Universidade Federal do Espírito Santo
Programa de Pós-Graduação em Informática

Uma Arquitetura para Integração de Processos de Negócio e Processamento de Situações Contextuais

Luis Augusto Melo Rohten

Vitória - ES, Junho de 2019

Luis Augusto Melo Rohten

Uma Arquitetura para Integração de Processos de Negócio e Processamento de Situações Contextuais

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática da UFES, como parte dos requisitos necessários para a obtenção do Título de Mestre em Informática.

Universidade Federal do Espírito Santo – UFES

Programa de Pós-Graduação em Informática - PPGI

Orientador: José Gonçalves Pereira Filho

Vitória - ES

Junho de 2019

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Rohten, Luis Augusto Melo, 1989-
R737a Uma Arquitetura para Integração de Processos de Negócio e Processamento de Situações Contextuais / Luis Augusto Melo Rohten. - 2019.
155 f. : il.

Orientador: José Gonçalves Pereira Filho.
Dissertação (Mestrado em Informática) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Negócios - Processamento de dados. 2. Negócios - Programas de computador. 3. Armazenamento de dados. 4. Programação de sistemas (Computação). 5. Análise de sistemas. I. Pereira Filho, José Gonçalves. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 004

Luis Augusto Melo Rohten

Uma Arquitetura para Integração de Processos de Negócio e Processamento de Situações Contextuais

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática da UFES, como parte dos requisitos necessários para a obtenção do Título de Mestre em Informática.

Trabalho aprovado. Vitória - ES, 18 de Junho de 2019:

José Gonçalves Pereira Filho
Orientador

Vinicius Mota
Membro Interno

Pedro Frosi Rosa
Membro Externo

Vitória - ES
Junho de 2019

A minha mãe Maria e minha irmã Louise.

Agradecimentos

Agradeço primeiramente a Deus, por me proporcionar a continuação dos meus estudos e, me permitir mais uma conquista na minha vida.

Agradeço aos meus pais, Claúdio (In memoriam), Maria de Lourdes e minha irmã Louise pelo apoio, carinho e amor que foram fundamentais em minha vida e jornada acadêmica.

Agradeço a minha companheira, Maria Eduarda por todo amor, carinho e paciência nas etapas finais deste trabalho.

Agradeço ao meu orientador Prof. Dr. José Gonçalves Pereira Filho, que me orientou nesta jornada e me apoiou para que eu pudesse finalizar mais esta etapa.

Apesar de ser difícil citar todos, agradeço aos amigos, demais estudantes, professores, funcionários da universidade, que fizeram parte da minha vida de alguma forma e foram importantes em minha formação acadêmica e convivência profissional.

"A ciência nunca
resolve um problema
sem criar pelo
menos outros dez."
(George Bernard Shaw)

Resumo

Nos últimos anos, mecanismos de detecção de situações contextuais vêm sendo explorados para análise de domínios específicos dentro das organizações, permitindo que sistemas de informação (SI) passem a trabalhar com dados *sensíveis a contexto*. Esses sistemas buscam detectar um particular estado de interesse da realidade (por exemplo, alterações em condições do domínio) e projetar uma reação à situação detectada. Um obstáculo para uma representação de domínio é o seu entendimento efetivo como um todo, sendo muitas vezes necessário que várias pessoas dentro da organização passem a entender o domínio para se propor uma solução. Uma notação comumente utilizada dentro das organizações para a representação de domínios é a Business Process Model and Notation (BPMN).

Neste trabalho, é descrita uma arquitetura conceitual focada na representação de situações contextuais por meio de *regras de negócio* que tiram proveito de processos de negócio. Uma abordagem de repositório de processos foi introduzida na arquitetura com o objetivo de armazenar processos, dados de processos e serviços da organização. Como prova de conceito, um experimento empírico foi realizado para validar à abordagem proposta. O trabalho desenvolvido mostrou que aliar a notação de processos de negócio BPMN com abordagens sensíveis à situação, pode diminuir a complexidade de sistemas e dos processos de negócio, bem como aumentar a confiabilidade no modelo de representação do domínio.

Palavras-chaves: Processos de Negócio. Regras de Negócio. BPMN. Detecção de Situações. Sensibilidade a Situações. Repositório de Processos de Negócio.

Abstract

In recent years, the concept of situation-awareness has been exploited by organizations, allowing their information systems to work with context-sensitive data. These systems seek to detect a particular state of interest in reality, i.e., a *contextual situation*, and react accordingly, thus contributing to improve business adaptability to changes in domain conditions.

In this work, we describe a conceptual architecture focused on the representation of contextual situations through *business rules*, and their integration with business processes. As a proof of concept, an empirical experiment was leaded to validate the proposal. The work developed showed that bringing situation-awareness to information systems can reduce the complexity of business processes, as well as increase the reliability of the domain representation model.

Keywords: Business Process. Business Rules. BPMN. Situation Detection. Situation-Awareness. Business Process Repository.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

TIC	Tecnologias da Informação e Comunicação
BPMS	Business Process Management Systems
BPM	Business Process Management
BRMS	Business Rule Management Systems
ECA	Evento-Condição-Ação
BPMN	Business Process Model and Notation
SOA	Service-Oriented Architecture
BPMS	Business Process Management Systems
BP	Business Process
CBOK	Guide to the Business Process Management Body of Knowledge
BPMI	Business Process Management Initiative
OMG	Object Management Group
BPD	Business Process Diagram
WM	Working Memory
RB	Rule Base
KB	Knowledge Base
IE	Inference Engine
RE	Rule Engine
EE	Execution Engine
SA	Situation-awareness
CEP	Complex Event Processing
DRL	Drools Rule Language
DBP	Dynamic Business Process

GD	Gerenciamento de Desastres
SI	Sistema de Informação
CBP	Collaborative Business Process
BPI	Business Process Integration
RBAC	Role-Based Access Control
JWT	JSON Web Token
SVG	Scalable Vector Graphics
LHS	Left Hand Side
RHS	Right Hand Site
PVM	Process Virtual Machine
API	Application Programming Interface
NLP	Natural Language Processsing
EPC	Event-driven Process Chain
UML	Unified Modeling Language

Sumário

1 Introdução

Este capítulo apresenta a motivação, os objetivos e a organização deste trabalho. São introduzidas as áreas de pesquisa nas quais o trabalho se apoia e descrito o problema central a ser investigado. As tecnologias usadas na solução são brevemente apresentadas, as quais são detalhadas nos capítulos que seguem.

1.1 Motivação

As últimas décadas presenciaram um aumento significativo do embate competitivo entre as organizações coorporativas, particularmente em busca de mercados transnacionais e mais rentáveis. Esta competição ficou cada vez mais acirrada visto que, em uma perspectiva global, a evolução tecnológica e os meios de comunicação permitiram que as empresas competissem cada vez mais entre si independentemente da sua localização geográfica, aumentando cada vez mais a dependência tecnológica para dar suporte a toda gestão dos seus processos de negócio.

Com isso, tornou-se primordial para as organizações a busca sistemática por ferramentas de auxílio à melhoria contínua de processos apoiadas na aplicação de modernas técnicas de Administração Científica (ROSEMANN, 2006) e no uso intensivo de sistemas de TIC (Tecnologias de Informação e Comunicação) (ZOET et al., 2011). Por exemplo, na busca por melhorias de processos suportados por novas tecnologias, consolidaram-se conceitos de reengenharia de processos, sistemas integrados de gestão, benchmarkings, gerenciamento total da qualidade, downsizing e, de forma mais recente, os Sistemas de Gerenciamento de Processos de Negócio (Business Process Management Systems – BPMS) (BALDAM et al., 2007; MAIO; SALATINO; ALIVERTI, 2014; FIORINI; LEITE; LUCENA, 2001; POLPINIJ; GHOSE; DAM, 2015; ROSEMANN, 2006)

Um outro fator importante no meio organizacional, e foco de muitas pesquisa nos anos recentes, são as regras de negócio. As regras de negócio são declarações usadas para definir ou restringir alguma ação nos processos de negócio. Elas guiam comportamentos, descrevendo como determinadas operações devem ser realizadas e se há algum limite que precisa ser aplicado. As regras de negócios estão rapidamente ganhando popularidade como meio de separar a lógica de negócios dos processos e aplicativos operacionais. Eles permitem a especificação do conhecimento comercial de uma maneira que seja compreensível pelo negócio, mas também executável pelos mecanismos ou “engines” de regras, contemplando, assim, a lacuna entre os negócios e a tecnologia (VASILECAS; KALIBATIENE; LAVBIČ, 2016).

As regras de negócios e a modelagem de processos de negócios (BP – Business Process) se tornaram um tópico de pesquisa importante nas últimas duas décadas, dando origem a muitas ferramentas e ambientes que fornecem recursos para melhor gerenciar as regras em sistemas baseados em processos de negócio (HERBST, 1996; STRUCK, 1999; ANTHES, 2003; TEMPORA..., 1994; MORIARTY, 2000; ROSEMANN, 2006). Os Business Rule Management Systems (BRMS) são sistemas de software usados para definir, desenvolvedor, executar, monitorar e manter a variedade e a complexidade da regra de negócio em uma organização ou empresa (BALI, 2009). Alguns BRMS inseridos no mercado, suportam a integração com ferramentas BPMS, permitindo que regras de negócio tirem proveito de processos de negócio ou vice-versa.

As regras de negócio normalmente são definidas nos BRMS's como regras descritas por meio do padrão ECA (Evento, Condição, Ação), onde na ocorrência de um evento e na satisfação uma dada condição, uma determinada ação é gerada, a qual executa rotinas pré-definidas no SI, podendo ser a execução de um serviço, ou até mesmo de processo de negócio definido em alto nível e representado em linguagem de Business Process Model and Notation (BPMN) por meio de BPMS definidos dentro da organização.

As regras de negócio são muito sensíveis às mudanças nos negócios, o que exige um tratamento explícito durante o desenvolvimento do SI para garantir esta agilidade. Alguns dos problemas comumente encontrados são: (i) regras de negócio modeladas de forma arbitrária, sem um cuidado sistemático e um entendimento completo de suas necessidades, propagam regras que não refletem as condições reais do ambiente de negócios e, consequentemente, são desenvolvidos aplicativos que não atendem às necessidades da organização; (ii) falta de documentação sobre as regras de negócios; (iii) regras de negócios embutidas no código do programa, não ficando claro que tipo de regra rege um aplicativo, em que momento as regras são acionadas e como elas são implementadas; (iv) dificuldade de manter a lógica de negócios pois as regras podem estar distribuídas pela lógica do aplicativo; (v) regras de negócios difíceis de controlar, já que não há um repositório de propósito comum e único para elas.

A integração entre regras de negócio e processos de negócio permite que se passe a executar processos de negócio automatizados de forma recorrente, gerando cada vez mais dados e informações a serem analisados pela organização. Fatores importantes, conhecidos como indicadores de processos, tais como o desempenho, a eficácia, a eficiência, entre outros, também passam a ser analisados mais de perto pelas aplicações e pela organização como um todo e, com isso, ganha importância o armazenamento dos modelos de processos de negócios e seus dados em repositórios de processos (vide adiante).

Mais recentemente, como uma extensão aos sistemas baseados de regras, o conceito de situation-awareness vem ganhando destaque na comunidade de processos de negócio. Uma situação é uma abstração de eventos do mundo real derivado de contextos relevantes

e os relacionamentos entre eles, e hipóteses sobre como um particular estado da realidade se relaciona com os interesses dos stakeholders e aplicações. Em outras palavras, uma situação é um estado das coisas da realidade ("state of affairs") que é de particular interesse de usuários e aplicações (ADI; BOTZER; ETZION, 2000; YE; DOBSON; MCKEEVER, 2012).

Diferentemente de eventos, que se iniciam e terminam num dado instante e são detectados somente quando ocorrem – e disparam ações em um sistema de gerenciamento de regras tradicional - uma situação pode ser detectada no primeiro instante em que ela ocorre, sem necessariamente se saber quando ou se ela terminará. Eventos não suportam o conceito de "acontecendo", que é parte intrínseca do conceito de situação. Quando ações são derivadas do uso desta tecnologia, processos e usuários podem tomar decisões baseadas num monitoramento contínuo da realidade observada.

A tecnologia de situation awareness tráz, portanto, para dentro das operações das organizações e, particularmente, para o seu ambiente de processos de negócio, uma visibilidade real-time e continuada do fragmento de realidade de interesse. Incorporar, portanto, uma plataforma de gerenciamento de situações ao ambiente de processo de negócios é acrescentar uma camada de inteligência no topo dos atuais sistemas coorporativos baseados em regras.

