

UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
ESCOLA DE INFORMÁTICA APLICADA
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Identificação automática de serviços a partir de processos de negócio em BPMN

Nome das autoras:

Bruna Christina Pinto Brandão

Juliana Carvalho Silva

Nome do Orientador:

Leonardo Guerreiro Azevedo

Março/2013

Identificação automática de serviços a partir de processos de negócio em BPMN

Projeto de Graduação apresentado à
Escola de Informática Aplicada da
Universidade Federal do Estado do Rio
de Janeiro (UNIRIO) para obtenção do
título de Bacharel em Sistemas de
Informação.

Nome das Autoras:

Bruna Christina Pinto Brandão

Juliana Carvalho Silva

Nome do Professor:

Leonardo Guerreiro Azevedo

Identificação automática de serviços a
partir de processos de negócio em
BPMN

Aprovado em ____/____/____

BANCA EXAMINADORA

Prof. Leonardo Guerreiro Azevedo, D.Sc. (UNIRIO)

Prof. Márcio de Oliveira Barros, D.Sc. (UNIRIO)

Prof. Gleison dos Santos Souza, D.Sc. (UNIRIO)

Henrique Prado Sousa, M.Sc. (PUC-Rio)

O autor deste Projeto autoriza a ESCOLA DE INFORMÁTICA APLICADA da UNIRIO a divulgá-lo, no todo ou em parte, resguardando os direitos autorais conforme legislação vigente.

Rio de Janeiro, ____ de ____ de ____

Bruna Christina Pinto Brandão

Juliana Carvalho Silva

Agradecimentos

Aos nossos pais, que, com muito carinho e apoio, não mediram esforços para que nós chegássemos até esta etapa de nossas vidas.

As nossas irmãs, que foram fundamentais em nossa formação pessoal e profissional e que sempre estiveram conosco nos incentivando e apoiando.

Ao nosso professor e orientador Leonardo Azevedo, pelo convívio, pelo apoio, pela compreensão e pela paciência na orientação e incentivo que tornaram possível a conclusão desta monografia.

A todo o corpo docente da UNIRIO, pelo carinho, dedicação e qualidade de ensino demonstrado ao longo do curso.

Aos nossos amigos de faculdade, que foram essenciais nesses anos de graduação, nos auxiliando, apoiando, rindo e nos divertindo. Temos certeza que os levaremos conosco em toda a nossa jornada.

E, finalmente agradecemos a todos aqueles que de alguma forma estiveram e estão próximos de nós, nos incentivando a sempre a conquistar os nossos sonhos.

Sumário

Capítulo 1: Introdução.....	12
1.1 Motivação	12
1.2 Objetivo	14
1.3 Estrutura do texto	14
Capítulo 2: Principais Conceitos.....	15
2.1 Modelagem de Processos	15
2.1.1 Definição	15
2.1.2 BPMN.....	16
2.1.3 EPC.....	18
2.2 SOA	20
2.2.1 Definição	20
2.2.2 Serviços	22
2.3 Relação de SOA com BPM	23
2.4 Arquivo JSON	24
2.5 RPST	25
Capítulo 3: Identificação automática de serviços	29
3.1 Identificação de serviços	29
3.2 Heurísticas para a identificação de Serviços Candidatos	31
3.2.1 Heurística de identificação de serviços candidatos a partir de Regras de Negócio	31
3.2.2 Heurística de identificação de serviços candidatos a partir de Requisitos de Negócio.....	33
3.2.3 Heurística de identificação de serviços candidatos a partir de Cluster	34
3.2.4 Heurística de identificação de serviços candidatos a partir de atividades sequenciais	35
3.2.5 Heurística de identificação de serviços candidatos a partir de <i>workflow</i> AND	37
3.2.6 Heurística de identificação de serviços candidatos a partir de <i>workflow</i> XOR	38
3.2.7 Heurística de identificação de serviços candidatos a partir de <i>workflow</i> OR	39
3.2.8 Heurística de identificação de serviços candidatos a partir de <i>LOOP</i> de atividades	40
3.2.9 Heurística de identificação de serviços candidatos a partir de interface de processo	41
3.2.10 Heurística de identificação de serviços candidatos a partir de atividades de múltipla instância.....	42
3.3 Heurísticas de Consolidação	43
3.3.1 Heurística de consolidação de eliminação de serviços candidatos	43
3.3.2 Heurística de associação de serviços com papéis	44

3.3.3	Heurística de associação de serviços com atividades	45
3.3.4	Heurística de consolidação de serviços identificados a partir de fluxo	45
Capítulo 4: Automatização da Identificação de Serviços em BPMN.....		47
4.1	Ferramenta de Modelagem de Processos - Signavio.....	47
4.2	Estrutura do Projeto	48
4.3	Implementação da Proposta.....	53
4.3.1	Identificação de workflows	53
4.3.2	Heurísticas de Identificação de serviços candidatos	54
4.3.3	Heurística de consolidação de serviços	58
Capítulo 5: Avaliação da Proposta.....		60
5.1	– Processo utilizado	60
5.2	– Produtos gerados da identificação automática de serviços.....	61
5.3	– Diferenças encontradas entre EPC e BPMN	64
Capítulo 6: Conclusão		66
Referências		68

LISTA DE FIGURAS

Figura 1 – Exemplo de processo modelado em BPMN [VALLE <i>et al.</i> , 2012].....	17
Figura 2 – Exemplo de processo modelado em EPC	19
Figura 3 – Exemplo de diagrama VAC [SOUSA <i>et al.</i> , 2011]	19
Figura 4 – Exemplo em EPC de cluster.	20
Figura 5 – Exemplo para json	25
Figura 6 – Exemplo do código json	25
Figura 7 – Exemplo de processo abstrato (adaptado de POLYVYANNY <i>et al.</i> [2009]).	27
Figura 8 – RPST do processo.....	27
Figura 9 – Exemplo em EPC de regra de negócio.	32
Figura 10 – Exemplo em BPMN de regra de negócio.	32
Figura 11 – Exemplo em EPC de requisito de negócio.	33
Figura 12 – Exemplo em EPC de <i>cluster</i>	34
Figura 13 – Exemplo em BPMN de <i>cluster</i>	35
Figura 14 – Exemplo em EPC de atividades sequenciais.	36
Figura 15 – Exemplo em BPMN de atividades sequenciais	36
Figura 16 – Exemplo em EPC de <i>workflow</i> AND com fluxos paralelos sincronizados	38
Figura 17 – Exemplo em BPMN de <i>workflow</i> AND com fluxos paralelos sincronizados	38
Figura 18 – Exemplo em EPC do <i>workflow</i> XOR	39
Figura 19 – Exemplo em BPMN do <i>workflow</i> XOR	39
Figura 20 – Exemplo em EPC do <i>workflow</i> OR	40
Figura 21 – Exemplo em BPMN do <i>workflow</i> OR	40
Figura 22 – Exemplo em EPC de <i>LOOP</i> de atividades	41
Figura 23 – Exemplo em BPMN de <i>LOOP</i> de atividades	41
Figura 24 – Exemplo em EPC de interface de processo	42
Figura 25 – Exemplo em EPC de atividade de múltipla instância	43
Figura 26 – Exemplo em BPMN de atividade de múltipla instância.....	43
Figura 27 – Modelagem de serviço candidato com perfil correspondente	44
Figura 28 – Serviço de Loop com Subfluxos.....	46
Figura 29 – Diagrama de Pacotes	49
Figura 30 – Diagrama de Atividades	51
Figura 31 – Exemplo real de processo estruturado em fragmentos de RPST (adaptado de POLYVYANNY <i>et al.</i> [2009]).....	71
Figura 32 – Processo Analisar de Pedido de Crédito em BPMN [Adaptação de SOUSA, 2012].	72

LISTA DE ALGORITMOS

Algoritmo 1 – Identificação dos serviços a partir das heurísticas de <i>workflow</i>	54
Algoritmo 2 – Identificação do serviço a partir de Regra de Negócio.....	55
Algoritmo 3 – Identificação do serviço a partir de <i>Cluster</i>	55
Algoritmo 4 – Identificação do serviço a partir de Atividades Sequenciais	55
Algoritmo 5 – Identificação de atividades sequenciais em função recursiva	56
Algoritmo 6 – Identificação do serviço a partir de <i>Workflow Loop</i>	56
Algoritmo 7 – Identificação se o nó é loop.....	56
Algoritmo 8 – Identificação do serviço a partir de <i>Workflow AND</i>	57
Algoritmo 9 – Identificação se o nó possui AND.....	57
Algoritmo 10 – Identificação do serviço a partir de <i>Workflow XOR</i>	57
Algoritmo 11 – Identificação se o nó possui XOR	57
Algoritmo 12 – Identificação do serviço a partir de <i>Workflow OR</i>	57
Algoritmo 13 – Identificação do serviço a partir de <i>Workflow OR</i>	58
Algoritmo 14 – Identificação do serviço a partir de Múltipla Instancia	58
Algoritmo 15 – Eliminação de serviços.....	58
Algoritmo 16 – Associação serviços x atividades	59
Algoritmo 17 – Associação serviços x papéis	59
Algoritmo 18 – Contar número de raias.....	59
Algoritmo 19 – Contar número de subfluxos.....	59

LISTA DE TABELAS

Tabela 1 – Serviço consolidado – Eliminação de serviços	44
Tabela 2 – <i>Template</i> dos serviços criados.....	53
Tabela 3 – Tabela de serviço de cluster	61
Tabela 4 – Tabelas de informações de fluxos (flows).....	62
Tabela 5 – Tabela de Associação de Serviços x Atividades	63
Tabela 6 – Tabela de Associação de Serviços x Perfis	63
Tabela 7 – Serviços identificados na heurística de Regras de Negócio	73
Tabela 8 – Serviços identificados na heurística de Múltipla Instância	73
Tabela 9 – Serviços identificados na heurística de <i>Cluster</i>	77
Tabela 10 – Serviços identificados na heurística de Atividades Sequenciais	77
Tabela 11 – Serviços identificados na heurística de <i>LOOP</i>	78
Tabela 12 – Serviços identificados na heurística de XOR	79
Tabela 13 – Relatório de <i>Flows</i>	79
Tabela 14 – Relatório Serviços x Perfil	80
Tabela 15 – Relatório Serviços x Atividades Parte I	82
Tabela 16 – Relatório Serviços x Atividades Parte II	83
Tabela 17 – Relatório Serviços x Atividades Parte III.....	84
Tabela 18 – Relatório Serviços x Atividades Parte IV	85

LISTA DE ABREVIATURAS

BPMI – *Business Process Management Initiative*

BPMN – *Business Process Modeling Notation*

CORBA – *Common Object Request Broker Architecture*

EPC – *Event-driven Process Chain*

IDEF – *Integrated Definition*

JSON – *JavaScript Object Notation*

REST – *Representational State Transfer*

RPST – *Refined Process Structure Tree*

SGBD – Sistema de Gerenciamento de Banco de Dados

SOA – *Service Oriented Architecture*

SOAP – *Simple Object Access Protocol*

TI – Tecnologia da Informação

UDDI – *Universal Description, Discovery and Integration*

XML – *Extensible Markup Language*

WSDL – *Web Service Description Language*

RESUMO

O uso da arquitetura orientada a serviços (SOA) cresce cada vez mais nas organizações. Uma modelagem de processo de negócio bem definida e detalhada auxilia a sua utilização e a identificação de serviços nesta arquitetura. Este trabalho descreve e implementa uma proposta para identificar automaticamente serviços candidatos a partir de um processo modelado utilizando a notação BPMN. A automatização teve como base trabalhos anteriores. As heurísticas de identificação e consolidação de serviços candidatos propostas por Azevedo *et al.* [2009a, 2009b] foram empregadas para construção dos algoritmos. Além disso, também foi base para este trabalho a automatização realizada por Sousa *et al.* [2011] e Azevedo *et al.* [2009c] para identificação de serviços candidatos a partir de processos modelados utilizando a notação EPC dentro da plataforma ARIS. Este trabalho estendeu os trabalhos anteriores com a automatização de heurísticas que não haviam sido automatizadas anteriormente para identificação de serviços compostos a partir dos padrões de workflow AND, XOR e OR. A identificação de serviços compostos empregou a árvore RPST como base para a automatização das heurísticas correspondentes. A proposta foi avaliada em um modelo de processo fictício de análise de pedido de crédito. Os resultados demonstraram a utilidade da proposta na identificação de serviços candidatos a partir de modelos de processos em BPMN.

Palavras-chave: SOA, Identificação automática de serviços, BPMN, modelagem de processos.

Capítulo 1: Introdução

O objetivo deste capítulo é contextualizar o assunto tratado neste trabalho, assim como apresentar a sua motivação, o seu objetivo e a estrutura de capítulos.

1.1 Motivação

Com os novos requisitos do mercado, buscando cada vez mais a eficácia e a eficiência, surgiu a necessidade das empresas implementarem um novo método de fabricação de software para responder a demanda do mercado de forma efetiva e rápida [JOSUTTIS, 2007].

Uma das arquiteturas de software que as empresas estão adotando atualmente denomina-se SOA (*Service-Oriented Architecture* ou Arquitetura Orientada a Serviços). Existem diferentes definições para SOA. Josuttis [2007] define SOA como um paradigma para a realização e manutenção de processos de negócio em um grande ambiente de sistemas distribuídos que são controlados por diferentes proprietários. Já MARKS e BELL [2006] apresentam que SOA é uma arquitetura conceitual onde a funcionalidade do negócio ou a lógica da aplicação é disponibilizada para usuários SOA, ou consumidores, como serviços compartilhados e reutilizáveis em uma rede de TI.

SOA apresenta várias vantagens para as empresas que adotam essa arquitetura de software. Algumas dessas vantagens são as reduções de custos, de riscos, do tempo de desenvolvimento, reutilização de códigos e a possibilidade de um melhor alinhamento com o negócio, ou seja, a equipe de negócio pode visualizar como a empresa é construída em termos de tecnologia [JOSUTTIS, 2007].

SOA tem como princípio fundamental implementar as funcionalidades das aplicações como serviços. Serviços são pedaços de funcionalidades que possuem interfaces expostas que são invocados via mensagens [MARKS e BELL, 2006]. Ou seja, serviços são pedaços de software construídos de forma que possam ser facilmente vinculados a outros componentes de software. O conceito de serviços engloba a ideia de que é possível definir partes dos códigos em porções significativas o suficiente para serem compartilhadas e reutilizadas em diversas áreas da empresa. A forma mais utilizada para a distribuição de serviços é via *web services* [ERL,

2005]. No entanto, existem outras tecnologias para implementar serviços, tais como, CORBA (*Common Object Request Broker Architecture*) [ERL, 2005] e REST (*Representational State Transfer*) [RICHARDSON e RUBY, 2007].

MARKS e BELL [2006] apresentam que a identificação de serviços pode ser feita de várias formas, como entrevistas com especialistas do negócio, a partir de funcionalidades já implementadas nas aplicações, análise das entidades principais do negócio, a partir de serviços pré-existentes e a partir de modelos de processos de negócio. Porém vários autores apontam que a melhor forma para a identificação de serviços é a partir de modelos de processo, como enfatizado por Azevedo *et al.* [2009a, 2009b].

A identificação de serviços deve ser feita a partir de um processo bem definido, sistematizado e detalhado, não importando a sua notação para a implementação dos serviços [JOSUTTIS, 2007]. A modelagem de processos de negócio é a atividade de representação dos processos de uma empresa, de modo que permite que o processo atual seja analisado e melhorado. Normalmente a modelagem de processos é realizada por analistas de negócios e gestores que estão buscando melhorar a eficiência do processo e a sua qualidade.

A modelagem é uma etapa importante, pois é onde os processos são descobertos e detalhados. É também nessa etapa que podem ser feitas algumas alterações no processo visando à sua otimização [VALLE *et al.*, 2012]. BPMN (*Business Process Modeling Notation*) é uma notação para modelagem de processo. BPMN foi desenvolvida pela *Business Process Management Initiative* (BPMI) sendo atualmente mantida pelo *Object Management Group* (OMG). A BPMN foi apoiada por várias empresas mundialmente conhecidas tornando-se a notação mais utilizada [VALLE *et al.*, 2012].

O uso de SOA está cada vez mais comum e frequente nas empresas devido aos benefícios de sua utilização resultando no aumento da demanda da identificação de serviços. Sendo assim, para organizações orientadas a processos, ou seja, que possuem seus processos desenhados empregando notação de modelo de processos como a BPMN, torna-se cada vez mais necessária a identificação de serviços a partir dos modelos de processos de negócio de forma sistemática, como a proposta de

AZEVEDO *et al.* [2009a, 2009b]. SOUSA *et al.* [2011] e [AZEVEDO, 2009c] implementaram a identificação automática de serviços usando a notação EPC. Como BPMN é a notação mais utilizada nas empresas atualmente, a identificação automática de serviços utilizando essa notação na modelagem dos processos apresenta-se como um fator preponderante para a implantação de SOA.

1.2 Objetivo

Este trabalho tem como objetivo a automatização da heurística de identificação de serviços candidatos a partir de modelos de processo de negócio em BPMN (*Business Process Modeling Notation*) propostas por AZEVEDO *et al.*, [2009a, 2009b] e implementadas por SOUSA *et al.* [2011] e AZEVEDO *et al.* [2009c] para modelos de processos desenhados com EPC (*Event-driven Process Chain*).

Dessa forma, para execução deste trabalho, as heurísticas propostas por AZEVEDO *et al.* [2009a, 2009b] foram analisadas comparando as notações EPC e BPMN. Em seguida, algoritmos foram definidos a fim de tratar as especificidades da notação e tendo como base as implementações realizadas anteriormente por SOUSA *et al.* [2011] e AZEVEDO *et al.* [2009c].

1.3 Estrutura do texto

Este trabalho está dividido da seguinte forma. O Capítulo 1 corresponde a presente introdução. O Capítulo 2 apresenta definições e conceitos necessários para o entendimento sobre modelagem de processo, SOA, BPMN, EPC, JSON e RPST. O Capítulo 3 apresenta as heurísticas utilizadas para a identificação e consolidação de serviços candidatos. O Capítulo 4 apresenta a automatização da identificação de serviços candidatos. O Capítulo 5 apresenta a avaliação da proposta e o Capítulo 6 apresenta a conclusão do trabalho.

Capítulo 2: Principais Conceitos

Este capítulo tem como objetivo apresentar os principais conceitos e definições de modelagem de processos de negócio e da arquitetura orientada a serviços (SOA).

2.1 Modelagem de Processos

Esta seção apresenta os principais conceitos empregados neste trabalho sobre modelagem de processos e descreve a notação BPMN (*Business Process Modeling Notation*) e EPC (*Event-driven Process Chain*).

2.1.1 Definição

Segundo Davenport (1994, p. 6) “Um processo é simplesmente um conjunto de atividades estruturadas e medidas, destinadas a resultar num produto especificado para um determinado cliente ou mercado. Um processo é, portanto, uma ordenação específica das atividades de trabalho no tempo e no espaço, com um começo, um fim e entradas e saídas claramente identificadas: uma estrutura para a ação.” [DAVENPORT, 1994].

Um processo de negócio é descrito por um ou mais procedimentos que, em conjunto, realizam um objetivo de negócio. A execução de um processo de negócio possui condições muito bem definidas de início e término, e pode combinar procedimentos automáticos e manuais [WfM, 1999].

Empresas orientadas a processos de negócios apresentam uma estrutura horizontal, onde sua característica principal é o foco no cliente, operando sob uma estrutura matricial, onde os gerentes hierárquicos são substituídos pelos *process owners*, estes operam com autonomia e responsabilidade por todo o processo, independente da estrutura hierárquica. A organização horizontal, orientada a processos de negócios, torna a operação mais flexível, centrada nos propósitos da organização e com maior proximidade do consumidor final [OSTROFF, 1999]. Por outro lado, organizações verticais são aquelas estruturadas por funções, onde a pirâmide hierárquica é a sua grande característica. Níveis sobrepostos e superpostos

de decisões são comuns em organizações desse gênero, criando atritos frequentes com clientes finais e fornecedores.

A modelagem de processos de negócio é o conjunto de práticas ou tarefas que as empresas podem executar para descrever visualmente todos os aspectos de um processo de negócio, incluindo o seu curso, controle e pontos de decisão, gatilhos e condições para execução das atividades, o contexto em que uma atividade executa e os recursos associados [JOSUTTIS, 2007 *apud* BLOOMERGSCHMELZER, 2006].

Um modelo é apenas uma representação do processo que permite às empresas documentar, simular, compartilhar, implementar, avaliar e continuamente melhorar suas operações [JOSUTTIS, 2007 *apud* BLOOMERGSCHMELZER, 2006].

As atividades de análise e modelagem de processos podem ser realizadas utilizando ferramentas disponíveis no mercado. Existem cerca de 300 softwares que oferecem uma variedade de características conforme o produto escolhido [OLIVEIRA *et al.*, 2006].

