neste caso, analisando-se o mapa de processos da organização, é possível identificar possíveis serviços de negócio a partir de atividades do processo que possam fazer uso da infra-estrutura de TI.

Pode-se abordar a identificação de serviços pela **análise de entidades**, que é uma perspectiva mais orientada a modelagem de dados. Nesta abordagem, serviços candidatos de negócio são identificados a partir de modelos de dados documentados e diagramas entidade-relacionamento. Primeiro, determinam-se as entidades centrais da organização, ou seja, as entidades com maior importância do ponto de vista do negócio (cliente, produtos, eventos, etc.). A seguir, deve ser feito um processo de *brainstorming* para identificar os processos e eventos relacionados a estas entidades, isto é, quais são os verbos que atuam sobre os substantivos (entidades) identificados. Tais verbos são candidatos a se tornar serviços de negócio.

Para auxiliar na identificação dos serviços, os autores propõem o uso da **experiência no domínio de negócio**, que pode ser representada por um indivíduo específico especialista no assunto ou pelo conhecimento da organização como um todo. A experiência e conhecimento sobre problemas e dificuldades mais comuns, pontos críticos do negócio e influências externas podem ser utilizados durante a análise de processo de negócio ou de entidade para guiar a identificação de serviços mais eficazes para a organização.

A organização pode já possuir um conjunto de **serviços pré-existentes** provenientes de projetos anteriores, projetos-piloto ou até mesmo serviços

desenvolvidos independentemente sem o conhecimento da organização. Estes serviços devem ser levados em consideração para otimizar investimentos passados. Serviços independentes devem ser publicados para toda a organização em um registro ou repositório. Deste modo, os serviços já existentes podem ser reusados, compostos e reprojatados de acordo com os requisitos do negócio.

Por fim, outra possível fonte de serviços candidatos a ser considerada são as **aplicações existentes** (sistemas de informação corporativos, aplicações legadas e bases de dados). Funções existentes nestas aplicações podem ser aproveitadas, agregadas ou fragmentadas para dar origem a serviços utilizáveis pelo negócio.

3.4.2.2 Análise de Serviços Candidatos

Uma vez identificados os serviços de negócio candidatos, estes devem ser analisados e projetados para implementação. A análise de serviços candidatos proposta por Marks e Bell consiste de operações lógicas de transformação sobre os candidatos para a formação de serviços de negócio finais. O foco principal desta atividade é controlar a granularidade e o potencial de reuso dos serviços desenvolvidos.

O primeiro passo é a análise de granularidade dos serviços de negócio candidatos. Cada candidato identificado na etapa anterior deve ter sua funcionalidade avaliada para a determinação de sua granularidade esperada. Como resultado, obtém-se um mapa de granularidade, exemplificado na Figura 3.15. Este mapa categoriza os serviços de negócio candidatos por nível de granularidade, posicionando os serviços com granularidade mais alta acima dos de granularidade mais baixa, formando uma hierarquia.

Figura 3.15: Mapa de granularidade de serviços (MARKS; BELL, 2006)

A partir daí, o segundo passo é a aplicação de uma série de operações lógicas sobre os serviços de negócio candidatos de acordo com suas funções e níveis de granularidade. Tais operações são transformações baseadas na teoria de conjuntos, a saber: união, interseção, decomposição, subconjunto e subtração. Estas operações lógicas são aplicadas de modo a agregar e fragmentar os serviços de negócio candidatos, reorganizando suas operações e dando origem aos serviços de negócio finais.

Na operação de **união**, dois ou mais serviços relacionados são agrupados e têm suas funções consolidadas para dar origem a um único serviço resultante. Esta operação não elimina nenhuma função existente, mas apenas agrupa as funções de serviços baseada em suas similaridades. Deve-se reavaliar a granularidade do serviço resultante, que tenderá a ser maior que a dos serviços unificados.

A operação de **interseção** é aplicada quando são identificadas funções comuns em dois ou mais serviços. As funções que se sobrepõem em vários serviços são retiradas dos serviços originais e agrupadas para dar origem a um novo serviço. Uma característica importante deste serviço gerado é a reuso, uma vez que ele já nasce com mais de um consumidor. Deve-se apenas atentar para não utilizar a interseção em serviços de granularidade muito baixa.

Alguns serviços de negócio candidatos podem possuir granularidade alta demais, o que dificultaria sua implementação e gerenciamento. Para estes casos, pode-se utilizar a operação de **decomposição**, que fragmenta o serviço inicial em diversos serviços desacoplados com menor granularidade. Deve-se tomar cuidado para que os serviços resultantes não apresentem interdependências entre si.

Na operação de **subconjunto**, um ou mais serviços de negócio candidatos são agregados a outro serviço de negócio candidato já previamente identificado. Esta operação trata serviços individuais como subconjuntos de um serviço maior. Esta agregação baseia-se no princípio de composição, por meio de técnicas de orquestração ou coreografia, onde os serviços menores suportam a operação interna do serviço de maior granularidade.

Por fim, a operação de **subtração** é aplicada no caso de serviços de negócio candidatos que possuem funções desnecessárias que podem dificultar sua implementação, reuso ou gerenciamento. A subtração remove função e lógica de negócio desnecessárias, para que o serviço final possa focar somente em funções mais relevantes. Em alguns casos, as funções removidas podem dar origem a um novo serviço se ainda possuírem valor para a organização. Caso contrário, elas são eliminadas completamente da arquitetura.

Após a aplicação das operações lógicas, analisa-se seu impacto sobre a granularidade dos serviços resultantes, atualizando o mapa de granularidade com os serviços de negócio finais.

3.4.3 Atividades de Projeto

Na fase de projeto de serviços, são criados os serviços físicos a partir dos serviços de negócio obtidos na fase de identificação e análise. O projeto começa com um mapa de granularidade contendo os serviços de negócio finais.

É importante ressaltar que as atividades de projeto não são focadas em detalhes da tecnologia de implementação. Em vez disso, o projeto preocupa-se com composições de serviços, lógica de negócio encapsulada por eles e reusabilidade de serviços.

Os passos da fase de projeto segundo o método proposto podem ser visualizados na Figura 3.16.

Figura 3.16: Processo de projeto de serviços (MARKS; BELL, 2006)

No primeiro passo, os serviços de negócio resultantes da fase de análise devem ser examinados sob o ponto de vista da viabilidade e da aderência à estratégia de SOA da organização. Assim, deve-se criar uma escala de prioridade dos serviços baseados em seus aspectos técnicos e de negócio e nos níveis de granularidade já estabelecidos. Esta escala de priorização servirá de base para selecionar os serviços que serão implementados como serviços físicos de solução.

A partir deste mapa de granularidade de serviços selecionados de acordo com a escala de prioridade, são construídos os mapas de demarcação, exemplificados na Figura 3.17. Neste mapa, são indicados grupos de serviços que serão transformados em serviços de solução físicos. O processo de demarcação é conceitual e agrupa serviços baseado em seu contexto comum, ou seja, domínio de negócio, padrões de consumo, requisitos de SLA ou ambiente de execução. Tal agrupamento deve iniciar pelos serviços de granularidade mais alta para os de mais baixa. Deste modo, serviços com potencial de reuso são detectados através da sobreposição de grupos de demarcação.

Figura 3.17: Mapa de demarcação (MARKS; BELL, 2006)

3.4.3.1 Projeto de Serviços Físicos

Com a elaboração dos mapas de demarcação de serviços, pode-se iniciar o projeto dos serviços de solução, que são os serviços físicos, tangíveis que executam suas funções estabelecidas e resolvem problemas e questões do negócio.

Na atividade de projeto são realizadas algumas operações sobre os serviços de negócio representados e agrupados no mapa de demarcação, com o intuito de estabilizar os níveis de granularidade, generalizar serviços para uma granularidade mais alta ou demovê-los para um nível de granularidade mais baixo. Estas operações focam em direcionar o atendimento das metas de negócio e de TI através da manipulação de grupos de serviços, em vez de serviços individuais. As operações possíveis de serem aplicadas são: promoção, demoção, importação, exportação, criação e remoção.

A **promoção** eleva o nível de granularidade de um serviço e é utilizada para aumentar o escopo de um determinado serviço e permitir que ele contenha outros serviços de granularidade mais baixa. Já a **demoção** tem efeito contrário e diminui o nível de granularidade de um serviço. É utilizada no caso de serviços que possuem

escopo muito pequeno para serem mantidos em um nível alto de granularidade, ou quando um serviço pode ser encapsulado por outro mais genérico.

A **importação** de serviços permite a expansão do escopo de um determinado grupo de serviços de maneira horizontal, isto é, sem elevar ou reduzir seu nível de granularidade. A importação é aplicada quando se pretende adicionar a um determinado grupo de serviços uma função representada por um serviço pertencente a outro grupo. A importação pode ser exclusiva, movendo-se o serviço de um grupo para outro, ou compartilhada, mantendo o serviço como parte dos dois grupos. A **exportação** é a operação análoga, que representa o ponto de vista do grupo de onde um serviço pode ser retirado para compor outro grupo. Também pode ser exclusiva ou compartilhada.

Já a **criação** e a **remoção** flexibilizam o projeto permitindo, respectivamente, que se incluam e excluam serviços do mapa de demarcação conforme a necessidade.

3.4.3.2 Realização de Serviços de Solução

O passo final do processo de projeto consiste na atividade de realização dos serviços, obtendo-se os serviços físicos de solução. Nesta atividade, o foco passa a ser na modelagem física dos serviços e em sua arquitetura. Os serviços de negócio representados nos mapas de granularidade e de demarcação elaborados nas etapas anteriores são transformados em elementos concretos, que podem ser serviços de solução, componentes físicos ou processos de serviço. Os três tipos de transformação são representados na Figura 3.18.

Figura 3.18: Tipos de transformação lógico-físico (MARKS; BELL, 2006)

- A transformação de serviços de negócio para serviços de solução é direta, somente passando do campo conceitual para o concreto, não sendo necessária nenhuma mudança de nível de granularidade. Este tipo de transformação é geralmente utilizado nos serviços de negócio de granularidade mais alta.
- A transformação de serviços de negócio para componentes físicos é geralmente aplicada em serviços de menor granularidade, ou pode demover serviços para um nível mais baixo.
- Já a transformação para processos de serviço é usada quando a função encapsulada possui um escopo realmente pequeno, sendo mapeada para um processo pertencente a um serviço de solução.

Estas decisões de transformação de lógico para físico devem ser baseadas em aspectos de negócio, tecnológicos e organizacionais. Diferentes diretrizes para a definição de granularidade e particularidades do ambiente tecnológico podem produzir diferentes tipos de cenários de transformação.

3.4.4 Papéis

O método descrito por Marks e Bell não identifica papéis para a execução das atividades.

3.4.5 Artefatos

O mapa de granularidade é o diagrama produzido durante a atividade de análise do método proposto por Marks e Bell. Neste diagrama, os serviços são divididos em níveis de granularidade, representando-se os níveis mais altos de granularidade na parte de cima e os níveis mais baixos na parte de baixo. O mapa de granularidade permite uma visualização gráfica das relações de granularidade dos serviços e possibilita estabelecer-se uma hierarquia de prioridade entre eles.

Já o mapa de demarcação, utilizado nas atividades de projeto, é obtido a partir do mapa de granularidade e representa agrupamentos lógicos de serviços por meio de demarcações. Serviços com similaridades no contexto do negócio e da tecnologia são considerados como parte de um mesmo grupo e são representados dentro da mesma linha de fronteira.

3.4.6 Análise do Método

O método de Marks e Bell procura aliar a abordagem *top-down* de identificação de serviços com a abordagem *bottom-up* para a realização do serviço. No entanto, não são descritas explicitamente atividades para analisar alternativas de realização para cada serviço.

São oferecidas diversas fontes para identificação de serviços, sendo uma delas a análise de processo de negócio, o que permite a sua aplicação em um cenário com modelagem de processos. Porém, outros aspectos do BPM, como a orquestração de serviços, não são considerados.

Na proposta, é possível também identificar serviços candidatos a partir de aplicações e serviços pré-existentes. Tal característica possibilita o

reaproveitamento de investimento já realizado, através do reuso de recursos existentes. Mas como não há uma atividade para a tomada de decisões de realização, este reuso não se torna possível para serviços identificados a partir de outras fontes.

O conceito de serviço candidato é utilizado como parte do ciclo de vida de serviços definido pelo método. São definidos três estados para um serviço durante este ciclo de vida: serviço de negócio candidato, serviço de negócio final e serviço de solução (físico). No momento em que é identificado, um serviço de negócio é considerado candidato. Após a análise de granularidade e aplicação das operações lógicas, os serviços de negócio são classificados como finais. Finalmente, após as atividades de projeto e realização, obtêm-se os serviços finais que serão implementados.

Marks e Bell dão bastante ênfase ao controle da granularidade dos serviços a serem desenvolvidos. Na análise de serviços candidatos, o método propõe o uso dos mapas de granularidade para classificar os serviços candidatos de acordo com seu nível de granularidade. No passo seguinte, são utilizadas operações lógicas derivadas da teoria dos conjuntos (união, interseção, decomposição, subconjunto e subtração) para manipular os serviços de forma que eles tenham níveis adequados de granularidade.

Além da granularidade dos serviços, o método considera também os princípios de reusabilidade e composição. A reusabilidade é levada em conta durante as operações lógicas na análise de serviços candidatos, enquanto o princípio da composição é exercitado durante a realização dos serviços de solução.

O método não descreve alternativas concretas para especificação dos contratos dos serviços, de seu comportamento funcional ou de seus requisitos não funcionais.

O método de Marks e Bell consiste em uma alternativa mais abstrata de desenvolvimento orientado a serviços, pois não se prende a uma abordagem específica como o BPM nem a uma tecnologia específica, como *Web Services*.

3.5 Considerações Finais

O Capítulo 3 apresentou uma descrição e análise dos métodos existentes para desenvolvimento orientado a serviços. Foram analisadas as boas práticas utilizadas em cada método. Além disso, soluções que cada método adota para solucionar cada requisito podem ser levadas em conta na proposição de uma nova abordagem.

Pode-se constatar pela análise que *Business Process Management* e a tecnologia *Web Services* são aplicados em conjunto com SOA com tanta frequência que alguns métodos tomam como premissa o uso dessas abordagens, como as propostas de Erl e Papazoglou e Van Den Heuvel.

Outra característica predominante nos métodos é a análise de recursos existentes, seja para a identificação ou para a realização dos serviços. Sendo o reuso um dos principais benefícios esperados da orientação a serviços, é fundamental considerar os ativos existentes na organização.

Os métodos analisados são as principais referências para a elaboração de uma nova proposta de método. O método proposto deve procurar unificar as abordagens analisadas contemplando boas práticas encontradas em cada uma delas. A partir da análise das propostas, é possível também levantar os requisitos de análise e projeto de serviços.

4 REQUISITOS DE UM MÉTODO DE ANÁLISE E PROJETO ORIENTADO A SERVIÇOS

A partir da análise dos métodos da literatura, foram identificadas boas práticas a partir das quais é possível inferir um conjunto de requisitos que um método de análise e projeto orientado a serviços deveria satisfazer. Estas boas práticas consistem em soluções propostas pelos métodos para atender às necessidades de análise e projeto de serviços.

4.1 Descrição dos Requisitos

Nas seções a seguir, são descritos os requisitos levantados. Os requisitos são identificados com códigos R1, R2, R3, etc. para referência ao longo do texto.

4.1.1 Abordagem *meet-in-the-middle* (R1)

Um método de análise e projeto orientado a serviços não pode limitar-se a propor somente uma abordagem *top-down*, levando em conta apenas os requisitos e objetivos do negócio para a identificação e especificação de serviços. Tampouco deve ser uma abordagem puramente *bottom-up*, apenas encapsulando funcionalidade de aplicações legadas na forma de serviços. Um método deve de alguma forma equilibrar as duas alternativas, resultando em uma abordagem *meet-in-the-middle*.

A identificação de serviços deve ser guiada pelos requisitos e pela modelagem de negócio, assim como estar alinhada às suas metas e estratégias de negócio. No entanto, os serviços identificados a partir do negócio devem ser analisados e revisados de modo a ajustar-se à realidade dos serviços já existentes na organização e das aplicações legadas. Desta maneira, obtém-se o reuso de recursos existentes e cria-se valor a partir do investimento já realizado.