Embora necessário, o conjunto de conhecimentos sobre esta integração ainda é limitado (WANG; INDULSKA; SADIQ, 2014). Com isso, mecanismos de detecção de situações vêm sendo desenvolvidos para suportar a definição de situações a partir de regras de negócio (COSTA et al., 2016), tendo em vista a sua grande importância e aplicabilidade nos inúmeros domínios a serem explorados dentro das organizações. Sistemas sensíveis a situação (situation-aware systems) buscam detectar, em tempo real ou quase isso, um particular estado de interesse da realidade – por exemplo, alterações em condições do domínio – e projetar uma reação à situação detectada (MOREIRA et al., 2015b), permitindo que processos de negócio sejam melhorados continuamente e que sejam aderentes a mudanças de contexto, constituindo uma funcionalidade interessante para a necessidade estratégica e uma vantagem competitiva para as organizações. Em (Pereira; 2013) (PEREIRA; COSTA; ALMEIDA, 2013) é proposta uma abordagem de suporte à situações contextuais por meio de definições de regras, que permite a análise, ao longo do tempo, de estados de interesse da realidade, ou seja, de situações do mundo real monitoradas por usuários e aplicações.

Observe que a integração entre regras de negócio e processos de negócio permite que se passe a executar processos de negócio automatizados de forma recorrente, gerando cada vez mais dados e informações a serem analisados pela organização. Com isso, ganha importância o armazenamento dos modelos de processos de negócios e seus dados em repositórios de processos. Ferramentas de software (CHOI; KIM; JANG, 2007; FIORINI;

LEITE; LUCENA, 2001) foram desenvolvidas para ajudar a pré-formatar tais tarefas; no entanto, elas foram especializadas para armazenar processos de negócios usando modelos conceituais, por exemplo, esquemas de bancos de dados, que são específicos do processo, e definindo interfaces específicas deste particular processo. Essas interfaces poderiam, por exemplo, assumir a forma de uma interface de serviço da Web ou uma API, nas quais modelos de processo pudessem ser importados ou exportados "livremente". Sistemas BPMS e BRMS poderiam tirar vantagens de abordagens orientadas a serviços (SOA - Service-Oriented Architecture) na implementação de repositórios de processos de negócio, a partir dos quais eles realizariam o consumo de processos de negócio por meio de requisição/resposta a serviços (BALCER, 2006). Como vantagens desta abordagem, seria possível uma maior separação de papéis entre os especialistas de processos de negócio e os desenvolvedores. Os primeiros se preocupariam apenas com a modelagem dos processos de negócio, ou seja, as questões referentes ao entendimento dos processos estariam diretamente nas mãos dos profissionais que possuem a expertise dos processos e do conhecimento do domínio, enquanto os últimos – os desenvolvedores – ficariam com os aspectos de baixo nível necessários para o disparo destes processos, ou seja, os desenvolvedores se preocupariam apenas com aspectos de implementação, que mediante a ocorrência de um evento ou situação de interesse, realizariam uma requisição à API de processos e daria início à execução do processo em questão.

Em suma, o cenário descrito acima traz questões importantes a serem discutidas: (i) as aplicações que envolvem regras de negócio e processos de negócio tem de lidar com muitos aspectos no processo de desenvolvimento das aplicações, envolvendo diferentes níveis de abstração; (ii) a participação dos especialistas em processos de negócio, bem como sua interação no processo de desenvolvimento deve ser feita de forma constante ao longo de todo o processo de desenvolvimento do software, uma vez que nem sempre os desenvolvedores possuem conhecimento em modelos de processos de negócio e profundo conhecimento do domínio; (iii) os serviços e processos com mesma representação de domínio dentro das organizações normalmente são redundantes e variam de aplicação para aplicação, deixando de lado questões que agregam na cultura da organização, como a melhoria continuada do processo, ao passo que eles poderiam ser mapeados e consumidos, em tese, por quaisquer aplicações que tivessem a mesma representatividade e sempre atualizados e melhorados, representando e sendo reaproveitados e domínios de aplicações distintas.

Para abordar os itens (i) e (ii) uma das estratégias é a "separação das preocupações" ("separation of concerns"), que é muito utilizada pela comunidade de engenharia de software (MITCHELL, 1990), considerando, duas visões: a primeira se refere às preocupações dos desenvolvedores com relação às regras de negócio em baixo nível para a criação de soluções de domínios específicos, se utilizando muitas vezes de dados advindos de web services, redes de sensores, feedback de membros da organização, entre outros; a segunda está relacionada especificamente com os modeladores de domínio, ou especialistas de processos

de negócio, que possuem como foco principal entender as necessidades dos stakeholders acerca do negócio e propor soluções eficientes para problemas específicos. Esta separação de preocupações permite distinguir claramente os papéis de desenvolvedor e especialistas de processos de negócio, o que poderia criar uma maior independência entre as áreas e potencializar aspectos de desenvolvimento e de melhoria dos processos. Evidentemente, os especialistas de processos de negócio atuariam apenas nas especificações de domínio, modelagem de processos, melhoramento continuado dos processos, análise de desempenho, monitoramento e cuidariam de novas demandas

A questão abordada em (iii) levanta um ponto importante: a necessidade do reaproveitamento de processos de negócio para evitar processos redundantes. Uma forma de evitar a criação de processos redundantes é quebrar processos longos em pequenos processos para serem consumidos/disparados um ao término do outro, evitando a necessidade de reimplementação de processos similares e com pequenas variações.

Assumindo que os processos de negócio são armazenados de maneira adequada nos repositórios e que os processos que são executados de forma automatizada pela organização acabam gerando um grande volume de dados para satisfazer demandas decorrentes da execução das aplicações desenvolvidas, essas demandas de domínios potencialmente distintos acabam por ampliar o universo de armazenamento dos processos alvo e dos dados gerados pelos mesmos, permitindo um maior reaproveitamento e evitando implementações redundantes ou desnecessárias

Em tal cenário de demanda crescente por processos e serviços, a implantação de uma visão computacional baseada em serviços pode preparar a infraestrutura de suporte a processos de negócio para problemas futuros que envolvem escalabilidade. Os BPMS's ou BRMS's podem e devem tirar vantagens de abordagens orientadas a serviços (SOA - Service-Oriented Architecture), por exemplo, na implementação de repositórios de processos de negócio, a partir dos quais sistemas BRMS's ou BPMS's realizam o consumo de processos de negócio por meio de requisição/resposta a serviços. (ROSEN et al., 2012; BALCER, 2006).

Um mapeamento sistemático da literatura buscando infraestruturas de apoio à processos de negócio, realizado nesta dissertação, mostrou que, embora várias pesquisas tenham sido conduzidas em direção à integração de processos e regras de negócio, como pode ser visto em (MATTOS et al., 2014; MAIO; SALATINO; ALIVERTI, 2014; GATTI; SANTORO; NUNES, 2010; HUANG; TAO, 2004; MOREIRA et al., 2015b; VASILECAS; KALIBATIENE; LAVBIČ, 2016), poucas contam com uma infraestrutura para definição e controle de situações contextuais em design-time e runtime aliada a uma abordagem de repositórios de processos de negócio (ROSA et al., 2011; LAZARTE et al., 2013; POLPINIJ; GHOSE; DAM, 2015; YAN; DIJKMAN; GREFEN, 2012). Os trabalhos discutidos acima pouco exploram os seguintes aspectos de maneira integrada, são eles: (i) a separação de

preocupações entre as fases de desenvolvimento e modelagem de processos; (ii) o uso de uma abordagem baseada em situações para descrever regras de negócio, que permita monitorar eventos que ocorrem ao longo do tempo; (iii) o uso de ferramentas drag-and-drop para a modelagem de processos de negócio em alto nível que seja capaz de gerar código XML que possa ser interpretado por aplicações de baixo nível, por exemplo, jBPM, Oracle BPM, Activiti, entre outros; e (iv) o uso de conceitos de Service-Oriented Architecture (SOA) para tratar aspectos de armazenamento de processos de negócio, seus dados e serviços, permitindo um mapeamento sistemático da organização por meio destes de maneira escalável e distribuída.

1.2 Objetivos

O objetivo geral deste trabalho é definir uma arquitetura conceitual de apoio ao desenvolvimento de aplicações de Sistemas de Informação levando em consideração os aspectos acima mencionados, os quais foram derivados do mapeamento sistemático da literatura da área. O trabalho está centrado na proposta de uma arquitetura conceitual flexível e escalável que integra as abordagens de processos de negócio e sensibilidade à situação e visa simplificar a construção de aplicações organizacionais que consomem dados de diversos domínios para a tomada de decisão. A organização da arquitetura é tal, que evita o crescimento desnecessário de processos e serviços redundantes, isto é, os processos passam a ser modelados e entregues em tamanhos menores, permitindo um maior reaproveitamento dos processos já existentes, bem como sua melhoria ao longo do tempo.

A arquitetura deverá propor uma abordagem de separação de responsabilidades entre os desenvolvedores e os especialistas de processos de negócio envolvidos na compreensão do domínio, ou seja, a proposta deverá tomar como base o conceito de separação de papéis, neste caso centrada na separação das responsabilidades pela definição de aspectos de alto nível dos modelos de processos de negócio e aspectos de baixo nível presentes na definição de análise de ocorrência de eventos e situações de interesse.

A implementação da arquitetura deverá ser baseada na abordagem SOA e utiliza uma plataforma de gerenciamento de situações como ambiente de apoio à definição de regras de negócio, além de empregar uma engine workflow de processos e uma infraestrutura programática para a definição de serviços e repositório de processos de negócio para o armazenamento de modelos e dados dos processos.

1.3 Organização do Trabalho

Além desta Introdução, o trabalho está organizado em mais cinco capítulos, a saber:

• Capítulo 2: apresenta um breve referencial teórico abordando os principais conceitos e tecnologias usadas no desenvolvimento da dissertação.

• Capítulo 3: discute os trabalhos relacionados.

• Capítulo 4: descreve o projeto da arquitetura conceitual e detalhes da implementação da arquitetura proposta.

• Capítulo 5: apresenta um cenário de uso do protótipo desenvolvido.

• Capítulo 6: conclui a dissertação, apresentando as considerações finais e as perspectivas de continuidade do trabalho.

Parte I

Preparação da pesquisa

2 Referencial Teórico

Este capítulo introduz os principais tópicos de pesquisa relacionadas ao trabalho. O capítulo está organizado da seguinte forma: as Seções 2.1 a 2.3 apresentam conceitos e definições relacionadas aos processos de negócio, ao gerenciamento de processos de negócio e à notação BPMN, padrão de facto de linguagem usada pelas organizações para descrever processos de negócio em alto nível; a Seção 2.4 apresenta uma breve visão sobre os repositórios de processos, que são de suma importância para o armazenamento dos processos dentro da organização; a Seção 2.5 discute as arquiteturas orientadas a serviços (SOA), as quais possibilitam que micro serviços sejam instanciados e rodem de maneira independente, permitindo que sejam consumidos por outras aplicações ou usuários como uma caixa preta no que tange sua implementação; as Seções 2.6 e 2.7 introduzem, respectivamente, os sistemas baseados em regras e as situações contextuais, paradigma e tecnologia de programação usados nesta dissertação; e, por fim, a Seção 2.8 traz as considerações finais do capítulo.

2.1 *Business Process*

O conceito de Processo de Negócio (*Business Process*) pode ser introduzido separando os dois termos, processo e negócio:

(i) Processo nada mais é do que o conjunto de atividades ou comportamentos desempenhados por máquinas ou humanos para se alcançar metas. Um processo pode ser disparado ou executado por um ou mais eventos específicos, realizando um conjunto de tarefas e por fim se encerrar ou disparar um outro processo.

(ii) Negócio por sua vez, é a realização de uma atividade que entrega valor ao cliente trazendo retorno a(s) parte(s) interessada(s), podendo ser com ou sem fins lucrativos.

Este conceito é basicamente uma definição em alto nível, associada a atividades finalísticas de uma organização ou empresa. Esta definição demonstra que um processo de negócio nada mais é do que uma atividade ponta-a-ponta, que entrega valor ao cliente.

Um processo é um conjunto de atividades que objetivam transformar insumos (entradas), por meio de procedimentos, em bens ou serviços (saídas), com o objetivo de entregar valor ao cliente (CRUZ, 2003). De acordo com (BALDAM et al., 2007), há diversas perspectivas com relação ao termo processo, que está presente nas mais diversas áreas e atividades do cotidiano; contudo, em todas elas, o propósito de qualquer processo é transformar uma entrada qualquer em uma ou mais saídas. Entretanto, quando o processo é aplicado a um negócio ou organização, torna-se mais usual a utilização dos termos

processo de negócio ou business process, no qual o conjunto de atividades produzem valor para um grupo de interessados.