Exemplos de ferramenta são: Aris (<http://www.softwareag.com/corporate/default.asp>), BizAgi (<http://www.bizagi.com/>), Bonita (<http://www.bonitasoft.com/>) e Signavio (<http://signavio.com/>).

Dentre as notações mais difundidas atualmente para modelagem de processos, estão a BPMN (*Business Process Modeling Notation*), UML (*Unified Modeling Language*), IDEF (*Integrated Definition*) e EPC (*Event-driven Process Chain*). Após a escolha da notação e da ferramenta, deve-se identificar o processo, ou processos, que deseja modelar, realizando um levantamento detalhado do processo para descobrir o fluxo de trabalho, quem inicia o processo, quem faz a próxima atividade, entre outros detalhes [VALLE *et al.*, 2012].

2.1.2 BPMN

BPMN (*Business Process Modeling Notation*) é uma notação padrão para modelagem de processo de negócio e é o resultado de um acordo entre diversas empresas de ferramentas de modelagem de processos, com o objetivo de criar uma

linguagem única e padrão, visando facilitar o entendimento e treinamento do usuário final [VALLE *et al.*, 2012].

Esta notação foi criada pela BPMI (*Business Process Management Initiative*) que hoje está integrada ao *Object Management Group* (OMG). As duas organizações se fundiram no ano de 2005. Atualmente, mais precisamente em Janeiro de 2011, foi lançada a versão 2.0 da BPMN [VALLE *et al.*, 2012].

Em BPMN, pode-se construir três tipos de diagramas: diagramas de colaboração, diagramas de processo e diagramas de coreografia. Este trabalho foca em diagramas de processo. O que torna a notação BPMN uma das mais completas e promissoras atualmente são os elementos utilizados na modelagem de processos de negócio, os quais são: atividades, eventos, *gateways* (decisões) e sequência de fluxos (*Sequence Flows*) ou rotas.

A modelagem é uma etapa importante da automação porque é nesta etapa que os processos são descobertos e desenhados. É nesta etapa também que pode ser feita alguma modificação no “caminho” do processo buscando aperfeiçoá-lo.

A Figura 1 apresenta um exemplo de processo - Venda de Mercadoria - modelado em BPMN. Neste processo, inicialmente é executada a atividade de identificar a forma de pagamento da mercadoria, posteriormente é preciso receber o pagamento, que será dinheiro/cheque ou cartão de crédito. E por último, após receber o pagamento, é necessário preparar a embalagem da mercadoria.

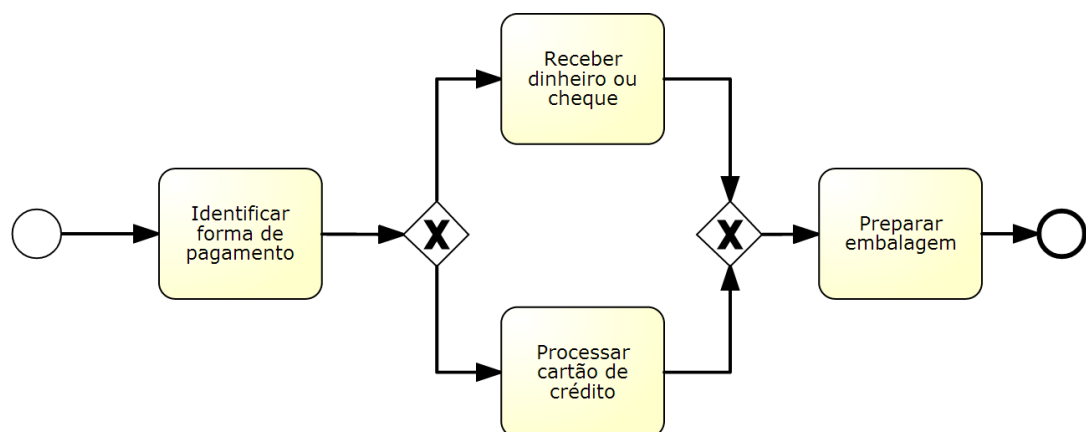


Figura 1 – Exemplo de processo modelado em BPMN [VALLE *et al.*, 2012]

2.1.3 EPC

EPC (*Event-Driven Process Chain*) é uma linguagem de modelagem que é utilizada para descrever processos de negócios e fluxos de trabalho [EPC NOTATION, 2011]. EPC foi desenvolvido em 1992 em um projeto de pesquisa e desenvolvimento do *Institute for Information Systems da Universidade de Saarland* na Alemanha [SCHEER *et al.*, 2005].

Seu foco é a modelagem de processos baseada no controle de fluxos de atividades e eventos, assim como as suas relações de dependência. Esta é uma das notações mais difundidas para modelagem e muito conhecida por ser utilizada na ferramenta ARIS (<http://www.ariscommunity.com/>). As ferramentas VISIO da Microsoft e EPC-Tools também oferecem a notação EPC [VALLE *et al.*, 2012].

EPC tem uma estrutura de “evento-atividade-evento”, ou seja, os processos modelados na notação EPC devem começar e terminar com eventos. Assim, um evento leva ao início da atividade e, após a sua execução, a atividade gera um novo evento [VALLE *et al.*, 2012]. Também há o modelo e-EPC (*Extended Event-driven Process Chain*), que estende o modelo EPC unindo objetos das visões organizacional, de dados e funcional.

A principal desvantagem da notação EPC, além da criação desnecessária de eventos que pode atrapalhar o entendimento de processos grandes, é que a notação não possui uma organização específica responsável pela padronização da notação [VALLE *et al.*, 2012].

As principais vantagens da notação é que ela permite mapear processos complexos, permite o uso de elementos de diferentes visões (organizacional, funcional e de dados). A ferramenta ARIS possibilita exportar o processo para várias extensões de arquivos. Mesmo a notação EPC não tendo uma organização responsável pela sua padronização, por ser o elemento principal da plataforma ARIS, a notação obteve grande sucesso [VALLE *et al.*, 2012].

Na notação EPC, o processo é modelado através do diagrama EPC, outros diagramas são o diagrama FAD (*Function Allocation Diagram*) e o diagrama VAC (*Valued Added Chain*). O diagrama EPC é o fluxo principal do processo da notação,

é onde são modeladas as atividades, os eventos e os gateways, com os respectivos perfis responsáveis pelas atividades. A Figura 2 representa o mesmo processo de Venda de Mercadoria, exemplificado e explicado detalhadamente na Figura 1, porém modelado em EPC.

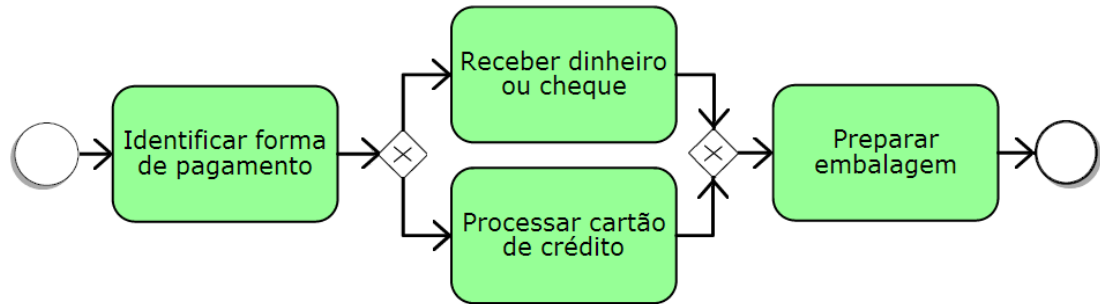


Figura 2 – Exemplo de processo modelado em EPC

O VAC é o diagrama para representar o processo no nível mais alto, ou seja, é utilizado para representar os macroprocessos da organização. Já o FAD é o diagrama para representar os detalhes operacionais de uma determinada atividade.

O diagrama VAC pode ser visto na Figura 3, no qual está modelado o macroprocesso “Realizar empréstimos para pessoas físicas” composto pelos macroprocessos “Gerir solicitações de pedido de crédito”, “Analisar pedido de crédito” e “Gerenciar contratos de crédito em vigor”.

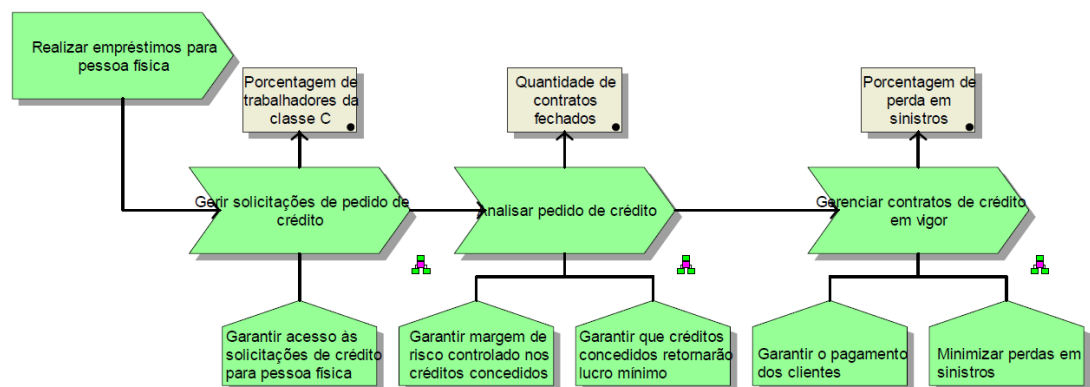


Figura 3 – Exemplo de diagrama VAC [SOUSA *et al.*, 2011]

uma rede com o objetivo de concluir uma tarefa em comum. Assim a arquitetura SOA estabelece uma comunicação entre os sistemas clientes (os que requerem os serviços, os consumidores) e os sistemas que implementam os serviços (provedores de serviços) [JOSUTTIS, 2007].

Porém, nem sempre seu uso é indicado, a empresa que deseja implementar uma arquitetura orientada a serviços deve verificar se possui alguns requisitos para utilizar eficientemente SOA. Um dos requisitos para um bom uso de SOA é a interoperabilidade entre diferentes sistemas e linguagens de programação. Ou seja, o uso de SOA é indicado, por exemplo, quando for necessária a integração de diferentes aplicações com diferentes plataformas.

Ao decidir se a abordagem SOA irá realmente ser necessária para a organização, a sua implementação traz vários benefícios, como a reutilização de códigos, assim aumentando a produtividade. SOA também aumenta a agilidade mesmo sem a reutilização dos serviços porque facilita a modificação de uma parte do sistema sem prejudicar e parar todo o acesso ao resto do sistema. Por exemplo, ao modificar um método que calcule a taxa de juros de um processo de imobiliária, SOA permite que a invocação do método modificado seja feito após ter sido terminado. Ou seja, os sistemas que invocam esse método irão continuar a invocar o método antigo até o término da modificação do método. Isso permite que os sistemas não sejam dependentes do outro, não comprometendo o funcionamento dos sistemas que invocam o método [SOUSA *et al.*, 2011 *apud* MARKS e BELL, 2006].

Um exemplo do uso da arquitetura orientada a serviços provendo uma maior agilidade mesmo sem a reutilização de serviços ocorreu na empresa ProFlowers.com. Na empresa não existem aplicativos redundantes ou várias unidades de negócio precisando de serviços. Porém foi identificada uma oportunidade de aplicar os conceitos de SOA com a divisão do processo de pedido de flores em pequenos serviços, onde cada componente pode ser isolado e modificado conforme a necessidade de lidar com os picos da demanda, como acontece em datas festivas, onde existe uma demanda muito maior de pedidos de flores sendo realizados. Quando a empresa possuía apenas um sistema integrado encarregado do processo, uma única alteração no processo ou um crescimento no volume das transações exigia

que o sistema inteiro fosse reformulado. Com SOA foi possível agregar valor à organização, facilitando a modificação de seu sistema [KOCH, 2006].

De acordo com o Kevin Hall, o superintendente de TI da empresa, com o novo sistema, os servidores reagem aos picos das atividades durante cada fase do processo de pedido, transferindo capacidade para o serviço específico. O sistema utilizando a abordagem SOA está mais previsível e não houve interrupções desde sua implementação. Desta maneira trazendo vários benefícios para a empresa, como a confiança dos clientes sabendo que o sistema não irá ficar inoperante em datas que a demanda será grande, e também trazendo lucro para a empresa, pois suas vendas não serão interrompidas [KOCH, 2006].

2.2.2 Serviços

Serviços são partes ou funções inteiras de um sistema que podem ser disponibilizadas para outro sistema. Os serviços devem funcionar de maneira independente de outros serviços, exceto quando houver serviços compostos, e devem possuir uma interface bem definida. A comunicação entre o sistema que requer o serviço (consumidor do serviço) e o sistema que disponibiliza o serviço (provedor de serviço) é realizada, principalmente, pela implementação dos serviços como web services. Há outras formas de implementar os serviços, porém a mais utilizada e vantajosa é web services [JOSUTTIS, 2007; ERL, 2005].

Os serviços são representações lógicas de uma atividade no processo de negócio que pode ser mapeada em entrada, processamento e saída. Eles devem estar alinhados ao negócio, atendendo a necessidade representada no processo, e funcionar de forma independentemente, ou seja, fornecer os mesmos resultados para as mesmas entradas. Os serviços permitem composições, ou seja, um serviço pode ser composto por outros serviços, como consultar o cadastro de cliente, e registrar o cadastro de cliente. Os dois serviços poderiam ser disponibilizados como um único serviço que os compõem, porém é importante que seja garantida a consistência das informações. Resumindo, os serviços devem garantir o seu reuso, a interoperabilidade e integração através dos processos de negócio e das plataformas tecnológicas [JOSUTTIS, 2007].

O provedor do serviço se compromete em realizar determinada tarefa com resultados já pré-estabelecidos, e o consumidor se compromete em usar o serviço da

forma acordada anteriormente. Cada serviço possui esse tipo de relacionamento entre o provedor do serviço e o consumidor do serviço o qual se denomina ‘contrato’. O contrato possui informações sobre determinado serviço, como especificação de atributos não funcionais, tempo de execução, desempenho e confiabilidade. Os serviços são invocados via mensagens [JOSUTTIS, 2007].

Para serviços implementados com a tecnologia de web services temos os seguintes padrões. SOAP (*Simple Object Access Protocol*) é um padrão que define o protocolo do web services. SOAP usa XML para trocar dados dos serviços em uma conexão via internet. Os serviços também utilizam o WDSL (*Web Services Description Language*). WSDL é usado para descrever os serviços, a sua assinatura (nome e parâmetros do serviço) e os detalhes para acesso (detalhes e localização). E o outro padrão fundamental para a implementação dos serviços via web services é o UDDI (*Universal Description, Discovery and Integration*). UDDI é padrão de gerenciamento de *web services*, ou seja, é onde os serviços são registrados e onde os serviços são procurados para serem usados [JOSUTTIS, 2007].

2.3 Relação de SOA com BPM

JOSUTTIS [2007] enfatiza que é necessário ter um bom entendimento dos processos de negócio para a identificação de serviços. Ele afirma que, em SOA, serviços são partes de um ou mais processos de negócios distribuídos. Esta afirmação é considerada a principal motivação para serviços serem identificados a partir de processos de negócio.

Na gestão de processos de negócio as atividades a serem desempenhadas pela organização são modeladas em processos de negócio, os quais são divididos em partes menores, como outras atividades e tarefas. Essas partes menores, quando podem ser executadas automaticamente ou apoiadas por sistemas, podem ser disponibilizadas como serviços em SOA [JOSUTTIS, 2007].

Em AZEVEDO *et al.* [2009a, 2009b] foram criadas heurísticas para identificação de serviços a partir de um modelos de processos de negócio. Porém, a execução destas heurísticas manualmente consumia muito tempo, levando a automatizar o processo, como proposto por AZEVEDO *et al.* [2009c] e SOUSA *et al.* [2011].

BECKER *et al.* [2011] apresenta que em SOA, os processos de negócio são apoiados por sistemas de informação, cujo sistemas de softwares são formados por serviços. Muitos trabalhos caracterizam processos de negócio como sendo sistemas intensivos, ou seja, são sistemas automatizados com pouca intervenção humana, e que os serviços são derivados a partir destes processos. Além disso, como uma mesma atividade pode aparecer em mais de um processo na organização, sendo implementada ou apoiada por vários sistemas de informação em diferentes áreas ou departamentos, a disponibilização de um serviço empregando as características apresentadas em uma SOA resulta em ganhos quanto a reuso, flexibilidade e agilidade do negócio.

SCHEER *et al.* [2005] afirma que há uma visão integrada dos processos de negócio da organização, onde cada processo é derivado de cadeias de valor da organização. Nesta visão integrada, as relações entre os processos e elementos de um mesmo processo são explícitas. Os processos e seus elementos são parte de um repositório comum, onde são associados uns aos outros por relações. Essas relações são chamadas de elementos da interface pela metodologia de BPM [SCHEER *et al.* 2005]. Elas têm em comum a representação de elementos globais, como regras de negócio, atividades de múltipla instância¹, informações de entrada e saída. Este repositório comum de elementos inter-relacionados deve orientar o desenvolvimento de serviços de ciclo de vida em uma abordagem SOA, a fim de alcançar os benefícios da arquitetura.

2.4 Arquivo JSON

Esta seção explica conceitos do arquivo no formato JSON. Este tipo de arquivo foi utilizado no projeto para exportar um arquivo de modelo de processo em BPMN, da ferramenta da modelagem de processos Signavio, para ser carregado dentro da ferramenta de automatização da identificação de serviços que foi implementada.

JSON (*Java Script Object Notation*) é um formato de arquivo para troca de dados leve. Sua sintaxe é considerada fácil para os seres humanos lerem e

¹ Múltipla Instância - representa uma ação executada repetidas vezes, que possui várias instâncias geradas em paralelo ou sequencialmente [OMG, 2011].

escreverem e também é considerada fácil para máquinas para analisarem e gerarem [ECMA, 2011].

Enquanto a maioria dos navegadores e ferramentas conseguem construir, enviar e analisar XML, JSON fornece um formato de troca de dados padronizado que é mais adequado para aplicações Ajax em estilo web. JSON pode ser utilizado em praticamente qualquer cenário onde as aplicações precisam trocar ou armazenar informações como texto estruturado [AZIZ, 2007].

JSON está constituído em uma coleção de pares nome/valor. Em várias linguagens isto é caracterizado como um *object*, *record*, *struct*, dicionário, *hash table*, *keyed list*, ou *arrays* associativos ou uma lista ordenada de valores [ECMA, 2011].

A Figura 6 representa o arquivo JSON correspondente à parte do processo modelado na Figura 5, o qual corresponde à atividade “Comunicar proposta não aprovada” executada pelo papel “Atendente”.

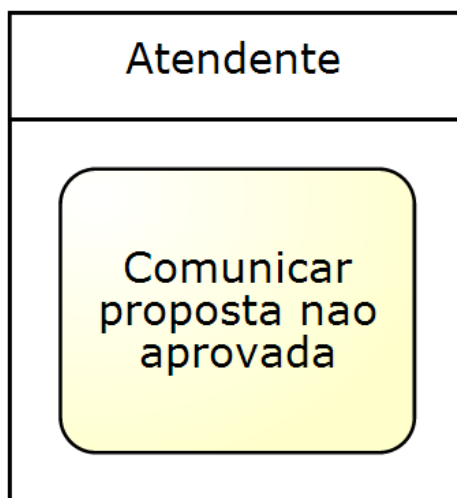


Figura 5 – Exemplo para json

```
[{"resourceId": "sid-D7DD418B-8147-44BF-9841-10F7E47D915B",
  "properties": {"type": "http://b3mn.org/stencilset/bpmn2.0#Pool",
  "properties": "", "name": "Atendente",
  "documentation": "", "status": "None",
  "processtype": "None",
  "bgcolor": "#ffffff",
  "processid": "sid-71F58409-C297-4004-9968-C11F417E0F18",
  "stencil": {"id": "VerticalPool"},
  "childShapes": [{"resourceId": "sid-DFBBF19C-5C66-4F30-927F-AC401E73E718",
  "childShapes": [{"resourceId": "sid-3AB62C14-A63E-4C9F-A6CC-7F5C35B4ACFE",
  "properties": {"processid": "",
  "name": "Comunicar proposta nao aprovada",
  "documentation": "",
  "tasktype": "None",
  "implementation": "webService",
  "looptype": "None",}]}
```

Figura 6 – Exemplo do código json

2.5 RPST

Processos de negócio são normalmente representados em grafos direcionados. RPST (*Refined Process Structure Tree*) é uma técnica de análise de fluxos. RPST

resulta uma árvore hierárquica de subgrafos do modelo de processo de negócio, possibilitando uma navegação mais simples para acessar partes do modelo para executar algoritmos [VANHATALO *et al.* 2009].

RPST é baseado no fato de que cada grafo pode ser decomposto em uma hierarquia de subgrafos independentes de forma lógica, contendo uma entrada e uma saída. A hierarquia é mostrada como uma árvore onde a raiz é a própria árvore, e as folhas são os que os desenvolvedores da técnica RPST chamam de fragmentos [LEOPOLD *et al.*, 2012].