O método de Papazoglou e Van Den Heuvel atende a este requisito, pois os serviços são propostos como parte do processo de negócio, mas são comparados com os recursos existentes na análise de *gap*.

O RUP *plugin for SOMA* também possui esta característica, pois procura alinhar os serviços aos requisitos de negócio e aos sistemas existentes.

4.1.2 Compatibilidade com Business Process Management (R2)

A adoção da arquitetura orientada a serviços em uma organização pode trazer maiores benefícios se realizada em conjunto com a adoção de práticas de *Business Process Management* (BPM). Com o surgimento do padrão WS-BPEL, a utilização conjunta de SOA e de BPM foi viabilizada e potencializada. O WS-BPEL permite a composição e orquestração de serviços a partir de definições de processos, criadas por analistas de negócio. Desta maneira, aumenta-se o alinhamento dos serviços desenvolvidos com as metas e estratégias de negócio.

Um método orientado a serviços deve levar em consideração esta sinergia e incorporar técnicas de BPM para a análise e o projeto de serviços (PAPAZOGLU; VAN DEN HEUVEL, 2006). Nas atividades de análise e projeto, dois aspectos do BPM são aplicáveis: a modelagem de processos e a automação de processos.

A técnica de modelagem de processos pode ser empregada para documentar o modelo de negócio e seus requisitos, que por sua vez, serão o ponto de partida para a identificação de serviços.

Já a automação de processos em uma arquitetura SOA pode ser realizada por meio da orquestração de serviços. Caso os processos sejam modelados em uma notação como o BPMN, eles poderão ser convertidos para uma linguagem de orquestração como WS-BPEL e executados em um servidor de aplicações.

Um método deve viabilizar projetos de BPM com SOA, mas não deve limitar-se a esta abordagem, permitindo também a realização de projetos de SOA somente.

Os métodos de Papazoglou e Van Den Heuvel e de Erl contemplam ambos os aspectos de BPM. Já o RUP *plugin for SOMA* e o método de Marks e Bell

descrevem somente a elaboração de modelos de processos e a identificação de serviços a partir deles, não abordando a orquestração de serviços.

4.1.3 Reuso de recursos existentes (R3)

Um dos princípios fundamentais da orientação a serviços é o reuso. Sendo assim, um método de análise e projeto orientado a serviços deve procurar maximizar o reaproveitamento de lógica já existente, esteja ela na forma de serviços disponíveis no portfólio da organização ou na forma de aplicações legadas, que ainda teriam que ser transformadas em serviços.

Segundo este requisito, durante o processo de análise e projeto de uma solução orientada a serviços, os recursos existentes na organização devem ser analisados como alternativas para identificação ou realização de serviços.

O método de Papazoglou e Van Den Heuvel oferece uma solução interessante para este requisito, que é a inclusão da atividade de análise de *gap*. Esta é uma atividade que possui exatamente o intuito de avaliar se há algum serviço ou sistema legado existente que atenda aos requisitos definidos durante a modelagem dos processos e serviços.

Já o RUP *plugin for SOMA* e o método de Marks e Bell analisam os recursos existentes como uma alternativa de identificação de serviços do tipo *bottom-up*.

A proposta de Erl não atende a este requisito, pois apesar de as aplicações legadas existentes serem identificadas, elas não são consideradas como alternativas para realização de serviços. Pelo contrário, tal identificação tem como propósito detectar as tarefas de um processo de negócio em que não serão utilizados serviços.

4.1.4 Especificação de requisitos não-funcionais (R4)

Um ponto bastante questionado de implementações de arquitetura SOA é o suporte a requisitos não-funcionais dos serviços, também chamados de requisitos de qualidade de serviço (QoS). Nesta área, existe uma série de padrões abertos sendo

desenvolvidos no sentido de especificar o tratamento de requisitos como segurança, desempenho e transações.

Um método proposto deve buscar formas de abordar requisitos não-funcionais, ou seja, incluir em seu fluxo de trabalho atividades em estes requisitos são definidos e documentados como parte da especificação de um contrato de serviço. Para tal, o método pode adotar padrões específicos, como WS-Security, WS-Policy ou WS-AtomicTransaction.

Dos métodos analisados, apenas o RUP *plugin for SOMA* e a abordagem de Papazoglou e Van Den Heuvel descrevem de maneira explícita a definição dos requisitos não-funcionais, em suas respectivas atividades de especificação de serviços.

4.1.5 Diferentes alternativas para identificação de serviços (R5)

Um método de análise e projeto orientado a serviços deve oferecer flexibilidade aos analistas no momento de identificar potenciais serviços necessários para a organização a partir da modelagem de negócio. Como ponto de partida para esta identificação, podem ser utilizados modelos de processos, descrições de casos de uso de negócio e modelos de dados para a abordagem *top-down* e, no caso *bottom-up*, aplicações legadas e serviços já existentes que podem ser reusados em uma composição.

O RUP *plugin for SOMA* permite identificar novos serviços a partir de diversas fontes, como processos de negócio, casos de uso de negócio, aplicações existentes, regras de negócio e modelos de dados.

Já no método de Marks e Bell, podem ser utilizadas as seguintes fontes: processos de negócio, entidades de dados, requisitos de negócio e aplicações existentes.

Os métodos de Eri e de Papazoglou e Van Den Heuvel permitem apenas a identificação de serviços a partir de processos de negócio.

4.1.6 Diferentes alternativas para realização de serviços (R6)

Assim como no caso da identificação, deve existir flexibilidade para a especificação de serviços, sendo considerados diversos cenários de realização. Um serviço identificado pode ser especificado como um novo componente a ser desenvolvido, como uma composição de serviços existentes, na forma de reuso de um serviço já existente com potenciais ajustes ou na forma de reuso de lógica de alguma aplicação legada, que pode ser disponibilizada por um serviço encapsulando ou por meio de alguma API ou adaptador.

Um método deve levar em conta os diversos cenários possíveis para realização de um serviço e analisar aspectos como custo, risco, esforço e ROI, o que permitirá ao analista que estiver criando a especificação decidir qual abordagem adotar.

A proposta de Papazoglou e Van Den Heuvel descreve a atividade de Análise de Realização, em que decide-se como cada serviço será desenvolvido com base nas alternativas disponíveis: *green-field*, *top-down*, *bottom-up* e *meet-in-the-middle*.

No RUP *plugin for SOMA*, uma análise semelhante ocorre na atividade Decisões de Realização, no entanto são considerados cenários de integração com aplicações existentes.

4.1.7 Conceito de serviço candidato (R7)

De acordo com este requisito, o ciclo de vida de um serviço deve possuir um passo intermediário entre o momento em que é concebido ou identificado e o momento em que ele é especificado. Neste passo intermediário, o serviço seria classificado como candidato. Em um método orientado a serviços, o conjunto de serviços de uma solução concebidos ou identificados não é necessariamente realizado e especificado em seu estado original. Por este motivo, ele é considerado candidato até ser passar por análises e revisões, que podem resultar em mudanças nas operações, reagrupamento de operações e redefinição do escopo funcional de cada serviço.

No RUP *plugin for SOMA*, o uso deste conceito viabiliza a abordagem *meet-in-the-middle*, pois os serviços identificados a partir da modelagem de negócio são

candidatos até serem revisados durante o Teste Litmus de Serviço, considerando possíveis restrições do ambiente tecnológico existente na organização ou outras limitações externas.

Já o método de Erl utiliza o conceito de serviço candidato durante toda a análise e projeto orientado a serviços. Desde sua identificação, os serviços são classificados como candidatos, deixando de ser candidatos apenas após sua especificação.

No método de Marks e Bell, a classificação de candidato é um dos vários passos do ciclo de vida de um serviço utilizados durante a análise e projeto. Segundo este ciclo de vida, a identificação dá origem a serviços de negócio candidatos. Estes por sua vez, passam a ser serviços de negócio finais após a análise de granularidade e a aplicação de operações lógicas, tornando-se serviços de solução após a especificação.

4.1.8 Aderência aos princípios de orientação a serviços (R8)

Um método de análise e projeto orientado a serviços deve assegurar que os serviços desenvolvidos sejam aderentes aos princípios de orientação a serviços, como reusabilidade, interoperabilidade, baixo acoplamento e autonomia. Esta aderência pode ser obtida inserindo-se no fluxo de trabalho do método pontos de verificação em que um conjunto de serviços é avaliado em relação aos princípios. Outra maneira seria por meio de orientações sobre como executar uma determinada atividade levando em conta os princípios.

O método proposto por Papazoglou e Van Den Heuvel inclui orientações na atividade de Especificação de Serviços sobre como projetar contratos de serviços de acordo com os princípios de alta coesão e baixo acoplamento.

O RUP *plugin for SOMA* possui uma atividade de verificação dos serviços, que é o Teste Litmus de Serviço. Nesta atividade, porém, são validados somente os princípios de reusabilidade e facilidade de composição.

No método de Erl, existem atividades para validação dos princípios de orientação a serviços tanto na fase de análise quanto na de projeto. Os princípios verificados são reusabilidade, autonomia, independência de estado e baixo acoplamento. De acordo

com a fase e com o tipo de cada serviço, diferentes conjuntos de princípios podem ser validados.

O método de Marks e Bell leva em conta a reusabilidade durante a análise de serviços candidatos e a facilidade de composição durante a realização dos serviços de solução. Além disso, o método dá ênfase ao controle da granularidade dos serviços.

4.1.9 Uso de padrões abertos (R9)

Caso um método utilize de maneira explícita linguagens e notações específicas para representar seus artefatos, estas linguagens e artefatos devem adotar padrões abertos e que sejam independentes de plataforma ou de fornecedor de ferramentas de implementação.

Exemplos de padrões abertos relacionados a análise e projeto de serviços são: o WSDL para descrição de interfaces de serviço, o XSD para definição de formato de dados, o WS-BPEL para orquestração de processos de negócio, o BPMN para modelagem de processos, o UML para documentar requisitos e arquitetura de software e os diversos padrões WS-* para uso com a tecnologia *Web Services*.

Os métodos de Erl e de Papazoglou e Van Den Heuvel, por serem alinhados à tecnologia *Web Services*, adotam de maneira explícita os padrões WSDL, XSD e WS-BPEL. Papazoglou e Van Den Heuvel citam ainda o uso do padrão WS-Policy para especificar requisitos não-funcionais de serviços. Para representar os serviços, Erl faz uso de uma notação própria e não-padronizada.

O RUP *plugin for SOMA* adota o UML como principal forma de documentar os artefatos elaborados, como modelos de casos de uso para representar o negócio, modelos de classe para representar contratos de serviços e mensagens e diagramas de seqüência para representar interações entre serviços.

Já o método de Marks e Bell não aborda técnicas específicas de implementação dos serviços, além de utilizar diagramas próprios e não padronizados, como os mapas de granularidade e demarcação, durante as atividades de análise e projeto.

4.1.10 Uso de técnicas existentes para implementação (R10)

Um método de análise e projeto deve ser estruturado de maneira tal que os seus produtos finais (as especificações de serviços) possam servir como entrada para atividades de uma metodologia de implementação já existente e consolidada (PAPAZOGLU; VAN DEN HEUVEL, 2006), como programação orientada a objetos ou desenvolvimento baseado em componentes (CBD).

Para que isto seja possível, é necessário que os serviços sejam especificados em um nível detalhado o suficiente para a realização de modelagem de classes e de interfaces de componentes.

Dos métodos analisados, apenas o RUP aborda explicitamente a passagem para atividades após a análise e projeto. Para representar os requisitos dos serviços especificados, o RUP *plugin for SOMA* utiliza diagramas em UML de classes, de sequência, de componentes, entre outros.

4.1.11 Conceito de camadas de serviços (R11)

Um método de análise e projeto deve classificar os serviços especificados em camadas, de acordo com sua função ou nível de granularidade. As camadas de serviços (ERL, 2005) definem uma maneira de se separar os tipos de lógica contidos em serviços e de se representar os relacionamentos dos serviços com os processos de negócio e com as aplicações existentes.

Dependendo da camada a que um serviço pertence ou de seu nível de granularidade, a análise e projeto devem ser levar em consideração aspectos específicos, como o nível de reusabilidade e presença de lógica de processo.

O método de Erl utiliza o conceito de camadas de serviço, separando os serviços nas camadas de orquestração, negócio e aplicação. Na fase de Projeto, são descritas atividades específicas para especificar serviços de cada camada.

O método de Marks e Bell não utiliza explicitamente o conceito de camadas, porém classifica os serviços de acordo com o nível de granularidade, definindo abordagens de realização específicas para cada nível.

4.1.12 Uso livre (R12)

O método deve ser livre, isto é, ser não-proprietário. O acesso à sua descrição e o seu uso devem ser livres, para que possa ser revisado e adaptado livremente por qualquer organização.

Todos os métodos analisados podem ser considerados de uso livre, com exceção do RUP *plugin for SOMA*, cuja utilização plena está vinculada à aquisição de licenças da ferramenta *Rational Method Composer* da IBM.

4.2 Aderência dos métodos analisados aos requisitos

Pode-se notar pela análise dos métodos existentes que nenhum deles atende por completo a todos os requisitos levantados, como pode ser visto na Tabela 4.1. Esta tabela resume a aderência dos métodos analisados aos requisitos descritos, permitindo a identificação da origem de cada requisito. A notação **X** significa que o requisito é plenamente atendido pelo método analisado, enquanto que a indicação **Parcial** denota um atendimento parcial do requisito. Quando a correspondência entre método e requisito estiver em branco, significa que o requisito não é atendido pelo método.

Como boas práticas a serem destacadas no método de Papazoglou e Van Den Heuvel estão a compatibilidade com BPM, abordando a modelagem de processos tanto para identificação de serviços quanto para a orquestração de processos, e as análise de *gap* e de realização, que possibilitam o reuso de recursos existentes.

Já o RUP *plugin for SOMA* possui maior flexibilidade para identificação e realização de serviços, descrevendo diversas abordagens para ambas as atividades, além de já ser integrado a técnicas de implementação orientadas a objetos e componentes.

O método proposto por Erl é bastante detalhado na validação dos princípios da orientação a serviços, possuindo atividades específicas para sua verificação, sendo também alinhado à abordagem BPM.

No método de Marks e Bell, podem ser destacadas a flexibilidade para identificação de serviços e as diversas etapas definidas para o ciclo de vida dos serviços.

Tabela 4.1: Resumo da aderência dos métodos analisados aos requisitos

	Requisito	Papazoglou & Van Den Heuvel	RUP <i>Plugin for</i> SOMA	Erl	Marks & Bell
(R1)	Abordagem <i>meet-in-the-middle</i>	X	X		Parcial
(R2)	Compatibilidade com BPM	X	Parcial	X	Parcial
(R3)	Reuso de recursos existentes	X	X	Parcial	X
(R4)	Especificação de requisitos não-funcionais	X	X		
(R5)	Diferentes abordagens de identificação de serviços		X		X
(R6)	Diferentes abordagens de realização serviços	X	Parcial		
(R7)	Conceito de serviço candidato		X	X	X
(R8)	Aderência aos princípios de orientação a serviços	Parcial	Parcial	X	Parcial
(R9)	Uso de padrões abertos	X	Parcial	X	
(R10)	Uso de técnicas existentes para implementação		X		
(R11)	Conceito de camadas de serviços			X	Parcial
(R12)	Uso livre	X		X	X

4.3 Considerações Finais

No Capítulo 4, foram apresentados os requisitos levantados para um método de análise e projeto orientado a serviços. Tais requisitos foram determinados a partir das características dos métodos estudados.

Não há como afirmar que o conjunto de requisitos levantado seja um conjunto completo, contemplando todos os requisitos possíveis para análise e projeto de serviços, mas apenas aborda os requisitos identificados como boas práticas nos métodos analisados.

O método proposto por este trabalho deve conter atividades e soluções para atender a cada um dos requisitos levantados, de modo a unificar os métodos existentes.

5 PROPOSTA DE MÉTODO

Como se pode inferir a partir da análise das propostas existentes em relação aos requisitos, torna-se necessário consolidar os métodos analisados em um método que unifique as boas práticas de cada um e que atenda aos requisitos de análise e projeto orientados a serviços.