Sendo assim, segundo (BENITEZ, 2006), um *Business Process* (BP) é uma representação formal do trabalho realizado por humanos ou sistemas de uma organização, objetivando gerar um serviço ou produto para as partes interessadas (*stakeholders*). Com isso, um BP pode ser compreendido como um conjunto de tarefas envolvendo pessoas e/ou recursos para se atingir objetivos previamente definidos, gerando assim, um produto ou serviço de valor para o cliente. O conjunto de tarefas pode ser descrito como um conjunto de atividades e passos unidos para o desenvolvimento de uma tarefa ou produto, podendo ser automatizado total ou parcialmente ou que necessite de alguma ação humana ao decorrer do processo para o prosseguimento do seu fluxo (ex. aprovação de um diretor, responsável, etc.), podendo variar o fluxo a ser executado dependendo dos dados ou informações fornecidos, ou até mesmo na dependência de execução de outro processo ou subprocesso no decorrer do processo.

Em (RUMMLER; BRACHE, 1995) os BP são definidos como uma série de etapas para produzir um serviço ou produto. Foram descritos dois tipos de processos chamados processos primários e processos de suporte. Os processos primários são os processos no qual o cliente externo é envolvido, já os processos de suporte são os processos descritos como invisíveis ao cliente externo, mas essenciais à gerência e eficácia do negócio. Como descrito e ilustrado em (LAURINDO; ROTONDARO, 2006), os BP são aqueles processos que caracterizam a atuação da empresa e são apoiados por outros processos internos, possibilitando como resultado um serviço ou produto de utilidade interna da própria organização que gera valor a parte interessada ou um resultado que é recebido por um cliente externo. Um BP envolve diversas funções e atividades necessárias para se produzir um resultado.

Um BP pode ser definido também como um conjunto de atividades projetadas para a produção de uma saída específica para um cliente, ou até mesmo para um negócio específico, como descrito em (DAVENPORT, 1993). O autor ainda descreve que um processo é especificado ao longo do tempo e espaço, com um começo e fim, e com entradas e saídas definidas.

Desta forma, é notória que na definição de BP alguns conceitos devem ser considerados imprescindíveis, como ter entradas e saídas bem definidas, partes interessadas, as transformações que ocorrerem ao longo do tempo e espaço que agregam valor a atividade ou produto, responsável claramente definido, podem ser automatizados por máquinas e/ou executados por ações humanas e devem acompanhar os negócios da organização.

A perspectiva de um processo deve demonstrar uma visão horizontal do negócio, podendo envolver toda a organização ou parte dela, permitindo ou não melhorias significativas ao longo do tempo, fazendo com que o resultado de um processo possa dar início

a outro processo, tomando como base o resultado do processo anterior como entrada do processo subsequente, possibilitando que o processo passe por várias etapas de controles dentro de uma organização.

2.2 *Business Process Management*

O Gerenciamento de Processos de Negócio (BPM – *Business Process Management*), de acordo com o BPM CBOK, nada mais é do que uma abordagem de descoberta, projeto e a entrega de processos de negócio. Seguindo as etapas de identificação, desenho, execução, documentação, medição, monitoramento, controle e melhoria de processos, tais processos podem ser automatizados ou não para se obter resultados alinhados com as metas estratégicas de uma organização. O objetivo principal do BPM é alinhar os processos de negócio da organização, otimizando a performance, eliminando a(s) redundância(s) de acordo com as necessidades das partes interessadas. O BPM envolve uma atuação de forma deliberada, colaborativa e cada vez mais assistida por tecnologias, melhorias, inovações e gerenciamento de processos ponta-a-ponta, objetivando atingir os resultados do negócio com maior agilidade (CBOK, 2013). O ciclo do BPM pode ser visto na Figura 1.

Figura 1 – Ciclo BPM - Fonte ABPMNP

Além dos conceitos básicos e fundamentais, o BPM pode ser descrito como um conjunto de tecnologias habilitadoras que abordam um trabalho ponta-a-ponta diferenciado entre conjuntos de subprocessos, tarefas, atividades e funções, sendo um conjunto contínuo

e em curso de processos com foco no gerenciamento de processos de negócio ponta-a-ponta nas organizações incluindo modelagem, análise, desenho e medição de processo da organização. Isto requer um compromisso significativo da organização que introduz novos papéis, responsabilidades e estruturas às organizações tradicionais orientadas a funções de forma frequente, através de ferramentas para se modelar, simular, automatizar, integrar, controlar e monitorar os BP junto a tecnologias de sistemas de informação que dão suporte à esses processos (CBOK, 2013).

Tomando como início um contexto histórico, estudos realizados em 2003 por (SMITH; FINGAR, 2003) caracterizaram o *Business Process Management* (BPM) em três ondas, no qual a primeira onda se deu em meados de 1920 por Frederick Winslow Taylor através de seu movimento conhecido como administração científica ou gerenciamento científico. Os funcionários e patrões possuíam papéis bem definidos dentro das organizações, os processos eram implícitos na execução dos trabalhos e a ênfase da gestão era acerca da produtividade de cada funcionário. Neste contexto, Taylor acreditava que caso os funcionários recebessem treinamentos, aumentariam sua produtividade e sua qualidade, através da sua especialização, criando determinadas habilidades na execução de suas tarefas. Este princípio foi tomado como base para a estratégia do sistema de produção em massa de Ford, que devido a sua escalabilidade foi necessário um maior controle, surgindo, assim, a supervisão funcional focada em verificar o tempo das tarefas executadas. A segunda onda se deu por volta de 1990, em um trabalho de Michael Hammer que teve como foco a reengenharia, tendo como ideia central a melhoria drástica do desempenho das empresas através de mudanças radicais nas operações, inclusive por meio do termo que ajudou a popularizar, o Downsizing. A popularização do Business Process Reengineering disseminou-se durante a década de 90, assim como outras técnicas de melhorias de processos e *workflows* centrados em documentos (*kaizen*, *just-in-time*, *kanban*, etc). Entretanto, a reengenharia carecia de flexibilidade e agilidade para atender as mudanças internas e externas. Por fim, em 2003, (SMITH; FINGAR, 2003) lançaram um trabalho conhecido como *Business process management: the third wave*, que tratou de um modelo que possibilita que organizações e colaboradores criem e otimizem seus BP em tempo real, e no qual, através dos processos ágeis, as cadeias de valor podem ser continuamente monitoradas e melhoradas.

A terceira onda da BPM é cercada de ideias que são passíveis de entendimento. Sua palavra chave é flexibilidade. Nela, a habilidade para mudar o processo passa a ser mais relevante do que a habilidade para criá-lo, pois ela gera condições para que toda a cadeia de valor seja monitorada, continuamente melhorada e otimizada. Neste âmbito, o BPM deve possibilitar meios de colocar processos concebidos em prática, métodos sistemáticos e confiáveis para analisar impacto de inovações nos processos de negócio, habilidades para responder alterações no mercado, combinando e customizando processos.

De acordo com o *BPM CBOK*, que é um guia da ABPMP de conhecimento comum

sobre BPM, o BPM implica em gerenciamento contínuo dos processos da organização, que reflete mudanças constantes das atividades desempenhadas no dia-a-dia; sendo assim, a diversidade de fatores e os pilares do BPM são fundamentais para se alcançar o comprometimento organizacional com o BPM. O aspecto de liderança organizacional é um dos principais e talvez o mais importante desses pilares, podendo influenciar até mesmo na cultura da organização e possuir a autoridade para implementar mudanças (CBOK, 2013).

2.3 Modelo e Notação de Processos de Negócio

O *Business Process Model and Notation* (BPMN) (SCHILIT; THEIMER, 1994; TENG, 2009) ou Modelo e Notação de Processos de Negócio foi desenvolvida pela *Business Process Management Initiative* (BPMI) e é atualmente mantido pela *Object Management Group* (OMG), devido à fusão das duas organizações em 2005. A versão atual do BPMN é a 2.0, lançada em março de 2011. O BPMN é uma notação padrão que representa processos de negócios por meio de diagramas de processos de negócio (*Business Process Diagram* - BPD). Dizemos também que a notação é orientada para uso humano, devido a fácil compreensão do diagrama formado, pois este faz uso de ícones padrões que facilitam o entendimento. A norma BPMN num todo está restrita aos conceitos de modelagem aplicáveis aos processos de negócios, conforme a sua sigla nos indica, ou seja, outro tipo de modelo criado para outros fins não está no escopo dessa nomenclatura.

Desenhos de estruturas organizacionais, modelos de dados e organogramas, não podem ser feitos seguindo a notação BPMN, dado que essa linguagem não está voltada para este fim, mesmo que as ferramentas e a própria linguagem nos dê elementos como as atividades e eventos. Estes elementos tem uma finalidade e não tem o propósito de substituir, por exemplo, elementos do desenho de uma estrutura organizacional.

Basic Categories:	Flow Objects	Data	Connecting Objects	Swimlanes	Artifacts
Elements	Events, Activities and Gateways	Data Objects, Data Inputs, Data Outputs and Data Stores	Sequence Flows, Message Flows, Associations and Data Associations	Pools and Lanes	Group and Text Annotations
Sample symbols	Start Event End Event Task Gateway Collapsed Sub-Process	Data Object Data Store	Sequence Flow Message	Pool	Text Annotation Group

Figura 2 – Cinco categorias básicas dos elementos da notação BPMN 2.0 (TENG, 2009)

.

O BPMN é um padrão e uma notação gráfica que foi proposto inicialmente em 2004 com o objetivo de permitir a modelagem de BP de forma simples, flexível, consistente e mais adequada a modelagem da camada lógica dos negócios de tal forma que pode ser facilmente utilizada por profissionais não especialistas da área de TI. BPMN usa uma linguagem gráfica estruturada por meio de um conjunto de símbolos padronizados para representar os processos por diagramas. A Figura 2 apresenta cinco categorias básicas de elementos da notação BPMN 2.0 (TENG, 2009).

O BPMN é uma linguagem mundialmente adotado e projetado para a modelagem de BP. Por se tratar de um padrão já consolidado e bastante utilizado, existem várias soluções de software que auxiliam o desenho e automação de BP, principalmente, integrado às soluções e arquiteturas empresariais utilizadas pelos grandes fornecedores de solução de ERP como a SAP e TOTVS.

2.4 Repositório de Processos de Negócio

Atualmente a maior parte das organizações precisa lidar com o gerenciamento de centenas e até mesmo milhares de processos de negócio. De acordo com (ROSA et al., 2011), "os modelos de processos de negócios estão se tornando disponíveis em grande número devido ao seu amplo uso em muitas aplicações industriais, como projetos de engenharia de qualidade e de empresas".

Gerenciar essa grande quantidade de processos de uma organização é uma tarefa difícil e uma conquista para as organizações que conseguem realizar esta tarefa com excelência. Os processos fazem parte das atividades do dia-a-dia da organização e são frequentemente modificados, atualizados, executados, acessados. Os softwares podem auxiliar no desenvolvimento destas tarefas, possibilitando funções para o gerenciamento de modelos de processos armazenados, assim como a busca e seus versionamentos. As organizações possuem processos de negócio públicos e privados, e com o apoio de programas conhecidos como repositórios de modelos de processos de negócio, esta tarefa é facilitada. Entretanto, como mencionado acima, conquistar tal expertise nesta área não é uma tarefa fácil para as organizações, devido ao grande número de profissionais envolvidos nas rotinas de cada processos de negócio e a grande quantidade de processos de negócio dentro das organizações. Por exemplo, o modelo de referência SAP contém mais de 600 modelos de processos de negócios e uma coleção de modelos de processos de negócios para o governo local holandês contém um número semelhante de modelos de processos de negócio (YAN; DIJKMAN; GREFEN, 2012). Com isso, é possível notar o grau de dificuldade e complexidade para se gerenciar este grande número de processos de negócio dentro das organizações, controle de versionamento, manutenção consistente destes modelos quando muitas pessoas modificam um determinado processo simultaneamente. Devido a isso,

novas oportunidades de estudo nesta área podem crescer significativamente, uma vez que processos, além de serem gerenciados, dizem e representam muito para as organizações, podendo ser definidos como a base de conhecimento do funcionamento das organizações.

Uma boa capacidade de gerenciamento dos modelos de processos de negócio pode prover até mesmo a capacidade de extração de informação dos seus modelos de processos de negócio, bem como a extração de informação dos seus produtos gerados e informações primordiais sobre o próprio modelo de processo de negócio descrito. Para que isto seja possível, torna-se necessário que os processos estejam armazenados e estruturados de forma adequada dentro das organizações. Em uma organização com uma determinada gama de processos de negócio, as informações extraídas dos modelos podem, por exemplo, orientar uma melhoria dos processos, ajustes na performance ou na produtividade dos processos. Tal abordagem torna possível uma maior maturidade sobre os processos da organização e o enriquecimento na sua cultura de gestão destes processos, alinhando de forma mais eficaz os objetivos da organização, e possibilitando uma melhoria continuada na manutenção de seus processos.