VANHATALO *et al.* [2009] classificam esses fragmentos como triviais (T), *bonds* (B), polígonos (P) e rígidos (R). Fragmentos triviais correspondem a dois nós conectados por um único arco. Os *bonds* são conjuntos de fragmentos que compartilham dois nós em comum. Em processos em BPMN, os fragmentos *bonds* normalmente são resultantes dos *gateways* de bifurcação e de junção. Polígonos correspondem a sequências de outros fragmentos. Fragmentos que não se encontram nas classificações dos fragmentos triviais, *bonds* ou polígonos são classificados como rígidos [POLYVYANNY *et al.*, 2009].

A Figura 7 exemplifica um processo abstrato, onde o fragmento *a* é um fragmento trivial. Este fragmento conecta os nós correspondentes ao evento inicial e ao vértice seguinte a ele. Os fragmentos B1 e B2 são *bonds*. E os nós P1, P2, P3, P4, P5, P6 e P7 são fragmentos classificados como polígonos. No Anexo I, Figura 31 é possível ver este mesmo exemplo em um processo que representa um cenário próximo da realidade.

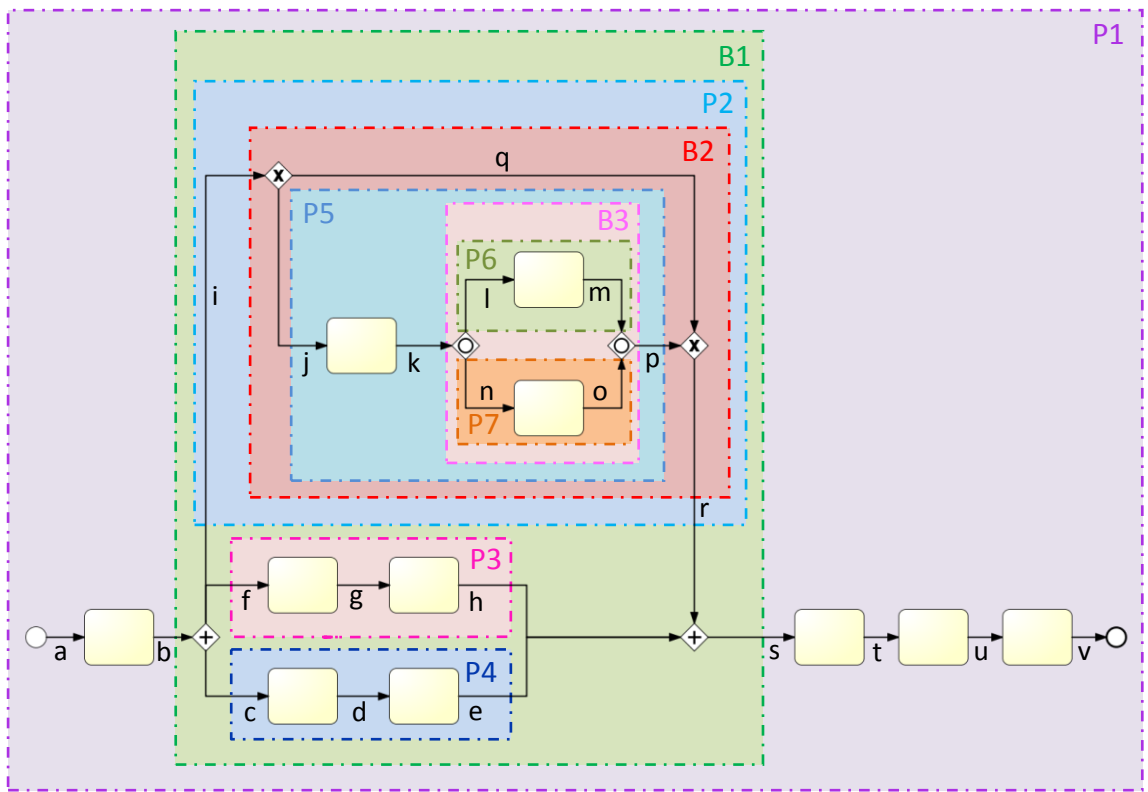


Figura 7 – Exemplo de processo abstrato (adaptado de POLYVYANNY *et al.* [2009]).

A Figura 8 exemplifica a árvore RPST correspondente ao processo da Figura 7.

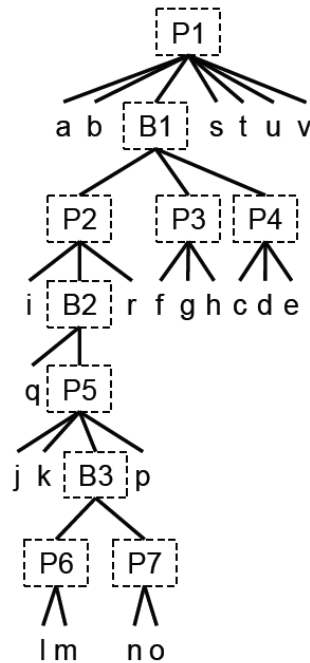


Figura 8 – RPST do processo.

O algoritmo RPST original foi elaborado para fluxos que tenham apenas uma entrada e uma saída. Porém, Artem Polyvyanny, Jussi Vanhatalo, e Hagen Völzer

evoluíram o RPST para retirar essa restrição. Assim o RPST pôde ser aplicado em grafos direcionais que possuem mais de um evento inicial e/ou evento final desde que cada um dos nós esteja em um caminho ligando uma fonte (evento inicial) ao uma saída (evento final) [POLYVYANNY *et al.*, 2009].

Neste trabalho, a utilização do RPST foi necessária para a identificação dos serviços provenientes das heurísticas de fluxo.

Capítulo 3: Identificação automática de serviços

Este capítulo tem como objetivo contextualizar como ocorre a identificação de serviços, as heurísticas utilizadas para a identificação e as diferenças entre a identificação de serviços com a modelagem de dados feita em EPC e em BPMN.

3.1 Identificação de serviços

Primeiramente, para ser possível a identificação automática de serviços, é necessário definir um conceito fundamental para tal implementação: o conceito de serviço candidato.

“Serviço candidato é uma abstração (não implementada) de serviço a qual, durante a fase de projeto em um modelo de ciclo de vida de serviço, pode ser escolhida para ser implementada como um serviço ou como uma funcionalidade de uma aplicação.” [ERL, 2005].

Existem duas abordagens de identificação de serviços, a *top-down* e a *bottom-up*. A primeira abordagem também é conhecida como decomposição de processos, pois consiste em dividir um processo em partes. Já a abordagem *bottom-up* consiste na composição de serviços existentes visando construir um processo de negócio [JOSUTTIS, 2007].

Porém, utilizar uma abordagem puramente *top-down* ou puramente *bottom-up* não é aconselhável, pois pode-se chegar a implementações muito complicadas, não correspondentes com a realidade. Na abordagem *top-down* fica claro quais serviços são necessários e onde fazer as separações das atividades. Porém, pode não considerar características técnicas, levando a serviços difíceis de serem implementados na prática. O ponto negativo da abordagem *bottom-up* é a proposta de detalhes técnicos e restrições em alto nível do processo, que pode resultar em um processo inflexível e não condizente com a realidade da organização. Por estes motivos, considera-se a melhor abordagem mesclar as duas já citadas [JOSUTTIS, 2007].

Serviços candidatos também podem ser identificados através de aplicações atuais do negócio, análise das entidades principais do negócio, via iniciativas

orçadas, conhecimento especializado do negócio, conhecimento de serviços pré-existent e análise dos processos de negócio [MARKS e BELL, 2006]. A identificação de serviços candidatos através de aplicações existentes e sistemas legados se deve pela disponibilização de funcionalidades do negócio por serviços, seja para replicação de funcionalidades ou visando facilitar o reuso e tornar processos mais flexíveis [JOSUTTIS, 2007].

Para uma organização que já possui modelagem de processos implementada, a identificação de serviços candidatos pode ser realizada da seguinte maneira: analisar a cadeia de valores da organização, mapear os processos da organização em alto nível e realizar a identificação de serviços candidatos partindo do mapeamento de processos [JOSUTTIS, 2007].

Os serviços podem ser identificados através do fluxo de processos (estruturas do fluxo e padrões que se repetem com frequência no processo) e das atividades (informações de entrada e saída, regra de negócio e requisitos do negócio) [AZEVEDO, 2009a].

Os serviços candidatos podem ser classificados em três tipos distintos: serviços candidatos de dados, serviços candidatos de lógica e serviços candidatos utilitários. Os serviços candidatos de dados realizam operações CRUD. Os serviços candidatos de lógica implementam operações que tratam regras de negócios. E os serviços candidatos utilitários representam padrões baseados no aspecto da estrutura organizacional e em funções recorrentes em fluxos de processos de negócios [JOSUTTIS, 2007].

O método de identificação de serviços se resume a três etapas: (i) seleção de atividades; (ii) identificação e classificação de serviços candidatos; (iii) consolidação de serviços candidatos [SOUSA *et al.*, 2011 apud AZEVEDO, 2009a].

Na etapa de "Seleção de atividades" identificam-se as atividades que são manuais, automáticas, automatizáveis e apoiadas por sistemas. Serviços candidatos são identificados a partir de elementos envolvendo os três últimos tipos de atividades [AZEVEDO, 2009a]. Na segunda etapa utilizam-se heurísticas para identificação dos serviços candidatos. Na terceira etapa, são utilizadas heurísticas para consolidar informações sobre os serviços, como, por exemplo, eliminar serviços iguais ou para

juntar serviços que contenham funções semelhantes. As heurísticas de identificação e consolidação são o escopo deste trabalho e são detalhadas a seguir.

3.2 Heurísticas para a identificação de Serviços Candidatos

Esta seção descreve as seguintes heurísticas para identificação de serviços candidatos: heurística de regra de negócio, heurística de requisito de negócio, heurística de informações de entrada e saída, heurística de workflow de atividades sequenciais, heurística de workflow AND, heurística de workflow XOR, heurística de workflow OR, heurística de workflow LOOP, heurística de interface de processo e heurística de múltipla instância. Vale ressaltar que a heurística de entrada e saída corresponde a uma evolução dos trabalhos de Azevedo *et al.* [2009a, 2009b].

A fim de explicitar as diferenças entre diagramas BPMN e EPC, a seguir são apresentadas a definição de cada heurística e a diferença entre diagramas elaborados empregando estas notações.

3.2.1 Heurística de identificação de serviços candidatos a partir de Regras de Negócio

Heurística: Toda regra de negócio deve ser identificada como um serviço candidato.

A Figura 9 apresenta um exemplo de modelo FAD. Na notação EPC é neste modelo que as regras de negócio são modeladas. Neste exemplo, as regras de negócio correspondem aos elementos "Cadastro de cliente desatualizado" e "Cliente novo" os quais correspondem às regras de negócio referentes à identificação de que o cadastro de um cliente está desatualizado e a de um cliente novo, respectivamente. A identificação de serviços candidatos, neste caso, corresponde a identificar estes elementos no FAD e a gerar os serviços candidatos para os mesmos, ou seja, foram identificados dois serviços candidatos, um para cada regra de negócio "Cadastro de cliente desatualizado" e "Cliente novo".

3.2.2 Heurística de identificação de serviços candidatos a partir de Requisitos de Negócio

Heurística: Todo requisito de negócio deve ser considerado um serviço candidato.

A Figura 11 apresenta um exemplo de modelo FAD. Na notação EPC é neste modelo que os requisitos de negócio são modelados. Neste exemplo, os requisitos de negócio correspondem aos elementos “Aprovar créditos concedidos” e “Consultar créditos concedidos”, os quais se referem aos requisitos de negócio de aprovação e consulta de créditos concedidos. A identificação de serviços candidatos identifica estes elementos no FAD e gera os serviços candidatos para os mesmos, ou seja, foram identificados dois serviços candidatos, um para cada requisito de negócio “Aprovar créditos concedidos” e “Consultar créditos concedidos”.

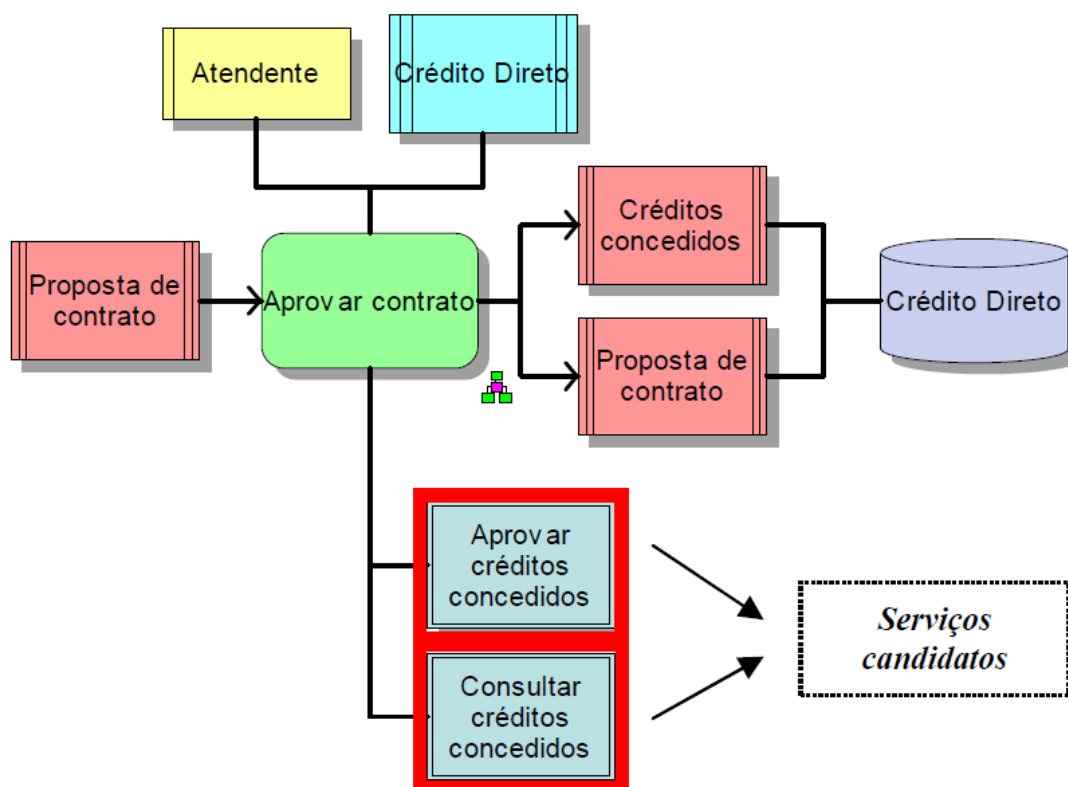


Figura 11 – Exemplo em EPC de requisito de negócio.

Em BPMN não existe símbolo para representar requisito de negócio [PAVLOVSKI *et al.*, 2008]. Logo, esta heurística não foi tratada neste trabalho.

3.2.3 Heurística de identificação de serviços candidatos a partir de Cluster

Heurística: Toda informação de entrada ou saída de uma atividade associada a um portador de informação deve ser considerada como um serviço candidato.

A Figura 12 apresenta um exemplo de modelo FAD. Na notação EPC é neste modelo que informações de entrada e saída (chamadas de *clusters* na notação) e portadores de informação são modelados. Neste exemplo, os *clusters* associados a portadores de informação correspondem aos elementos “Taxa de juros” e “Taxas de juros do cliente” os quais correspondem aos clusters referentes aos dados de taxas de juros e taxas de juros do cliente, respectivamente. Estes elementos estão associados ao portador de informação “Crédito Direto”. A identificação de serviços candidatos corresponde a identificar estes elementos no FAD e a gerar os serviços candidatos. Neste caso, foram identificados dois serviços candidatos, um para cada *cluster* “Taxa de juros” e “Taxas de juros do cliente”.

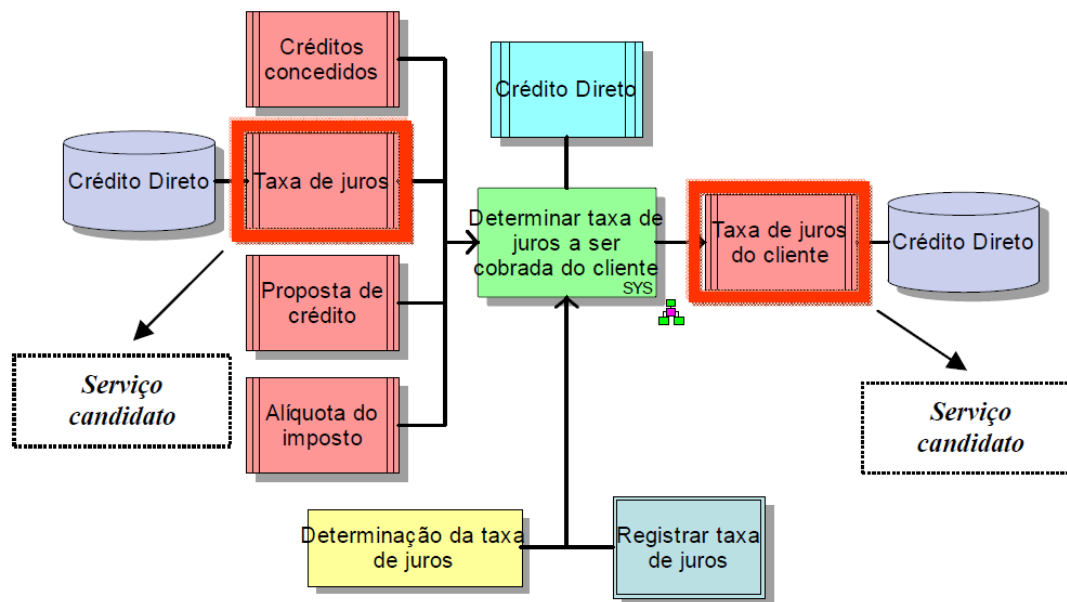




Figura 12 – Exemplo em EPC de *cluster*.

Em BPMN as informações de entrada e saída são modeladas no diagrama de processo e podem ser identificadas nas informações de entrada e saídas das atividades associadas a um portador de informação (*data store*), representado pelo

símbolo . As informações de entradas e de saídas das atividades são representadas em BPMN pelo objeto de dados (*data object*), cujo símbolo é .

A modelagem do exemplo apresentado na Figura 12 empregando notação BPMN é ilustrada na Figura 13. Neste caso, os *clusters* “Taxa de juros” e “Taxas de juros do cliente” são explícitos no modelo de forma semelhante à representada no FAD, mas são apresentadas no diagrama de processo. Porém, existem outros serviços identificados a partir de *clusters* na figura abaixo.

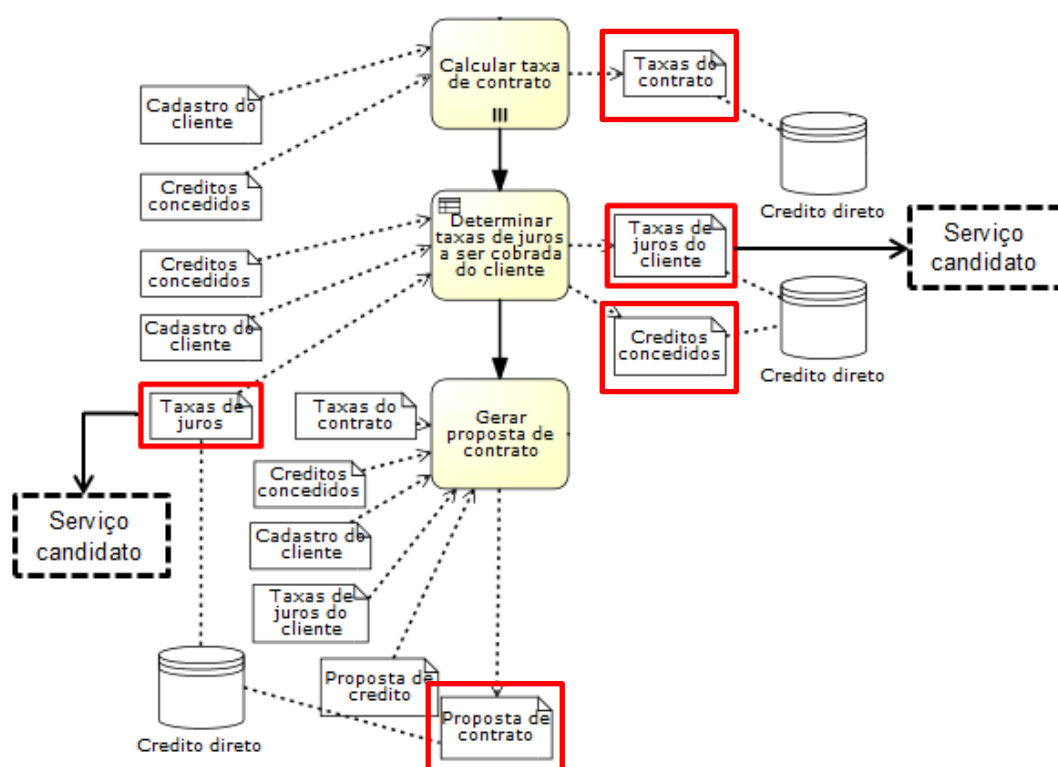


Figura 13 – Exemplo em BPMN de *cluster*.