Este capítulo descreve o Método de Análise e Projeto Orientado a Serviços (MAPOS) (FUGITA; HIRAMA, 2008a) (FUGITA; HIRAMA, 2008b), uma proposta de método elaborada a partir do estudo e análise dos métodos existentes e dos princípios de orientação a serviço.

A descrição do método segue a mesma estrutura das descrições do Capítulo 3, com a adição de algumas atividades que antecedem e sucedem o método no ciclo de vida.

5.1 Ciclo de Vida

O método procura oferecer flexibilidade em sua aplicação por meio de iterações durante a fase de Análise e de Projeto. O ciclo de vida descrito inicia-se com a Modelagem de Negócio, passa pelas fases de Análise e Projeto (abrangidas pelo método), seguindo adiante com atividades de Construção, como Implementação e Testes, conforme pode ser visualizado na Figura 5.1.

Figura 5.1: Método de Análise e Projeto Orientado a Serviços

A seguir são descritas as atividades de todo o ciclo de vida, a partir da fase de Modelagem de Negócio que precede a Análise e o Projeto Orientados a Serviços, até a fase de Construção subsequente. Para cada atividade, são explicitados o papel da pessoa que a desempenha, os artefatos utilizados em sua execução e os gerados.

5.2 Atividades de Modelagem de Negócio

A Modelagem de Negócio ocorre antes das atividades de análise do método propriamente dito. Nesta fase, busca-se um entendimento sobre as necessidades de negócio e levantam-se os requisitos. As atividades de Modelagem de Negócio são executadas por pessoas da área de negócio ou da área de processos da organização.

5.2.1 Modelagem de Processo *as-is* e *to-be*

Na Modelagem de Processo *as-is*, é levantado o estado atual dos processos de negócio envolvidos no projeto. É elaborado um modelo representando o estado do negócio antes do projeto da solução orientada a serviços.

Na Modelagem de Processo *to-be*, o analista de processos pode realizar avaliações e simulações e propor modificações ao processo com o intuito de promover melhorias de desempenho ou reduzir custos. Tais modificações do processo podem ser mudanças em sua própria estrutura (fluxo de trabalho) ou propostas de automatização de parte ou de todo o processo. Como produto desta atividade, é elaborado um novo modelo de processo alterado com relação ao *as-is*.

Os modelos podem ser representados na forma de casos de uso de negócio ou preferencialmente utilizando uma notação de processo, como BPMN, que pode ser convertida diretamente para a linguagem WS-BPEL por diversas ferramentas existentes no mercado.

O uso de notações baseadas em padrões abertos, como casos de uso e BPMN, atende ao requisito **R9**.

Papéis: Analista de Processo

Artefato utilizado: Requisitos de Negócio

Artefatos gerados: Modelo de Processo *as-is*, Modelo de Processo *to-be*

5.3 Atividades de Análise

Na fase de Análise, são identificados serviços candidatos a partir dos modelos de negócio fornecidos. Ainda nesta fase, decide-se como cada serviço será realizado, analisando-se possibilidades de reuso para cada um, chegando a um conjunto de serviços finais.

5.3.1 Identificação de Serviços Candidatos

A atividade de identificação de serviços candidatos é a primeira atividade de Análise e recebe como entrada um modelo de processo *to-be* ou modelo de casos de uso de negócio produzido durante a Modelagem de Negócio.

O modelo de processo *to-be* deve ser decomposto em uma série de passos de baixa granularidade e então devem ser identificados quais os passos poderão ser automatizados. Passos não computacionais (execução de tarefas humanas) do processo não serão considerados nesta etapa, pois não poderão ser executados por operações de serviços.

Em um modelo de casos de uso, deve-se analisar as descrições dos fluxos de trabalho de cada caso de uso, separando os passos executados pelo ator usuários dos passos automatizados, ou seja, executados por algum sistema ou aplicação.

O Analista de Serviços deve listar os passos que serão automatizados, de modo que cada um destes passos poderá ser mapeado para uma operação de um serviço sendo invocada. As operações relacionadas devem então ser agrupadas, de modo que cada grupo de operações pertença a um serviço candidato.

Devem ser agrupadas operações que compartilhem um mesmo contexto lógico (semântico) ou que trabalhem com as mesmas entidades de dados (ERL, 2005). Em caso de operações identificadas a partir de um modelo de processo, podem ser agrupadas operações executadas por um mesmo papel. Em caso de operações identificadas de um caso de uso, podem ser agrupadas operações relacionadas a um mesmo ator do caso de uso.

Por utilizar um modelo de processo como fonte para identificação de serviços, o método procura atender ao requisito **R2**, ou seja, ser compatível com a abordagem BPM. E o requisito **R5** também é atendido, pois é possível identificar serviços a partir de processos ou casos de uso. Ao utilizar o conceito de serviços e operações candidatos, o método proposto atende ao requisito **R7**.

Papel: Analista de Serviços

Artefato utilizado: Modelo de Processo *to-be*

Artefato gerado: Modelo de serviços candidatos

5.3.2 Análise de Gap

A Análise de *Gap* tem como objetivo comparar os serviços candidatos identificados com os recursos já existentes na infra-estrutura de TI da organização e avaliar as possibilidades de reuso (PAPAZOGLU; VAN DEN HEUVEL, 2006). Para cada operação candidata (pertencente a um serviço candidato), deve-se verificar se já existe algum serviço ou aplicação de software legada que execute total ou parcialmente sua funcionalidade. Para realizar esta busca, pode ser consultado um repositório de serviços e recursos reusáveis ou documentações de aplicações existentes.

Assim, para cada operação analisada, pode-se identificar um dos seguintes cenários de reuso:

- Operação realizada por um serviço: A funcionalidade da operação avaliada já está implementada por um serviço existente no repositório de serviços da organização, que pode então ser reusado.
- Operação realizada parcialmente por um serviço: A funcionalidade da operação avaliada é parcialmente realizada por um serviço existente. O serviço pode ser reusado, mas será necessário alterar o serviço existente ou implementar uma mediação para utilizar o serviço.
- Operação realizada por uma aplicação legada: A funcionalidade da operação avaliada já está implementada por uma aplicação legada. Para poder ser usada pelo processo, esta funcionalidade deverá ser encapsulada na forma de um serviço.
- Operação realizada parcialmente por uma aplicação legada: A funcionalidade da operação avaliada é parcialmente realizada por uma aplicação legada. Para poder ser usada pelo processo, esta funcionalidade deverá ser encapsulada na forma de um serviço e deverá ser implementado algum tipo de lógica de mediação ou integração.
- Operação não existente: A funcionalidade da operação não existe em nenhum recurso existente e deve, portanto, ser realizada por um novo serviço a ser criado ou por uma nova operação a ser criada para um serviço existente.

O agrupamento das operações candidatas em serviços pode necessitar ser revisado em vista do resultado da Análise de *Gap*. Serviços candidatos podem ser alterados caso haja sobreposição com algum serviço já existente. Por exemplo, uma operação que já é realizada por um serviço existente passará a pertencer a ele. Ou ainda, uma nova operação a ser criada pode pertencer ao mesmo contexto lógico de um serviço existente, sendo portanto acrescentada a ele.

Esta atividade busca atender ao requisito **R3** ao considerar os recursos existentes para a realização das operações dos serviços candidatos. Por confrontar os serviços candidatos identificados de modo *top-down* com os recursos existentes levantados de modo *bottom-up*, a atividade caracteriza uma abordagem *meet-in-the-middle*, atendendo ao requisito **R1**. Juntamente com a atividade de Identificação de Serviços Candidatos, atende ao requisito **R5** permitindo que serviços sejam identificados a partir de modelos de processos e aplicações existentes.

Papel: Analista de Serviços

Artefatos utilizados: Modelo de serviços candidatos, Documentação de aplicações existentes

Artefato gerado: Não aplicável

5.3.3 Análise de Realização

Uma vez identificados os cenários de reuso para cada operação candidata em relação aos recursos existentes, devem então ser analisadas as alternativas de realização possíveis para cada uma delas e decidir qual será adotada para realizar o serviço.

A Análise de Realização tem por objetivo avaliar as alternativas para realização de cada operação candidata em termos de viabilidade técnica, custos, riscos técnicos e de projeto, esforço e retorno de investimento. Para cada cenário de reuso identificado, as seguintes considerações devem ser feitas:

- Operação realizada por um serviço: Não é necessário especificar o serviço, uma vez que ele poderá ser utilizado diretamente pelo processo de negócio sem a necessidade alteração ou integração. Este é o cenário com menor

risco de ser realizado, acarretando apenas no custo de reuso. É o cenário que aumenta o retorno sobre o investimento já realizado anteriormente.

- Operação realizada parcialmente por um serviço: Neste cenário, duas opções devem ser analisadas. Pode-se alterar o serviço existente, o que traz um risco a ser considerado (impacto em outros processos e serviços existentes), ou implementar uma lógica de mediação para reusar o serviço em seu estado atual.
- Operação realizada por uma aplicação legada: Este cenário exige a criação de um serviço que encapsule a funcionalidade da aplicação legada. Para tal, é necessário verificar a existência de APIs ou adaptadores para acessar o legado e, caso contrário, avaliar a viabilidade e o custo de se implementar este acesso.
- Operação realizada parcialmente por uma aplicação legada: Este cenário levanta as mesmas questões que o cenário anterior, mas também oferece a possibilidade de se alterar o legado para corresponder à operação candidata, sendo necessária apenas a criação do serviço encapsulador. Deve-se avaliar se os custos e o risco tornariam esta alternativa viável.
- Operação não existente: Como a operação deverá ser implementada a partir do zero, o risco é baixo, pois se tem controle sobre sua especificação e realização. O custo resume-se ao esforço necessário para se especificar e implementar a nova funcionalidade.

Para cada cenário de reuso, ainda podem ser levantadas outras alternativas de realização específicas de cada caso analisado.

Após a análise de todas as operações candidatas, o Analista de Serviços deve decidir se será viável a realização dos serviços candidatos conforme as considerações feitas. Para a tomada da decisão final, pode ser necessário o apoio do Gerente de Projeto, pois são avaliados aspectos como risco, custo e esforço, e do Arquiteto Corporativo, pois pode haver alterações em aplicações e serviços existentes.

Caso seja definida uma alternativa de realização viável, o método prossegue com a especificação dos serviços. Caso contrário, os resultados da Análise de Realização devem ser passados para o Analista de Processos para que a modelagem de processo *to-be* seja revisada e as atividades de Análise repetidas.

Esta atividade busca atender ao requisito **R6** ao oferecer diversos cenários de realização de serviços, dependendo da disponibilidade de recursos reusáveis.

Papéis: Analista de Serviços, Gerente de Projeto, Arquiteto Corporativo

Artefatos utilizados: Modelo de serviços candidatos, Documentação de aplicações existentes

Artefato gerado: Não aplicável

5.4 Atividades de Projeto

Definidos os serviços, inicia-se a fase de Projeto de serviços. O objetivo da fase de Projeto é criar especificações de serviços que satisfaçam aos requisitos do negócio e sejam aderentes aos princípios de orientação a serviços (ERL, 2005). Como produtos desta fase, temos a orquestração dos serviços e especificações técnicas de serviços que servem como base para a Construção.

5.4.1 Especificação de Serviços

A primeira atividade de Projeto é a Especificação de Serviços, na qual as interfaces dos serviços serão definidas e suas operações (ou sua lógica) serão especificadas.

Para cada serviço candidato, deverão ser definidas todas as suas operações e as mensagens de entrada e saída envolvidas em cada operação. Caso a tecnologia utilizada seja *Web Services*, os formatos das mensagens podem ser definidos na forma de esquemas XSD e as definições de interfaces, representadas na linguagem WSDL, atendendo ao requisito **R9**.

Para cada serviço, será definido um artefato de Especificação de Serviço, composto por definição de interface, esquemas de dados de mensagens, especificação funcional de cada operação e políticas de serviço.

As mensagens de um serviço serão definidas com base nas informações necessárias como entrada e saída de suas operações e com base em modelos de dados existentes. Deve-se procurar especificar mensagens padronizadas segundo

um modelo de dados organizacional, para que uma mesma mensagem possa ser utilizada em diversos serviços.

Dependendo do cenário, a elaboração da interface do serviço pode ser realizada de modo *top-down*, *bottom-up* ou *meet-in-the-middle*.

- Operação realizada por um serviço: Neste cenário, o serviço já existe e possui interface definida, não sendo necessário nenhum tipo de especificação adicional.
- Operação realizada parcialmente por um serviço: Caso tenha se optado por alterar o serviço existente, pode ser necessário alterar sua interface e/ou a especificação funcional da operação a ser chamada. Caso seja utilizada uma mediação, deve-se definir a interface que será exposta por ela e especificar sua lógica de integração. Em ambos os casos, a especificação da interface é feita de modo *top-down*, sendo realizada a partir do serviço candidato.
- Operação realizada por uma aplicação legada: Neste cenário, a interface é elaborada de maneira *bottom-up*, tendo suas mensagens e operações definidas a partir das características da aplicação legada sendo reusada.
- Operação realizada parcialmente por uma aplicação legada: A interface pode ser elaborada de modo *bottom-up* como no cenário anterior, ou caso se tenha optado por alterar o legado, a interface será derivada de modo *bottom-up*, mas deverá ser especificada como o legado deverá ser alterado de modo *top-down*, caracterizando uma especificação *meet-in-the-middle*.
- Operação não existente: Neste cenário, tanto a elaboração da interface quanto da especificação funcional das operações são elaboradas de modo totalmente *top-down*.

As especificações funcionais de cada operação podem ser definidas na forma de casos de uso de sistema do tipo caixa-branca (COCKBURN, 2000), que descrevem um determinado processamento expondo detalhes de sua implementação. Podem-se usar diagramas de atividades para detalhar a lógica do serviço e diagramas de sequência ou de colaboração para representar as interações com outros serviços ou aplicações legadas. Por utilizarem casos de uso e diagramas da UML, os serviços especificados podem ser diretamente implementados utilizando técnicas de orientação a objetos e desenvolvimento baseado em componentes, atendendo ao requisito **R10**.

As políticas de serviço especificam os requisitos não-funcionais relacionados com o serviço, como segurança, comportamento transacional e registro de auditoria, e requisitos de nível de serviço, por meio de contratos de SLA. SLAs definirão os requisitos de desempenho, confiabilidade e disponibilidade acordados entre provedor e consumidor de serviço. A documentação de especificações de requisitos não-funcionais visa suprir o requisito **R4** de um método orientado a serviços.

Nesta atividade, o Projetista de Serviços deve determinar também a qual camada pertence cada um dos serviços, dividindo-os entre a camada de serviços de negócio e a camada de serviços de aplicação de acordo com seu nível de granularidade. Separam-se os serviços com nível alto e baixo de granularidade.

Os serviços de negócio são aqueles com nível mais alto de granularidade. Geralmente, suas funções representam entidades do modelo de negócio ou tarefas que podem ser mapeadas diretamente para um processo de negócio. Serviços de negócio podem ser compostos por serviços de aplicação.

Os serviços de aplicação possuem nível mais baixo de granularidade e representam funções de infra-estrutura ou funções de aplicações legadas. Devem ser os mais genéricos possíveis, não contendo lógica específica de um processo ou domínio de negócio, para que possam ser utilizados por diversos serviços de negócio e de orquestração.

A classificação em camadas atende ao requisito **R11**.

Papel: Projetista de Serviços

Artefato utilizado: Modelo de serviços candidatos

Artefato gerado: Especificação de serviço

5.4.2 Verificação dos Princípios

Nesta atividade, uma série de verificações é realizada sobre as especificações de serviços com o intuito de avaliar sua conformidade com os princípios de orientação a serviços e atender ao requisito **R8**.

Para garantir a reusabilidade de um serviço, deve-se verificar se não é necessário adicionar operações ou alterar as já existentes ou especificadas para que o serviço possa vir a ser útil em outro contexto que não o do processo de negócio sendo desenvolvido (ERL, 2005). Mesmo que nem todas as operações não venham a ser implementadas em um primeiro momento, é recomendável que elas já estejam especificadas para minimizar a necessidade de mudanças posteriores na interface do serviço.