Sendo assim, como ressaltado também nos trabalhos de (ROSA et al., 2011; YAN; DIJKMAN; GREFEN, 2012; LAZARTE et al., 2013; WEBER et al., 2011; OCA et al., 2015), o armazenamento de processos de negócio de forma inteligente é de grande valia para as organizações. Em grandes organizações há uma grande dependência de aplicações BPMS e de repositórios de processos de negócio bem estruturados, não apenas para a implementação e manutenção de processos, mas também para a agregação na cultura da empresa. Tal abordagem faz com que a organização se conheça melhor, não apenas na questão dos processos em si, mas também em como melhorá-los e como se extrair informações relevantes dos modelos para a organização.

2.5 *Arquitetura Orientada a Serviços*

A *Service-Oriented Architecture* (SOA) ou Arquitetura Orientada a Serviços é uma estratégia que organiza as funções discretas contidas em aplicativos corporativos em serviços interoperáveis, baseados em padrões e independentes de plataformas, que o usuário pode combinar e reutilizar rapidamente para atender às necessidades de negócios. Mas para torná-lo realidade, o usuário precisa entender melhor a organização, suas capacidades de SOA e ter um sólido roteiro para implementação. De acordo com (ERL, 2005; CONNER; ROBINSON, 2007), SOA pode realmente fornecer um arcabouço tecnológico para reduzir os riscos e custos associados a projetos de integração, tanto atuais quanto futuros. A Figura 3 dá uma perspectiva com relação ao conceito de uma abordagem SOA.

Em SOA, “Arquitetura” pode significar tanto um padrão de alto nível para o arranjo e dependências de módulos técnicos (BOSCH, 2000) e uma abordagem abrangente

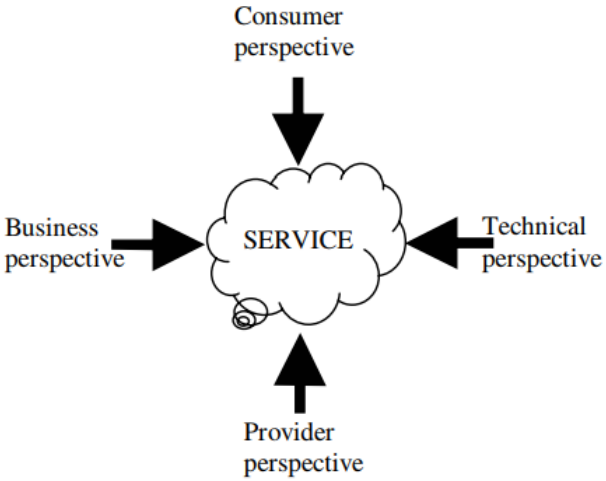

Figura 3 – Perspectiva Orientada a Serviços (PERREY; LYCETT, 2003)

.

para o desenvolvimento e instanciação das unidades funcionais que compõem um sistema (MARY; DAVID, 1996). Muito trabalho foi feito nesta área em serviços da Web, portanto, é compreensível que haja um viés substancial na literatura em relação aos detalhes da arquitetura da Web. Aqui, a arquitetura de serviços e serviços não é especificamente orientada para a Web, mas sim mais sobre as abstrações usadas em desenvolvimento de sistemas de software grandes e geralmente distribuídos.

Na **Implementação BPM com SOA** de acordo com Marc J. Balcer (BALCER, 2006) um modelo de processos de negócio deve interagir com um sistema ou dado de forma "direta", entretanto, nos cenários típicos eles interagem com vários sistemas e dados diferentes, em um mesmo processo de negócio. Em algumas situações se torna um tanto quanto inviável manter esta mecânica de interconexão.

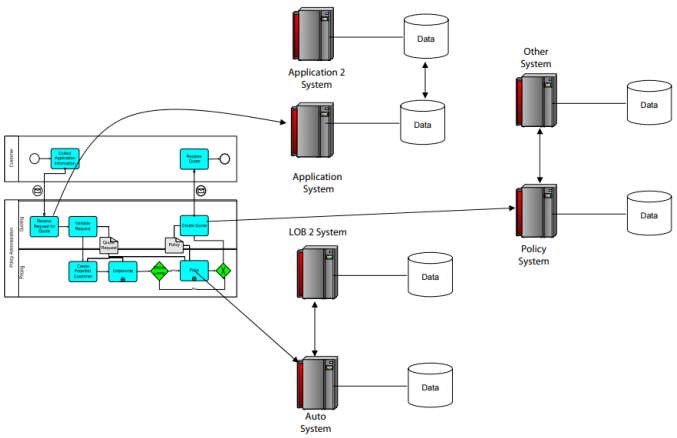

Figura 4 – Abordagem BPM com SOA aceitável

.

O modelo expresso na Figura 4 mostra como tal abordagem pode ir crescendo e tomar grandes proporções no que tange as conexões entre sistemas. Dependendo da complexidade e tamanho do modelo, um sistema pode passar a depender de outro e assim

sucessivamente, até não se ter mais o controle de onde os dados do modelo realmente vem, como pode ser visto na Figura 5.

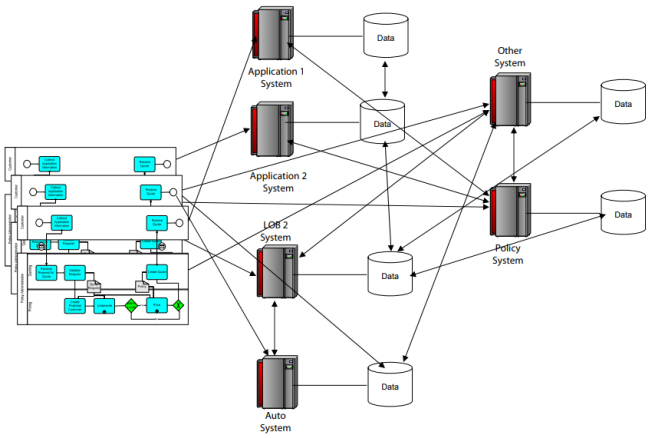

Figura 5 – Abordagem BPM com SOA inaceitável

.

Sendo assim, torna-se necessário usar "tipos certos" de serviços para se construir sistemas com modelos BPM ou BPMS. Organizar o conjunto completo de serviços a serem usados em um determinado contexto é fundamental para realizar o design de forma sistemática, por exemplo, através de uma abordagem *Model Driven Architecture* (MDA). Os modelos de processos de negócio devem estar em linguagem de negócios. Os passos do BPM deveram mapear os serviços e proverem um funcionamento equivalente do negócio. Os serviços de negócios devem ser construídos sobre camadas de serviços que vão até as tecnologias de implementação, possibilitando assim que os modelos sejam compreendidos tanto em alto quanto em baixo nível, por todos que com ele interagem.

2.6 Sistemas baseados em Regras

Um sistema baseado em regras pode ser descrito como um sistema que se baseia no padrão Evento-Condição-Ação (ECA) para mensurar a relevância dos parâmetros e determinar o seu grau de prioridade em um dado domínio, e posteriormente executar uma determinada ação. Em um sistema baseado em regras, várias regras ECA são definidas. A medida que os dados "casam" a partir dos fatos dados como entrada, eventos são gerados e verificados com base no condicional das regras pertinentes a cada domínio, gerando assim uma ou mais ações que são tomadas e/ou executadas pelo sistema. À medida que esses fatos mudam (removidos, adicionados ou atualizados) outras decisões podem ser tomadas a partir das regras previamente definidas.

Esta abordagem, que constitui um paradigma de programação declarativo, tem por objetivo representar através de regras o conhecimento relativo ao domínio do problema, possibilitando a conclusão sobre fatos a partir de suas premissas (HILL, 2003; KAMBONA;

BOIX; MEUTER, 2015; BALI, 2009). Comumente as técnicas baseadas em regras possuem modelos de desenvolvimento de sistemas compreendido em alto nível de abstração, possibilitando uma maior facilidade de compreensão acerca do modelo de negócio. Os sistemas baseados em regras possuem como objetivo prover uma separação clara entre a lógica de sistemas e outras partes pertinentes ao sistema, facilitando a manutenção do sistema e da lógica de negócio.

Tipicamente um sistema baseado em regras consiste em 3 (três) componentes, que são (Figura 6): (i) *Working Memory* (WM), ou Memória de Trabalho, onde se reserva espaço de memória para o armazenamento sobre os fatos conhecidos pelo domínio, ou seja, contém informações sobre uma instancia particular do problema; (ii) *Rule Base* (RB)/*Knowledge Base* (KB), ou Base de Regras/Base de Conhecimento, é onde se determina o conjunto de regras sobre os módulos de conhecimento sobre um determinado domínio; (iii) a *Inference Engine* (IE)/*Rule Engine* (RE), ou Motor de Inferência/Motor de Regras. Este componente é responsável por avaliar a instância dos fatos de acordo com as regras definidas, na qual é um processo conhecido como *Pattern Matching* ou Casamento de Padrões, após realizar esse casamento de regras e fatos a RE, chama uma Agenda, na qual será agendada uma execução daquela ocorrência assim que possível. Posteriormente a RE chamará o *Execution Engine* (EE) que executará a regra agendada. Nesta etapa será executada uma ou mais ações pelo motor das regras, das quais as ações foram definidas no RHS da regra.

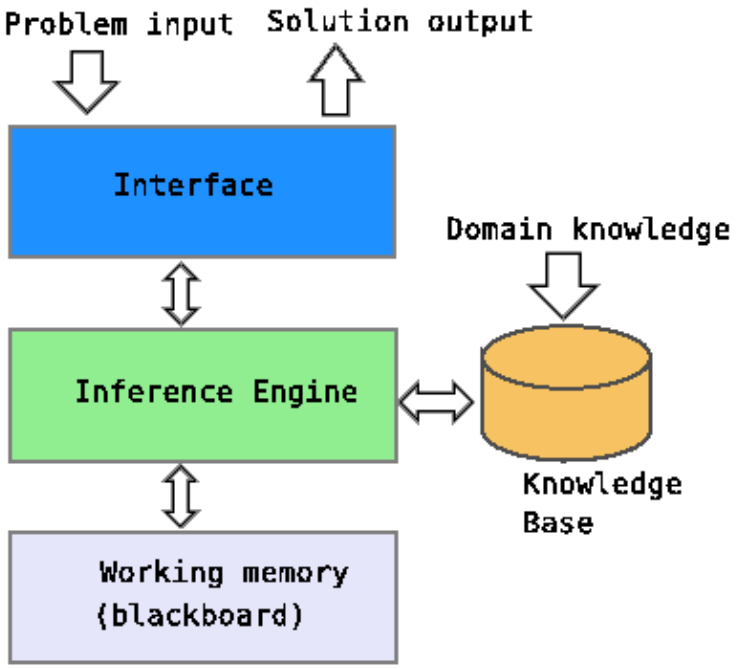

Figura 6 – Típicos sistemas baseados em regras

.

Existem várias implementações de máquinas de regras no mercado. Algumas implementações são JESS (HAYES-ROTH, 1985), Drools (HILL, 2003; BALI, 2009) e

Nools (KAMBONA; BOIX; MEUTER, 2015). Algumas dessas ferramentas vêm sendo implementadas e atualizadas constantemente pelos seus desenvolvedores e fornecem um ambiente rico e agradável de desenvolvimento para soluções no âmbito de negócios. O Drools por exemplo suporta além da definição de regras de negócio, a integração com processos de negócios, como será descrito mais à frente neste trabalho.

2.7 Ciência da Situação (*Situation-Awareness*)

Uma situação pode ser definida como uma interpretação de um estado da realidade, definida pelos seus respectivos dados pertinentes ao domínio de interesse (ou seja, o contexto), e que contribui com algum significado para o usuário pessoa ou sistema que os consome ou executa (COSTA, 2007). De forma similar, (PEREIRA; COSTA; ALMEIDA, 2013) define uma situação como sendo uma abstração de eventos do mundo real derivados do contexto e hipóteses sobre como um contexto observado se relaciona com os interesses de usuários e aplicações. Em palavras mais simples, uma situação define um estado de algo na vida real que é de interesse para usuários e aplicações.