3.2.4 Heurística de identificação de serviços candidatos a partir de atividades sequenciais

Heurística: Toda sequência de duas ou mais atividades identificadas no processo deve ser considerada um serviço candidato.

A Figura 14 apresenta um exemplo de modelo EPC. Nesta notação é no modelo principal que as atividades sequenciais são representadas. No exemplo

apresentado, as atividades sequenciais correspondem aos elementos “Comprometer limite de crédito”, “Calcular alíquota de imposto”, “Determinar taxa de juros a ser cobrada do cliente”, “Gerar proposta de contrato” e “Analisar contrato”. A identificação de serviços candidatos corresponde a identificar uma sequência de atividades no modelo EPC e a gerar um serviço candidato para cada sequência existente, ou seja, no exemplo foi identificado um serviço candidato, pois existe apenas uma sequência de atividades.

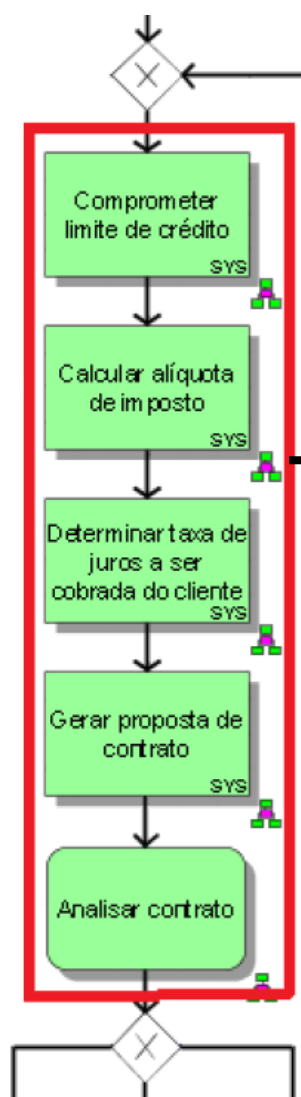


Figura 14 – Exemplo em EPC de atividades sequenciais.

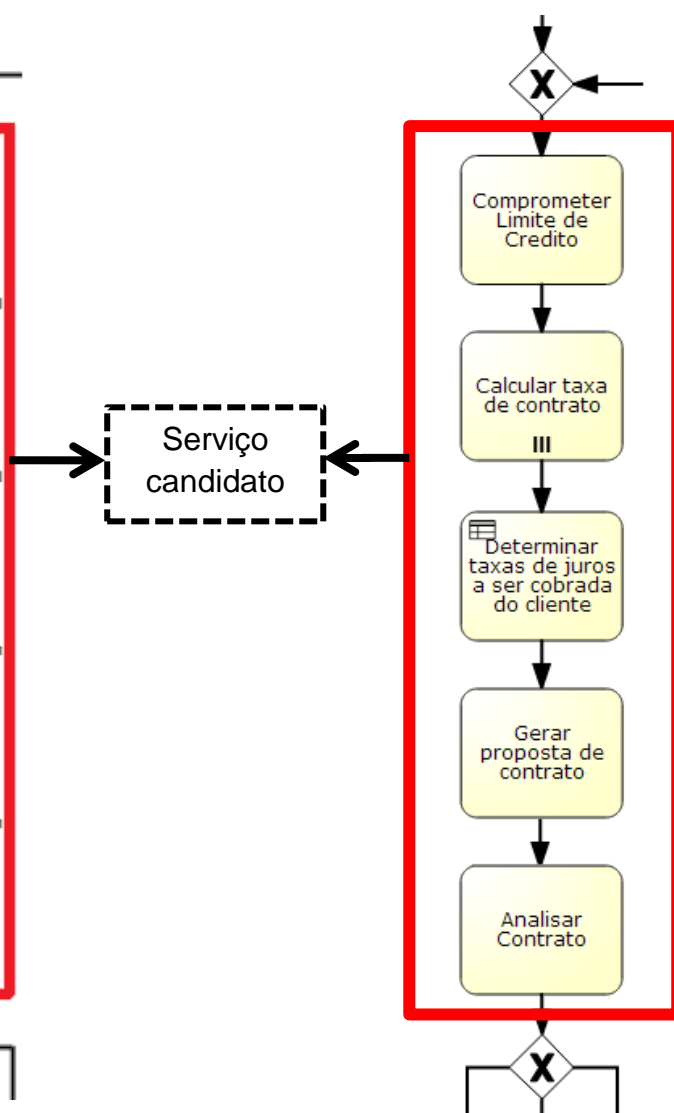




Figura 15 – Exemplo em BPMN de atividades sequenciais

Em BPMN as atividades sequenciais são modeladas no diagrama de processo e podem ser identificadas quando houver duas ou mais atividades representadas em sequência.

A modelagem do exemplo apresentado na Figura 14, empregando notação BPMN, é ilustrada na Figura 15. Neste caso, as atividades sequenciais são explícitas no modelo de processo como na notação EPC.

3.2.5 Heurística de identificação de serviços candidatos a partir de *workflow* AND

Heurística: Todo fluxo paralelo deve ser considerado como um serviço candidato, incluindo o seu início na bifurcação até a sua junção ou término em evento(s) final(is).

A Figura 16 apresenta um exemplo na notação EPC. Neste exemplo, o *workflow* AND corresponde a todos os elementos existentes entre o *gateway* AND  de abertura e o de fechamento. Estes *gateways* são responsáveis por controlar a execução das atividades em paralelo. O fechamento deste fluxo também poderia ter ocorrido através de um evento final . A Figura 17 corresponde ao mesmo serviço na notação BPMN. A identificação de serviços candidatos corresponde a identificar estes elementos de *gateway* AND de abertura e o seu fechamento ou evento final, e gerar o serviço candidato composto pelo conjunto de atividades que se encontram dentre estes elementos. No exemplo foi identificado um serviço candidato para todo o fluxo paralelo.

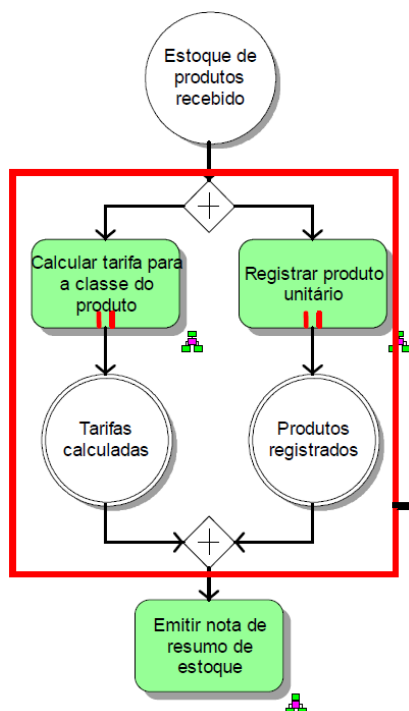


Figura 16 – Exemplo em EPC de *workflow* AND com fluxos paralelos sincronizados

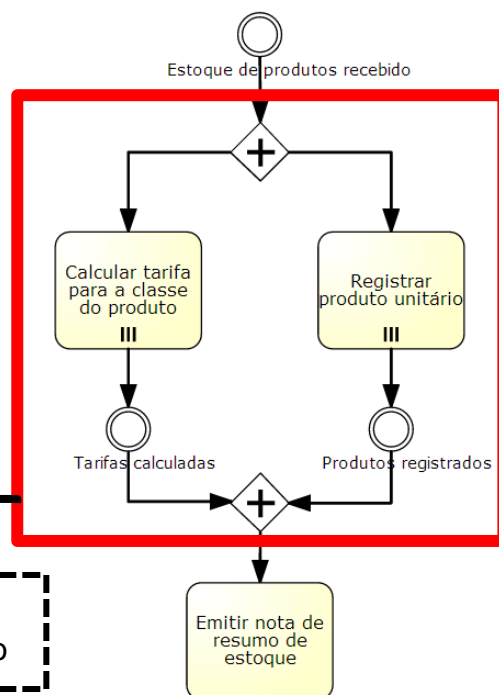


Figura 17 – Exemplo em BPMN de *workflow* AND com fluxos paralelos sincronizados

3.2.6 Heurística de identificação de serviços candidatos a partir de *workflow* XOR

Heurística: Todo fluxo de ou-exclusivo (XOR) identificado no processo deve ser considerado como um serviço candidato incluindo o seu início na bifurcação até a sua junção ou seu término em evento(s) final(is).

A Figura 18 apresenta um exemplo na notação EPC. Neste exemplo, o *workflow* XOR corresponde a todos os elementos existentes entre o *gateway* XOR de abertura e o de fechamento. Estes *gateways* são responsáveis por controlar a execução das atividades de forma exclusiva. O fechamento deste fluxo também poderia ter ocorrido através de um ou mais eventos finais . A Figura 19 corresponde ao mesmo serviço na notação BPMN. A identificação de serviços candidatos corresponde a identificar este elemento de *gateway* XOR de abertura e o seu fechamento ou eventos finais e gerar serviço candidato para o conjunto de

atividades entre estes elementos. No exemplo foi identificado um serviço candidato com todo o fluxo XOR.

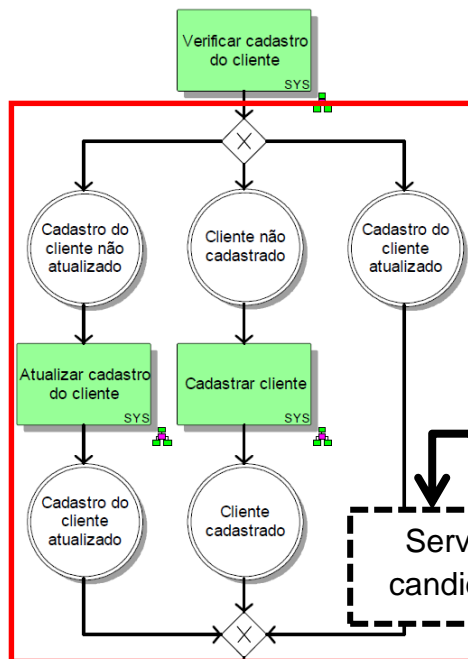


Figura 18 – Exemplo em EPC do *workflow* XOR

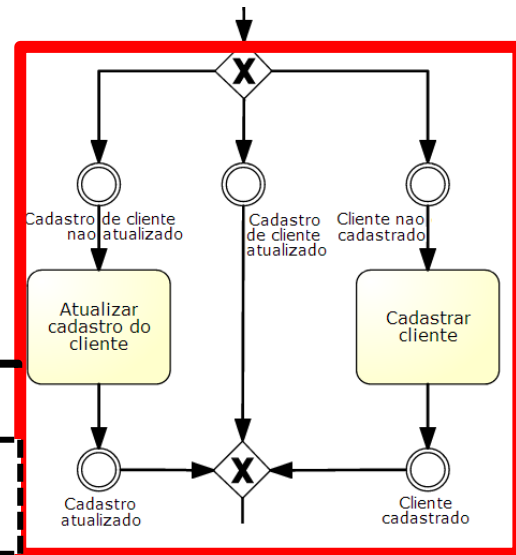




Figura 19 – Exemplo em BPMN do *workflow* XOR

3.2.7 Heurística de identificação de serviços candidatos a partir de *workflow* OR

Heurística: Todo fluxo OR deve ser considerado como um serviço candidato, incluindo o seu início na bifurcação até a sua junção ou seu término em evento(s) final(is).

A Figura 20 apresenta um exemplo na notação EPC. Neste exemplo, o *workflow* OR corresponde a todos os elementos existentes entre o *gateway* OR  de abertura e o de fechamento. Estes *gateways* são responsáveis por controlar a execução das atividades que podem ser executadas ao mesmo tempo. O fechamento deste fluxo também poderia ter ocorrido através de um evento final . A Figura 21 corresponde ao mesmo serviço na notação BPMN. A identificação de serviços candidatos corresponde a identificar este elemento de *gateway* OR de abertura e o seu fechamento ou eventos finais e a gerar serviço candidato para as atividades entre

estes elementos. No exemplo foi identificado um serviço candidato com todo o fluxo OR.

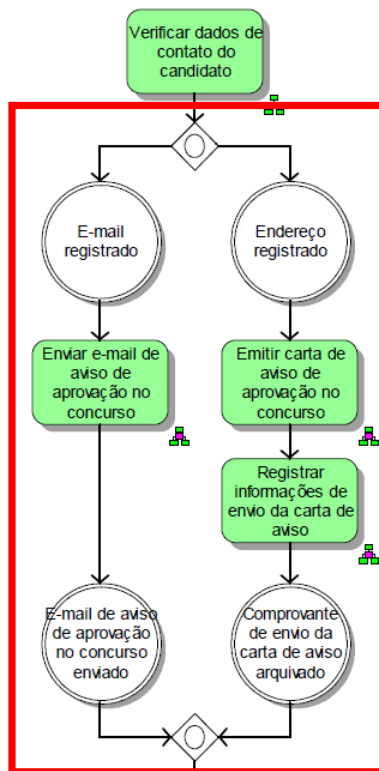


Figura 20 – Exemplo em EPC do *workflow* OR

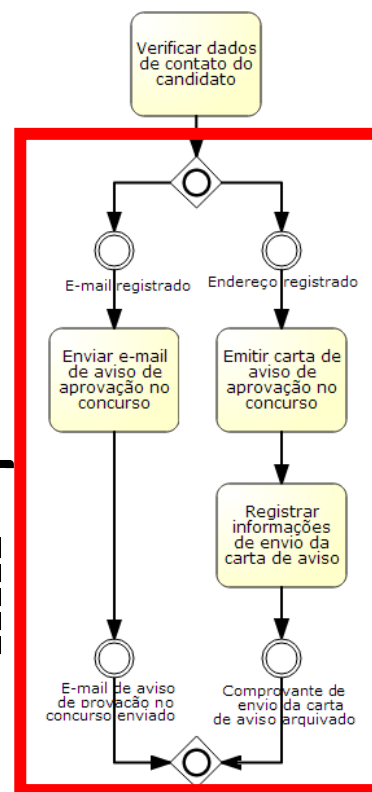


Figura 21 – Exemplo em BPMN do *workflow* OR

3.2.8 Heurística de identificação de serviços candidatos a partir de *LOOP* de atividades

Heurística: Toda estrutura de *workflow* onde duas ou mais atividades são executadas repetidas vezes deve ser considerada um serviço candidato.

A Figura 22 apresenta um exemplo de modelo EPC onde um *loop* de atividades é modelado. Neste exemplo, o *loop* de atividades corresponde às atividades que repetem no fluxo devido ao retorno de uma atividade para o início da sequência. A identificação de serviços candidatos corresponde a identificar a repetição de atividades no processo e a gerar o serviço candidato para as mesmas, ou seja, foi identificado um serviço candidato de *loop* de atividades.



Figura 22 – Exemplo em EPC de *LOOP* de atividades

Serviço candidato

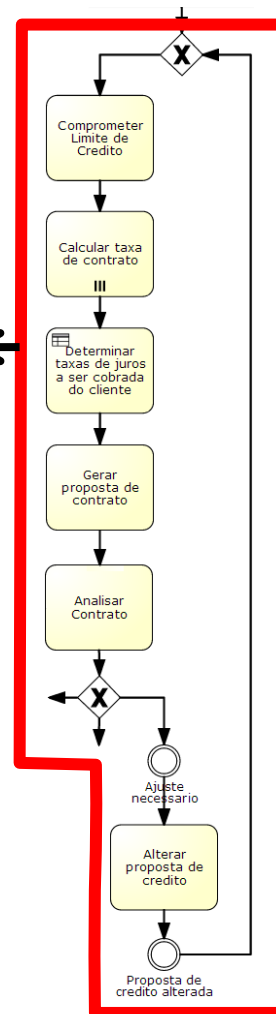


Figura 23 – Exemplo em BPMN de *LOOP* de atividades

Na notação BPMN o *loop* de atividades é modelado no diagrama de processo de forma semelhante a da notação EPC. A modelagem do exemplo apresentado na Figura 22 empregando notação BPMN é ilustrada na Figura 23.

3.2.9 Heurística de identificação de serviços candidatos a partir de interface de processo

Heurística: Toda interface de processo precedente e antecedente de uma atividade automatizada deve ser considerada como um serviço candidato.

A Figura 24 apresenta um exemplo de modelo EPC. Neste exemplo, as interfaces de processo correspondem aos elementos “Analisar pedido de crédito” e “receber proposta de crédito”. A identificação de serviços candidatos corresponde a identificar estes elementos no modelo EPC, verificar se os mesmos são precedentes ou antecedentes de uma atividade automatizada e a gerar os serviços candidatos para os mesmos, ou seja, foi identificado um serviço candidato para cada interface de processo.

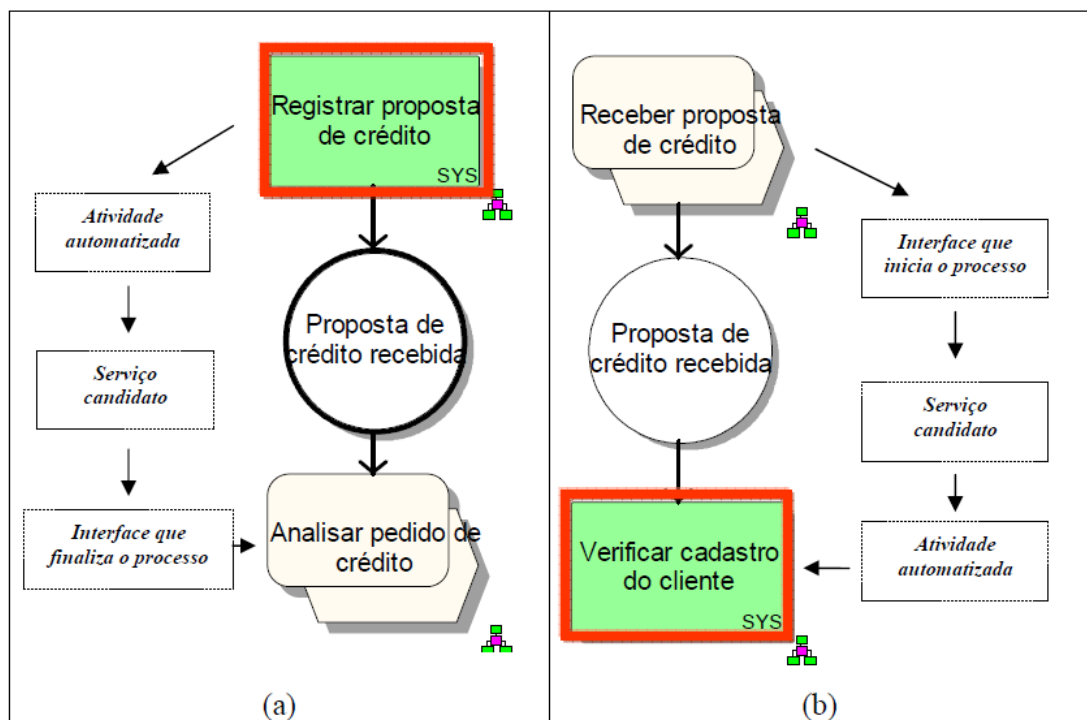


Figura 24 – Exemplo em EPC de interface de processo


Em BPMN não existe elemento para representar interfaces de processo [OMG, 2011].

3.2.10 Heurística de identificação de serviços candidatos a partir de atividades de múltipla instância

Heurística: Toda atividade de múltipla instância deve ser considerada um serviço candidato.

Foi verificado que toda atividade de múltipla instância é considerada um serviço candidato, pois representa uma ação executada repetidas vezes, que possui várias instâncias geradas em paralelo ou sequencialmente [OMG, 2011].

A Figura 25 apresenta um exemplo de modelo EPC. Neste exemplo, a atividade de múltipla instância corresponde ao elemento "Calcular taxa de contrato". A identificação de serviços candidatos corresponde a identificar este elemento no fluxo principal do modelo EPC e a gerar o serviço candidato para o mesmo, ou seja, foi identificado um serviço candidato para a atividade de múltipla instância "Calcular taxa de contrato".

Em BPMN a atividade de múltipla instancia é modelada no diagrama de processo através de atividade marcada com o símbolo  [OMG, 2011]. A modelagem do exemplo apresentado na Figura 25 empregando notação BPMN é ilustrada na Figura 26.