Deve-se verificar se o nível de granularidade do serviço (a quantidade de lógica encapsulada) está de acordo com a camada a que pertence e com nível de reusabilidade esperado. Serviços de aplicação devem possuir nível baixo de granularidade, ou seja, devem ser genéricos para que possam ter maior reusabilidade. Serviços de negócio podem ter nível mais alto de granularidade, pois têm necessidade menor de reusabilidade e devem poder ser utilizados diretamente pelos serviços de orquestração.

Para se obter autonomia em um serviço, deve-se garantir que a lógica encapsulada é auto-contida, isto é, não possui dependências com relação a outros serviços ou processos.

Deve-se verificar se o contrato (interface) do serviço está abstraindo detalhes de implementação, ocultando-os dos potenciais consumidores. O serviço deve funcionar como uma “caixa-preta”, expondo em seu contrato apenas as informações essenciais para que possa ser utilizado.

Para se garantir o baixo acoplamento dos serviços, deve-se verificar se o contrato do serviço não está limitado por características específicas de um processo de negócio ou de sua lógica e tecnologia de implementação. O contrato deve ser o mais desacoplado possível de elementos externos.

E para se garantir que os serviços sejam independentes de estado, deve-se procurar minimizar a quantidade de informação de estado armazenada pelo serviço, armazenando-a nas mensagens trocadas ou na lógica de orquestração do processo de negócio.

Papel: Arquiteto Corporativo

Artefato utilizado: Especificação de Serviço

Artefato gerado: Não aplicável

5.4.3 Revisão dos Serviços

A verificação da aderência aos princípios de orientação a serviços pode demandar alterações nos serviços especificados. Caso isto seja necessário, executam-se novamente alguns passos da atividade de Especificação de Serviços para que os contratos dos serviços sejam reprojatados conforme necessário para se adequar aos princípios de orientação a serviços.

Pode ser necessário rever também a alocação dos serviços nas camadas de negócio e de aplicação, caso tenha havido alteração no escopo funcional ou na granularidade dos serviços.

Papel: Projetista de Serviços

Artefato utilizado: Especificações de Serviço

Artefato gerado: Não aplicável

5.4.4 Orquestração de Serviços

Uma vez especificados os serviços e suas interfaces WSDL, é realizada a orquestração dos serviços. Se o Modelo de Processo *to-be* estiver representado na notação BPMN, ele pode ser convertido em um modelo na linguagem WS-BPEL. Caso contrário, a lógica de processo pode ser descrita manualmente em WS-BPEL ou em outra linguagem. Tal lógica deve ser especificada de modo que as chamadas dos serviços sejam devidamente orquestradas com as interfaces de serviços definidas e revisadas.

O objetivo desta atividade é definir a lógica de orquestração necessária para executar o processo, como manipulação de variáveis de processo, gerenciamento de informação de estado, tratamento de exceções e compensação.

O processo orquestrado deverá ser encapsulado por um serviço da camada de orquestração composto por serviços das camadas de negócio e aplicação. Por isto,

deverá ser descrito em sua própria Especificação de Serviço, que conterà a lógica de orquestração definida.

Esta atividade atende aos requisitos **R2** ao viabilizar a abordagem de BPM e **R9** ao utilizar o padrão WS-BPEL.

Papel: Projetista de Serviços

Artefatos utilizados: Modelo de Processo *to-be*, Especificações de Serviço

Artefato gerado: Especificação de Serviço de orquestração

5.5 Atividades de Construção

Após a Análise e o Projeto, o ciclo de vida prossegue com atividades de Construção e posteriormente de Implantação. Na fase de Construção, são desempenhadas atividades com o intuito de se obter os serviços executáveis devidamente integrados e testados, prontos para serem implantados.

5.5.1 Implementação e Teste

As fases de Implementação e Teste são executadas por Desenvolvedores, que receberão as especificações dos serviços e implementarão os *Web Services* e componentes necessários. A partir destas fases, pode ser seguido o ciclo tradicional de desenvolvimento orientado a objetos e/ou baseado em componentes.

Papéis: Desenvolvedor, Testador

Artefatos utilizados: Especificações de Serviço

Artefato gerado: Componentes de serviço executáveis

5.6 Papéis

O **Analista de Processo** é o responsável pela elaboração dos modelos de processo de negócio que serão implementados ou automatizados em um projeto. Este papel

deve possuir conhecimentos sobre levantamento de requisitos, modelagem de processos ou elaboração de casos de uso de negócio.

O **Analista de Serviços** é o papel que realiza a transição dos artefatos gerados pela modelagem de negócio para uma solução orientada a serviços. Este papel exige conhecimentos nos princípios de orientação a serviços e análise de sistemas.

O **Projetista de Serviços** tem como função principal criar as especificações dos serviços a serem implementados, projetando suas interfaces e comportamento funcional. Para tal, são necessários conhecimentos em orientação a serviços, integração de aplicações e padrões de projeto.

O **Arquiteto Corporativo** é o responsável pelo portfólio de serviços da organização, definindo uma arquitetura de referência e padrões a serem seguidos nos projetos. Deve possuir conhecimentos em orientação a serviços, governança, integração e arquitetura de aplicações e padrões de projeto.

O **Gerente de Projeto** é o responsável por conduzir o projeto, controlando aspectos como escopo, prazo, custo e riscos.

Os papéis de **Desenvolvedor** e **Testador** são os responsáveis pela construção dos serviços especificados. O perfil necessário para estes papéis é o mesmo de um método de desenvolvimento tradicional.

5.7 Artefatos

O **Modelo de Processo *as-is*** e o **Modelo de Processo *to-be*** correspondem, respectivamente, à representação do estado atual do negócio e dos requisitos da solução em uma notação como o BPMN. No método, é possível utilizar um modelo de casos de uso ao invés dos modelos de processo, ou mesmo as duas formas associadas.

O **Modelo de Serviços** é utilizado para documentar o processo de análise orientada a serviços, que dá origem a um conjunto de serviços identificados a partir de um modelo de negócio, na forma de processo ou caso de uso. Ele é preenchido ao longo das seguintes atividades:

- Identificação de Serviços Candidatos

- Análise de *Gap*
- Análise de Realização

A Figura 5.2 apresenta os principais aspectos contidos no modelo de serviços, abstratamente, e o relacionamento entre eles e as atividades que os definem.

Figura 5.2: Modelo de Serviços

A **Especificação de Serviço** fornece a especificação estrutural e comportamental de um serviço. Esta especificação estabelece um contrato entre provedor e consumidor do serviço, além de servir como documentação de referência para o desenvolvedor responsável por implementar o serviço. Contém a descrição da interface do serviço, a definição das mensagens de entrada e saída, a especificação do comportamento de cada operação e os requisitos não-funcionais envolvidos, além de indicar a camada a que pertence o serviço.

5.8 Análise do Método

Trata-se de um método desenvolvido para ser de uso livre, satisfazendo ao requisito **R12** descrito anteriormente.

O método foi elaborado de modo a conter atividades e orientações que atendessem a todos os requisitos de análise e projeto orientados a serviços levantados anteriormente.

Além de atender aos requisitos, a definição do MAPOS procurou unificar as boas práticas identificadas nos métodos existentes.

Do método de Papazoglou e Van Den Heuvel, o MAPOS tomou como referência as atividades de Análise de *Gap* e Análise de Realização que tratavam do reuso de recursos existentes.

Do RUP *plugin for SOMA*, foram usadas as práticas de identificação de serviços a partir de casos de uso e o artefato Modelo de Serviços, além da definição de alguns papéis que serviram como referência para definir os papéis do MAPOS.

O MAPOS baseou a atividade de Identificação de Serviços Candidatos nas técnicas de identificação do método de Erl. Ainda do método de Erl, foi adotada a prática de verificação dos princípios de orientação a serviços.

A inclusão da atividade Orquestração de Serviços na fase de Projeto tomou como referência as atividades Especificação de Processos de Papazoglou e Van Den Heuvel e Projeto de Processo de Negócio Orientado a Serviços de Erl.

Por fim, a classificação de serviços em camadas (na atividade Especificação de Serviços) foi baseada na análise de granularidade de Marks e Bell e nas camadas definidas por Erl.

Além destas boas práticas, o MAPOS introduz o conceito de cenários de reuso, que permite orienta a identificação de oportunidades de reuso de recursos existentes a tomada de decisões de realização.

O método proposto contém recomendações e práticas que o tornam compatível com BPM e a utilização da tecnologia *Web Services*, mas não limitado a estas abordagens. Tais práticas e recomendações não são obrigatórias de modo a permitir que o método seja aplicado a cenários sem BPM e com utilização de outras tecnologias de implementação.

5.9 Considerações Finais

Neste Capítulo 5, foi apresentada a proposta de método elaborada por este trabalho de pesquisa, o MAPOS. O MAPOS foi elaborado de modo a atender aos requisitos

levantados no Capítulo 4 e tomando como referência os métodos estudados no Capítulo 3.

O método proposto emprega técnicas e boas práticas utilizadas em cada um dos métodos estudados. Além disso, foi definido de modo a ser compatível com a abordagem BPM e com a tecnologia *Web Services*, como parte dos métodos analisados. No entanto, sua aplicabilidade não deve ser restrita a projetos com essas características.

Torna-se necessária então a realização de estudos de caso aplicando o método no intuito verificar sua aplicabilidade e avaliar a eficiência das boas práticas propostas.

6 ESTUDOS DE CASO

Para avaliar o método proposto e verificar o atendimento aos requisitos levantados, foram realizados estudos de caso aplicando o método em uma empresa de consultoria denominada IVS, especializada na implantação de arquitetura SOA com abordagem BPM utilizando ferramentas de software WebSphere da IBM.

O método proposto foi aplicado em projetos envolvendo SOA e opcionalmente a abordagem BPM, no qual um modelo de negócio é elaborado para então ser automatizado na forma de uma solução orientada a serviços. O MAPOS foi aplicado para, a partir do modelo de negócio, realizar a análise e o projeto dos serviços que dão apoio aos processos de negócio automatizados.

Tais estudos de caso levaram em conta os requisitos de negócio reais da organização, assim como a infra-estrutura existente de serviços e aplicações legadas. Foi avaliado como o método proposto trata esses aspectos de um projeto orientado a serviços.

Por meio destes estudos de caso foi possível realizar uma análise crítica do método proposto e avaliar sua aderência aos requisitos de análise e projeto orientados a serviços.

6.1 Caso Banco BES

6.1.1 Contexto do Estudo

O primeiro estudo de caso realizado consiste no projeto executado pela empresa IVS no Banco BES.

O Banco BES, procurando suprir as necessidades de informação e o aumento da demanda de seus serviços e produtos, busca modernizar sua área de TI adotando novas práticas e tecnologias. O banco recentemente adotou a prática da modelagem de processos devido a exigências internas e externas de conformidade e à necessidade de documentação das regras de negócio.

Há, portanto, um direcionamento para a cultura de gerenciamento de processos de negócios (BPM), porém ainda não generalizado. Vale ressaltar que é focado mais o aspecto de modelagem, mesmo porque ainda não há infra-estrutura para suportar outros aspectos do BPM, como simulação de processos, automação de fluxo de trabalho e monitoração de processos.

Neste cenário, surgiu no Banco BES um projeto para implantar uma infra-estrutura de BPM apoiada por uma arquitetura SOA, automatizando processos através da composição de serviços integrados em um barramento (ESB). Com a adoção de SOA e BPM, o Banco BES pretende obter maior reuso e flexibilidade, reduzindo custos e ganhando mais velocidade nas idas/respostas ao mercado.

Para fornecer a solução de SOA e BPM, o Banco BES selecionou as ferramentas da IBM e algumas empresas de consultoria para executar o projeto de desenvolvimento e implantação. As ferramentas selecionadas abrangiam desde o ambiente de execução de serviços e processos de negócio até as ferramentas de *Computer-Aided Software Engineering* (CASE) para o desenvolvimento e integração de aplicações, passando por ferramentas de gerência de requisitos e gerência de configuração.

Os objetivos deste projeto são:

- Implantação das ferramentas.
- Definição de um processo de desenvolvimento de software de acordo com a nova arquitetura.
- Desenvolvimento um projeto piloto.

Tal projeto foi executado pela IVS em conjunto com outras empresas de consultoria, sendo cada consultoria responsável por uma parte do escopo do projeto. A parte de responsabilidade da IVS foi a implantação das ferramentas de SOA e BPM, a definição do processo de desenvolvimento de software (em conjunto com a empresa QSP).

As empresas IVS e QSP foram incumbidas de auxiliar o banco a definir o novo Processo de Desenvolvimento de Software do Banco BES, baseado no arcabouço do RUP. A QSP contribuiu com o conhecimento sobre o arcabouço e as disciplinas do RUP, enquanto a IVS contribuiu com o conhecimento na abordagem BPM com arquitetura SOA, definindo as atividades de Modelagem de Negócio (processos) e

de Análise e Projeto orientados a serviços, contexto em que foi aplicado o MAPOS. Ao Banco BES foi incumbida a definição da disciplina de Gerenciamento de Projetos. Foi formada uma equipe com profissionais da IVS, da QSP e do Banco BES com o intuito de combinar os materiais elaborados por cada uma em um processo de desenvolvimento integrado para guiar o desenvolvimento de software dentro do Banco BES seguindo a nova arquitetura corporativa, orientada a serviços.

Com o propósito de validar o processo elaborado, as ferramentas implantadas e a arquitetura proposta, o Banco BES organizou um projeto piloto. Neste projeto piloto, profissionais do Banco BES seriam responsáveis por executar as tarefas do processo, seguindo a documentação publicada, apoiados pelos consultores da IVS e da QSP. Neste contexto, o autor deste trabalho foi responsável por acompanhar e orientar a execução das atividades de análise e projeto e validar os artefatos produzidos, de modo a documentar o estudo de caso. As atividades correspondentes ao MAPOS foram realizadas no período de outubro a dezembro de 2008.

6.1.1.1 Empresas Envolvidas

A IVS é uma empresa, especializada na prestação de serviços de consultoria em arquitetura SOA, abordagem BPM, integração de sistemas e desenvolvimento de software. Nas atividades descritas do projeto, foram alocados os seguintes recursos humanos da IVS:

- Um especialista em SOA (o autor deste trabalho), que atuou na definição do processo de desenvolvimento e orientou a execução do MAPOS.
- Um analista de processos, que auxiliou os analistas do Banco BES durante as atividades de modelagem de processo.
- Um especialista nas ferramentas da plataforma de BPM, que apoiou a equipe do Banco BES no uso das ferramentas.
- Um gerente de projetos.

A QSP é uma empresa que presta serviços de fábrica de software, fábrica de testes, consultoria em Engenharia de Software e implantação de metodologia e ferramentas de desenvolvimento de software. Participaram as seguintes pessoas da QSP:

- Um especialista em metodologias de desenvolvimento de software e no RUP, que atuou na definição do processo de desenvolvimento.
- Três especialistas nas ferramentas CASE, que implantaram as ferramentas e apoiaram a equipe do Banco BES no seu uso.
- Um gerente de projetos.

Já o Banco BES contou com os seguintes recursos:

- Dois analistas de processos, responsáveis pela modelagem.
- Dois arquitetos, que atuaram como Analistas de Serviços, Projetistas de Serviços, Arquitetos Corporativos e realizaram atividades de implementação.
- Dois desenvolvedores, que também realizaram atividades de implementação.

6.1.1.2 Ferramentas Utilizadas

O processo de desenvolvimento do Banco BES foi documentado com a ferramenta *Rational Method Composer* da empresa IBM. A ferramenta permitiu também a publicação do material produzido para a organização.

Para a representação dos artefatos produzidos nas atividades de análise e projeto, foi utilizada a ferramenta de modelagem UML *Rational Software Architect* da IBM.

O IBM *WebSphere Business Modeler* foi a ferramenta usada para modelagem, simulação e publicação de modelos de processo de negócio. A ferramenta suporta o desenho dos diagramas de processos seguindo uma notação própria ou a notação BPMN e permite exportar o modelo de processo desenvolvido na linguagem WS-BPEL.

Descrições dos processos e dos requisitos foram documentadas na ferramenta de gerenciamento de requisitos da IBM, o *Rational RequisitePro*.

Para implementar a orquestração do processo em WS-BPEL, foi utilizado o IBM *WebSphere Integration Developer*. Com o uso desta ferramenta, foi possível também integrar o processo aos serviços e componentes desenvolvidos.