Sensibilidade à Situação ou Ciência da Situação (*Situation-awareness* - SA), por sua vez, refere-se à percepção de contextos e eventos ambientais levando em consideração sua relação no espaço e no tempo, compreendendo seu significado e a projeção de seu estado geral em um certo "domínio do discurso". As aplicações sensíveis ao contexto podem tirar proveito das situações contextuais, podendo realizar o compartilhamento destes elementos de contexto de alto nível para a redução da carga de processamento. Os fatores temporais são levados em consideração na análise das situações, uma vez que as janelas de tempo de ocorrência dessas situações e o momento em que elas começaram e terminaram são de suma importância para o estabelecimento de relações temporais. Embora os sistemas convencionais baseados em *Complex Event Processing* (CEP), por exemplo, ESPER (MATHEW, 2014) e Drools (BALI, 2009), sejam capazes de agregar, compor, raciocinar e derivar eventos compostos, poucos são capazes de representar e gerenciar situações de um domínio específico de maneira expressiva e amigável ao usuário. Os eventos são vistos como cidadãos de primeira classe dos sistemas de processamento de eventos e são detectados apenas quando ocorrem, ou seja, quando termina ao mesmo tempo ou antes do momento presente. Uma situação, por outro lado, pode ser detectada no primeiro instante em que aparece, sem saber quando ou se terminará. Um evento não suporta os conceitos de “acontecendo”, no entanto, isto é uma parte intrínseca do conceito de situação. Ou seja, o CEP está aquém da sua capacidade de apoiar o conceito de ciência da situação por si só. Como visto, uma situação contextual pode ser definida coletando contextos relevantes, revelando correlações significativas entre eles e rotulando-os com um nome descritivo. O nome descritivo pode ser chamado de ‘*situation definition*’ ou ‘*situation type*’. Isso se refere a como um *stakeholder* define um estado de coisas na realidade (ADI;

BOTZER; ETZION, 2000; YE; DOBSON; MCKEEVER, 2012). Exemplos de tipos de situações incluem “O freezer no consultório médico está descongelando”, “o caixa eletrônico está sendo assaltado” etc.

Uma expressão lógica do contexto correlacionando predicados é conhecido como uma "especificação lógica de um tipo de situação"ou ‘*situation rule*’. A inferência de uma situação para esse tipo ocorre sempre que seus predicados de contexto forem verdadeiros. Como eventos compostos, as situações herdam as dimensões do evento, tais como: composição, agregação e correlação temporal com outras situações ou eventos. Além disso, as situações apresentam uma dimensão de atividade. Uma situação é considerada ativa enquanto essas propriedades satisfazem predicados capturados na expressão lógica do tipo de situação. Ele deixa de existir quando essas propriedades não satisfazem mais os predicados definidos. Neste caso, a situação é considerada uma situação passada.

SCENE é uma plataforma baseada em regras proposta em (PEREIRA; COSTA; ALMEIDA, 2013) como uma solução para conectar abordagens baseadas em CEP e aquelas baseadas em situações, alavancando o sistema de regras gerais Drools e o módulo *Drools Fusion* CEP como mecanismo de detecção de situação e gerenciamento do ciclo de vida. O módulo *Fusion* oferece um esquema de eventos, processamento de fluxo, correlação temporal e operadores de janela de tempo, perfeitamente integrado à sua linguagem de regras (*Drools Rule Language* - DRL). Na abordagem SCENE, os ’*situation types*’ são especificados por meio de ’*situation rules*’. SCENE usa o ’*situation scheme*’ para definir quais entidades podem estar envolvidas em um determinado tipo de situação e rotula as funções ou partes para que essas entidades possam participar dela. Quando uma ’*situation rule*’ é totalmente correspondida (isto é, as condições são satisfeitas), todos os fatos ligados pelo identificador de LHS que se referem às ‘*situation parts*’ (*@part annotation*) declaradas pelo esquema, são referidas como ‘*situation cast*’. A ‘*situation cast*’ é o conjunto de todas as entidades que participam da situação, incluindo outras situações em ‘*situation types*’ compostas.

A Figura 7 ilustra a especificação em SCENE de uma situação bastante simples, de alerta de temperatura alta em um compartimento de armazenamento de produtos médicos.

2.8 Considerações do Capítulo

Este capítulo apresentou conceitos básicos relacionados a áreas distintas que apresentam grande potencial de integração em aplicações organizacionais. Nesta dissertação, defende-se a ideia de que a convergência dessas áreas é bastante oportuna aos propósitos da universalização das aplicações de BPM em ambientes organizacionais, podendo essas tecnologias serem convenientemente combinadas de modo a se criar infraestruturas que potencializem a construção de aplicações que integrem sistemas inteligentes e modelos

```
Line Instruction
01   declare HighTemperatureAlert extends Situation
02     container: ProductContainer @part
03   end
04
05   rule "HighTemperatureAlert"
06     @role(situation)
07     @schema(HighTemperatureAlert)
08       when
09         doctor: Doctor()
10         container: ProductContainer(owner == doctor,
11                                     temperature > 0)
12       then
13           SituationHelper.situationDetected(drools)
14   End
```

Figura 7 – Exemplo de especificação de uma situação em SCENE.

.

de processos de negócio, utilizando de forma correta metodologias de armazenamento de modelos de processos de negócio de forma gerenciável, possibilita que organizações criem uma maior maturidade acerca dos seus processos.

Iniciativas que propõem essa integração, porém, ainda são incipientes. O próximo capítulo realiza uma breve descrição de alguns trabalhos que tratam desta tentativa de integração. A proposta de integração desenvolvida nesta dissertação é apresentada na sequência, no Capítulo 4.

3 Trabalhos Relacionados

Este capítulo apresenta alguns trabalhos correlatos que implementam soluções que relacionam análise de contexto e situações, modelos de processos de negócio e repositório de processos. São descritas abordagens que propõem algum tipo de infraestrutura que integre o conceito de processos de negócio e sensibilidade a contexto/situação para facilitar o desenvolvimento ou uso dos processos de negócio dentro das organizações, bem como algumas propostas de repositórios de processos de negócio para dar apoio ao armazenamento correto dos processos organizacionais. O capítulo está estruturado da seguinte maneira: a Seção 3.1 apresenta exemplos de infraestrutura de integração entre processos de negócio e sistemas inteligentes; a Seção 3.2 descreve algumas infraestruturas voltadas a ambientes de armazenamento de processos de negócio em repositórios. Em ambas as seções são identificados requisitos e apresentadas tabelas comparativas entre os trabalhos analisados. A Seção 3.3 traz as considerações finais do capítulo.

3.1 Infraestruturas de Integração entre Sistemas Sensíveis ao Contexto e Aplicações BPM

Um mapeamento sistemático da literatura foi conduzido com o objetivo de investigar trabalhos que integrassem sensibilidade ao contexto e ambiente de processos de negócio. Trabalhos propondo abordagens de repositórios para o gerenciamento de modelos de processos de negócio também foram observados. O método de pesquisa adotado no mapeamento foi baseado em (KITCHENHAM, 2004; BRERETON et al., 2007) e (KITCHENHAM; BUDGEN; BRERETON, 2011), e inspirado por outros trabalhos (NARDI; FALBO; ALMEIDA, 2013).

Alguns detalhes como string de busca e maiores resultados desta pesquisa podem ser vistos no Apêndice A. Com isso, pôde-se constatar que ao longo dos últimos anos, foi possível notar um aumento no número de pesquisas e um grande interesse acerca de infraestruturas de apoio a integração entre sistemas inteligentes e abordagens que utilizem processos de negócios dentro das organizações. Este aumento se deu devido à dificuldade de gerenciar conhecimento e o grande número de processos dentro das organizações. Tendo em vista um aumento crescente na utilização dos modelos de processos de negócio, o valor e o que estes processos representam, tornou-se importante unir tal conhecimento a uma base tecnológica que possibilite a extração de conhecimento acerca do domínio no qual estes processos estão inseridos, ou até mesmo conhecimento sobre dos próprios processos.

Tal interesse é justificado pela promessa dessas infraestruturas oferecerem as

organizações um ambiente capaz de prover uma maior flexibilidade, dinamicidade, interoperabilidade, capacidade de modificações em tempo real, extração do conhecimento, análise de domínios e situações, entre outros. Essas abordagens por sua vez, contribuem para um crescimento cultural das organizações, representando uma conquista para o seu próprio autoconhecimento, uma maior agilidade na execução de determinados processos, a capacidade de integração dos processos com outras áreas do conhecimento e departamentos dentro das organizações e, outros fatores importantes.

Na literatura estudada, vários são os trabalhos que investigam esses aspectos. Alguns deles, tais como: (ABBASI; SHAIKH, 2009), (BUCCHIARONE; MEZZINA; PISTORE, 2013), (VASILECAS; KALIBATIENE; LAVBIČ, 2016), (MOREIRA et al., 2015b) e (BAJEC; KRISPER, 2005), (SANTRA; CHOUDHURY, 2018), (NEUMANN et al., 2019), (CELESTRINI et al., 2019) são brevemente discutidos a seguir:

3.1.1 Trabalho 1 - *A Conceptual Framework for Smart Workflow Management*

Em (ABBASI; SHAIKH, 2009) é descrita uma tecnologia de workflow como um meio eficaz de modelagem e gerenciamento de processos de negócios complexos que está sendo empregada em outros domínios diversificados. O autor argumenta que, embora a tecnologia de fluxo de trabalho seja bastante madura e os produtos comerciais de gerenciamento de workflow estejam prontamente disponíveis, observa-se que a tecnologia não possui certos componentes e conceitos relacionados ao contexto quando aplicada a domínios como comércio eletrônico e computação móvel.

Os workflows inteligentes ou sensíveis ao contexto são aqueles que operam de acordo com as informações de contexto reunidas a partir do conhecimento do ambiente ou do domínio. Para tornar o gerenciamento do fluxo de trabalho sensível ao contexto, extensões e modificações são necessárias em arquiteturas e padrões existentes de sistemas de gerenciamento de workflow.

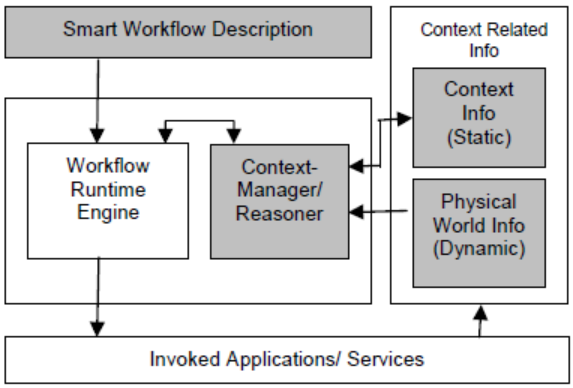

Figura 8 – Arquitetura proposta por (ABBASI; SHAIKH, 2009)

A arquitetura proposta é mostrada na Figura 8 e tem como objetivo principal

gerenciar fluxos de trabalho com reconhecimento de contexto que englobam diferentes aspectos da mudança de condições no domínio observado. O trabalho indica as alterações e extensões necessárias na tecnologia atual para facilitar a integração do contexto nos workflows para permitir sua execução inteligente. O trabalho apresenta ainda uma técnica simplificada de representação do workflow com a introdução da arquitetura de atividade-contexto tomando como base o *framework* proposto.

3.1.2 Trabalho 2 - *CAptLang e ASTRO-CAptEvo* – Linguagem e *Framework* para execução de processos sensíveis ao contexto

Em (BUCCHIARONE et al., 2012) é proposta uma estrutura para a adaptabilidade de aplicativos baseados em serviços, que explora o conceito de fragmentos de processo como uma forma de modelar o conhecimento de processo reutilizável e permitir a composição dinâmica e incremental do contexto desses fragmentos em aplicativos baseados em serviços adaptáveis. É descrito um *framework* que fornece um conjunto de mecanismos de adaptação que combinados por meio de estratégias de adaptação são capazes de resolver problemas complexos de adaptação. Uma implementação da solução proposta é apresentada e avaliada em um cenário real do domínio de logística.

Em um outro dos seus trabalhos (BUCCHIARONE; MEZZINA; PISTORE, 2013) os autores apresentaram a adaptabilidade em tempo de execução como uma característica chave de ambientes de negócios dinâmicos, em que os processos precisam ser constantemente aprimorados e reestruturados para lidar com as mudanças de contexto. Os autores apresentam a CAptLang, uma linguagem para modelar processos de negócios sensíveis ao contexto e adaptáveis, onde a principal característica é a possibilidade de deixar o manuseio de situações extraordinárias ou improváveis para o tempo de execução.

É apresentada a sintaxe e semântica formal da linguagem, mostrando como sua semântica foi usada para guiar a implementação de um mecanismo de execução de processos de negócios baseado em Java, o componente de adaptação *ASTRO-CAptEvo framework*, mostrado na Figura 9.

Por sua vez, os autores (RAIK et al., 2012) implementaram o *ASTRO-CAptEvo*, e salientaram que a adaptabilidade é uma característica fundamental para explorar plenamente as vantagens oferecidas pelo paradigma de serviço. Os autores propuseram uma infraestrutura que pode ser usada para definir e suportar processos de negócios baseados em serviços altamente sensíveis ao contexto. Na abordagem utilizada, segundo os autores, todas as tarefas relacionadas à adaptação, desde detectar um problema até encontrar uma solução, são executadas automaticamente. O *framework* foi implementado e aplicado a um cenário do mundo real do domínio de logística.