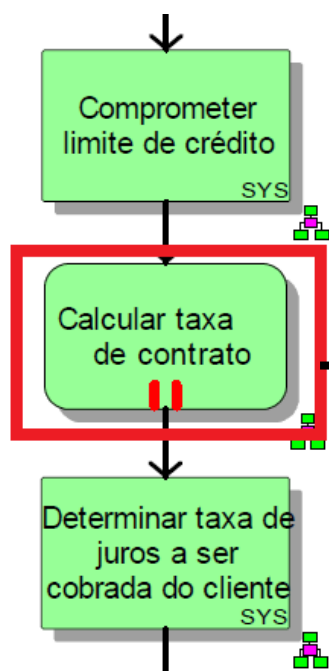


Figura 25 – Exemplo em EPC de atividade de múltipla instância

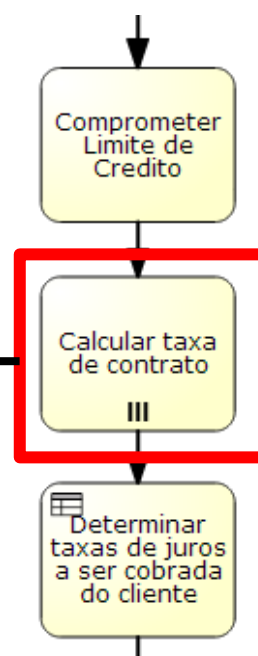


Figura 26 – Exemplo em BPMN de atividade de múltipla instância

3.3 Heurísticas de Consolidação

Nesta seção são apresentadas as heurísticas utilizadas para a consolidação de serviços candidatos.

3.3.1 Heurística de consolidação de eliminação de serviços candidatos

Heurística: Deve ser mantido apenas um serviço quando existirem serviços candidatos duplicados.

Esta heurística de consolidação é utilizada quando existem dois ou mais serviços com nomes iguais e mesma origem. Na Tabela 1 é apresentado o serviço consolidado “Obter Créditos Concedidos”, o qual poderia ter sido consolidado a partir do serviço "Obter créditos concedidos" e "Recuperar créditos", por exemplo.

<u>Informações</u>	<u>Serviços</u>
Nome	Obter creditos concedidos
Tipo	Dados (leitura)
Entrada	Creditos concedidos
Saída	
Origem	Cluster
Atividades	[Comprometer Limite de Credito, Cancelar contrato, Determinar taxas de juros a ser cobrada do cliente, Cancelar contrato de risco]
Descrição	A partir dos dados creditos concedidos deve-se obter creditos concedidos.
Perfil	[Credito Direto, Atendente, Analista de Credito]

Tabela 1 – Serviço consolidado – Eliminação de serviços

3.3.2 Heurística de associação de serviços com papéis

Heurística: Todo serviço candidato deve ser associado aos papéis (raias) em que suas atividades estão localizadas.

Na Figura 27 pode-se observar o serviço candidato identificado da regra de negócio “Verificar cadastro do cliente”, que está localizado na raia Crédito Direto. Logo, este serviço deve estar associado ao perfil de Crédito Direto.

Esta heurística é essencial para o gerenciamento e restrição de acesso as funcionalidades do processo.

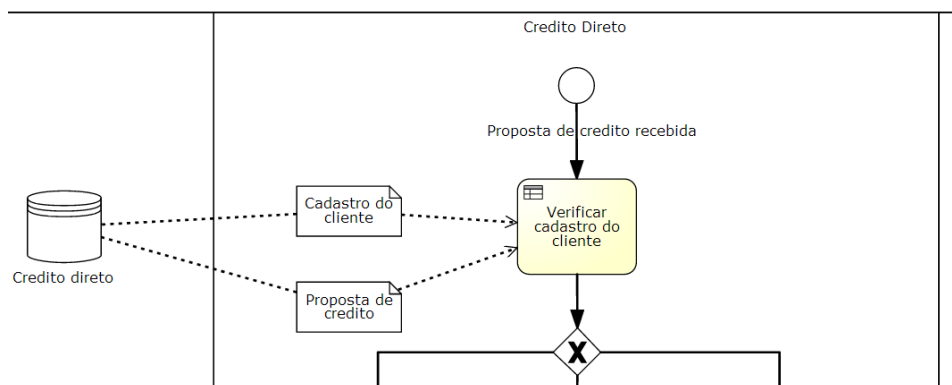


Figura 27 – Modelagem de serviço candidato com perfil correspondente

3.3.3 Heurística de associação de serviços com atividades

Heurística: Todo serviço candidato deve ser associado às atividades a partir do qual foi identificado.

Na Figura 23 pode-se observar o serviço candidato identificado empregando a heurística de loop. Este serviço inclui seis atividades. Logo, a partir da heurística de associação de serviços com atividades, este serviço deve estar associado a estas seis atividades.

3.3.4 Heurística de consolidação de serviços identificados a partir de fluxo

Nesta seção são apresentadas as heurísticas utilizadas para a consolidação dos serviços candidatos envolvendo fluxos de processos.

3.3.4.1 Número de raias envolvidas no fluxo

Heurística: Os papéis envolvidos no serviço de fluxo devem ser contabilizados, ou seja, toda raia que contiver atividades presentes nos serviços candidatos de fluxo identificados deve ser contabilizada.

Na Figura 23 pode-se observar o serviço candidato de loop de atividades, que possui seis atividades distintas distribuídas em duas raias. Logo, nesta heurística, este serviço candidato deve contabilizar duas raias.

3.3.4.2 Número de subfluxos

Heurística: Todo fluxo de processo existente dentro de outro fluxo deve ser contabilizado.

Na Figura 28 pode-se observar que existe um subfluxo de XOR dentro do fluxo de XOR principal. O fluxo de XOR principal é o fluxo iniciado pelo gateway de XOR identificado pela letra “a”. Este fluxo se divide em dois caminhos, um é finalizado por um evento final e o outro caminho possui um subfluxo iniciado com o gateway de XOR identificado pela letra “b”. O subfluxo iniciado em "b" se divide em dois caminhos e ambos terminam com um evento final. Logo, de acordo com a

aplicação desta heurística, o serviço candidato identificado a partir do fluxo iniciado em "a" deve contabilizar um subfluxo.

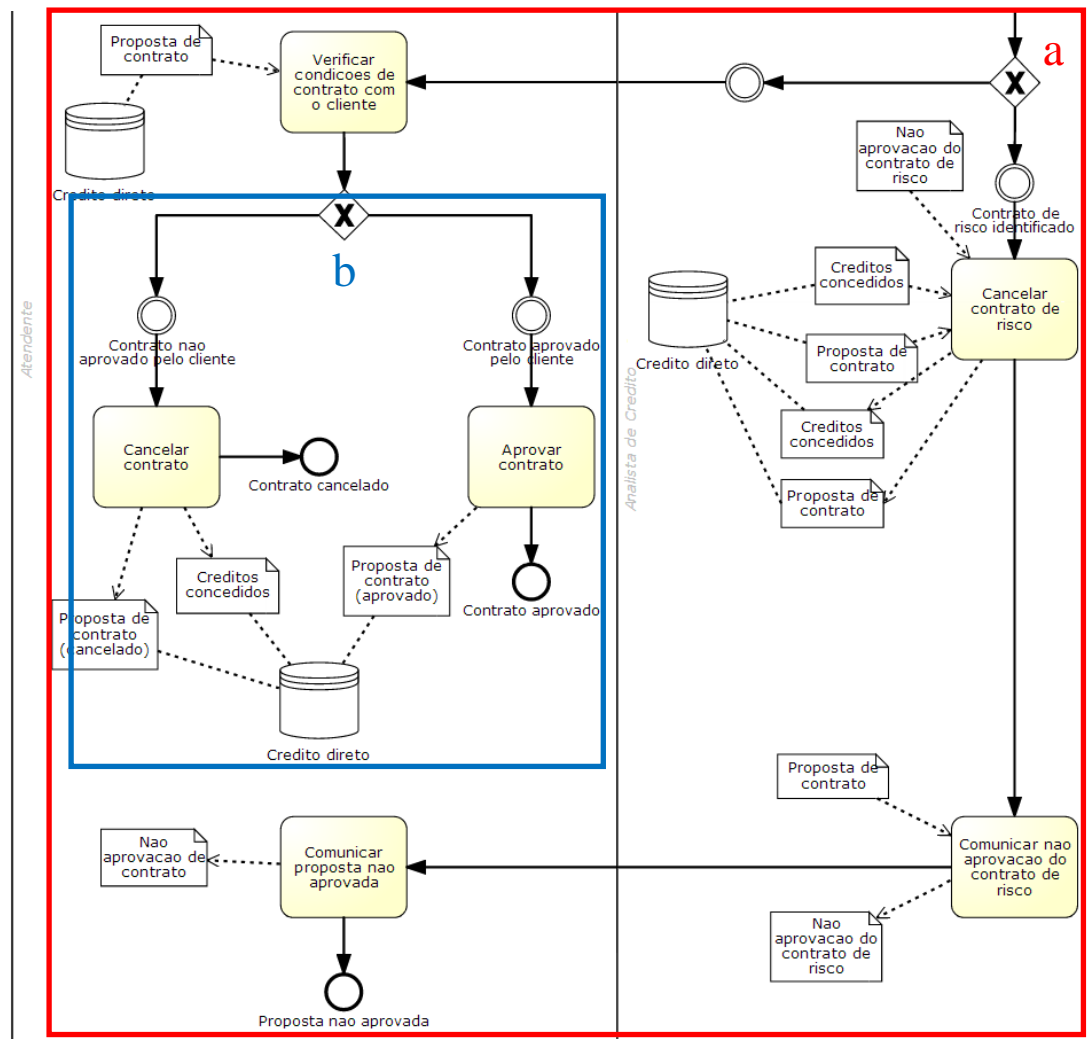


Figura 28 – Serviço de Loop com Subfluxos

Capítulo 4: Automatização da Identificação de Serviços em BPMN

Neste capítulo são apresentados a ferramenta de modelagem de processos, ambiente, estrutura do projeto e os algoritmos utilizados na parte prática deste trabalho para a identificação automática dos serviços candidatos.

4.1 Ferramenta de Modelagem de Processos - Signavio

A ferramenta utilizada neste trabalho foi o Signavio Process Editor (<http://www.signavio.com>) que foi desenvolvida por uma empresa alemã com o mesmo nome da ferramenta - Signavio. A empresa Signavio é composta por membros do grupo do Business Process Technology do Hasso Plattner Institute da Universidade de Potsdam na Alemanha [KUNZE *et al.*, 2010].

O Signavio Process Editor é uma ferramenta de análise e de modelagem de processos baseada no Oryx, que é outra ferramenta de modelagem que faz parte de um projeto acadêmico de software livre [KUNZE *et al.*, 2010]. O Signavio Process Editor é disponibilizado como software independente ou como SaaS (software como serviço), ou seja, o fornecedor do software é responsável pela estrutura necessária para disponibilizar a ferramenta. Esses fornecedores são responsáveis pelos servidores, conectividade e segurança; o usuário a utiliza via web pagando um valor determinado [TURNER *et al.*, 2003].

A ferramenta se encontra até o desenvolvimento deste trabalho na versão 6.2, e permite elaborar modelos de processos empregando as notações BPMN 2.0, EPC, diagramas de classe e diagramas de caso de uso da UML e outros. Também é compatível com as extensões XPD, BPMN 2.0, SGX, ARIS AML, permitindo a importação de modelos de processos salvos nestes formatos para o Signavio. A ferramenta exporta para as extensões PNG, SVG, PDF, SGX, BPMN 2.0, XML, JSON [SIGNAVIO, 2013].

O uso do Signavio como software independente (local na máquina do usuário) necessita de alguns pré-requisitos: o processador precisa ser no mínimo quadcore; os sistemas operacionais compatíveis são Windows e Linux; necessita de

um servidor web Apache Tomcat; e os servidores de banco de dados podem ser MySQL, Oracle e MS SQL [SIGNAVIO, 2013].

A versão SaaS do Signavio também precisa que alguns requisitos sejam atendidos, porém como tudo necessário para seu funcionamento é responsabilidade do seu fornecedor, a única exigência são as versões dos navegadores: Internet Explorer 9, Mozilla Firefox 13, Safari 5.1, Google Chrome 20, ou versões superiores. A empresa fornecedora também disponibiliza o Signavio Thin Client 1.0 como alternativa para rodar a ferramenta [SIGNAVIO, 2013].

O ambiente usado para a implementação e execução da proposta foi no sistema operacional Windows, com o uso do navegador Google Chrome versão 25 para o uso da ferramenta Signavio, e Netbeans versão 7.0.1 com a plataforma Java (jdk) 1.7.

4.2 Estrutura do Projeto

A Figura 29 ilustra o diagrama de pacotes que representa os principais pacotes e classes utilizadas para implementação da automatização da identificação de serviços candidatos, assim como a maneira com que esses pacotes e classes se relacionam.

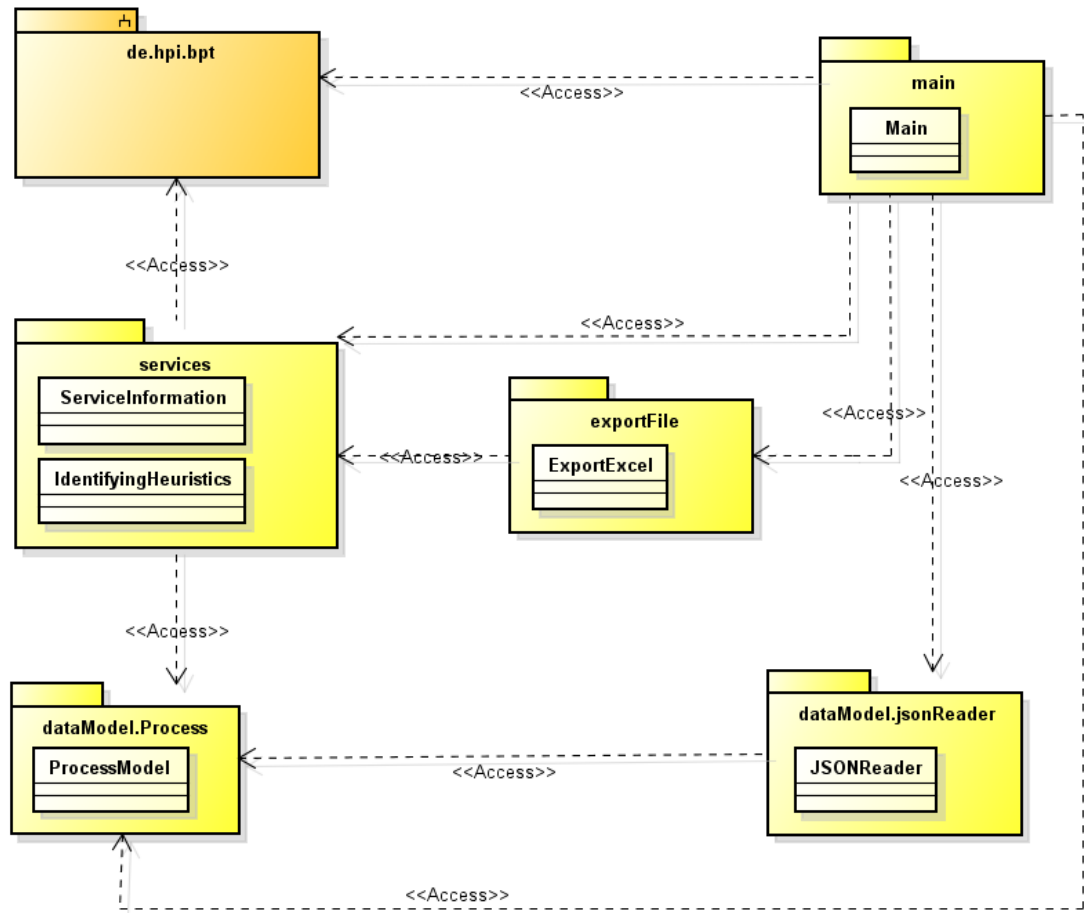


Figura 29 – Diagrama de Pacotes

Os pacotes existentes neste diagrama estão brevemente explicados abaixo:

1. de.hpi.bpt – Contém as classes da biblioteca RPST;
2. services – Possui a classe ServiceInformation, que corresponde à estrutura dos serviços candidatos, e a classe IdentifyHeuristics, que contém os algoritmos correspondentes às heurísticas de identificação e consolidação de serviços;
3. exportFile – Biblioteca responsável por escrever os resultados da identificação e consolidação de serviços candidatos no Excel;
4. main – Possui a classe main, responsável pela invocação dos métodos desenvolvidos;
5. dataModel.Process – Armazena as classes necessárias para transformar o modelo de negócio para a estrutura de *hashmaps*²;

² HashMaps – Estrutura de dados que armazena os elementos em uma tabela, onde há uma chave correspondente ao elemento armazenado desejado [DROZDEK, 2002].

6. `dataModel.jsonReader` – Armazena as classes responsáveis por mapear as informações presentes no arquivo do processo gerado (arquivo *json*) para a sua manipulação no projeto.

A Figura 30 ilustra o diagrama de atividades do processo de identificação de serviços candidatos. Este diagrama representa as atividades realizadas sequencialmente em todo o processo até a sua finalização, assim como a identificação dos responsáveis pela execução de cada atividade citada no diagrama.

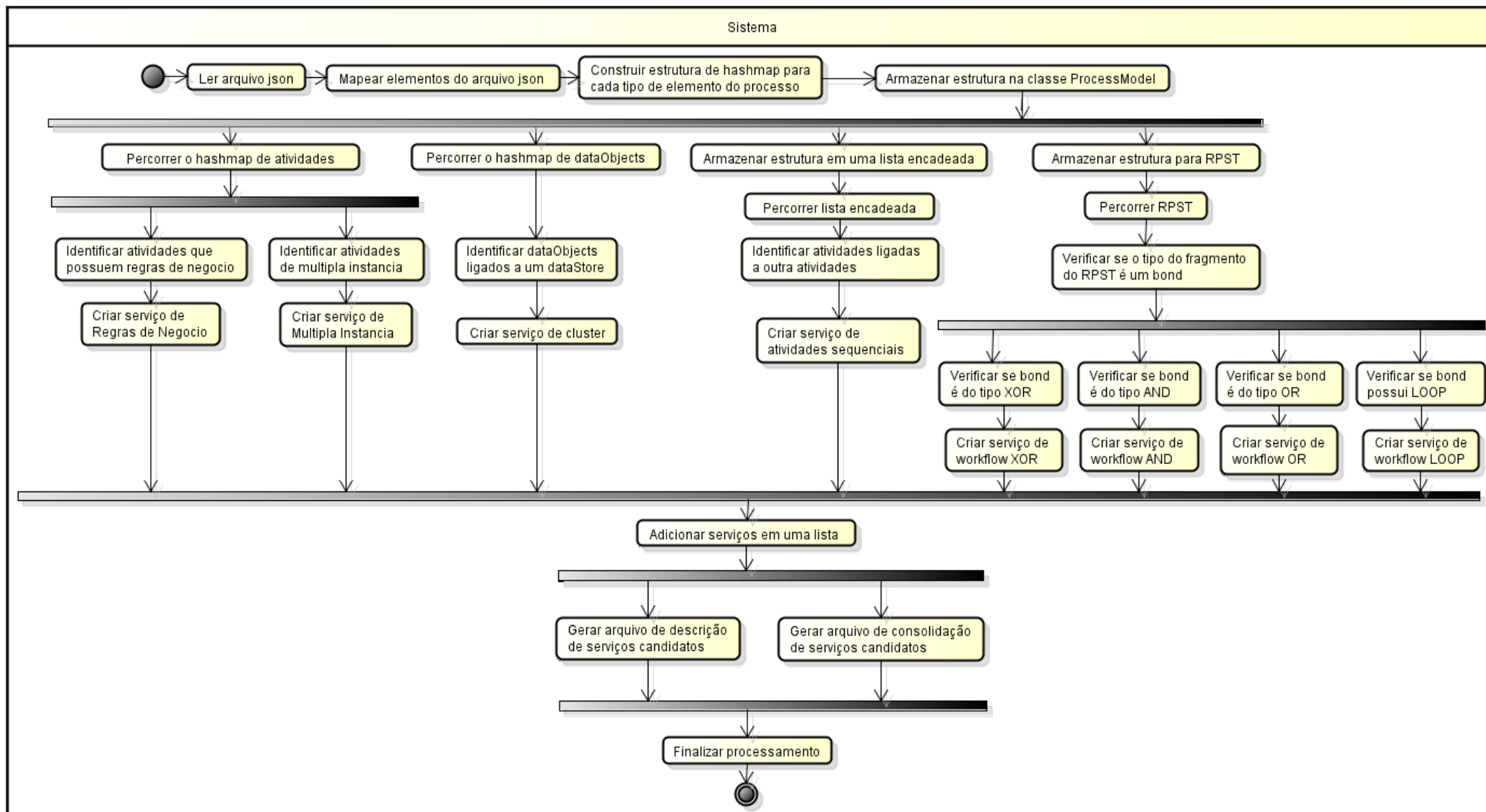


Figura 30 – Diagrama de Atividades

Para melhor entendimento do fluxo de atividades existente nesta implementação foi detalhado o diagrama de atividades, descrevendo seus elementos e funções. Primeiramente o usuário da ferramenta de automatização da identificação de serviços realiza o upload do arquivo do modelo de processo no formato JSON. Em seguida, o sistema lê esse arquivo e mapeia os elementos existentes no json para a estrutura de *hashmaps*, criando um *hashmap* para cada tipo de elemento do modelo de processo (atividades, *gateways*, eventos e outros). Essa nova estrutura é armazenada na classe `ProcessModel` do pacote `dataModel.Process`. Posteriormente, é iniciado o processamento das heurísticas de identificação.