Para os serviços desenvolvidos em Java, foi utilizado o *Rational Application Developer* da IBM, enquanto os serviços em .NET foram implementados no *Microsoft Visual Studio*.

O ambiente de execução era composto por:

- IBM *WebSphere Process Server*, para execução dos processos de negócio em WS-BPEL
- IBM *WebSphere Message Broker*, como barramento de integração (ESB)
- JBoss *Application Server*, para execução dos serviços implementados em Java
- Microsoft *Internet Information Services*, para execução dos serviços implementados em .NET

6.1.2 Aplicação do Método

Esta seção relata a execução de cada atividade do método MAPOS, descrevendo os produtos gerados.

6.1.2.1 Modelagem de Processo *as-is* e *to-be*

Esta tarefa teve como objetivo elaborar os modelos de processo de negócio, isto é, representar na forma de uma notação o processo de negócio na sua forma atual (*as-is*) e o processo futuro (*to-be*), contendo as mudanças propostas.

No projeto piloto, o processo de negócio escolhido para modelagem e automação foi o processo “Inclusão de cliente pessoa física”. Trata-se de um processo de pré-cadastro de pessoas físicas, pelo qual potenciais novos clientes devem passar para abrir uma conta, obter um financiamento ou solicitar um cartão de crédito.

O modelo de processo *as-is* foi levantado por meio de entrevistas com os usuários de negócio e de aplicação de questionários. Um consultor da IVS atuou junto a profissionais do Banco BES, todos exercendo o papel de Analistas de Processo. O diagrama do processo foi produzido utilizando-se o *WebSphere Business Modeler*.

As necessidades dos usuários de negócio foram registradas na forma de requisitos, utilizando-se o *Rational RequisitePro*.

Após o levantamento do processo atual, foram detectados pontos de melhoria e oportunidades de automação em algumas atividades. Desta maneira, foi proposto um modelo de processo *to-be*, contendo as melhorias e tarefas automatizadas que visaram o atendimento às necessidades de negócio levantadas. Para se chegar à versão final do modelo de processo *to-be*, foi novamente utilizado o *WebSphere Business Modeler* para a criação do diagrama do processo e execução de simulações. O diagrama do processo *to-be* pode ser visualizado na Figura 6.1.

Figura 6.1: Diagrama do processo "Incluir cliente pessoa física"

No processo desenvolvido, foi utilizado somente o formato de processos para a modelagem de negócio, não sendo utilizada a abordagem de casos de uso de

negócio. Sendo assim, o modelo de negócio era composto por: diagrama do processo de negócio, documento de Especificação de Tarefas e descrição textual de regras de negócio.

O documento de Especificação de Tarefas tem como propósito complementar o diagrama do processo elaborado na ferramenta. Tal documento foi criado contendo a descrição detalhada de cada tarefa do processo de “Inclusão de cliente pessoa física”. A descrição de cada tarefa continha as seguintes informações:

- Objetivo da tarefa.
- Tipo da tarefa: tarefa totalmente automatizada ou tarefa que envolve interação com o usuário.
- Fluxo de trabalho da tarefa: Para descrever o fluxo de trabalho executado em cada tarefa do processo, foi utilizada uma estrutura de passos, semelhante à descrição de um caso de uso.
- Regras de negócio envolvidas na execução da tarefa.
- Fluxos de trabalho alternativos e de exceção.

O documento de Especificação de Tarefas, bem como os documentos das regras de negócio, foi armazenado no *Rational RequisitePro*, de modo que cada tarefa e regra de negócio pudessem ser referenciadas como requisitos. A Tabela 6.1 apresenta um exemplo de especificação de tarefa criada durante esta modelagem.

Tabela 6.1: Especificação da tarefa “Reagendar Visita”

Tipo de informação	Descrição
Objetivo	O objetivo dessa tarefa é registrar o reagendamento da visita do cliente à agência. Esse reagendamento é necessário, a partir do momento no qual é identificada alguma pendência, o que impede a continuidade do processo.
Tipo de tarefa	Tarefa com interação humana
Fluxo de trabalho	<ol style="list-style-type: none"> 1. A tarefa é iniciada quando é identificado que há pendências que impedem a continuidade do processo 2. O ator informa dados de contato e data na qual o cliente pode comparecer à agência [FE001] [FE002] [FA001] 3. O sistema apresenta a seguinte mensagem “Formulário enviado com sucesso.” 4. A tarefa é finalizada
Fluxo de exceção FE001	<ol style="list-style-type: none"> 1. Caso a data informada não esteja em conformidade com a RNV29 2. Sistema retorna ao passo 2 do fluxo de trabalho
Fluxo de exceção FE002	<ol style="list-style-type: none"> 1. Caso os dados de e-mail não sejam válidos 2. Caso os dados de telefone não sejam válidos 3. Sistema retorna ao passo 2 do fluxo de trabalho

continua

Tipo de informação	Descrição
Fluxo alternativo FA001	<ol style="list-style-type: none"> 1. Caso o ator não informe o reagendamento da visita no prazo de um dia após a data previamente agendada 2. Sistema prorroga a data do reagendamento automaticamente para o próximo dia útil
Fluxo alternativo FA002	<ol style="list-style-type: none"> 1. A qualquer momento, enquanto o processo está parado nessa tarefa aguardando o reagendamento, o gerente de agência pode informar que deseja encerrar o processo.

6.1.2.2 Identificação de Serviços Candidatos

Após a elaboração do modelo de processo *to-be*, composto pelo diagrama do processo mais as descrições das tarefas, foi iniciada a identificação de serviços candidatos. Esta atividade recebe como entrada o modelo de processo *to-be* e tem como objetivo definir um conjunto inicial de serviços, identificados a partir do processo.

Esta atividade foi executada por um profissional do Banco BES designado como Analista de Serviços, suportado por um consultor da IVS e um da QSP, sendo o consultor da IVS responsável por auxiliar no processo de identificação e o da QSP, responsável por auxiliar no uso da ferramenta *Rational Software Architect*. O *Rational Software Architect* foi utilizado para documentar os serviços candidatos.

As operações candidatas foram identificadas a partir da análise do documento de Especificação de Tarefas, seguindo a seqüência das tarefas determinada pelo diagrama do processo. No texto de descrição de cada tarefa do processo, era possível separar os passos em passos executados pelo ator e passos automatizados, ou seja, executados pelo sistema. Em alguns casos, a tarefa inteira foi considerada uma operação candidata, enquanto em outros, apenas alguns passos de uma tarefa foram considerados operações candidatas. As operações identificadas nas tarefas do processo são listadas na Tabela 6.2.

Tabela 6.2: Operações candidatas do processo "Inclusão de cliente pessoa física"

Tarefa	Operação Candidata
Consultar CPF	<ul style="list-style-type: none"> • Validar número de CPF • Validar existência do cliente
Preencher Dados	<ul style="list-style-type: none"> • Consultar agência • Validar dados iniciais do cliente • Validar existência do cliente

Tarefa	Operação Candidata
Preencher Dados Internet	<ul style="list-style-type: none"> Listar agências Consultar agência Validar dados iniciais do cliente Validar existência do cliente
Conferir Documentos	<ul style="list-style-type: none"> Pesquisar e-mail por grupo Enviar e-mail
Consultar Situação do CPF	<ul style="list-style-type: none"> Validar situação do CPF
Gerar Pendência	<ul style="list-style-type: none"> Consultar documentos obrigatórios
Reagendar Visita	<ul style="list-style-type: none"> Obter próximo dia útil
Complementar Dados	<ul style="list-style-type: none"> Listar ocupações Listar tipos de cliente Obter endereço a partir do CEP Validar dados do cliente
Registrar Dados no SIC	<ul style="list-style-type: none"> Gravar dados do cliente
Enviar Notificação Cliente Potencial	<ul style="list-style-type: none"> Enviar e-mail

Operações registradas usando a ferramenta *Rational Software Architect*, na forma de uma classe “OperacoesCandidatas”, que pode ser vista na Figura 6.2.

Figura 6.2: Operações candidatas identificadas

Após a identificação, as operações foram agrupadas em serviços candidatos, que por sua vez foram representados como interfaces em um diagrama de classes.

No entanto, para o Analista de Serviços do Banco BES, houve dúvida no momento de definir o agrupamento das operações candidatas em serviços candidatos. Segundo ele, na descrição do método poderia haver diretrizes mais claras para o agrupamento de operações em serviços.

Decidiu-se criar um serviço com as operações utilitárias chamado “SvcCorporativo”. Já as operações relacionadas a cliente foram separadas em um serviço

“SvcClientePF”, com operações de manipulação de informações de clientes do tipo pessoa física, e um serviço “InfraCliente”, com operações utilitárias comuns a todos os tipos de cliente.

A Figura 6.3 mostra as interfaces modeladas para representar os serviços candidatos identificados.

Figura 6.3: Serviços candidatos identificados

Como as operações foram registradas como métodos de uma classe, percebeu-se a necessidade de se definirem neste momento os parâmetros das operações, permitindo a identificação de algumas mensagens. Por exemplo, para a operação “consultarCliente” do serviço “SvcCliente”, definiu-se que um parâmetro de entrada da operação seria o número de CPF do cliente e o parâmetro de saída (ou de retorno) seria uma mensagem do tipo “Cliente”, contendo em seus atributos informações sobre o cliente consultado.

Desta maneira, decidiu-se registrar as mensagens utilizadas nas operações de serviços na forma de classes com o estereótipo “*message*”. As classes de mensagens documentadas foram definidas a partir dos objetos de negócio modelados juntamente com o fluxo de trabalho do processo. A Figura 6.4 mostra alguns exemplos de classes de mensagens definidas neste momento.

Figura 6.4: Exemplos de classes de mensagens

Após a definição dos serviços candidatos, foram elaboradas as descrições de cada operação e serviço, sendo que os textos descritivos foram registrados utilizando-se a própria ferramenta de modelagem UML, na seção de documentação das classes candidatas.

6.1.2.3 Análise de Gap

Após a identificação dos serviços candidatos, o método prosseguiu com a Análise de Gap, quando se buscaram nas aplicações existentes as funções necessárias para os serviços candidatos, chegando a um cenário de reuso para cada operação candidata. A atividade foi conduzida pelo Arquiteto Corporativo do Banco BES, que por sua vez foi auxiliado pelo consultor da IVS.

Para analisar os cenários de reuso, o método orienta a consultar um repositório de serviços e recursos reusáveis ou fontes de documentação das aplicações existentes. Como o repositório de serviços ainda não estava implantado durante a execução desta atividade, foi necessário recorrer à documentação do legado, que era escassa e se mostrou incompleta para a análise de gap. Deste modo, foi necessário contatar

os analistas e desenvolvedores responsáveis pelas aplicações legadas para obter as informações necessárias para análise.

Como resultado desta análise, foram obtidos os seguintes apresentados na Tabela 6.3.

Tabela 6.3: Cenários de reuso para operações candidatas

Serviço Candidato	Operação Candidata	Cenário de Reuso
SvcClientePF	gravarDadosCliente	Operação realizada por aplicação legada (componente GerenciadorPessoa do Sistema de Informações de Cliente implementado em Java)
	validarDadosCliente	Operação não existente
	validarDadosIniciais	Operação não existente
	validarExistenciaCliente	Operação realizada por aplicação legada (componente GerenciadorPessoa do Sistema de Informações de Cliente implementado em Java)
	validarSituacaoCPF	Operação não existente
SvcCorporativo	consultarAgencia	Operação parcialmente realizada por aplicação legada (dados existentes no sistema Automação)
	enviarEmail	Operação realizada por aplicação legada (classe Util da biblioteca Corporativo implementada em .NET)
	listarAgencias	Operação parcialmente realizada por aplicação legada (dados existentes no sistema Automação)
	listarOcupacoes	Operação não existente
	obterEnderecoCEP	Operação realizada por aplicação legada (componente GerenciadorEndereco do Sistema de Informações de Cliente implementado em Java)
	obterProximoDiaUtil	Operação realizada por aplicação legada (classe RotDias da biblioteca Corporativo implementada em .NET)
	pesquisarEmailGrupo	Operação não existente
	validarCPF	Operação realizar por aplicação legada (classe Util da biblioteca Corporativo implementada em .NET)
InfraCliente	consultarDocumentosObrigatorios	Operação não existente
	listarTiposCliente	Operação não existente

A lógica necessária para as operações “gravarDadosCliente” e “validarExistenciaCliente” do serviço “SvcClientePF” já eram realizadas por uma aplicação legada, neste caso um componente Java “GerenciadorPessoa”

pertencente ao Sistema de Informações de Clientes (SIC). O componente em questão era um *Enterprise Java Bean* do tipo *Session*.

As classes utilitárias “Util” e “RotDias” da biblioteca corporativo, implementada em .NET, continham as funções necessárias para as operações “enviarEmail”, “validarCPF” e “obterProximoDiaUtil” do serviço “SvcCorporativo”.

Já a funcionalidade da operação “obterEnderecoCEP” do serviço “SvcCorporativo” foi encontrada no componente Java “GerenciadorEndereco” do SIC.

Para as operações “consultarAgencia” e “listarAgencias” do serviço “SvcCorporativo”, foi identificado que os dados de agência necessários já existiam no sistema Automação, mas não havia no sistema a lógica para buscar as informações necessárias.

Para retratar de modo mais claro este tipo de caso, notou-se que o MAPOS, ao analisar o reuso de recursos existentes, poderia verificar separadamente a existência da lógica e dos dados necessários para cada operação candidata. Para esta operação, poderia ser considerado que os dados existissem, mas a lógica não.

Os cenários de reuso analisados foram documentados na mesma ferramenta de modelagem UML, na seção de documentação das classes candidatas, juntamente com as descrições das operações.

6.1.2.4 Análise de Realização

Na Análise de Realização, cada operação candidata é avaliada quanto ao seu cenário de reuso identificado anteriormente. Nesta etapa, deve ser decidido como cada operação de serviço será realizada.

As possibilidades de realização dos serviços candidatos foram discutidas pelo Analista de Serviços responsável pelas análises, o consultor da IVS e os analistas responsáveis pelas aplicações. As decisões de realização resultantes da análise foram registradas no diagrama de classes de serviços. Para cada interface de serviço, foi indicada na forma de uma classe qual seria sua realização.

Foram obtidas as seguintes decisões de realização.

Serviço “SvcCliente”

Como o “GerenciadorEndereco” pertencia ao mesmo sistema que o “GerenciadorPessoa”, decidiu-se mover a operação “obterEnderecoCEP” do serviço “SvcCorporativo” para o serviço “SvcClientePF”. Deste modo, seria possível manter as implementações dos serviços em plataformas separadas e a coesão seria mantida, uma vez que as informações de clientes possuíam relacionamento com informações de endereço.

Para realizar este serviço, seria então criado o novo *Web Service* “SvcCliente” em Java, que seria responsável por acessar os *Enterprise Java Beans* “GerenciadorPessoa” e “GerenciadorEndereco” do SIC e implementar as funções das operações não existentes. A plataforma Java permite a criação deste *Web Service* sem a necessidade de adaptadores.

A realização do serviço “SvcClientePF” pode ser visualizada na Figura 6.5.

Figura 6.5: Realização do serviço "SvcClientePF"

Serviço “SvcCorporativo”

Operações implementadas por um novo *Web Service* “SvcInfraestrutura” na plataforma .NET responsável por encapsular as classes “Util” e “RotDias” e

acessar os dados do sistema Automação. Como o arcabouço .NET também possui suporte nativo a *Web Services*, não seria necessário utilizar nenhuma mediação ou adaptador.

A realização do serviço “SvcCorporativo” pode ser visualizada na Figura 6.6.

Figura 6.6: Realização do serviço "SvcCorporativo"

Serviço “InfraCliente”

Como as operações deste serviço não existiam nas aplicações legadas, seria implementado um novo *Web Service* em Java que acessaria uma nova base de dados e executaria a consulta de dados conforme lógica necessária.

A realização do serviço “SvcCorporativo” pode ser visualizada na Figura 6.7.

Figura 6.7: Realização do serviço "InfraCliente"

6.1.2.5 Especificação de Serviços

A atividade de especificação de serviços tem como objetivo detalhar as classes de mensagens e as interfaces de serviços, inclusive com a descrição técnica de cada operação, para chegar aos artefatos que comporão o contrato do serviço.