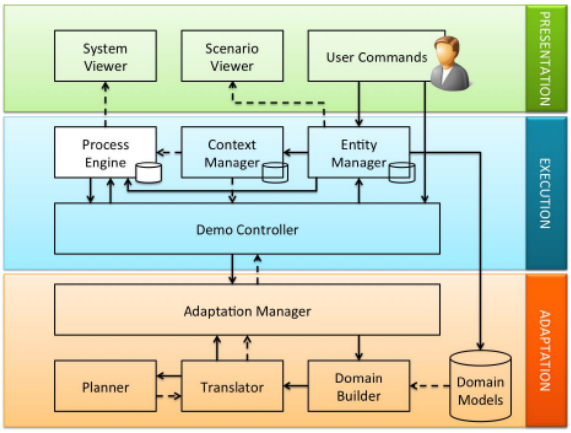

Figura 9 – Arquitetura do *ASTRO-CAptEvo* (RAIK et al., 2012)

3.1.3 Trabalho 3 - *Rule and context-based dynamic business process modelling and simulation*

Em (VASILECAS; KALIBATIENE; LAVBIČ, 2016) os autores destacam que as abordagens tradicionais usadas para implementar processos de negócios nos sistemas de informação atuais não abrangem mais as necessidades reais dos negócios, os quais mudam dinamicamente, surgindo assim a necessidade de uma nova abordagem de modelagem e simulação de processos de negócios dinâmicos (*Dynamic Business Process* - DBP).

Ainda segundo os autores, até o momento as abordagens existentes para modelagem e simulação de DBP estão incompletas, não possuindo teoria ou estudo de caso - ou ambos. Além disso, ressaltam que não existe uma definição comumente aceita de DBP. As atuais ferramentas de modelagem de processos de negócio (BP) são adequadas quase que exclusivamente para a modelagem e simulação de BP estáticos, que prescrevem estritamente quais atividades e em qual sequencia executar.

Geralmente, um DBP não é definido estritamente no início de sua execução e é alterado sob novas condições no tempo de execução. O trabalho propôs seis requisitos do DBP e uma abordagem para modelagem e simulação de DBP baseada em regras e contexto. A abordagem é baseada na mudança de regras BP, ações BP e suas sequências no tempo de execução da instância de processo, de acordo com o novo contexto do sistema de negócios.

Com base na abordagem proposta, foi desenvolvida uma arquitetura de referência, vista na Figura 10, e um protótipo de uma ferramenta de simulação de DBP. A modelagem e a simulação foram realizadas usando esse protótipo, e o estudo de caso mostra a correspondência com as necessidades de mudanças dinâmicas nos negócios, bem como as possibilidades de modelagem e simulação da BDP.

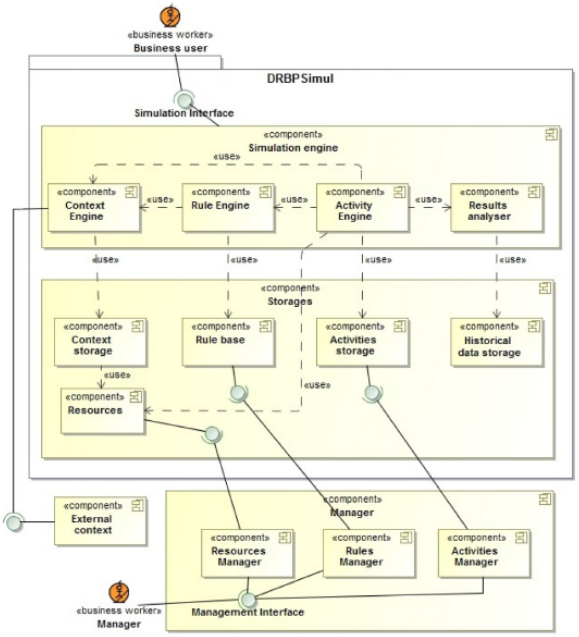

Figura 10 – Arquitetura proposta por (VASILECAS; KALIBATIENE; LAVBIČ, 2016)

3.1.4 Trabalho 4 -*Ontology-driven and Distributed Rule-Based Platform for Disaster Management*

Em (MOREIRA et al., 2015a) os autores destacam que inúmeras aplicações de tecnologia da informação e comunicação (TIC) com mecanismos para detectar situações foram desenvolvidas para apoiar o gerenciamento de desastres (GD) nos últimos anos, sendo um campo de grande importância econômica e social. Essas aplicações *situation-aware* (SA) tentam perceber e compreender, quase em tempo real, algum tipo de situação de interesse possibilitando projetar uma reação à circunstância detectada, por exemplo: epidemias de doenças, pessoas doentes isoladas etc.

Segundo os autores, um obstáculo para a modelagem de aplicações de SA é a falta de construções estruturais e temporais bem fundamentadas. Os autores alegam que a modelagem conceitual baseada em ontologia tem sido aplicada com sucesso para superar esse problema, onde a análise ontológica baseada em uma ontologia fundacional suporta a modelagem de conceitos dentro de um campo específico como uma ontologia central bem fundamentada. O foco de discussão do trabalho é a importância de uma ontologia central bem fundamentada para o GD visando suportar a especificação de aplicativos SA. Com base nos requisitos identificados no trabalho, os autores utilizam o *framework* mostrado na Figura 11 para introduzir e exemplificar os conceitos de seu trabalho.

Em (MOREIRA et al., 2015b) os autores descrevem um *framework* com o objetivo de melhorar a interoperabilidade e a produtividade no desenvolvimento de aplicativos sensíveis a situações para o gerenciamento de desastres, em que destacam a necessidade de mecanismos e diretrizes apropriadas, em especial a nessidade de promover a semântica

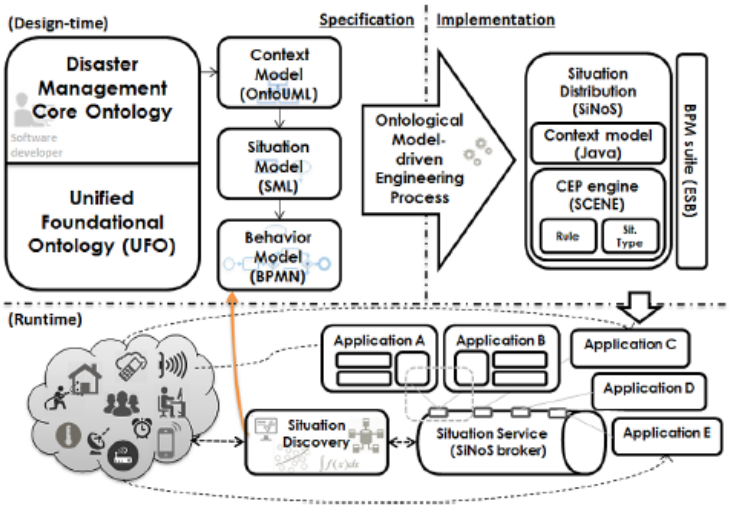

Figura 11 – Framework para o desenvolvimento de aplicações SA de (MOREIRA et al., 2015b)

na modelagem de situações de emergência. Além disso, destacam que a natureza sempre mutável e imprevisível dos cenários de desastres apresenta desafios para o processamento e a colaboração de informações. Neste sentido, o *framework* proposto combina os seguintes elementos:

(i) Uma ontologia fundacional para a conceitualização temporal;

(ii) Especificações bem fundamentadas de modelos estruturais e comportamentais;

(iii) Um mecanismo CEP apoiado em uma plataforma distribuída baseada em regras para o gerenciamento de situações;

(iv) Uma abordagem baseada em modelos.

Em ambos os trabalhos os autores englobam conceitos de integração entre situation-awareness e behavior model para a descrição de estruturas e comportamentos, como pode ser visto na arquitetura da Figura 11.

3.1.5 Trabalho 5 - *A Methodology and Tool Support for Managing Business Rules in Organisations*

O trabalho apresentado em (BAJEC; KRISPER, 2005) propõe uma metodologia que visa ajudar empresários e desenvolvedores a manter regras de negócios no nível comercial alinhadas com as regras implementadas no nível do sistema. Em contraste com várias abordagens existentes, que se concentram principalmente nas regras de negócios no escopo de um aplicativo, a metodologia proposta aborda todo o sistema de informação (SI) de uma organização. Descreve também os requisitos para um suporte de ferramenta que seria apropriado para apoiar a metodologia.

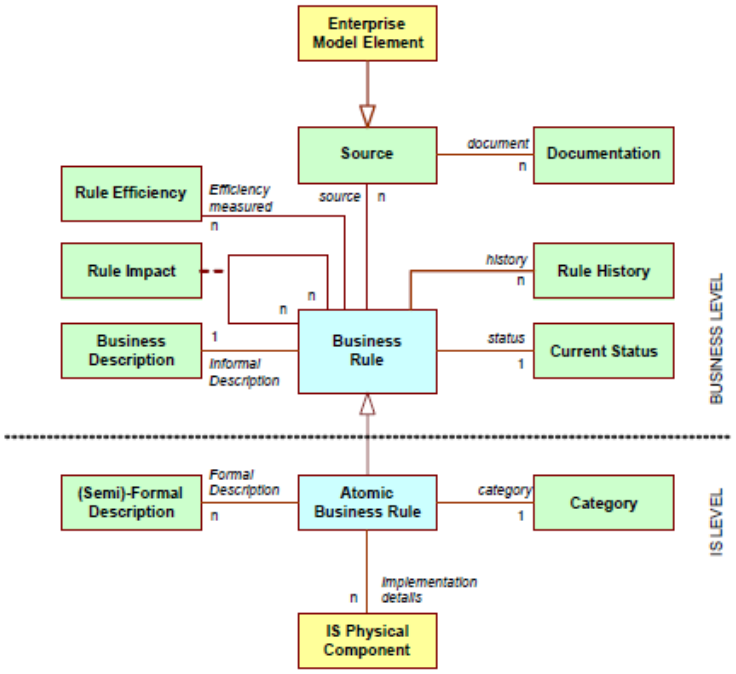

Figura 12 – Componentes do modelo de regra de negócios de (BAJEC; KRISPER, 2005)

O meta-modelo da Figura 12 ilustra as informações mais importantes, que vale a pena acompanhar as regras de negócios. O meta-modelo é dividido em duas seções, uma que compreende elementos de nível de negócios e a outra que representa conceitos interessantes no nível de SI. No nível comercial, cada regra de negócios é descrita numa linguagem de negócios (*Business Description*) que é compreensível para os executivos. Entre as regras de negócios há muitos relacionamentos, por exemplo, uma regra de negócios suporta outra regra, se uma regra de negócios está em conflito com outra regra, etc (Impacto da Regra).

Cada regra tem um histórico (*Rule History*) que informa quando ela foi colocada em operação e como ela mudou ao longo do tempo. O *Current Status* informa a posição atual da regra, por exemplo, sugerida, aceita, colocada em operação, recusada etc, sendo importante manter a informações sobre fontes que explicam o motivo ou a motivação para a existência da regra. As fontes, como políticas, regulamentos ou alguns outros atos administrativos, são frequentemente documentadas (*Document*). Uma fonte de uma regra de negócios também pode ser um elemento modelado em modelos corporativos, como um objetivo de negócios, um processo de negócios, etc (*Enterprise Model Element*). Uma informação adicional, mas desejada, é a eficiência das regras (*Rule Efficiency*), que diz quão eficiente é a regra em termos de alcançar seus objetivos. A eficiência é normalmente medida em relação aos elementos que a regra restringe ou aciona.

No nível SI, as regras precisam ser atômicas, no sentido de que não podem ser decompostas, perdendo seu significado (*Atomic Business Rule*). Como as descrições de regras precisam ser mais rigorosas neste nível, linguagens formais devem ser empregadas (*Formal Description*). Para simplificar a formalização de regras e para suportar a sua

implementação, as regras devem ser categorizadas (*Category*). Por fim, cada regra no nível do sistema vincula-se a um ou vários componentes físicos do SI. Esta é uma informação essencial, pois informa onde e como a regra é implementada.