Para a heurística de regras de negócio, é percorrido todo o *hashmap* de atividade e identificadas as atividades que são regras de negócios. Essas atividades são consideradas serviços candidatos.

Para a heurística de múltipla instância, o procedimento é parecido com o anterior. O *hashmap* de atividades é percorrido e, para cada elemento, é verificado se é uma atividade de múltipla instância. Caso positivo, essas atividades também são consideradas serviços candidatos.

Para a identificação da heurística de *cluster*, o *hashmap* de `dataObjects` é percorrido e é verificado se este está ligado a algum *data store*. Caso positivo, esses elementos são considerados serviços candidatos.

Para atividades sequenciais é necessário armazenar os dados em uma estrutura de lista encadeada, o que possibilita visualizar o elemento e todos os demais elementos que estão diretamente ligados a ele. Esta visão é essencial, pois é necessário identificar atividades que estão diretamente ligadas a outras atividades. Neste caso, para cada atividade ligada à outra atividade, se houver uma sequência de duas ou mais atividades seguidas, então ela é considerada um serviço candidato.

As demais heurísticas de identificação de serviços são heurísticas de *workflow*. Para tratar esses casos, foi necessário armazenar os dados na estrutura de RPST. Esta estrutura é apresentada no Capítulo 2. Posteriormente, foi percorrida esta estrutura e identificados os fragmentos de XOR, OR, AND e *Loop* de atividades. Todos estes fragmentos identificados são considerados serviços candidatos.

Após a identificação de todos os serviços são gerado os arquivos finais de serviços candidatos e de consolidação de serviços candidatos. Estes arquivos são disponibilizados para a visualização do usuário.

4.3 Implementação da Proposta

Nesta seção são apresentados os algoritmos utilizados para a identificação automática dos serviços candidatos a partir de modelos de processos desenhados com a notação BPMN e os algoritmos utilizados para a consolidação destes serviços.

O resultado correspondente à identificação de cada serviço candidato foi armazenado em uma tabela de acordo com o *template* apresentado na Tabela 2.

<u>Informações</u>	<u>Serviço</u>
Nome	
Tipo	
Entrada	
Saída	
Origem	
Atividades	
Descrição	
Perfil	

Tabela 2 – *Template* dos serviços criados

4.3.1 Identificação de workflows

Os algoritmos utilizados para a identificação das heurísticas de *workflow* foram implementados usando a RPST. O algoritmo usado para a identificação é apresentado no Algoritmo 1.

```

encontraHeuristicaWorkflow(raizRPST, rpst, nivel, modeloProcesso,
listaServiços)

    ordenarNosPorNivel(raizRPST, rpst)

    Para cada nó da rpst do nível atual
        profundidade recebe getProfundidade(nó, rpst)
        identificarHeuristicaWorkflowLoop(modeloProcesso, root,
                                           rpst, nível, serviço)
        identificarHeuristicaWorkflowXOR(modeloProcesso, root, rpst,
                                         nível, serviço)
        identificarHeuristicaWorkflowOR(modeloProcesso, root, rpst,
                                         nível, serviço)
        identificarHeuristicaWorkflowAND(modeloProcesso, root, rpst,
                                         nível, serviço)
        se éLoop
            encontraHeuristicaWorkflow(raizRPST, rpst, nivel,
                                       modeloProcesso, listaServiços)

```

```

    se éXORSplit ou éORSplit ou éEventoSplit
        caminhos recebe sortTreeLevel(nó, entrada do nó, rpst)
        para cada caminho
            encontraHeuristicaWorkflow(caminho, rpst, nível+1,
                                       modeloProcesso, listaServiços)
    se éANDSplit
        nósAND recebe sortTreeLevel(nó, entrada do nó, rpst)
        se o tamanho do nósAND é maior que 2
            topNós recebe sortTreeLevel(nó, entrada do nó, rpst)
            caminho1 recebe o primeiro elemento do topNós
            caminho2 recebe o segundo elemento do topNós
            encontraHeuristicaWorkflow(caminho1, rpst, nível,
                                       modeloProcesso, listaServiços)
            encontraHeuristicaWorkflow(caminho2, rpst, nível,
                                       modeloProcesso, listaServiços)
        senão
            caminhos recebe sortTreeLevel(nó, entrada do nó,
                                           rpst)
            para cada caminho em caminhos
                encontraHeuristicaWorkflow(caminho, rpst,
                                           nível+1, modeloProcesso, listaServiços)
    senão
        se profundidade maior que 0
            encontraHeuristicaWorkflow(nó, rpst, nível,
                                       modeloProcesso, listaServiços)

```

Algoritmo 1 – Identificação dos serviços a partir das heurísticas de *workflow*

Nas seções seguintes o algoritmo será decomposto em cada heurística de *workflow* para uma melhor compreensão.

4.3.2 Heurísticas de Identificação de serviços candidatos

Nesta seção serão apresentados os algoritmos utilizados para a identificação automática dos serviços candidatos.

Com a exceção dos algoritmos que utilizam a estrutura RPST, todos os outros algoritmos foram executados em cima da estrutura de *HashMaps*, construído a partir do arquivo JSON gerado pela ferramenta de modelagem Signavio.

4.3.2.1 Heurística de identificação de serviço a partir de Regras de Negócio

Para a identificação das regras de negócio presentes no modelo de processo, o algoritmo implementado percorre todas as atividades do processo e verifica se o tipo é regra de negócio (Algoritmo 2).

```

identificarHeuristicaRegrasDeNegocio(modeloProcesso)
  Para cada atividade do modeloProcesso faça
    Se o tipo da atividade for igual a Regra de Negocio
      Cria serviço candidato Regra de Negocio

```

Algoritmo 2 – Identificação do serviço a partir de Regra de Negócio

4.3.2.2 Heurística de identificação de serviços a partir de *Cluster*

Para a identificação de *clusters* que podem se tornar serviços candidatos presentes no modelo de processo, o algoritmo implementado percorre todos os portadores de informações (*data store*) do processo e verifica este elemento está ligado a um *cluster* (*data object*). Posteriormente, verifica se este *data object* está ligado a uma atividade (Algoritmo 3).

```

identificarHeuristicaCluster(modeloProcesso)
  Para cada dataStore do modeloProcesso faça
    Se dataStore estiver ligado a um dataObject
      Se dataObject for o alvo de uma atividade
        Cria serviço candidato Cluster de escrita
      Senão //Se o dataObject for a origem
        Cria serviço candidato de Cluster de leitura

```

Algoritmo 3 – Identificação do serviço a partir de *Cluster*

4.3.2.3 Heurística de identificação de serviços a partir de Atividades Sequenciais

Para a identificação de atividades sequenciais presentes no modelo de processo, o algoritmo implementado percorre todas as atividades do processo e verifica se esta atividade está associada à outra atividade (Algoritmo 4 e Algoritmo 5).

```

identificarHeuristicaAtividadesSequenciais(modeloProcesso)
  Para cada elemento do modeloProcesso faça
    Se elemento for uma atividade (elemento1)
      Ligados recebe lista dos elementos ligados à atividade
      Para cada elemento em ligados (elemento2)
        Se elemento for uma atividade
          Adiciona elemento1 e elemento2 na lista de ativSeq
          obterAtivsSeq(modeloProcesso, ativSeq, elemento2)
        Se elemento for um evento
          Adiciona elemento1 na lista de ativSeq
          obterAtivsSeq(modeloProcesso, ativSeq, elemento2)

```

Algoritmo 4 – Identificação do serviço a partir de Atividades Sequenciais

```

obterAtivsSeq(modeloProcesso, ativSeq, elemento2)
  ligados recebe lista dos elementos ligados à elemento2
  Para cada elemento em ligados (elemento3)
    Se elemento for atividade

```

```

        Adiciona elemento3 a lista ativSeq
        obterAtivsSeq(modeloProcesso, ativSeq, elemento3)
    Se elemento for evento
        obterAtivsSeq(modeloProcesso, ativSeq, elemento3)

```

Algoritmo 5 – Identificação de atividades sequenciais em função recursiva

4.3.2.4 Heurística de identificação de serviços a partir de *Loop* de Atividades

Para a identificação de Loops presentes no modelo de processo, o algoritmo implementado percorre todos os nós da árvore RPST e verifica para cada nó se o nó é um *loop*, ou seja, verifica se há um arco saindo do *gateway* de saída do nó e entrando no *gateway* de entrada do respectivo nó (Algoritmo 6 e Algoritmo 7).

```

identificarHeuristicaWorkflowLoop(modeloProcesso, root, rpst, nível, se
rviço)
    Para cada nó do rpst faça
        Se nó é do tipo bond
            Se éLoop(nó)
                Cria serviço candidato Loop (adiciona nó a serviço)

```

Algoritmo 6 – Identificação do serviço a partir de *Workflow Loop*

```

éLoop(nóBond)
    Se nóBond é do tipo bond
        e primeiro elemento do nóBond é um gateway e é do tipo XOR
        Para cada nó filho do nóBond
            Se nó éTrivial e entrada do nó é igual a saída do nóBond
                e saída do nó é igual a entrada do nóBond
                e nó é um gateway
                Retorna verdade
            Se entrada do nóBond é igual saída do nó
                e entrada do nó é gateway
                Retorna verdade
    Retorna falso

```

Algoritmo 7 – Identificação se o nó é loop

4.3.2.5 Heurística de identificação de serviços a partir de AND

Para a identificação de *workflows* AND presentes no modelo de processo, o algoritmo implementado percorre todos os nós da árvore RPST e verifica para cada fragmento classificado como *bond* se possui *gateway* do tipo AND (Algoritmo 8 e Algoritmo 9).


```

identificarHeuristicaWorkflowAND(modeloProcesso, root, rpst, nível, ser
viço)
    Para cada nó do rpst faça
        Se nó é do tipo bond
            Se possuiAND(nó)
                Cria serviço candidato AND

```

Algoritmo 8 – Identificação do serviço a partir de *Workflow* AND

```

possuiAND(nóBond)
    Se nóBond é bond e primeiro elemento do nóBond é um gateway
        Se gateway é do tipo AND
            Retorna verdade

```

Algoritmo 9 – Identificação se o nó possui AND

4.3.2.6 Heurística de identificação de serviços a partir de XOR

Para a identificação de *workflows* XOR presentes no modelo de processo, o algoritmo implementado percorre todos os nós da árvore RPST e verifica para cada fragmento classificado como *bond* se possui *gateway* do tipo XOR (Algoritmo 10 e Algoritmo 11).

```

identificarHeuristicaWorkflowXOR(modeloProcesso, root, rpst, nível, ser
viço)
    Para cada nó do rpst faça
        Se nó é do tipo bond
            Se possuiXOR(nó)
                Cria serviço candidato XOR

```

Algoritmo 10 – Identificação do serviço a partir de *Workflow* XOR

```

possuiXOR(nóBond)
    Se nóBond é bond e primeiro elemento do nó é um gateway
        Se gateway é do tipo XOR
            Retorna verdade

```

Algoritmo 11 – Identificação se o nó possui XOR

4.3.2.7 Heurística de identificação de serviços a partir de OR

Para a identificação de *workflows* OR presentes no modelo de processo, o algoritmo implementado percorre todos os nós da árvore RPST e verifica para cada fragmento classificado como *bond* se possui *gateway* do tipo OR (Algoritmo 13 e Algoritmo 14).

```

identificarHeuristicaWorkflowOR(modeloProcesso, root, rpst, nível, serv
iço)
    Para cada nó do rpst faça
        Se nó é do tipo bond
            Se possuiOR(nó)
                Cria serviço candidato OR

```

Algoritmo 12 – Identificação do serviço a partir de *Workflow* OR

```
possuiOR (nóBond)
  Se nóBond é bond e primeiro elemento do nó é um gateway
  Se gateway é do tipo OR
  Retorna verdade
```

Algoritmo 13 – Identificação do serviço a partir de *Workflow OR*

4.3.2.8 Heurística de identificação de serviços a partir de Múltipla Instância

Para a identificação de atividades de múltiplas instâncias presentes no modelo de processo, o algoritmo implementado percorre todas as atividades do processo e verifica se esta atividade é do tipo múltipla instância (Algoritmo 14).

```
identificarHeuristicaMultiplaInstancia (modeloProcesso)
  Para cada atividade do modeloProcesso faça
    Se LoopType da atividade for diferente de "none"
      Cria serviço candidato Múltipla Instância
```

Algoritmo 14 – Identificação do serviço a partir de Múltipla Instancia

4.3.3 Heurística de consolidação de serviços

Nesta seção são apresentados os algoritmos utilizados para a consolidação dos serviços candidatos.

4.3.3.1 Heurística de eliminação de serviços candidatos duplicados

Para a eliminação de serviços candidatos presentes na lista de serviços, o algoritmo implementado percorre todos os serviços da lista e verifica se este serviço possui nome igual a outro serviço da lista de serviços (Algoritmo 15).

```
eliminarServiços (listaServiços)
  Para cada serviço da listaServiços faça
    Se nome do serviço for igual
      ao nome de outro serviço na listaServiços
      Eliminar serviço
```

Algoritmo 15 – Eliminação de serviços

4.3.3.2 Heurística de associação de serviços com atividades

Para a associação de serviços com atividades, o algoritmo implementado lista todos os serviços existentes na lista de serviços, e para cada um deles, são verificadas as atividades que ele possui (Algoritmo 16).

```
Para cada serviço na listaServiços faça
  Para cada atividade na listaAtividades do serviço faça
    Imprimir associação na tabela de serviços x atividades
```

Algoritmo 16 – Associação serviços x atividades

4.3.3.3 Heurística de associação de serviços com papéis

Para a associação de serviços com papéis, o algoritmo implementado lista todos os serviços existentes na lista de serviços e para cada um deles, são verificadas as raiais em que seus elementos estão localizados (Algoritmo 17).

```
Para cada serviço na listaServiços faça
  Para cada perfil na listaPerfis do serviço faça
    Imprimir associação na tabela de serviços x papéis
```

Algoritmo 17 – Associação serviços x papéis

4.3.3.4 Heurística de consolidação de serviços identificados a partir de fluxo

Nesta seção são apresentados os algoritmos utilizados para a consolidação dos serviços candidatos identificados envolvendo fluxos de processos.

4.3.3.4.1 Número de raiais envolvidas no fluxo

Para a contagem do número de raiais presentes na lista de serviços de *workflows*, o algoritmo percorre cada workflow da lista e conta as raiais em que cada atividade do *workflow* está localizada (Algoritmo 18).

```
Para cada serviço na listaServiços faça
  Contar número de perfis envolvidos no fluxo
```

Algoritmo 18 – Contar número de raiais

4.3.3.4.2 Número de subfluxos

Para a contagem do número de subfluxos presentes na lista de serviços de *workflows*, o algoritmo implementado percorre todos os *workflows* da lista e verifica se as atividades de um *workflow* estão contidas nas atividades de outro *workflow* da lista (Algoritmo 19).

```
contarSubfluxos(listaServiçosWorkflows)
  Para cada workflow da listaServiçosWorkflows faça
    Se atividades do workflow estiver contida em outro workflow
    da listaServiçosWorkflows
      Contar subfluxo
```

Algoritmo 19 – Contar número de subfluxos

Capítulo 5: Avaliação da Proposta

Esta seção apresenta uma análise da proposta deste trabalho.

5.1 – Processo utilizado

A avaliação da proposta de automatização da identificação de serviços candidatos foi realizada executando a ferramenta implementada no modelo de processo “Analisar pedido de crédito” modelado em BPMN adaptado de [SOUSA, 2012] e apresentado no Anexo II. Este processo possui um evento inicial, quatro eventos finais, dezoito atividades, quinze *data stores*, treze eventos intermediários, quarenta e nove *data objects* e seis *gateways*.

No processo modelado é feita a análise de uma proposta de crédito recebida, verificado se crédito pode ser concedido ao cliente requisitante. Primeiramente, ao receber uma proposta de crédito, é verificado o cadastro do cliente. Caso este cliente não possua cadastro, deverá ser realizado o cadastramento. Caso contrário, se as informações do cliente estiverem desatualizadas, apenas deverá ser feita uma atualização das informações. Caso contrário, segue-se para o próximo passo. Posteriormente, é verificado o limite de crédito deste cliente. Caso o limite não seja aprovado, a proposta de crédito é cancelada. Se o limite de crédito for aprovado, deve-se comprometer o limite de crédito, calcular a taxa de contrato e de juros e gerar a proposta de contrato. O analista de crédito deve analisar o contrato e realizar ajustes necessários. Ele poderá cancelar o contrato, caso o considere um contrato de risco. Em seguida, o cliente deverá verificar as condições do contrato e cancelar ou aprovar o contrato, finalizando o processo.

Neste processo foram identificados dois serviços de regras de negócio, um serviço de múltipla instância, dezenove serviços de *clusters*, três serviços de atividades sequenciais, um serviço de *LOOP* e quatro serviços de *workflow XOR*, totalizando 30 serviços.

Os resultados obtidos com o estudo de caso analisado estão detalhados no Anexo II.

5.2 – Produtos gerados da identificação automática de serviços

O projeto proposto possui como produtos finais a geração de dois relatórios no Microsoft Excel, com o formato .xls: Relatório de Serviços e Relatório de Serviços Consolidados. Esses serviços foram identificados a partir do processo presente no Anexo II.

O primeiro arquivo gerado possui os serviços identificados na automatização da identificação de serviços baseados nas heurísticas descritas anteriormente e suas principais informações, como nome do serviço, tipo (serviço de dados ou de negócio), entrada, saída, origem (heurística a partir do qual o serviço foi identificado), atividades (atividade(s) que originaram o serviço), descrição e perfil (nome dos perfis executores da(s) atividade(s) em que o serviço foi identificado).

As informações deste relatório devem servir de base para o analista responsável avaliar quais serviços candidatos devem ser implementados, dependendo das particularidades do projeto.

Este relatório possui uma tabela para cada tipo de heurística implementada, totalizando oito tabelas. São elas: Tabela de Serviços de Regras de Negócio, Tabela de Serviços de Múltipla Instância, Tabela de Serviços de *Cluster*, Tabela de Serviços de Atividades Sequenciais, Tabela de Serviços de *Loop* de Atividades, Tabela de Serviços de *Workflow* XOR, Tabela de Serviços de *Workflow* OR e Tabela de Serviços de *Workflow* AND. Todas as tabelas apresentam os serviços segundo o *template* apresentado na Tabela 2. A Tabela 3 apresenta um exemplo de serviço candidato identificado a partir da heurística de *cluster*.

<u>Informações</u>	<u>Servicos</u>
Nome	Obter Taxas do contrato
Tipo	Dados (leitura)
Entrada	Taxas do contrato
Saída	
Origem	Cluster
Atividades	[Calcular taxa de contrato]
Descrição	A partir dos dados taxas do contrato deve-se obter taxas do contrato.
Perfil	[Credito Direto]

Tabela 3 – Tabela de serviço de cluster

Outros exemplos são apresentados no Anexo III.

O segundo arquivo gerado apresenta a consolidação dos serviços identificados na automatização da identificação de serviços baseados nas heurísticas descritas anteriormente.

Este relatório possui três tabelas: Tabela de Informações de Fluxo, Tabela de Associação de Serviços x Atividades e Tabela de Associação de Serviços x Perfis.

Segue detalhamento de cada tabela existente neste relatório:

- Tabelas de informações de fluxos (flows)

Nesta tabela existe a listagem de todos os serviços de fluxos (LOOP, XOR, OR e AND) identificados no processo. Para cada um deles é detalhado o número de subfluxos existentes, o número de raias envolvidas no serviço e os clusters associados, seja de entrada ou de saída de informações. O nome dos serviços foi definido como a concatenação dos nomes das atividades presentes no fluxo. Esta forma de nomenclatura deve ser melhorada.

<u>Serviços</u>	<u>Nº de Subflows</u>	<u>Nº de Raias</u>	<u>Cluster Associados</u>
Cadastrar cliente_Atualizar cadastro do cliente	0	1	[Proposta de credito]
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	3	3	[Cadastro do cliente, Nao aprovacao do contrato de risco]
Comprometer Limite de Credito_G	0	2	[Cadastro do cliente]
Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco	1	2	[Nao aprovacao do contrato de risco]
Cancelar contrato_Aprovar contrato	0	1	[]

Tabela 4 – Tabelas de informações de fluxos (flows)

Na Tabela 4 pode-se observar que existem cinco serviços de fluxos identificados. Para o primeiro serviço identificado, “Cadastrar cliente_Atualizar cadastro do cliente”, observa-se que este serviço não possui nenhum subfluxo, possui

apenas uma raia (perfil) envolvida e possui o cluster “Proposta de Crédito” associado.