Durante esta atividade, um dos arquitetos do Banco BES atuou como Projetista de Serviços realizando necessários ajustes e informações adicionais nas classes, tais como:

- Especificação do tipo de dado de cada atributo das classes de mensagens.
- Especificação do tipo de dados de cada parâmetro de operação.
- Seleção dos parâmetros das operações com base na lógica a ser implementada.
- Especificação de requisitos não-funcionais, como segurança (autenticação e autorização).

Uma vez especificadas as classes, foram gerados esquemas XSD e interfaces WSDL, utilizando a ferramenta *Rational Software Architect*. A ferramenta possui uma função de transformação de classes em WSDL e XSD, sendo que as classes ou interfaces com o estereótipo “*serviceSpecification*” são convertidas em interfaces WSDL e classes com estereótipo “*message*” são convertidas em esquemas XSD.

O serviço “SvcClientePF” foi classificado como pertencente à camada de serviços de negócio por possuir maior granularidade e representar uma entidade de negócio, enquanto “SvcCorporativo” e “InfraCliente” foram classificados como serviços de aplicação por proverem funções técnicas de menor granularidade.

6.1.2.6 Verificação dos Princípios

O objetivo desta atividade é verificar se cada especificação de serviço criada atende aos princípios de orientação a serviços e se estão adequados a integrarem o portfólio de serviços da organização.

A verificação dos princípios foi realizada em uma reunião envolvendo o Analista de Serviços responsável por identificar e especificar os serviços, o segundo arquiteto do Banco BES atuando como Arquiteto Corporativo, o consultor da IVS e o gerente do projeto do Banco BES.

A presença do gerente foi importante para obter o aval gerencial sobre as decisões de novas implementações, que poderiam demandar um esforço além do planejado para o projeto.

Abaixo, são descritos os resultados obtidos da verificação de cada um dos serviços projetados.

Serviço “SvcCliente”

A operação “validarExistenciaCliente” foi considerada como possuindo alto potencial de reuso pelos Arquitetos, que citaram a existência de diversos processos de negócio que poderiam vir a utilizá-la. Entretanto, para tornar o serviço mais reusável, foi recomendada a adição de outras operações à interface, já existentes no componente “GerenciadorPessoa” encapsulado pelo serviço, como “removerPessoa”, “obterPessoaPorNome”, “consultarRiscoCliente”, “obterSituacaoCredito” e “alterarSituacaoCredito”.

Decidiu-se que as operações seriam especificadas na interface do serviço, mas que não seriam implementadas imediatamente no projeto piloto. As atividades de implementação do projeto piloto teria como foco somente as operações necessárias para suportar o processo de negócio “Inclusão de Cliente Pessoa Física”.

O nível de granularidade do serviço foi considerado adequado. Já a autonomia do serviço foi considerada um ponto de atenção, uma vez que os componentes encapsulados pertencem a um sistema externo e possuem relacionamentos com outros componentes.

Serviço “SvcCorporativo”

Este serviço foi projetado desde o início de forma a ser altamente reusável, com operações utilitárias de baixa granularidade, aplicáveis a diversos contextos de negócio. Portanto, eventuais ajustes poderiam incluir apenas a adição de novas operações, já presentes nas classes encapsuladas “Util” e “RotDias”.

Devido ao fato de os dados acessados por este serviço pertencerem a uma aplicação externa (Automação), a autonomia do serviço poderia estar comprometida. A recomendação foi analisar o relacionamento destes dados com outros componentes do sistema para verificar possíveis impactos com relação à autonomia.

Serviço “SvcAgencia”

O serviço foi considerado com nível de granularidade adequado para reuso. A avaliação do serviço constatou sua autonomia, pois toda sua lógica estava contida em um novo componente de serviço, sem dependência com outros sistemas externos.

Além da validação dos princípios, foi verificada a aderência dos serviços especificados à arquitetura de referência definida para o Banco BES. A arquitetura de referência determina diversos padrões a serem aplicados às novas aplicações desenvolvidas no banco, incluindo padrões de nomenclatura e de projeto.

6.1.2.7 Revisão dos Serviços

Na revisão, quaisquer não conformidades detectadas durante a verificação dos princípios devem ser corrigidas nos artefatos que compõem a especificação dos serviços.

A revisão consistiu apenas na realização de ajustes nas interfaces de serviços especificadas, adicionando-se as operações recomendadas durante a Verificação dos Princípios. Esta atividade foi executada pelo Projetista de Serviços do Banco BES.

6.1.2.8 Orquestração de Serviços

Durante a orquestração de serviços, o modelo de processo *to-be* elaborado no início do projeto é convertido em um modelo de implementação, que deve ser trabalhado para tratar todas as regras de negócio modeladas e adequar-se aos serviços especificados.

Para executar esta atividade, foi designado o Projetista de Serviços do Banco BES, auxiliado mais uma vez pelo consultor da IVS. Antes de ser exportado em formato WS-BPEL, o modelo de processo criado na ferramenta *WebSphere Business Modeler* passou por uma modelagem técnica. A modelagem técnica consistiu na realização de ajustes e adição de informações necessárias para que o processo pudesse ser convertido de maneira adequada para WS-BPEL. Após os ajustes, o processo foi exportado em formato WS-BPEL para a ferramenta de integração de processos *WebSphere Integration Developer*.

A partir daí, foi realizada implementação do processo em WS-BPEL. Uma vez que a estrutura principal do fluxo de trabalho e as variáveis de processo já vieram prontas da exportação, foi necessário implementar no processo os seguintes aspectos:

- Lógica de tratamento de dados
- Lógica de decisão nos nós do tipo “*Choice*” (*if-then-else*) e nos loops do tipo “*While*”
- Dados de correlação de instâncias de processo
- Tempo de expiração de atividades com tal requisito
- Mapeamento das interfaces geradas no WS-BPEL para as interfaces WSDL dos serviços utilizados pelo processo

Após implementado, o processo em WS-BPEL foi encapsulado pelo serviço de orquestração “*InclusaoClientePF*”. Para tal, foi necessário também definir a interface WSDL e as mensagens em XSD para o serviço de orquestração.

6.1.2.9 Implementação e Teste

A partir da construção da lógica de orquestração do processo em WS-BPEL, foram conduzidas as atividades de implementação dos serviços, integração dos serviços ao processo e testes de todos os elementos desenvolvidos.

O estudo de caso não contemplou o acompanhamento de todas as atividades de implementação, mas somente o início da implementação dos serviços, segundo as definições de realização especificadas durante a Análise e Projeto.

Os testes foram especificados por um analista do Banco BES, com o auxílio de um consultor da QSP. As especificações abrangiam desde testes unitários das operações dos serviços até testes integrados dos vários cenários de negócio possíveis para o processo.

6.1.3 Resultados

A aplicação do método no projeto piloto do Banco BES pôde ser considerada bem-sucedida, pois permitiu identificar, definir e implementar um conjunto de serviços seguindo uma seqüência lógica de atividades. Até o fim do estudo de caso, a solução não havia sido implantada em ambiente de produção, mas encontrava-se em fase de testes de homologação.

Durante a Análise de *Gap*, observou-se que neste momento é possível identificar novas operações candidatas para os serviços analisados. São operações que podem ser levantadas principalmente nas aplicações legadas examinadas, permitindo a identificação *bottom-up* de novas operações para o serviço. Pelo fato de terem sido identificadas de modo *bottom-up*, é baixa a probabilidade de estas operações serem utilizadas pelos processos de negócio ou casos de uso sendo analisados. Entretanto, a identificação delas neste momento evitaria uma revisão após a Verificação dos Princípios, como de fato ocorreu no projeto piloto do Banco BES.

Devido a mudanças ocorridas durante a execução do projeto piloto, o método foi aplicado com diversas iterações da fase de Análise. Após a execução da Análise de

Realização, apesar de as decisões de realização terem sido aprovadas, foi necessário voltar à atividade de Modelagem de Processo *to-be* para efetuar mudanças no modelo de processo de negócio, a pedido dos usuários de negócios. Com isso, houve uma nova iteração das atividades Identificação de Serviços Candidatos, da Análise de *Gap* e da Análise de Realização. Nesta nova iteração, o modelo de processo alterado foi revisto para possivelmente identificar novas operações e serviços candidatos e posteriormente analisá-los. Pelo menos duas mudanças deste tipo ocorreram durante a execução do projeto, gerando novas iterações. Esta experiência durante o projeto piloto permitiu verificar a aplicabilidade da abordagem iterativa da fase de Análise do MAPOS. Originalmente, as iterações haviam sido propostas para resolver restrições de realização, porém constatou-se que elas podem ser executadas para suportar mudanças nos requisitos de negócio.

Pela realização do estudo de caso, verificou-se que a condução do desenvolvimento de uma solução baseada em SOA e BPM torna-se eficiente com o uso de uma plataforma integrada de ferramentas. Tal integração facilitou a transição da fase de Modelagem de Negócio e para a Análise e a transição da fase de Projeto para a Construção.

O estudo de caso permitiu constatar que é possível adaptar o MAPOS para aplicá-lo junto a um processo completo de desenvolvimento, como o RUP. No caso do Banco BES, as atividades do MAPOS foram incluídas no início do fluxo de trabalho da disciplina de Análise e Projeto, seguidas pelas atividades de análise e projeto de classes e componentes.

6.2 Caso NF-e

6.2.1 Contexto do Estudo

O Projeto Nota Fiscal eletrônica (NF-e) é uma iniciativa da Secretaria da Receita Federal brasileira em conjunto com as Secretarias da Fazenda estaduais, que visa substituir o atual documento fiscal emitido em papel por um equivalente em formato eletrônico. Segundo o projeto, a nota fiscal passa a ser um documento emitido e armazenado digitalmente, que documenta uma operação de comercialização de

mercadorias ou prestação de serviços. Sua validade jurídica e fiscal é obtida através de um processo de assinatura digital do emitente e da Autorização de Uso dada pela Secretaria da Fazenda (ENCONTRO..., 2006).

O projeto NF-e prevê a implantação, em cada Secretaria da Fazenda estadual, de uma aplicação para gerenciar o ciclo de vida das notas fiscais eletrônicas, desde seu recebimento a partir do contribuinte, passando por sua validação, armazenamento e disponibilização para consultas. Por sua vez, os contribuintes emissores de notas fiscais deverão ser capazes de acessar o sistema da Secretaria da Fazenda de seu estado e enviar suas notas fiscais.

A arquitetura é composta de um lado servidor e um lado cliente. O lado servidor, implementado pela Secretaria da Fazenda de cada estado, deve disponibilizar serviços da forma de *Web Services* para que as empresas sejam capazes de registrar suas notas fiscais eletrônicas. Os sistemas das Secretarias da Fazenda deverão disponibilizar os seguintes serviços para os contribuintes:

- Recepção de lote de NF-e
- Consulta de processamento de lote
- Cancelamento de NF-e
- Inutilização de numeração de NF-e
- Consulta de situação atual de NF-e
- Consulta do status do serviço
- Consulta cadastro de contribuintes

Os requisitos foram definidos pelo Encontro Nacional de Coordenadores e Administradores Tributários Estaduais (ENCAT), composto por membros das Secretarias da Fazenda Estaduais e da Secretaria da Receita Federal, e publicados no Manual de Integração do Contribuinte (ENCONTRO..., 2006). Com base nos requisitos deste manual, a IVS desenvolveu a aplicação que implementa o lado servidor, de responsabilidade da Secretaria da Fazenda estadual. O Servidor NF-e foi estruturado em serviços integrados por uma ferramenta de ESB. Para atender a suas necessidades decorrentes do Projeto NF-e, uma Secretaria da Fazenda estadual, denominada SEFAZ, adquiriu o Servidor NF-e da IVS juntamente com a ferramenta de ESB, que serviria de infra-estrutura para futuras soluções orientadas a serviços.

Os serviços da aplicação original foram definidos sem o uso de um método bem definido, que garantisse a aderência aos princípios de orientação a serviços. Deste modo, foi proposta a aplicação do MAPOS para realizar a reengenharia dos serviços. Para tal, dois profissionais foram treinados no MAPOS e responsabilizados por aplicar o método para identificar e especificar os serviços que compõem a aplicação a partir de seus requisitos originais. Para garantir validade deste estudo de caso, foram selecionados para executar o método dois profissionais sem qualquer conhecimento prévio sobre a versão original do Servidor NF-e. Deste modo, os serviços resultantes formariam um conjunto totalmente novo, definido somente de acordo com os requisitos e com o método aplicado. As atividades deste estudo de caso foram executadas entre março e abril de 2009.

6.2.1.1 Empresas Envolvidas

A IVS foi a empresa responsável por desenvolver o Servidor NF-e e realizar a posterior revisão de sua arquitetura. Nas atividades deste estudo de caso, participaram os seguintes recursos humanos da IVS:

- Um especialista em SOA (o autor deste trabalho), que orientou a execução do MAPOS e atuou como arquiteto corporativo.
- Um analista de sistemas, que atuou como analista de processos e analista de serviços.
- Um arquiteto de software, que atuou como analista de serviços e projetista de serviços.

O cliente que adquiriu o Servidor NF-e da IVS foi a SEFAZ, uma Secretaria de Fazenda estadual que buscava uma solução para o Projeto NF-e e uma infraestrutura para iniciar a adoção de arquitetura orientada a serviços.

6.2.1.2 Ferramentas Utilizadas

O *Rational Method Composer* da IBM foi empregado para documentar o método e disponibilizá-lo para os profissionais, tanto para treinamento quanto para referência.

Para edição de documentos de texto, como o Modelo de Serviços, foi usado o Microsoft Word.

O *IBM Rational Software Architect* foi utilizado para criar os documentos de mensagens e de interface dos serviços, bem como os diagramas UML que representavam sua estrutura e comportamento.

O *Enterprise Architect* da Sparx Systems foi usado para modelar os processos na notação BPMN e gerenciar os requisitos.

O ambiente de execução do Servidor NF-e utilizado foi o *IBM WebSphere Enterprise Service Bus*, ferramenta com as funções de barramento ESB e servidor de aplicações. Como IDE de desenvolvimento, foi usado o *IBM WebSphere Integration Developer*.

6.2.2 Aplicação do Método

Para redefinir os serviços que compõem o Servidor NF-e, os profissionais da IVS empregaram o MAPOS.

6.2.2.1 Modelagem de Processo *as-is* e *to-be*

Esta tarefa teve como objetivo elaborar os modelos de processo de negócio, isto é, representar na forma de uma notação o processo de negócio na sua forma atual (*as-is*) e o processo futuro (*to-be*), contendo as mudanças propostas.

Os requisitos para o projeto NF-e estão definidos e documentados no Manual de Integração do Contribuinte, sendo que cada um dos serviços a serem providos pela Secretaria da Fazenda estão especificados com suas mensagens, interfaces e comportamento funcional. O comportamento de cada serviço está descrito na forma de processo *to-be* ilustrado por um diagrama de atividade, exemplificado na Figura 6.8.

Figura 6.8: Diagrama do processo Recepção de Lote de NF-e (ENCONTRO..., 2006)

Por possuírem alta granularidade e já estarem modelados em uma notação de processo, tais serviços (Recepção de Lote de NF-e, Processamento de Lote de NF-e, Retorno de Recepção de Lote de NF-e, Consulta de NF-e, Cancelamento de NF-e, Inutilização de Numeração de NF-e e Consulta de Status de Serviço) foram classificados como serviços de orquestração. Pertenceriam à camada de serviços de orquestração, sendo que cada um encapsularia a lógica de um processo de negócio. O objetivo do estudo de caso seria então a análise e projeto dos serviços de negócio e de aplicação que comporiam os serviços de orquestração já definidos.

Como os processos *to-be* já estavam definidos, esta atividade consistiu somente na criação dos diagramas em BPMN representando a lógica especificada na documentação.

6.2.2.2 Identificação de Serviços Candidatos

Recebendo como entrada os modelos *to-be* dos processos de negócio (encapsulados pelos serviços de orquestração), esta atividade teve como objetivo identificar um conjunto inicial de serviços candidatos de negócio e de aplicação.

Nesta atividade, tanto o analista de sistemas quanto o arquiteto de software atuaram como analistas de serviços, enquanto o autor deste trabalho forneceu orientações sobre o método. Analisando-se a documentação de cada processo de negócio, foram identificados os serviços exibidos na Tabela 6.4.