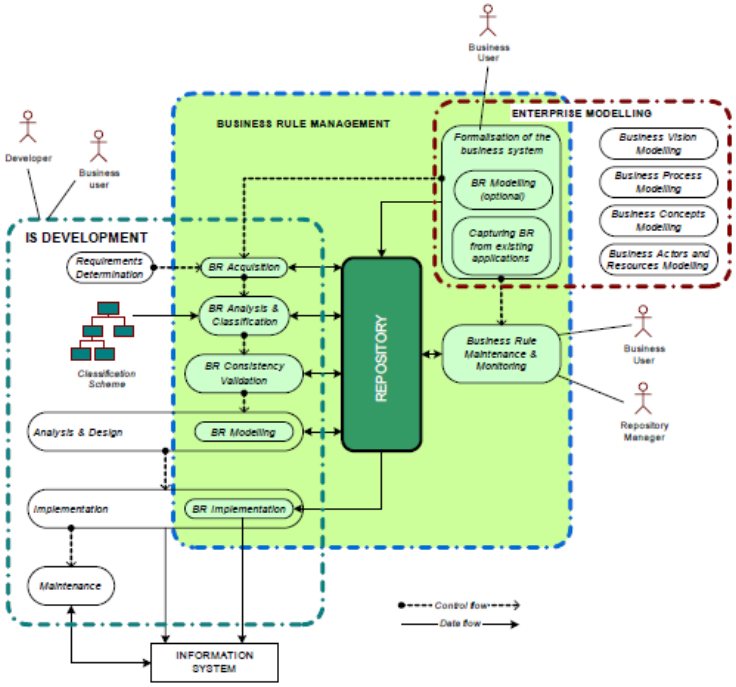

Figura 13 – Principais funções e atividades para gerenciar regras de negócio descritas por (BAJEC; KRISPER, 2005)

Foi criado um cenário, visto na Figura 13, que define as principais atividades e funções necessárias para gerenciar regras de negócios para toda a organização. Além das atividades dedicadas à gestão de regras nos negócios e no nível de SI, o cenário determina as atividades responsáveis pela manutenção e monitoramento de regra. A atividade mais importante no nível do negócio é a formalização do ambiente de negócios. O objetivo é capturar elementos de negócios que podem atuar como uma fonte para as regras de negócios. Por exemplo, metas de negócios, problemas, processos de negócios, funções de negócios, unidades organizacionais e conceitos de negócios que definem a terminologia de negócios. Informações adicionais que devem ser capturadas nesse estágio são a arquitetura de *software* e *hardware* IS. Esses elementos são elaborados e armazenado no repositório, que representa um armazenamento físico para os elementos do modelo de regra de negócios.

3.1.6 Trabalho 6 - *C-BPMN: A Context Aware BPMN for Modeling Complex Business Process*

Com um desafio de uma modelagem de processos de negócios que inclui a sensibilidade do contexto na própria modelagem, o trabalho apresentado em (SANTRA; CHOUDHURY, 2018) propõe uma abordagem para expressar a ciência de contexto na modelagem de processos. O autor salienta que as linguagens EPC, UML e BPMN, não

são capazes de expressar tal conscientização; com isso, é proposto um modelo de contexto que amplia o modelo BPMN existente e adiciona um novo construtor que integra os componentes que permitem a consciência de contexto. O autor define um Contexto de Processos de Negócio como a informação do impacto de design ou execução de um processo de negócio, ainda que externamente ou internamente. O contexto de um processo de negócio pode ser *static, steady or dynamic*, dependendo da aplicação. Um contexto estático (*static context*) é uma informação de contexto com valores fixos toda as vezes. Este tipo de contexto de informação não pode ser todo modificado em razão do tempo. Um contexto firme (*steady context*) é uma informação que vem sendo modificada após um certo intervalo de tempo. O intervalo de duração será específico da aplicação. Este tipo de contexto de informação permanece estático ou fixo por algum tempo e é modificado ocasionalmente. Um contexto dinâmico (*dynamic context*) é um valor modificado frequentemente em um intervalo curto de 1 (um) minuto ou próximo disto. Um contexto basicamente reflete as circunstâncias de modificação durante a execução de um processo de negócio. Segundo estudos realizadas pelo autor, muitas pesquisas vêm identificando vários tipos de fatores de contexto que afetam os processos de negócio.

Com isso, decorrente da baixa representatividade do BPMN para a sensibilidade a contexto, é proposto o C-BPMN, que é uma extensão do BPMN com sensibilidade a contexto. É proposto um modelo de contexto baseado em grafo que representa uma possibilidade natural de contexto e seus inter-relacionamentos para um ambiente de processos de negócio. Para validar o modelo é usado uma Rede de Petri Colorida.

3.1.7 Trabalho 7 -*Extending BPMN 2.0 for intraoperative workflow modeling with IEEE 11073 SDC for description and orchestration of interoperable, networked medical devices*

Com foco em workflows, o trabalho descrito em (NEUMANN et al., 2019) possui como objetivo implementar uma nova forma de assistência cirúrgica auxiliada por computadores e novas aplicações em automação de processos, conhecimento de situações e suporte a decisões. Os autores salientam que a configuração sensível ao contexto e a orquestração de dispositivos médicos interoperáveis em rede é um pré-requisito para uma redução efetiva da carga de trabalho dos cirurgiões, fornecendo o serviço e as informações corretas no momento certo. Segundo os autores, as informações sobre as situações cirúrgicas devem ser descritas como modelos de processo cirúrgicos e distribuídas para os dispositivos médicos e sistemas de TI na sala de cirurgia. É argumentado que as linguagens de modelagem disponíveis não são capazes de descrever processos cirúrgicos para tal aplicativo; com isso, é proposta uma linguagem de modelagem processos intraoperatórios denominada BPMNSIX tecnicamente aprimorada para o tempo de construção e o tempo de execução do fluxo de trabalho.

O trabalho dá atenção especial à integração da família padrão IEEE 11073 SDC, que foi recentemente publicada para arquiteturas orientadas a serviços de dispositivos médicos em rede. Além disso, são apresentados padrões de interação para configuração sensível ao contexto e orquestração de dispositivos. De posse dos padrões de interação identificados, estes foram implementados no BPMNSIX para um caso de uso oftalmológico e foi concluído que a modelagem de procedimentos cirúrgicos com o BPMNSIX permite a implementação de funcionalidades de assistência cirúrgica sensíveis ao contexto e flexibilidade em termos de orquestração de conjuntos de dispositivos que mudam dinamicamente no gerenciamento do fluxo de trabalho cirúrgico.

3.1.8 Trabalho 8 - *An Architecture and Its Tools for Integrating IoT and BPMN in Agriculture Scenarios*

Tendo em vista que as estufas agrícolas aumentam a produtividade de culturas específicas, (CELESTRINI et al., 2019) foca seus esforços para automação desses ambientes. Os autores notaram uma ausência de soluções que inclua a cadeia completa do processo de automação. Com isso, focado em oferecer uma solução de modernização de ambientes agrícolas que possibilite automatizar processos, foi desenvolvida uma infraestrutura para prever situações de interesse e, com isso, melhorar as atividades de produção. Dessa forma, foi proposta uma arquitetura para detectar episódios de interesse (i.e., situações contextuais) e atuar em ambientes agrícolas controlados, usando a modelagem de regras de negócios como o artefato central para a automação de cadeias de produção.

O design da infraestrutura é dividida em 5 (cinco) componentes, conforme pode ser visto na Figura 14: (i) o produtor *front-end*, para criar regras de negócios desejadas na cultura usando BPMN; (ii) *smart-core* para o processamento de dados e ativação de atuadores de acordo com as condições e tarefas; (iii) *virtual-environment*, contém um *dashboard* para exibir a coleta de dados e simulação da operação dos atuadores de acordo com os *smart-core*; (iv) *data-service*, para prover dados através do *middleware* GSN para o *smart-core*; e (v) *data-collector*, para coleta e agregação de dados dos sensores.

Esta arquitetura provê flexibilidade para a modificações na cultura nas *greenhouses* sem a necessidade de escrever um novo código, a partir da modelagem de regras de negócio usando uma notação padrão da indústria, o BPMN, cujo uso pode beneficiar usuários que não são experientes com plataformas IoT, uma vez que ele permite a especificação de cenários de negócio usando abstrações de alto nível. O uso do BPMN melhora a modelagem de processos de negócio, aumentando o controle do processo de produção e provendo informações aos produtores na tomada de decisão. Por sua vez, este aumento do controle no processo pode ajudar a reduzir de forma indiscriminada o uso de pesticidas, reduzindo prevalência das pestes ou doenças no cultivo enquanto reduz o impacto ecológico.

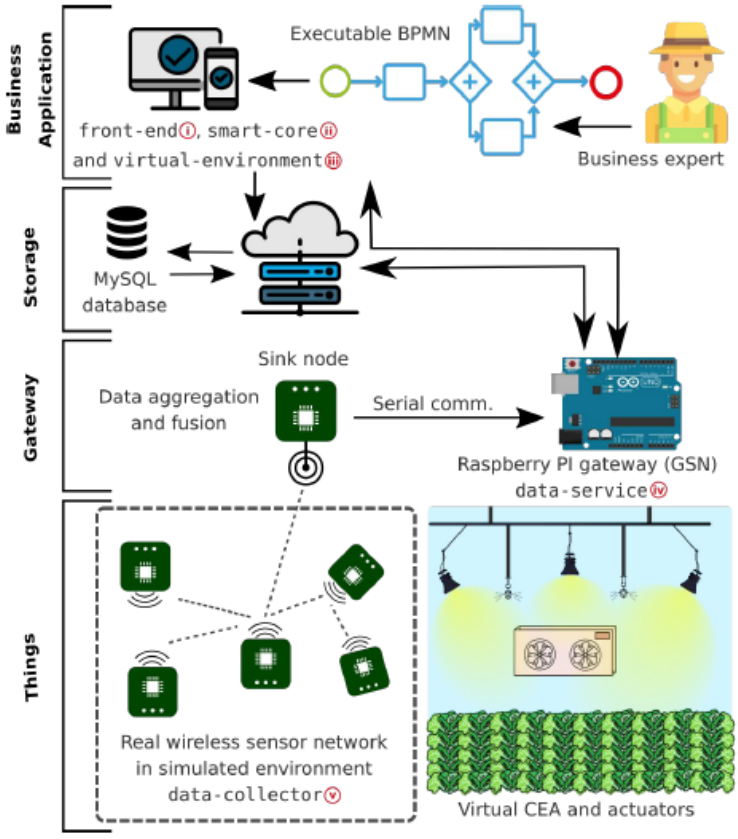

Figura 14 – Arquitetura proposta por (CELESTRINI et al., 2019)

Para a validação da solução proposta, foi realizado um estudo de caso que implementou uma ferramenta para apoiar a produção de vegetais. O estudo mostrou que a arquitetura é viável para monitorar as culturas, colaborar para maximizar a produtividade e controlar o uso de insumos e agroquímicos. Além disso, atende ao monitoramento da cultura em tempo real, oferecendo informações aos produtores para auxiliar na tomada de decisões.

3.1.9 Discussão

A Tabela 1 apresenta uma comparação dos trabalhos analisados. Na tabela, o símbolo "+" significa que o trabalho atende completamente ao requisito avaliado, o "–" indica que o trabalho atende apenas parcialmente, enquanto que o símbolo "•" representa o não atendimento ao requisito. Os requisitos estão assim identificados: Sensibilidade a Contexto com Processos de Negócio (R1); Suporte a situações (R2); Definição de *Workflows* (R3); Adaptabilidade em Tempo de Execução (R4); Foco em Organizações (R5); e Separação de papéis entre áreas de conhecimento (R6).

Tabela 1 – Comparação de trabalhos que integram sensibilidade ao contexto e BPM

Trabalhos	R1	R2	R3	R4	R5	R6
(BAJEC;KRISPER;2005)	-	•	+	-	+	+
(ABBASI;SHAIKH;2009)	+	•	+	•	+	•
(BUCCHIARONE;MEZZINA;PISTORE;2013)	+	•	•	-	+	-
(MOREIRA et al.,2015)	+	+	+	-	•	+
(VASILECAS;KALIBATIENE;LAVBIC; 2016)	+	•	+	+	+	-
(SANTRA;CHOUDHURY;2018)	+	•	+	+	•	•
(NEUMANN et al.;2019)	+	-	+	-	•	•
(CELESTRINI et al.;2019)	•	+	+	-	•	-

Conforme mostrado na Tabela 1, a maioria dos trabalhos analisados não atende de forma completa aos requisitos R2, R4, R5 e R6, demonstrando que existe uma carência de suporte integrado para o desenvolvimento de aplicações em ambientes que lidam com análise de situações, adaptabilidade de processos e regras em tempo de execução, foco em organizações e separação de papéis. O presente trabalho visa atender ao conjunto de requisitos R1, R2, R3, R4, R5 e R6 de forma completa e integrada, ou seja, todos requisitos descritos. Vale salientar que a pesquisa demonstrou que os requisitos R1 e R3 estão presentes na maioria dos trabalhos apresentados, tendo sido contemplados de forma completa. A maior parte dos requisitos não foram satisfeitos de forma integral em mais que 50% nos R2, R4, R5 e R6, demonstrando uma lacuna de pesquisa no que tange à integração entre estes temas.