- Tabelas de Associação de Serviços x Atividades

Nesta tabela é possível identificar as atividades que compõem cada serviço. Cada atividade que fizer parte do serviço é marcada com um X na tabela, para representar essa associação.

<u>Serviços</u>	<u>Calcular taxa de contrato</u>	<u>Verificar cadastro do cliente</u>	<u>Verificar limite de credito do cliente</u>	<u>Analisar Contrato</u>
Verificar cadastro do cliente		X		
Determinar taxas de juros a ser cobrada do cliente				
Calcular taxa de contrato	X			
Obter Taxas do contrato	X			
Obter Proposta de credito				

Tabela 5 – Tabela de Associação de Serviços x Atividades

Na Tabela 5 pode-se observar que o serviço “Verificar cadastro do cliente” possui a atividade “Verificar cadastro do cliente” associada.

- Tabelas de Associação de Serviços x Perfis

Nesta tabela é possível identificar os perfis que são responsáveis por executar as atividades pertencentes ao serviço. Cada perfil que fizer parte do serviço é marcado com um X na tabela, para representar essa associação.

<u>Serviços</u>	<u>Atendente</u>	<u>Analista de Credito</u>	<u>Credito Direto</u>
Verificar cadastro do cliente			X
Determinar taxas de juros a ser cobrada do cliente			X
Calcular taxa de contrato			X
Obter Taxas do contrato			X
Obter Proposta de credito		X	X
Obter Creditos concedidos	X	X	X
Obter Limite de credito do cliente		X	X

Tabela 6 – Tabela de Associação de Serviços x Perfis

Na Tabela 6 pode-se observar que o serviço “Verificar cadastro do cliente” possui apenas o perfil “Crédito Direto” associado a ele.

5.3 – Diferenças encontradas entre EPC e BPMN

Durante as implementações das heurísticas de identificação de serviços em BPMN foram observadas algumas diferenças com a identificação dos serviços em EPC de SOUSA *et al.* [2011] e Azevedo *et al.* [2009c].

Na heurística de regras de negócio foram encontradas poucas diferenças, pois a forma de identificar quais atividades tem regras de negócio associadas nas duas notações é semelhante. Como BPMN não possui o diagrama FAD, que obtém detalhes da atividade na notação EPC, as regras de negócio em BPMN são identificadas como um campo dentro das atividades. O mesmo ocorre com a heurística de identificação de *cluster*, por não haver o FAD. Em BPMN, os *clusters* são identificados a partir do fluxo principal do processo.

Nas heurísticas de identificação de atividades sequenciais e de múltipla instância, a diferença entre suas notações é pequena, pois a forma de representar atividades e atividades de múltipla instância é muito semelhante em BPMN e em EPC. A diferença é apenas a lógica para percorrer os elementos armazenados nas diferentes estruturas.

As heurísticas de identificação de serviços por requisitos de negócio e de interface de processo não puderam ser implementadas devido às limitações da notação BPMN.

Na heurística de identificação de workflow LOOP foi obtida uma diferença significativa em sua implementação. Em [SOUSA *et al.*, 2011], o autor utilizou uma função nativa da ferramenta ARIS, no qual foi modelado o processo em EPC, para encontrar ciclos no fluxo. Já nesse projeto, a identificação de loop foi obtida usando a estrutura de armazenamento de dados RPST.

As heurísticas de identificação de workflow AND, XOR, OR não foram implementadas por [SOUSA *et al.* 2011], conforme citado anteriormente. As implementações dessas heurísticas nesse projeto foram realizadas utilizando também a estrutura RPST.

Todas as heurísticas foram implementadas considerando algumas premissas: todas as atividades devem possuir nomes distintos, e de acordo com a especificação da notação BPMN, o elemento atividade só pode possuir uma transição de entrada e uma transição de saída.

Capítulo 6: Conclusão

A modelagem de processos é muito utilizada nas organizações para uma melhor compreensão de seus processos, facilitando sua análise e melhorias. Uma modelagem bem executada contém detalhes e informações relevantes sobre o processo. Essas informações são essenciais para o desenvolvimento de serviços em uma abordagem SOA.

Nos trabalhos de AZEVEDO [2009a, 2009b] foram desenvolvidas heurísticas para a identificação desses serviços, e AZEVEDO [2009c] e SOUSA *et al.* [2011] implementaram essas heurísticas em um processo de automatização da identificação de serviços usando notação EPC para modelagem do processo.

O benefício mais claramente identificado com a execução desta automatização foi a velocidade com que os serviços foram identificados, reduzindo riscos de erros humanos. Esta tarefa, quando executada manualmente, demanda tempo e esforço que foram poupados com a execução da ferramenta desenvolvida [AZEVEDO *et al.*, 2009c].

Outro benefício identificado foi a possibilidade de que qualquer alteração na modelagem do processo não demande muito tempo para realizar novamente a identificação dos serviços. Ou seja, a execução da ferramenta permite gerar rapidamente novos serviços, de acordo com os ajustes realizados.

Além dos benefícios citados, um dos ganhos da ferramenta implementada é que esta foi desenvolvida em Java podendo ser utilizada por qualquer ferramenta de modelagem de processos. Diferente da ferramenta desenvolvida por [SOUSA, 2010] que é desenvolvida dentro da plataforma de modelagem ARIS, sendo dependente da mesma.

O objetivo deste trabalho foi implementar as heurísticas desenvolvidas para a identificação automática de serviços usando notação BPMN, a qual se apresenta como uma das notações mais utilizadas atualmente. Há duas heurísticas desenvolvidas por AZEVEDO *et al.* [2009a] e AZEVEDO *et al.* [2009b] que não puderam ser implementadas devido às limitações da notação BPMN: heurística de identificação de serviços por requisitos de negócio; e heurística de identificação de

interface de processo. Na notação BPMN não há elementos para representar requisitos de negócio e interface de processo na modelagem do processo.

Nas heurísticas de consolidação de serviços foi encontrada limitação por não usar um repositório, apenas o arquivo de um processo. Dessa forma, não foi possível calcular o grau de reuso de cada serviço identificado, associar os serviços com os sistemas que os usam, associar os serviços aos requisitos da demanda, identificar a granularidade e dependência entre os serviços candidatos e executar a heurística de consolidação dos serviços utilitários.

Durante a realização desse trabalho, foram identificadas oportunidades de trabalhos futuros, tais como:

- Utilizar a ferramenta em um cenário real, avaliando sistematicamente o ganho com o uso da ferramenta em relação a tempo e qualidade;
- Usar processamento de linguagem natural para identificar informações adicionais do serviço;
- Implementar heurísticas considerando repositórios de processos;
- Nomear os serviços utilizando processamento de texto.

Referências

AZEVEDO, L., BAIÃO, F., SANTORO, F., SOUZA, J., REVOREDO, K., PEREIRA, V., HERLAIN, I. **Identificação de serviços a partir da modelagem de processos de negócio**. Simpósio Brasileiro de Sistemas de Informação (SBSI'09), Brasília, 2009a.

AZEVEDO, L. G.; SANTORO, F.; BAIÃO, F.; SOUZA, J. F.; REVOREDO, K.; PEREIRA, V.; HERLAIN, I. **A Method for Service Identification from Business Process Models in a SOA Approach**. In: 10th Workshop on Business Process Modeling, Development, and Support (BMDS'09), 2009, Amsterdam. Enterprise, Business-Process and Information Systems Modeling. Berlin Heidelberg: Springer, 2009. v. 29. p. 99-112, 2009b.

AZEVEDO, L., SOUSA, H. P., SOUZA, J. F., SANTORO, F. BAIÃO, F., **Identificação automática de serviços candidatos a partir de modelos de processos de negócio**. Conferência IADIS Ibero Americana WWW/INTERNET 2009 (CIAWI'09), Outubro, 21-23, Madrid, Espanha, 2009c.

AZIZ, S. M. **An Introduction to JavaScript Object Notation (JSON) in JavaScript and .NET**. Publicada em fevereiro de 2007, disponível em <http://msdn.microsoft.com/en-us/library/bb299886.aspx>. Acessado em 07 de Março de 2013.

BECKER, A., WIDJAJA, T., BUXMANN, P. **Value Potentials and Challenges of Service-Oriented Architectures - Results of an Empirical Survey from User and Vendor Perspective**. Business & Information Systems Engineering, Vol. 3: Iss. 4, 199-210. 2011.

BLOOMBERG, J., SCHMELZER, R. **Service Orient or Be Doomed!: How Service Orientation Will Change Your Business**. Hoboken, NJ: John Wiley & Sons. 2006.

DAVENPORT, T. **Reengenharia de processos**. Rio de Janeiro: Campus, p.6-8, 1994.

DROZDEK, A. **Estrutura de dados e algoritmos em C++**. Editora Cengage Learning. 2002.

ECMA. **ECMA Script Language Specification**. 5.1 Edition, June 2011.

ERL, **Service-Oriented Architecture (SOA): Concepts, Technology, and Design**. Upper Saddle River, NJ, USA: Prentice Hall, 2005.

EPC NOTATION. **Event-driven process chain (EPC)**, 2011. Disponível em <http://www.ariscommunity.com/event-driven-process-chain>. Acessado em 27 de fevereiro de 2013.

JOSUTTIS, N. **SOA in practice: The Art of Distributed System Design**, Beijing;Cambridge. O'Reilly, 2007, 324p.

KUNZE, M., WESKE, M. **Signavio-Oryx Academic Initiative**, Demo Session of the 8th International Conference on Business Process Management (BPM 2010). Hoboen, NJ, Setembro de 2010.

KOCH, C. **ABC da SOA**, publicada em 17 de Julho de 2006, disponível em http://cio.uol.com.br/tecnologia/2006/07/17/idgnoticia.2006-07-17.3732358054/paginador/pagina_3. Acessado em 04 de Março de 2013.

LEOPOLD, H., MENDLING J., POLYVYANYYY A., **Generating Natural Language Texts from Business Process Models**. 2012

OLIVEIRA, S. B. **Gestão pro Processos: fundamentos, técnicas e modelos de implementação**. Rio de Janeiro: Qualitymark, 2006. p. 291-305.

OMG, **Business Process Model and Notation**. Especificação da Notação BPMN, Janeiro de 2011.

OSTROFF, F. **The Horizontal Organization: what the organization of the future actually looks like and how it delivers value to customers**. 257p. USA: Oxford University Press, 1999.

PAVLOVSKI, C. J., ZOU, J. **Non-functional requirements in business process modeling**. In APCCM '08: Proceedings of the fifth on Asia-Pacific conference on Conceptual modelling, Darlinghurst, Australia, 2008.

POLYVYANNY, A., VANHATALO, J., VÖLZER, H. V., **Simplified Computation and Generalization of the Refined Process Structure Tree**. 2009.

RICHARDSON, L. e RUBY, S. (2007). **RESTful web services**. O' Reilly Media.

SIGNAVIO INFORMATION, 2013. **Feature Overview – Signavio Process Editor**. Disponível em <http://www.signavio.com/products/process-editor/process-modeling/>. Acessado em 05 de março de 2013.

SCHEER, A. W., THOMAS, O., ADAM, O. **Process Modeling Using Event-Driven Process Chains**, Cap 6 do livro: **Process-Aware Information Systems: Bridging People and Software Through Process Technology**. John Wiley & Sons, 2005.

SOUSA, H. P., AZEVEDO, L.G., SANTORO, F. **Identificação automática de serviços em uma abordagem SOA**. Relatórios Técnicos do DIA/UNIRIO (RelaTe-DIA), RT-0002/2011, 2011. Disponível (também) em: <http://seer.unirio.br/index.php/monografiasppgi>.

SOUSA, H. P. **Integrando modelagem intencional à modelagem de procesos.** Dissertação de mestrado, 2012.

TURNER, M., BUDGEN, D., BRERETON, P. **Turning Software into a Service.** Keele University, Staffordshire. 2003

VALLE, R., OLIVEIRA, S. B. *et al.*, **Análise e Modelagem de Processos de Negócio. Foco na Notação BPMN.** São Paulo: Atlas, 2012.

VANHATALO, J., VÖLZER H. V., KOEHLER, J., **The refined process structure tree.** 2009.

WfMC. **Workflow management coalition terminolog and glossary.** Relatório técnico WfMC-TC-1011, Workflow Management Coalition, Brussels, fevereiro de 1999.

Anexo I – Exemplo real do processo do exemplo RPST

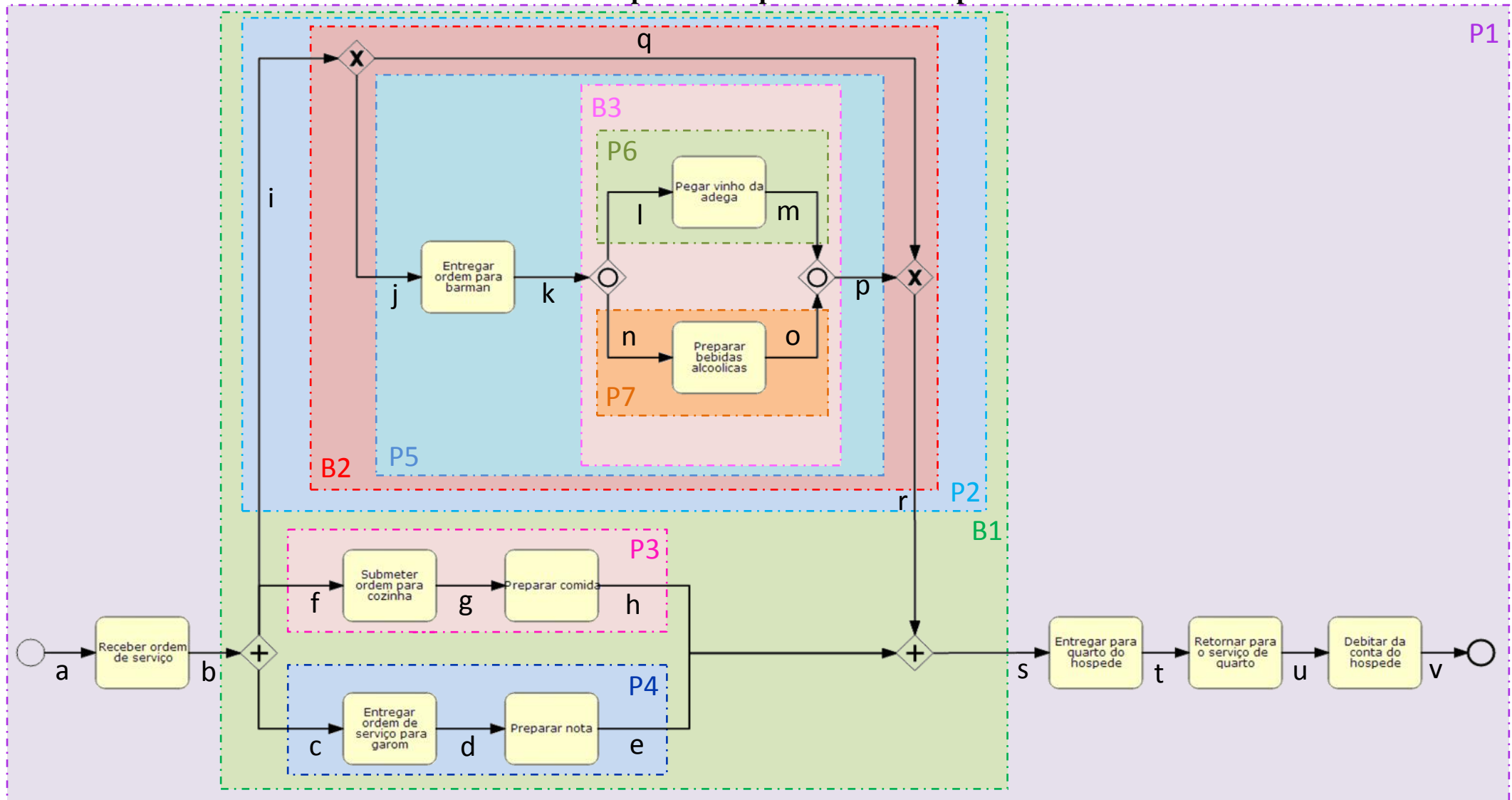


Figura 31 – Exemplo real de processo estruturado em fragmentos de RPST (adaptado de POLYVYANNY *et al.* [2009])

Anexo II – Modelagem do Estudo de Caso – Processo “Analisar de Crédito”

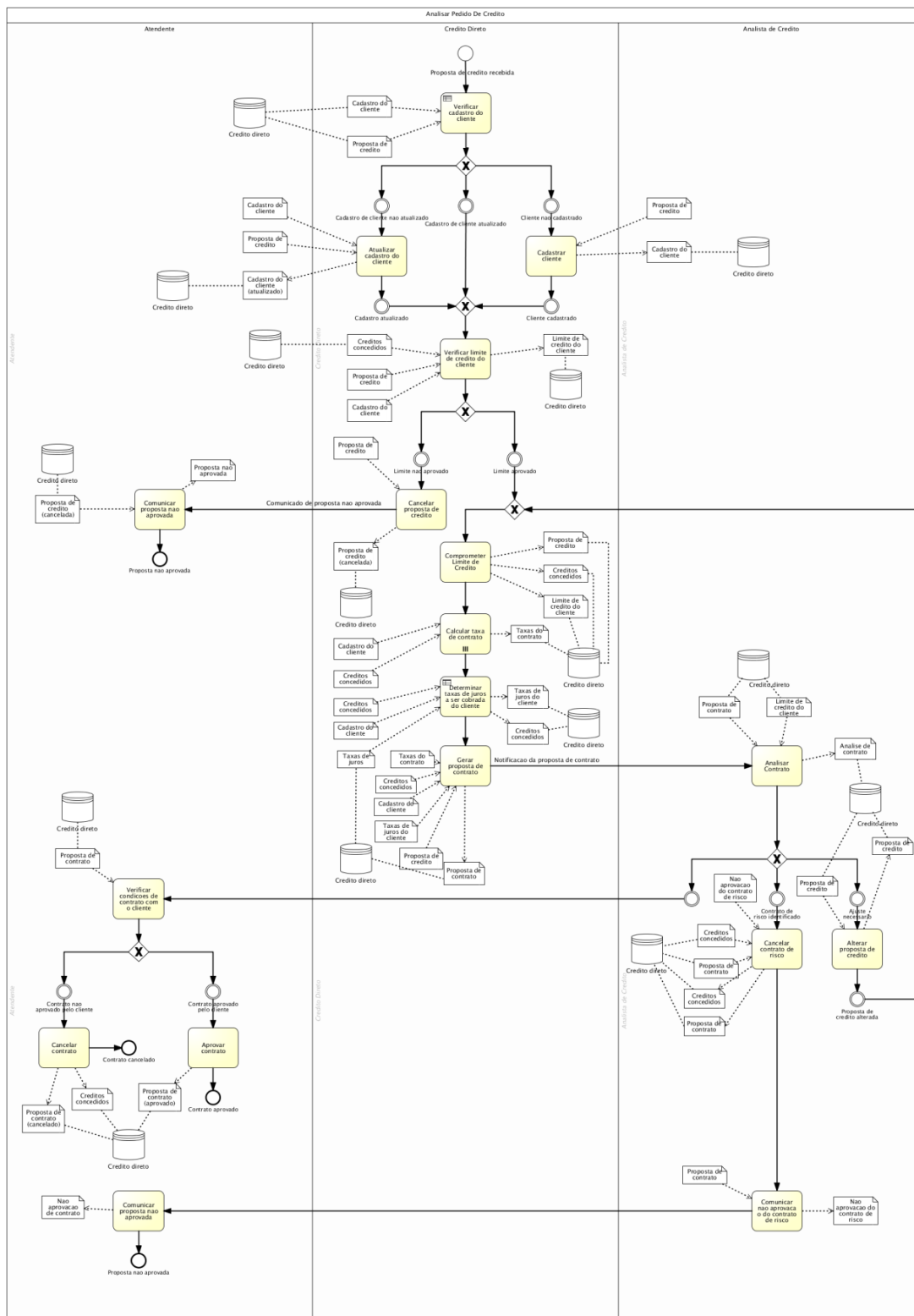


Figura 32 – Processo Analisar de Pedido de Crédito em BPMN [Adaptação de SOUSA, 2012].