Tabela 6.4: Operações candidatas identificadas

Processo de Negócio	Operação Candidata
Recepção de Lote de NF-e	<ul style="list-style-type: none"> • Identificar número de CNPJ do transmissor • Validar tamanho do lote • Gerar número de recibo para o lote • Obter tempo médio de resposta • Gravar lote • Gerar recibo • Enviar lote para fila de entrada
Processamento de NF-e	<ul style="list-style-type: none"> • Validar XML do lote • Validar assinatura e integridade de cada NF-e • Validar autorização do emissor • Identificar número de CNPJ do emissor na assinatura • Validar dígito verificador do número de CNPJ • Validar dígito verificador do número de IE • Validar data de emissão de NF-e • Validar existência de NF-e • Verificar regularidade fiscal do Emissor • Verificar regularidade fiscal do Destinatário • Gravar NF-e • Criar protocolo de status • Enviar lote para fila de saída
Retorno de Recepção de Lote de NF-e	<ul style="list-style-type: none"> • Validar XML do recibo • Verificar existência de lote na fila de saída • Verificar existência de lote na fila de entrada
Cancelar NF-e	<ul style="list-style-type: none"> • Validar XML do pedido de cancelamento de NF-e • Validar assinatura do cancelamento • Identificar número de CNPJ do emissor na assinatura • Validar autorização do emissor • Verificar existência de NF-e • Verificar existência de autorização de uso para NF-e • Verificar recebimento de NF-e pelo destinatário • Verificar existência de registro de trânsito para a NF-e • Verificar status da NF-e • Cancelar NF-e • Criar protocolo de status

continua

Processo de Negócio	Operação Candidata
Inutilizar Numeração de NF-e	<ul style="list-style-type: none"> Validar XML do pedido de inutilização Validar assinatura da inutilização Identificar número de CNPJ do emissor na assinatura Validar autorização do emissor Verificar regularidade fiscal do Emissor Validar limite de faixa de inutilização da NF-e Verificar existência de NF-e na numeração inutilizada Efetivar inutilização da numeração da NF-e Criar protocolo de status
Consultar Dados da NF-e	<ul style="list-style-type: none"> Validar XML do pedido de consulta de NF-e Verificar existência de NF-e
Consultar Status do Serviço	<ul style="list-style-type: none"> Validar XML do pedido de consulta de status Consultar status do serviço

Uma vez identificadas as operações candidatas, foi realizado o agrupamento delas em serviços candidatos. As operações candidatas foram agrupadas por similaridade quanto ao tipo de lógica executada e quanto ao tipo de informação manipulada. Durante o agrupamento, foi detectado que algumas operações poderiam ser eliminadas, uma vez que executavam a mesma lógica que outra operação identificada. Por exemplo, constatou-se que as operações “Verificar regularidade fiscal do Emissor” e “Verificar regularidade fiscal do Destinatário” executavam exatamente o mesmo tipo de validação, não havendo diferença para emissores ou destinatários de NF-e. Portanto, foram substituídas por uma única operação chamada “Verificar regularidade fiscal”.

O resultado deste agrupamento pode ser visualizado na Tabela 6.5.

Tabela 6.5: Operações candidatas agrupadas por serviço candidato

Serviço Candidato	Operação Candidata
Lote	<ul style="list-style-type: none"> Validar tamanho do lote Gerar número de recibo de lote Obter tempo médio de resposta Gravar lote Gerar recibo
Validar XML	<ul style="list-style-type: none"> Validar XML do lote Validar XML do recibo Validar XML do pedido de cancelamento de NF-e Validar XML do pedido de inutilização Validar XML do pedido de consulta de NF-e Validar XML do pedido de consulta de serviço

Serviço Candidato	Operação Candidata
Nota Fiscal	<ul style="list-style-type: none"> Validar data de emissão da NF-e Criar protocolo de status Gravar NF-e Verificar existência de autorização de uso para NF-e Verificar recebimento de NF-e pelo destinatário Verificar existência de registro de trânsito para a NF-e Verificar existência de NF-e Cancelar NF-e Verificar status da NF-e Verificar existência de NF-e na numeração inutilizada Validar limite de faixa de inutilização da NF-e Efetivar inutilização da numeração da NF-e
Assinatura	<ul style="list-style-type: none"> Validar assinatura e integridade de cada NF-e Validar assinatura do cancelamento Validar assinatura da inutilização Identificar número de CNPJ do certificado
Autorização	<ul style="list-style-type: none"> Validar autorização do emissor
Regularidade Fiscal	<ul style="list-style-type: none"> Verificar Regularidade Fiscal
IE/CNPJ	<ul style="list-style-type: none"> Validar Dígito Verificador do número de IE Validar Dígito Verificador do número de CNPJ
Verificar Status	<ul style="list-style-type: none"> Consultar status do serviço
Fila	<ul style="list-style-type: none"> Enviar lote para a fila de entrada Enviar lote para a fila de saída Verificar existência de lote na fila de entrada Verificar existência de lote na fila de saída

Após a definição do agrupamento, cada serviço com suas operações candidatas foram descritos no documento de Modelo de Serviços. Exemplos de descrição de serviço candidato e operação candidata podem ser vistos na Tabela 6.6 e na Tabela 6.7, respectivamente.

Tabela 6.6: Descrição do serviço candidata "Regularidade Fiscal"

Tipo de Informação	Descrição
Nome	Regularidade Fiscal
Descrição	Serviço responsável manter informações sobre a regularidade fiscal dos contribuintes emissores e destinatários de notas fiscais eletrônicas (NF-e).

Tabela 6.7: Descrição da operação candidata "Verificar Regularidade Fiscal"

Tipo de Informação	Descrição
Nome	Verificar Regularidade Fiscal
Descrição	Verificação a situação de regularidade fiscal de um contribuinte, podendo consultar órgãos responsáveis.
Dados de Entrada	Número de CNPJ do contribuinte a ser verificado
Dados de Saída	<p>Indicador de situação fiscal válida para o contribuinte com o número de CNPJ informado, considerando o domínio a seguir:</p> <ul style="list-style-type: none"> True (caso regular) False (caso irregular)

6.2.2.3 Análise de Gap

Após a identificação dos serviços candidatos, o método prosseguiu com a análise de *gap*, quando se buscaram nas aplicações existentes as funções necessárias para os serviços candidatos, chegando a um cenário de reuso para cada operação candidata.

Os profissionais foram responsáveis por realizar a análise baseada em informações fornecidas pelo autor a respeito da infra-estrutura existente na SEFAZ. Para não comprometer o estudo de caso, não foi considerado o reuso de nenhum recurso existente no Servidor NF-e, mas somente nos sistemas existentes na SEFAZ antes do Projeto NF-e. A Tabela 6.8 exibe os serviços em que foi possível detectar oportunidades de reuso nos recursos já existentes. Nos outros serviços analisados, o cenário de reuso obtido foi “Operação não existente”.

Tabela 6.8: Cenários de reuso para operações candidatas

Serviço Candidato	Operação Candidata	Cenário de Reuso
Nota Fiscal	Validar data de emissão da NF-e	Operação não existente
	Criar protocolo de status	Operação não existente
	Gravar NF-e	Operação não existente
	Verificar existência de autorização de uso para NF-e	Operação não existente
	Verificar recebimento de NF-e pelo destinatário	Operação parcialmente realizada por aplicação legada (Sistema de Trânsito de Mercadoria)
	Verificar existência de registro de trânsito para a NF-e	Operação realizada por aplicação legada (Sistema de Trânsito de Mercadoria)
	Verificar existência de NF-e	Operação não existente
	Cancelar NF-e	Operação não existente
	Verificar status da NF-e	Operação não existente
	Verificar existência de NF-e na numeração inutilizada	Operação não existente
	Validar limite de faixa de inutilização da NF-e	Operação não existente
	Efetivar inutilização da numeração da NF-e	Operação não existente
Regularidade Fiscal	Verificar Regularidade Fiscal	Operação realizada parcialmente pela aplicação Cadastro de Contribuintes
Validar IE/CNPJ	Validar Dígito Verificador do número de IE	Operação realizada por aplicação legada (biblioteca Brazil Utils)
	Validar Dígito Verificador do número de CNPJ	Operação realizada por aplicação legada (biblioteca Brazil Utils)

Para o serviço “Nota Fiscal”, devido aos cenários de reuso identificados, decidiu-se separar as duas operações realizadas pela aplicação legada Sistema de Trânsito de Mercadoria (“Verificar recebimento de NF-e pelo destinatário” e “Verificar existência de registro de trânsito para a NF-e”) em um novo serviço chamado “Registro de Trânsito”. Entendeu-se que as informações sobre o trânsito de mercadorias são independentes das informações de notas fiscais, possibilitando a criação de serviços separados para lidar com cada tipo de informação. A separação foi realizada também para facilitar as decisões de realização.

Os cenários de reuso analisados foram documentados no Modelo de Serviços, assim como as alterações no agrupamento dos serviços candidatos.

6.2.2.4 Análise de Realização

Na análise de realização, cada operação candidata é avaliada quanto ao seu cenário de reuso identificado anteriormente. Nesta etapa, deve ser decidido como cada operação de serviço será realizada.

Nesta atividade, todos atuaram como analistas de serviços e arquitetos corporativos, discutindo as alternativas de realização para os serviços candidatos.

Para os serviços candidatos com operações não existentes, decidiu-se que seriam realizados por componentes Java expostos na forma de *Web Services* executando no servidor de aplicações do *WebSphere Enterprise Service Bus*.

Para os outros serviços, foram tomadas as seguintes decisões de realização:

Serviço “RegistroTransitoService”

Os dados necessários para as operações deste serviço já existiam no Sistema de Trânsito de Mercadorias. Como o sistema é implementado em Java, optou-se por expor as funções correspondentes às operações na forma de *Web Services*.

Serviço “RegularidadeFiscalService”

O sistema de Cadastro de Contribuintes é implementado em tecnologia proprietária sem suporte a *Web Services*. No entanto, como há na Secretaria

da Fazenda a necessidade de expor outras funções deste sistema como serviços, além da regularidade, optou-se por implementar um conector no ESB que acessasse a base de dados do sistema para prover a operação. Futuramente, esse conector poderia ser expandido para prover outras operações relacionadas.

Serviço “IECNPJService”

A lógica necessária para as operações deste serviço já estava implementada na biblioteca Java Brazil Utils. Portanto, decidiu-se por criar um novo *Web Service* em Java que utilizasse a biblioteca para realizar as operações.

As decisões de realização foram documentadas no Modelo de Serviços.

6.2.2.5 Especificação de Serviços

A atividade de especificação de serviços tem como objetivo detalhar as classes de mensagens e as interfaces de serviços, inclusive com a descrição técnica de cada operação, para chegar aos artefatos que comporão o contrato do serviço.

Os esquemas XSD e interfaces WSDL dos serviços foram criados utilizando a ferramenta *Rational Software Architect*. As mensagens e interfaces foram definidas na forma de um modelo de classes, que foi então convertido em documentos WSDL e XSD.

A Figura 6.9 mostra alguns exemplos de interfaces de serviço modeladas na ferramenta.

Figura 6.9: Exemplos de serviços especificados

Os diagramas de classe foram complementados por descrições textuais de cada serviços e de cada operação de serviço. A descrição de cada operação detalhava a lógica de processamento a ser executada por ela e descrevia os requisitos não-funcionais aplicáveis.

Os serviços foram classificados de acordo com seu nível de granularidade, sendo separados nas camadas de serviços de negócio e de serviços de aplicação. A alocação dos serviços nas camadas pode ser resumida na Tabela 6.9.

Tabela 6.9: Alocação dos serviços em camadas

Serviço	Camada
LoteService	Serviço de negócio
ValidacaoXMLService	Serviço de aplicação
NotaFiscalService	Serviço de negócio
RegistroTransitoService	Serviço de negócio
AssinaturaService	Serviço de aplicação
AutorizacaoEmissorService	Serviço de negócio
RegularidadeFiscalService	Serviço de negócio
IECNPJService	Serviço de aplicação
VerificarStatusService	Serviço de aplicação
FilaService	Serviço de aplicação

Esta atividade foi realizada pelo arquiteto de software, que atuou como projetista de serviços.

6.2.2.6 Verificação dos Princípios

O objetivo desta atividade é verificar se cada especificação de serviço criada atende aos princípios de orientação a serviços e se estão adequados a integrarem o portfólio de serviços da organização.

Esta atividade foi realizada pelo arquiteto de software exercendo o papel de Arquiteto Corporativo, orientado pelo autor deste trabalho.

A seguir são descritas recomendações resultantes desta verificação.

Serviço “NotaFiscalService”

Verificou-se que as operações “validarDuplicidadeNfe”, “validarExistenciaNfe” e “obterStatusNfe” possuíam funções parecidas, sendo mantida apenas a operação “obterStatusNfe”.

Serviço “ValidarXML”

Este serviço continha seis operações de validação de XML, sendo um para cada tipo de mensagem, que correspondiam às mensagens de entrada dos serviços de orquestração do Projeto NF-e. Como esta característica geraria um acoplamento entre o serviço “ValidacaoXMLService” e os serviços de orquestração, decidiu-se criar uma única operação genérica “validarXMLMensagem” capaz de validar qualquer tipo de mensagem, dependendo do parâmetro passado. Além de reduzir o acoplamento, tal modificação aumentaria a reusabilidade do serviço.

Serviço “AssinaturaService”

Pelo mesmo motivo do serviço “ValidarXMLService”, foi decidido substituir as operações de validação de assinatura para mensagens específicas (“validarAssinaturaNfe”, “validarAssinaturaCancelamento” e “validarAssinaturaInutilizacao”) por uma única operação genérica “validarAssinaturaMensagem”.

Serviço “RegularidadeFiscalService”

Para promover a reusabilidade deste serviço, recomendou-se a alteração do escopo funcional do serviço para que ele passasse a validar a regularidade fiscal tanto de pessoas jurídicas quanto de físicas. Portanto, as operações deste serviço passariam a ser “verificarRegularidadeFiscalPJ” e “verificarRegularidadeFiscalPF”.

Serviço “IECNPJ”

Para tornar o serviço, mais reusável decidiu-se acrescentar a operação “validarCPF”, alterando seu nome para “ValidarIdentificadorService”.

6.2.2.7 Revisão dos Serviços

Na revisão, quaisquer não conformidades detectadas durante a verificação dos princípios devem ser corrigidas nos artefatos que compõem a especificação dos serviços.

A revisão consistiu apenas na realização de ajustes nas interfaces de serviços especificadas, adicionando-se ou alterando-se as operações conforme determinado durante a Verificação dos Princípios. Esta atividade foi executada pelo arquiteto de software atuando como Projetista de Serviços.

6.2.2.8 Orquestração de Serviços

Durante a orquestração de serviços, o modelo de processo *to-be* elaborado no início do projeto é convertido em um modelo de implementação, que devem se trabalhar para tratar todas as regras de negócio modeladas e adequar-se aos serviços especificados.

A lógica de orquestração de serviços já estava implementada na versão original do Servidor NF-e em código Java. Para o projeto de reengenharia de serviços, seria necessário somente substituir as invocações às antigas interfaces de serviços por

invocações às novas interfaces modeladas. No entanto, o estudo de caso não acompanhou a execução desta atividade.

6.2.2.9 Implementação e Teste

A partir da construção da lógica de orquestração de serviços, devem ser conduzidas as atividades de implementação dos serviços, integração dos serviços e testes de todos os elementos desenvolvidos.

Apesar de todas as especificações de serviços estarem definidas e revisadas, a execução destas atividades não foi provisionada até o fim da realização deste estudo de caso.

6.2.3 Resultados

Neste segundo estudo de caso, o método mostrou-se eficaz ao guiar o procedimento de identificação, análise e especificação de serviços. Por se tratar de um projeto de reengenharia, foi possível notar que a execução do MAPOS resultou em um conjunto de serviços mais coeso e aderente aos princípios de orientação a serviços que os serviços originais, que possuíam nível mais baixo de granularidade e menor autonomia, sendo mais semelhantes a componentes.