Anexo III – Relatórios obtidos no Estudo de Caso

Informações	Serviços
Nome	Verificar cadastro do cliente
Tipo	
Entrada	
Saída	
Origem	Regras de Negocio
Atividades	[Verificar cadastro do cliente]
Descrição	A partir dos dados deve-se verificar cadastro do cliente.
Perfil	[Credito Direto]

Nome	Determinar taxas de juros a ser cobrada do cliente
Tipo	
Entrada	Créditos concedidos, Cadastro do cliente
Saída	
Origem	Regras de Negocio
Atividades	[Determinar taxas de juros a ser cobrada do cliente]
Descrição	A partir dos dados créditos concedidos, cadastro do cliente deve-se determinar taxas de juros a ser cobrada do cliente.
Perfil	[Credito Direto]

Tabela 7 – Serviços identificados na heurística de Regras de Negócio

Informações	Serviços
Nome	Calcular taxa de contrato
Tipo	
Entrada	Cadastro do cliente, Créditos concedidos
Saída	
Origem	Múltipla Instância
Atividades	[Calcular taxa de contrato]
Descrição	A partir dos dados cadastro do cliente, créditos concedidos deve-se calcular taxa de contrato.
Perfil	[Credito Direto]

Tabela 8 – Serviços identificados na heurística de Múltipla Instância

Informações	Serviços
Nome	Obter Taxas do contrato
Tipo	Dados (leitura)
Entrada	Taxas do contrato
Saída	
Origem	Cluster
Atividades	[Calcular taxa de contrato]
Descrição	A partir dos dados taxas do contrato deve-se obter taxas do contrato.
Perfil	[Credito Direto]

Nome	Obter Proposta de credito
Tipo	Dados (leitura)
Entrada	Proposta de credito
Saída	
Origem	Cluster

Atividades	[Comprometer Limite de Credito, Alterar proposta de credito]
Descrição	A partir dos dados proposta de credito deve-se obter proposta de credito.
Perfil	[Credito Direto, Analista de Credito]

Nome	Obter Créditos concedidos
Tipo	Dados (leitura)
Entrada	Créditos concedidos
Saída	
Origem	Cluster
Atividades	[Comprometer Limite de Credito, Cancelar contrato, Determinar taxas de juros a ser cobrada do cliente, Cancelar contrato de risco]
Descrição	A partir dos dados créditos concedidos deve-se obter créditos concedidos.
Perfil	[Credito Direto, Atendente, Analista de Credito]

Nome	Obter Limite de credito do cliente
Tipo	Dados (leitura)
Entrada	Limite de credito do cliente
Saída	
Origem	Cluster
Atividades	[Comprometer Limite de Credito, Verificar limite de credito do cliente]
Descrição	A partir dos dados limite de credito do cliente deve-se obter limite de credito do cliente.
Perfil	[Credito Direto, Analista de Credito]

Nome	Armazenar Taxas de juros
Tipo	Dados (escrita)
Entrada	
Saída	Taxas de juros
Origem	Cluster
Atividades	[Determinar taxas de juros a ser cobrada do cliente]
Descrição	A partir dos dados deve-se armazenar taxas de juros.
Perfil	[Credito Direto]

Nome	Obter Proposta de contrato
Tipo	Dados (leitura)
Entrada	Proposta de contrato
Saída	
Origem	Cluster
Atividades	[Gerar proposta de contrato, Cancelar contrato de risco]
Descrição	A partir dos dados proposta de contrato deve-se obter proposta de contrato.
Perfil	[Credito Direto, Analista de Credito, Atendente]

Nome	Obter Cadastro do cliente
Tipo	Dados (leitura)
Entrada	Cadastro do cliente
Saída	
Origem	Cluster
Atividades	[Cadastrar cliente]
Descrição	A partir dos dados cadastro do cliente deve-se obter cadastro do cliente.
Perfil	[Analista de Credito, Credito Direto]

Nome	Armazenar Proposta de contrato
Tipo	Dados (escrita)
Entrada	
Saída	Proposta de contrato
Origem	Cluster
Atividades	[Verificar condições de contrato com o cliente, Analisar Contrato, Cancelar contrato de risco]
Descrição	A partir dos dados deve-se armazenar proposta de contrato.
Perfil	[Atendente, Analista de Credito]

Nome	Armazenar Proposta de credito (cancelada)
Tipo	Dados (escrita)
Entrada	
Saída	Proposta de credito (cancelada)
Origem	Cluster
Atividades	[Comunicar proposta não aprovada]
Descrição	A partir dos dados deve-se armazenar proposta de credito (cancelada).
Perfil	[Atendente, Credito Direto]

Nome	Obter Proposta de contrato (aprovado)
Tipo	Dados (leitura)
Entrada	Proposta de contrato (aprovado)
Saída	
Origem	Cluster
Atividades	[Aprovar contrato]
Descrição	A partir dos dados proposta de contrato (aprovado) deve-se obter proposta de contrato (aprovado).
Perfil	[Atendente]

Nome	Obter Proposta de contrato (cancelado)
Tipo	Dados (leitura)
Entrada	Proposta de contrato (cancelado)
Saída	
Origem	Cluster
Atividades	[Cancelar contrato]
Descrição	A partir dos dados proposta de contrato (cancelado) deve-se obter proposta de contrato (cancelado).
Perfil	[Atendente]

Nome	Armazenar Créditos concedidos
Tipo	Dados (escrita)
Entrada	
Saída	Créditos concedidos
Origem	Cluster
Atividades	[Verificar limite de credito do cliente, Cancelar contrato de risco]
Descrição	A partir dos dados deve-se armazenar créditos concedidos.
Perfil	[Credito Direto, Analista de Credito]

Nome	Armazenar Limite de credito do cliente
Tipo	Dados (escrita)
Entrada	

Saída	Limite de credito do cliente
Origem	Cluster
Atividades	[Analisar Contrato]
Descrição	A partir dos dados deve-se armazenar limite de credito do cliente.
Perfil	[Analista de Credito]

Nome	Obter Cadastro do cliente (atualizado)
Tipo	Dados (leitura)
Entrada	Cadastro do cliente (atualizado)
Saída	
Origem	Cluster
Atividades	[Atualizar cadastro do cliente]
Descrição	A partir dos dados cadastro do cliente (atualizado) deve-se obter cadastro do cliente (atualizado).
Perfil	[Atendente]

Nome	Obter Taxas de juros do cliente
Tipo	Dados (leitura)
Entrada	Taxas de juros do cliente
Saída	
Origem	Cluster
Atividades	[Determinar taxas de juros a ser cobrada do cliente]
Descrição	A partir dos dados taxas de juros do cliente deve-se obter taxas de juros do cliente.
Perfil	[Credito Direto]

Nome	Armazenar Cadastro do cliente
Tipo	Dados (escrita)
Entrada	
Saída	Cadastro do cliente
Origem	Cluster
Atividades	[Verificar cadastro do cliente]
Descrição	A partir dos dados deve-se armazenar cadastro do cliente.
Perfil	[Credito Direto]

Nome	Armazenar Proposta de credito
Tipo	Dados (escrita)
Entrada	
Saída	Proposta de credito
Origem	Cluster
Atividades	[Verificar cadastro do cliente, Alterar proposta de credito]
Descrição	A partir dos dados deve-se armazenar proposta de credito.
Perfil	[Credito Direto, Analista de Credito]

Nome	Obter Analise de contrato
Tipo	Dados (leitura)
Entrada	Analise de contrato
Saída	
Origem	Cluster
Atividades	[Analisar Contrato]
Descrição	A partir dos dados analise de contrato deve-se obter analise de contrato.

Perfil	[Analista de Credito]
Nome	Obter Proposta de credito (cancelada)
Tipo	Dados (leitura)
Entrada	Proposta de credito (cancelada)
Saída	
Origem	Cluster
Atividades	[Cancelar proposta de credito]
Descrição	A partir dos dados proposta de credito (cancelada) deve-se obter proposta de credito (cancelada).
Perfil	[Credito Direto]

Tabela 9 – Serviços identificados na heurística de *Cluster*

<u>Informações</u>	<u>Serviços</u>
Nome	Cancelar proposta de credito_Co
Tipo	
Entrada	[Proposta de credito]
Saída	[Proposta não aprovada]
Origem	Atividades Sequenciais
Atividades	[Cancelar proposta de credito, Comunicar proposta não aprovada]
Descrição	A partir dos dados [proposta de credito]deve-se obter cancelar proposta de credito_co.
Perfil	[Credito Direto, Atendente]
Nome	Cancelar contrato de risco_Comu
Tipo	
Entrada	[Não aprovação do contrato de risco]
Saída	[Não aprovação do contrato de risco, Não aprovação de contrato]
Origem	Atividades Sequenciais
Atividades	[Cancelar contrato de risco, Comunicar não aprovação do contrato de risco, Comunicar proposta não aprovada]
Descrição	A partir dos dados [não aprovação do contrato de risco]deve-se obter cancelar contrato de risco_comu.
Perfil	[Analista de Credito, Atendente]
Nome	Comprometer Limite de Credito_C
Tipo	
Entrada	[Cadastro do cliente]
Saída	[]
Origem	Atividades Sequenciais
Atividades	[Comprometer Limite de Credito, Calcular taxa de contrato, Determinar taxas de juros a ser cobrada do cliente, Gerar proposta de contrato, Analisar Contrato]
Descrição	A partir dos dados [cadastro do cliente]deve-se obter comprometer limite de credito_c.
Perfil	[Credito Direto, Analista de Credito]

Tabela 10 – Serviços identificados na heurística de Atividades Sequenciais

<u>Informações</u>	<u>Serviços</u>
Nome	Comprometer Limite de Credito_G
Tipo	
Entrada	[Cadastro do cliente]

Saída	[]
Origem	Workflow Loop
Atividades	[Comprometer Limite de Credito, Gerar proposta de contrato, Determinar taxas de juros a ser cobrada do cliente, Analisar Contrato, Alterar proposta de credito, Calcular taxa de contrato]
Descrição	A partir dos dados [cadastro do cliente]deve-se comprometer limite de credito_g.
Perfil	[Credito Direto, Analista de Credito]

Tabela 11 – Serviços identificados na heurística de *LOOP*

Informações	Serviços
Nome	Cadastrar cliente_Atualizar cadastro do cliente
Tipo	
Entrada	[Proposta de credito]
Saída	[]
Origem	Workflow XOR
Atividades	[Cadastrar cliente, Atualizar cadastro do cliente]
Descrição	A partir dos dados [proposta de credito]deve-se cadastrar cliente_atualizar cadastro do cliente.
Perfil	[Credito Direto]

Nome	Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito
Tipo	
Entrada	[Cadastro do cliente, Não aprovação do contrato de risco]
Saída	[Nao aprovacao do contrato de risco, Não aprovação de contrato, Proposta não aprovada, Não aprovação de contrato, Proposta não aprovada]
Origem	Workflow XOR
Atividades	[Cancelar contrato, Cancelar proposta de credito, Aprovar contrato, Comunicar não aprovação do contrato de risco, Calcular taxa de contrato, Gerar proposta de contrato, Verificar condições de contrato com o cliente, Cancelar contrato de risco, Comprometer Limite de Credito, Determinar taxas de juros a ser cobrada do cliente, Comunicar proposta não aprovada, Comunicar proposta não aprovada, Analisar Contrato, Alterar proposta de credito]
Descrição	A partir dos dados [cadastro do cliente, não aprovação do contrato de risco]deve-se cancelar contrato_cancelar proposta de credito_aprovar contrato_comunicar nao aprovacao do contrato de risco_calcular taxa de contrato_gerar proposta de contrato_verificar condicoes de contrato com o cliente_cancelar contrato de risco_comprometer limite de credito_determinar taxas de juros a ser cobrada do cliente_comunicar proposta nao aprovada_comunicar proposta nao aprovada_analisar contrato_alterar proposta de credito.
Perfil	[Atendente, Credito Direto, Analista de Credito]

Nome	Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco
Tipo	
Entrada	[Não aprovação do contrato de risco]
Saída	[Não aprovação do contrato de risco, Não aprovação de contrato, Proposta não aprovada]
Origem	Workflow XOR

Atividades	[Cancelar contrato, Aprovar contrato, Comunicar não aprovação do contrato de risco, Comunicar proposta não aprovada, Verificar condições de contrato com o cliente, Cancelar contrato de risco]
Descrição	A partir dos dados [não aprovação do contrato de risco]deve-se cancelar contrato_aprovar contrato_comunicar nao aprovacao do contrato de risco_comunicar proposta nao aprovada_verificar condicoes de contrato com o cliente_cancelar contrato de risco.
Perfil	[Atendente, Analista de Credito]

Nome	Cancelar contrato_Aprovar contrato
Tipo	
Entrada	[]
Saída	[]
Origem	Workflow XOR
Atividades	[Cancelar contrato, Aprovar contrato]
Descrição	A partir dos dados []deve-se cancelar contrato_aprovar contrato.
Perfil	[Atendente]

Tabela 12 – Serviços identificados na heurística de XOR

<u>Serviços</u>	<u>Nº de Subflows</u>	<u>Nº de Raias</u>	<u>Cluster Associados</u>
Cadastrar cliente_Atualizar cadastro do cliente	0	1	[Proposta de credito]
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	3	3	[Cadastro do cliente, Não aprovação do contrato de risco]
Comprometer Limite de Credito_G	0	2	[Cadastro do cliente]
Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco	1	2	[Não aprovação do contrato de risco]
Cancelar contrato_Aprovar contrato	0	1	[]

Tabela 13 – Relatório de *Flows*

<u>Serviços</u>	<u>Atendente</u>	<u>Analista de Credito</u>	<u>Credito Direto</u>
Verificar cadastro do cliente			X
Determinar taxas de juros a ser cobrada do cliente			X
Calcular taxa de contrato			X
Obter Taxas do contrato			X
Obter Proposta de credito		X	X
Obter Créditos concedidos	X	X	X
Obter Limite de credito do cliente		X	X
Armazenar Taxas de juros			X
Obter Proposta de contrato	X	X	X
Obter Cadastro do cliente		X	X

Armazenar Proposta de contrato	X	X	
Armazenar Proposta de credito (cancelada)	X		X
Obter Proposta de contrato (aprovado)	X		
Obter Proposta de contrato (cancelado)	X		
Armazenar Créditos concedidos		X	X
Armazenar Limite de credito do cliente		X	
Obter Cadastro do cliente (atualizado)	X		
Obter Taxas de juros do cliente			X
Armazenar Cadastro do cliente			X
Armazenar Proposta de credito		X	X
Obter Analise de contrato		X	
Obter Proposta de credito (cancelada)			X
Cancelar proposta de credito_Co	X		X
Cancelar contrato de risco_Comu	X	X	
Comprometer Limite de Credito_C		X	X
Cadastrar cliente_Atualizar cadastro do cliente			X
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	X	X	X
Comprometer Limite de Credito_G		X	X
Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco	X	X	
Cancelar contrato_Aprovar contrato	X		

Tabela 14 – Relatório Serviços x Perfil

<u>Serviços</u>	<u>Calcular taxa de contrato</u>	<u>Verificar cadastro do cliente</u>	<u>Verificar limite de credito do cliente</u>	<u>Analisar Contrato</u>	<u>Cancelar contrato de risco</u>
Verificar cadastro do cliente		X			
Determinar taxas de juros a ser cobrada do cliente					
Calcular taxa de contrato	X				
Obter Taxas do contrato	X				
Obter Proposta de credito					
Obter Créditos concedidos					X
Obter Limite de credito do cliente			X		
Armazenar Taxas de juros					
Obter Proposta de contrato					X
Obter Cadastro do cliente					
Armazenar Proposta de contrato				X	X
Armazenar Proposta de credito (cancelada)					
Obter Proposta de contrato (aprovado)					
Obter Proposta de contrato (cancelado)					
Armazenar Créditos concedidos			X		X
Armazenar Limite de credito do cliente				X	
Obter Cadastro do cliente (atualizado)					
Obter Taxas de juros do cliente					
Armazenar Cadastro do cliente		X			
Armazenar Proposta de credito		X			
Obter Analise de contrato				X	
Obter Proposta de credito (cancelada)					
Cancelar proposta de credito_Co					
Cancelar contrato de risco_Comu					X
Comprometer Limite de Credito_C	X			X	
Cadastrar cliente_Atualizar cadastro do cliente					
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	X			X	X
Comprometer Limite de Credito_G	X			X	
Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco					X
Cancelar contrato_Aprovar					

contrato					
----------	--	--	--	--	--

Tabela 15 – Relatório Serviços x Atividades Parte I

<u>Serviços</u>	<u>Verificar condições de contrato com o cliente</u>	<u>Atualizar cadastro do cliente</u>	<u>Cancelar proposta de credito</u>	<u>Comprometer Limite de Credito</u>
Verificar cadastro do cliente				
Determinar taxas de juros a ser cobrada do cliente				
Calcular taxa de contrato				
Obter Taxas do contrato				
Obter Proposta de credito				X
Obter Créditos concedidos				X
Obter Limite de credito do cliente				X
Armazenar Taxas de juros				
Obter Proposta de contrato				
Obter Cadastro do cliente				
Armazenar Proposta de contrato	X			
Armazenar Proposta de credito (cancelada)				
Obter Proposta de contrato (aprovado)				
Obter Proposta de contrato (cancelado)				
Armazenar Créditos concedidos				
Armazenar Limite de credito do cliente				
Obter Cadastro do cliente (atualizado)		X		
Obter Taxas de juros do cliente				
Armazenar Cadastro do cliente				
Armazenar Proposta de credito				
Obter Analise de contrato				
Obter Proposta de credito (cancelada)			X	
Cancelar proposta de credito_Co			X	
Cancelar contrato de risco_Comu				
Comprometer Limite de Credito_C				X
Cadastrar cliente_Atualizar cadastro do cliente		X		
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	X		X	X
Comprometer Limite de Credito_G				X
Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao	X			

aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco				
Cancelar contrato_Aprovar contrato				

Tabela 16 – Relatório Serviços x Atividades Parte II

<u>Serviços</u>	<u>Determinar taxas de juros a ser cobrada do cliente</u>	<u>Aprovar contrato</u>	<u>Comunicar não aprovação do contrato de risco</u>	<u>Cadastrar cliente</u>
Verificar cadastro do cliente				
Determinar taxas de juros a ser cobrada do cliente	X			
Calcular taxa de contrato				
Obter Taxas do contrato				
Obter Proposta de credito				
Obter Créditos concedidos	X			
Obter Limite de credito do cliente				
Armazenar Taxas de juros	X			
Obter Proposta de contrato				
Obter Cadastro do cliente				X
Armazenar Proposta de contrato				
Armazenar Proposta de credito (cancelada)				
Obter Proposta de contrato (aprovado)		X		
Obter Proposta de contrato (cancelado)				
Armazenar Créditos concedidos				
Armazenar Limite de credito do cliente				
Obter Cadastro do cliente (atualizado)				
Obter Taxas de juros do cliente	X			
Armazenar Cadastro do cliente				
Armazenar Proposta de credito				
Obter Analise de contrato				
Obter Proposta de credito (cancelada)				
Cancelar proposta de credito_Co				
Cancelar contrato de risco_Comu			X	
Comprometer Limite de Credito_C	X			
Cadastrar cliente_Atualizar cadastro do cliente				X
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	X	X	X	
Comprometer Limite de Credito_G	X			
Cancelar contrato_Aprovar		X	X	

contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco				
Cancelar contrato_Aprovar contrato		X		

Tabela 17 – Relatório Serviços x Atividades Parte III

<u>Serviços</u>	<u>Gerar proposta de contrato</u>	<u>Cancelar contrato</u>	<u>Alterar proposta de credito</u>	<u>Comunicar proposta não aprovada</u>
Verificar cadastro do cliente				
Determinar taxas de juros a ser cobrada do cliente				
Calcular taxa de contrato				
Obter Taxas do contrato				
Obter Proposta de credito			X	
Obter Créditos concedidos		X		
Obter Limite de credito do cliente				
Armazenar Taxas de juros				
Obter Proposta de contrato	X			
Obter Cadastro do cliente				
Armazenar Proposta de contrato				
Armazenar Proposta de credito (cancelada)				X
Obter Proposta de contrato (aprovado)				
Obter Proposta de contrato (cancelado)		X		
Armazenar Créditos concedidos				
Armazenar Limite de credito do cliente				
Obter Cadastro do cliente (atualizado)				
Obter Taxas de juros do cliente				
Armazenar Cadastro do cliente				
Armazenar Proposta de credito			X	
Obter Analise de contrato				
Obter Proposta de credito (cancelada)				
Cancelar proposta de credito_Co				X
Cancelar contrato de risco_Comu				X
Comprometer Limite de Credito_C	X			
Cadastrar cliente_Atualizar cadastro do cliente				
Cancelar contrato_Cancelar proposta de credito_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Calcular taxa de contrato_Gerar proposta de contrato_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco_Comprometer Limite de Credito_Determinar taxas de juros a ser cobrada do cliente_Comunicar proposta nao aprovada_Comunicar proposta nao aprovada_Analisar Contrato_Alterar proposta de credito	X	X	X	X

Comprometer Limite de Credito_G	X		X	
Cancelar contrato_Aprovar contrato_Comunicar nao aprovacao do contrato de risco_Comunicar proposta nao aprovada_Verificar condicoes de contrato com o cliente_Cancelar contrato de risco		X		X
Cancelar contrato_Aprovar contrato		X		

Tabela 18 – Relatório Serviços x Atividades Parte IV