Como os processos foram modelados em um nível menor de granularidade, a identificação de operações candidatas deu-se de maneira bastante direta. Na maioria dos casos, podia-se estabelecer uma relação de uma tarefa de processo para uma operação de serviço. Entretanto, nem todas as tarefas do processo deveriam ser executadas por serviços. Para resolver esta questão, foi constatado que poderia haver no método alguma verificação sobre as operações candidatas identificadas, de modo a descartar tarefas com nível muito baixo de granularidade e complexidade, principalmente no caso de regras de validação.

Mais uma vez, os participantes do estudo de caso manifestaram dúvidas no momento de agrupar as operações candidatas em serviços candidatos. Notou-se que um dos profissionais realizou o agrupamento mais orientado a tarefas, enquanto

o outro profissional procurou agrupar as operações de acordo com as entidades de dados manipuladas. Foram necessárias explicações por parte do autor, em complemento ao material de referência.

6.3 Análise dos Resultados

O fluxo de trabalho definido pelo método mostrou ser uma seqüência lógica de atividades eficiente como abordagem para análise e projeto orientados a serviços. Foi possível constatar que os passos a serem seguidos e a ordem proposta permitem a obtenção de um conjunto de especificações de serviços a partir de modelos e requisitos de negócio.

Verificou-se que a abordagem iterativa proposta pelo método durante a fase de Análise é aplicável. No caso de serem executadas várias iterações para atender a novos requisitos e processos, após a atividade de Análise de Realização, os serviços poderiam ser registrados em um repositório com status “em especificação”, por exemplo. Deste modo, na atividade de Análise de *Gap*, os serviços identificados em iterações anteriores seriam considerados como serviços já existentes no portfólio da organização que poderiam ser reusados.

A abordagem de cenários de reuso, utilizada nas atividades de Análise de *Gap* e Análise de Realização, foi bem sucedida em seu propósito de guiar a identificação de oportunidades de reuso e tomada de decisões de como cada serviço será realizado. Os cenários de reuso propostos permitiram classificar as situações encontradas durante a Análise de *Gap*, com uma única ressalva da possibilidade de separar cenários de reuso para funcionalidade e dados de cada operação candidata.

Apesar de o fluxo de trabalho e definição das atividades estarem adequados, um possível ponto de melhoria seria um aprofundamento das diretrizes sobre como identificar operações candidatas, agrupar as operações em serviços candidatos e verificar os princípios de orientação a serviços. Foram estes os pontos em que os participantes mostraram mais dúvidas e em que foram necessárias mais orientações por parte do autor. Um fator que pode ser levado em conta é que em ambos os estudos de casos, os participantes estavam aplicando o método pela primeira vez.

No caso do Banco BES, os modelos de processo foram elaborados com um nível mais alto de granularidade e por isso tornou-se necessário um detalhamento textual das tarefas. Deste modo, as operações candidatas poderiam ser identificadas a partir do diagrama de processo e das descrições textuais de cada tarefa.

Em nenhum dos dois estudos de caso realizados foi possível realizar a identificação a partir de casos de uso, porém o formato utilizado pelo Banco BES para descrever as tarefas é bastante semelhante a um fluxo de caso de uso. Sendo assim, pode-se afirmar que a identificação de serviços a partir de descrições textuais (como um caso de uso) foi exercitada no caso do Banco BES.

Já no caso NF-e, os processos estavam definidos em um nível mais baixo de granularidade, o que permite um mapeamento mais direto das tarefas do processo para operações de serviços.

Portanto, o método mostra que pode ser aplicado mesmo em casos em que os modelos de processo não chegam ao nível de granularidade de chamadas de serviços. Nestes casos, uma das duas alternativas pode ser escolhida: decompor o modelo de processo em tarefas de menor granularidade ou elaborar descrições textuais detalhadas de cada tarefa.

No caso do Banco BES, foi possível constatar que a inclusão de um ponto de verificação no método (a atividade de Verificação dos Princípios) pode funcionar como um instrumento para assegurar a governança dos serviços. Neste caso, além da validação dos princípios de orientação a serviços, foi realizada uma validação da aderência dos serviços aos padrões e à arquitetura de referência da organização.

Uma das características verificadas é a aderência do MAPOS à abordagem BPM, que foi verificada no caso do Banco BES. Este caso envolvia o projeto de uma solução completa de BPM com SOA. Entretanto, a aplicabilidade do método não se limitou a projetos desta natureza, pois foi verificada também em um projeto não baseado na abordagem BPM, o caso NF-e.

Um resultado obtido foi a verificação de que o MAPOS é aplicável tanto em projetos de desenvolvimento de novos serviços e processos, como o caso do Banco BES, quanto em projetos de reengenharia de serviços, como o caso NF-e.

Uma situação não testada pelos estudos de caso foi a aplicação do método sem a presença do autor. Como as equipes envolvidas não possuíam experiência em

métodos de análise e projeto de serviços e, no caso do Banco BES, não eram tão familiares com os conceitos de SOA, esta presença tornou-se necessária para a condução das atividades. No entanto, tal participação limitou-se a orientações sobre como executar as atividades, buscando não influenciar no conjunto de serviços resultantes.

6.4 Considerações Finais

Os estudos de caso documentados neste Capítulo 6 permitiram avaliar a aplicabilidade do MAPOS em dois contextos distintos.

Os estudos de caso permitiram verificar a aplicabilidade do MAPOS em um projeto de BPM com SOA e em um projeto de reengenharia de uma solução contemplando apenas orientação a serviços.

De forma geral, o método mostrou-se efetivo em seu propósito, orientando seus usuários durante as etapas de análise e projeto de serviços.

7 CONCLUSÃO

Este capítulo contém as considerações finais sobre o presente trabalho. São apresentadas as contribuições geradas pelo trabalho e sugestões para trabalhos futuros.

7.1 Contribuições do Trabalho

Este trabalho baseou-se em um estudo aprofundado sobre os conceitos relacionados a SOA, focando em um dos principais desafios enfrentados pelas organizações na adoção da orientação a serviços, que é a complexidade da análise e projeto dos serviços.

Um estudo de diversos métodos propostos para resolver tal desafio foi realizado, resultando em uma análise crítica e comparativa das abordagens. Para cada método, foi apresentado seu posicionamento dentro de um ciclo de vida de desenvolvimento e foram descritas suas atividades de análise e projeto, papéis e artefatos. Este estudo permitiu identificar pontos em comum nos métodos, as boas práticas propostas em cada um e suas eventuais limitações. Foi comparada também a maneira como cada método procura solucionar questões específicas de análise e projeto, como o reuso de recursos existentes e a aderência aos princípios de orientação a serviços.

A análise dos métodos traz como contribuição uma visão sobre o estado da arte em análise e projeto orientados a serviços. Mesmo métodos propostos após a publicação deste trabalho poderiam ser analisados seguindo o mesmo formato descrito e avaliando-se critérios semelhantes.

As boas práticas identificadas neste estudo serviram como base para definir um conjunto de requisitos demandados pelo paradigma orientação a serviços para as atividades de análise e projeto.

O conjunto de requisitos levantados não pode ser classificado como completo, isto é, não há como afirmar que são os únicos requisitos necessários para análise e projeto de serviços. É possível que, se fossem analisados mais métodos existentes,

identificassem-se requisitos adicionais. No entanto, esse conjunto de requisitos pode ser considerado uma contribuição válida por estabelecerem parâmetros para comparar métodos existentes e por terem guiado a elaboração de uma nova proposta de método.

A principal contribuição trazida pelo trabalho é o método proposto com base no estudo dos conceitos de SOA e na análise dos métodos existentes, o Método de Análise e Projeto Orientado a Serviços (MAPOS). O MAPOS foi elaborado de modo a unificar os métodos analisados e atender ao conjunto de requisitos levantado. Foram empregadas no método proposto boas práticas derivadas de todos os métodos analisados, sendo elas integradas em um fluxo de trabalho único.

A realização de dois estudos de caso permitiu verificar a aplicabilidade do MAPOS em projetos reais de contextos distintos. O método foi aplicado em um projeto de criação de uma nova solução orientada a serviços, com o uso da abordagem BPM, e em um projeto de reengenharia de serviços de uma solução sem BPM. Em ambos os projetos, foi empregada a tecnologia *Web Services* para especificar e implementar os serviços.

O MAPOS foi considerado eficiente como fluxo de trabalho para execução de análise e projeto de serviço, inclusive com a abordagem iterativa para as atividades de análise. Foi constatado apenas que poderiam haver mais diretrizes e orientações sobre como executar algumas atividades.

O método proposto mostrou-se especialmente efetivo na análise de reuso de recursos existentes e na manutenção da governança de serviços. Tais características suportam dois pontos críticos da orientação a serviços. Um deles é a redução do custo de projetos e aumento da agilidade na entrega de soluções, obtidos pelo reuso. E o outro é a necessidade de governança sobre o portfólio de serviços da organização, suprida pelas atividades de validação do método.

7.2 Trabalhos Futuros

O método proposto é uma versão inicial de uma abordagem de análise e projeto orientados a serviços. Possíveis complementações ao trabalho apresentado poderiam ser trazidas no intuito de evoluir o método.

Trabalhos futuros poderiam envolver novas aplicações do método em diversos tipos de cenários e projetos. Por exemplo, o MAPOS poderia ser aplicado em projetos que utilizassem somente casos de uso para a modelagem de negócio, projetos que utilizassem outras tecnologias de implementação de serviços além de *Web Services* ou projetos que utilizassem plataformas e ferramentas diferentes daquelas utilizadas neste trabalho. Novos estudos de caso contribuiriam para obter uma versão mais refinada e madura do MAPOS, inclusive adicionando novas orientações sobre como executar as atividades.

Melhorias ao método proposto poderiam ser sugeridas com base na análise de outros métodos de análise e projeto orientados a serviços que possam ter surgido posteriormente à elaboração deste trabalho. Este tipo de análise poderia inclusive dar origem a novos requisitos a serem atendidos pelo método.

Outra possibilidade de trabalho futuro seria estender a definição do método proposto de forma a constituir um ciclo completo de desenvolvimento de serviços, desde concepção do projeto até a implantação e gerenciamento dos serviços em produção.

REFERÊNCIAS²

BIEBERSTEIN, N. et al. *Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap*. Estados Unidos: IBM Press, 2005. 272 p. ISBN: 0131870025.

BRAY, T. et al. *Extensible Markup Language (XML) 1.1*, Second Edition. W3C Recommendation, 2006. [acesso em 27 de março de 2008]. Disponível em: <<http://www.w3.org/TR/2006/REC-xml11-20060816/>>

CHRISTENSEN, E. et al. *Web Services Description Language (WSDL) 1.1*. W3C Note, 2001. [acesso em 1 de dezembro de 2008]. Disponível em: <<http://www.w3.org/TR/wsdl>>

COCKBURN, A. *Writing Effective Use-Cases*. Boston: Addison Wesley, 2000. 304 p. ISBN: 0201702258.

ENCONTRO NACIONAL DE COORDENADORES E ADMINISTRADORES TRIBUTÁRIOS ESTADUAIS. *Projeto Nota Fiscal Eletrônica: Manual de Integração – Contribuinte*, versão 1.1.1. Brasil, 2006. [acesso em 6 de abril de 2009]. Disponível em: <<http://www.nfe.fazenda.gov.br/portal/integracao.aspx>>

ENDREI, M. et al. *Patterns: Service-Oriented Architectures and Web Services*. Estados Unidos: IBM Redbooks, 2004. 348 p. ISBN: 073845317X.

ERL, T. *Service-Oriented Architecture: Concepts, Technology and Design*. Estados Unidos: Prentice Hall, 2005. 792 p. ISBN: 0131858580.

ERL, T. *SOA: Principles of Service Design*. Estados Unidos: Prentice Hall, 2007. 608 p. ISBN: 0132344823.

FALLSIDE, D. C.; WALMSLEY, P. *XML Schema Part 0: Primer*, Second Edition. W3C Recommendation, 2004. [acesso em 1 de dezembro de 2008]. Disponível em: <<http://www.w3.org/TR/xmlschema-0/>>.

² De acordo com a International Organization for Standardization (ISO).

FUGITA, H. S.; HIRAMA, K. MAPOS: Método de Análise e Projeto Orientado a Serviços. In Anais do Congresso Internacional de Gestão de Tecnologia e Sistemas de Informação (CONTECSI), 5., São Paulo, 2008 (a).

FUGITA, H. S.; HIRAMA, K. Method for Service-Oriented Analysis and Design. In Proceedings of the International Conference on Software Engineering Research and Practice (SERP), Las Vegas, 2008. Las Vegas: CSREA Press, 2008 (b).

GORNIK, D. *IBM Rational Unified Process: Best Practices for Software Development Teams*. IBM Whitepapers, 2003. [acesso em 27 de maio de 2007]. Disponível em: http://ftp.software.ibm.com/software/rational/web/whitepapers/2003/rup_bestpractice_s.pdf

HUHNS, M. N.; SINGH, M. P. Service-Oriented Computing: Key Concepts and Principles, *IEEE Internet Computing*, v. 9, no. 1, pp. 75-81, jan-fev. 2005.

IBM. *Rational Method Composer*, version 7.2. IBM Rational, 2007.

JESTON, J.; NELIS, J. *Business Process Management: Practical Guidelines to Successful Implementations* (2nd Edition). Estados Unidos: Butterworth-Heinemann, 2008. 504 p. ISBN: 0750686561.

KEEN, M. et al. *Patterns: SOA with an Enterprise Service Bus*. Estados Unidos: IBM Redbooks, 2005. 386 p. ISBN: 073849058X.

KENNEY, L. F.; PLUMMER, D. C. *Magic Quadrant for Integrated SOA Governance Technology Sets, 2007*. Gartner RAS Core Research Note. Gartner, 2007.

KRAFZIG, D.; BANKE, K.; SLAMA, D. *Enterprise SOA*. Estados Unidos: Prentice Hall, 2004. 408 p. ISBN: 0131465759.

KRUTCHEN, P. *The Rational Unified Process: An Introduction* (2nd Edition). Boston, Estados Unidos: Addison Wesley, 2000. 272 p. ISBN: 0201707101.

MARKS, E. A.; BELL, M. *Service Oriented Architecture: A Planning and Implementation Guide for Business and Technology*. Hoboken, Estados Unidos: John Wiley & Sons, 2006. 384 p. ISBN: 0471768944.

OASIS SOA REFERENCE MODEL TC. *Reference Model for Service Oriented Architecture 1.0*. OASIS Open, 2006. 31 p.

OASIS WSBPEL TC. *Web Services Business Process Execution Language*, version 2.0. OASIS Open, 2007. [acesso em 1 de dezembro de 2008]. Disponível em: <<http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>>

THE OPEN GROUP. *Definition of SOA*, version 1.1. The Open Group, 2006. [acesso em 21 de agosto de 2007]. Disponível em: <<http://opengroup.org/projects/soa/doc.tpl?gdid=10632>>

PAPAZOGLU, M. P. Service-Oriented Computing: Concepts, Characteristics and Directions. In *Proceedings of the International Conference on Web Information Systems Engineering (WISE)*, 4., Roma, 2003. Roma: IEEE Computer Society, 2003. pp. 3-12.

PAPAZOGLU, M. P.; VAN DEN HEUVEL, W. J. Service-Oriented Design and Development Methodology. *International Journal of Web Engineering and Technology (IJWET)*, v. 2, no. 4, pp. 412-442, 2006.

RAMOLLARI, E.; DRANIDIS, D.; SIMONS, A. J. H. A Survey of Service Oriented Development Methodologies. In *Proceedings of the European Young Researchers Workshop on Service Oriented Computing (YR-SOC)*, 2., Leicester, 2007. pp. 75-80.

RUMBAUGH, J.; JACOBSON, I.; BOOCH, G. *The Unified Modeling Language Reference Manual* (2nd Edition). Estados Unidos: Addison Wesley, 2004. 752 p. ISBN: 0-32124562-8.

PERREY, R.; LYCETT, M. Service-Oriented Architecture. In *Proceedings of the Applications and the Internet Workshops*, Uxbridge, 2003. pp. 116-119.

SHUJA, A. K.; KREBS, J. *IBM Rational Unified Process Reference and Certification Guide: Solution Designer*. Estados Unidos: IBM Press, 2007. 307 p. ISBN: 0131562924.

ZIMMERMANN, O. et al. *Analysis and Design Techniques for Service-Oriented Development and Integration*. Stuttgart: IBM Press, 2005.