3.2 Infraestruturas de Repositórios para o Gerenciamento de Modelos de Processos de Negócio

Os últimos anos também viram crescer o número de trabalhos que exploram abordagens preocupadas com o armazenamento de modelos de processos de negócio, com destaque para o uso de arquiteturas conceituais e linguagens de consultas, os quais visam prover um maior gerenciamento às organizações. O interesse se justifica pela dificuldade real que existe nas organizações de se armazenar modelos de processos de negócio de maneira gerenciável. De fato, em grandes organizações, estes modelos crescem em larga escala, sendo usados para descrever rotinas realizadas diariamente, tornando difícil o gerenciamento e a melhoria continuada dos processos.

Observa-se que não é apenas uma questão de se armazenar os processos de negócio, e sim de se ter um mapeamento e um controle detalhado dos mesmos ao longo do seu ciclo de execução visando tomadas de decisão em cima de indicadores de processos que medem o seu desempenho, eficácia, qualidade, entre outros aspectos, o que permite determinar de maneira efetiva o quão otimizado e eficiente um processo está. Com isso, é importante para uma organização ter um controle detalhado dos processos que são executados por

ela, e o quão eficiente eles estão. As organizações que conseguem gerenciar de forma a realizar este mapeamento de processos e o seu controle, são tidas como organizações com alta maturidade em termos de processos de negócio, de tão complexo que é tornar um gerenciamento de processos eficaz e extrair informações do mesmo para sua melhoria continuada.

De acordo com (OCA et al., 2015) a modelagem de processos de negócios é uma parte essencial da compreensão e do redesenho das atividades que uma empresa típica usa para atingir suas metas de negócios. A qualidade de um modelo de processo de negócios tem um impacto significativo no desenvolvimento de qualquer empresa. Como os insights sobre o que constitui a qualidade da modelagem estão em constante evolução, não está claro se a pesquisa atual sobre a qualidade da modelagem de processos de negócios já abrange todos os principais aspectos da qualidade da modelagem. O trabalho apresentado em (POLPINIJ; GHOSE; DAM, 2015), por sua vez, afirma que o processo de negócios se transformou no principal ativo de muitas organizações e torna-se cada vez mais comum para a maioria das organizações de médio e grande portes ter coleções de centenas ou mesmo milhares de modelos de processos de negócios.

Na literatura estudada vários são os trabalhos que investigam esses aspectos, sendo alguns deles: (ROSA et al., 2011), (YAN; DIJKMAN; GREFEN, 2012), (LAZARTE et al., 2013), (AA et al., 2018), (POURMIRZA et al., 2019), (POLYVYANYY, 2019), (HUANG et al., 2019) os quais serão apresentados brevemente a seguir. Para efeitos de análise e comparação, formam o escopo da discussão os seguintes requisitos:

• Colaboração entre organizações: Em muitos casos é necessário ter um controle e uma estrutura sólida para o compartilhamento de informações entre organizações no que tange a área de processos e serviços. Esta questão torna-se ainda mais complexa uma vez que gerenciar processos e serviços dentro de uma única organização já é um trabalho complexo; entretanto, as abordagens colaborativas tornam-se cada vez mais necessárias para o compartilhamento destes tipos de dados e de informações entre as organizações, sejam elas entidades da mesma organização, ou organizações parceiras.

• Base de Dados Distribuída: Em abordagens que tira proveito de informações sobre processos e serviços armazenados em vários locais distintos do mundo, torna-se relevante pensar em bases de dados distribuídas. Sendo assim, é importante que a infraestrutura dê suporte ou pelo menos tenha a possibilidade de implementar a distribuição sem provocar impactos relevantes no funcionamento da organização como um todo.

• Gestão adequada de processos: As grandes organizações armazenam centenas e até milhares de processos em seus repositórios; com isso, uma gestão adequada de processos torna-se primorial para o esquema produtivo e melhoria continuada da organização.

• Armazenamento de dados de processos: A infraestrutura deve pensar e prover

não apenas o armazenamento dos modelos de processos de negócio, mas também os dados referentes a eles. Os dados podem ser tão importantes quanto os próprios processos, uma vez que podem tornar-se necessários para análises e auditorias futuras.

3.2.1 Trabalho 1 - *A distributed repository for managing business process models in cross-organizational collaborations*

Em (LAZARTE et al., 2013), é afirmado que as colaborações entre organizações exigem o gerenciamento de modelos para os processos de negócios colaborativos (CBPs), que definem o comportamento da colaboração, e para os processos de negócios de integração (BPIs), que definem o comportamento que suporta o papel que uma organização desempenha em um CBP. O trabalho destaca que gerenciar esses modelos de processos de negócios é uma tarefa complexa quando as organizações integram redes colaborativas e estabelecem várias colaborações entre organizações. Deste modo, o trabalho propõe um repositório distribuído que fornece as funcionalidades necessárias para gerenciar modelos conceituais de processos de negócios envolvidos em colaborações interorganizacionais.

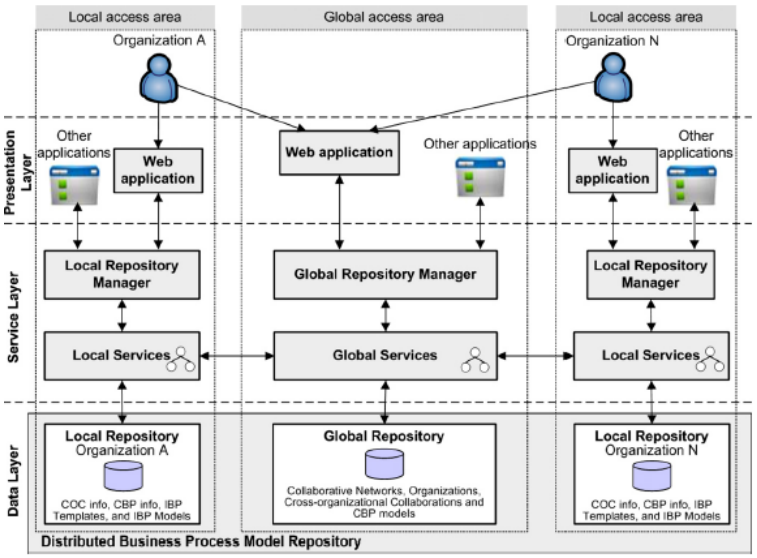

Figura 15 – SOA para repositório distribuído de (LAZARTE et al., 2013)

Uma arquitetura orientada a serviços, vista na Figura 15, é definida para o repositório distribuído. Essa arquitetura permite que as organizações acessem um repositório global para gerenciar redes colaborativas, colaborações entre organizações e seus modelos de CBP. As organizações também podem manter repositórios locais de modelos BPI, que são sincronizados e consistentes com os modelos CBP, preservando seus aspectos privados. Usando métodos de verificação e um método de arquitetura orientada a modelos, o repositório distribuído fornece serviços que suportam os requisitos de sincronização, consistência e interoperabilidade para os modelos CBP e BPI.

3.2.2 Trabalho 2 - APROMORE

Em seu trabalho, (ROSA et al., 2011) afirma que os modelos de processos de negócios estão se tornando disponíveis em grande número devido ao seu amplo uso em muitas aplicações industriais, como projetos de engenharia de qualidade e de empresas. Por um lado, isso levanta o seguinte desafio quanto a sua gestão adequada: como pode ser assegurado que o modelo de processo adequado está sempre disponível para o *stakeholder* interessado? Por outro lado, a riqueza de um grande conjunto de modelos de processos oferece grandes oportunidades, por exemplo, no que diz respeito à reutilização de peças de modelos existentes para a construção novos modelos.

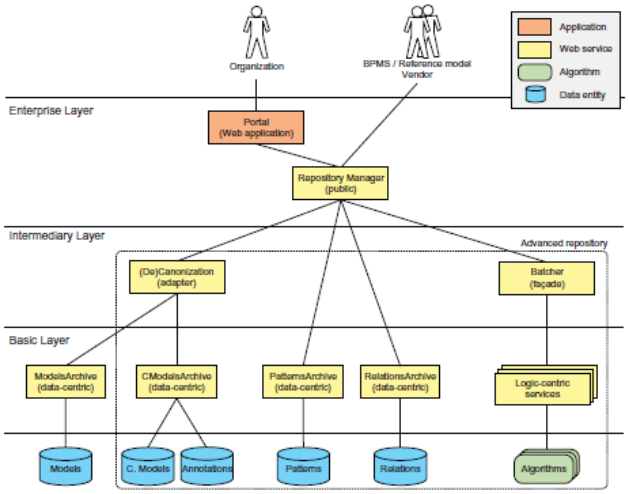

Figura 16 – Arquitetura APROMORE de (ROSA et al., 2011)

O trabalho descreve as funcionalidades e a organização de uma arquitetura de um repositório de modelo de processos, denominada APROMORE, vista na Figura 16. A ferramenta desenvolvida reúne um conjunto de recursos para análise, gerenciamento e uso de grandes coleções de modelos de processos, com base em pesquisas de ponta no campo da modelagem de processos. Um protótipo da plataforma é apresentado, demonstrando a sua viabilidade, bem como uma perspectiva sobre o desenvolvimento do APROMORE.

3.2.3 Trabalho 3 - *Business process model repositories – Framework and survey*

Em (YAN; DIJKMAN; GREFEN, 2012) é mencionado que em grandes organizações geralmente são executados centenas ou até milhares de processos de negócios diferentes. Também é salientado que gerenciar coleções tão grandes de modelos de processos de negócios é uma tarefa desafiadora. O estudo argumenta que um software de repositório de modelo de processo de negócios pode ajudar na execução dessa tarefa, oferecendo suporte a funções comuns de gerenciamento, como armazenamento, pesquisa e gerenciamento de

versões de modelos. Esse software também pode fornecer funções avançadas específicas para o gerenciamento de coleções de modelos de processo, como o gerenciamento da consistência de processos públicos e privados.

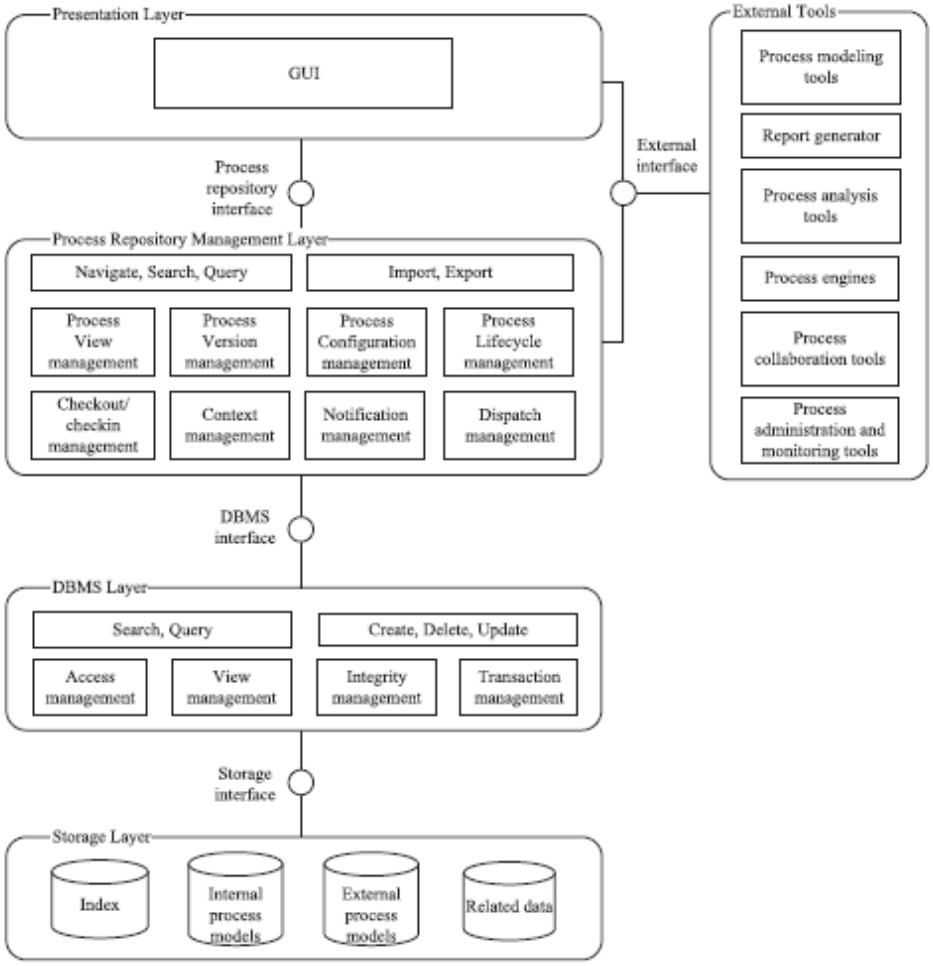

Figura 17 – Uma arquitetura de referência do repositório de modelo BP de (YAN; DIJKMAN; GREFEN, 2012)

É apresentado um *framework* para repositórios de modelos de processos de negócios, mostrado na Figura 17, que consiste em um modelo de gerenciamento e uma arquitetura de referência. O modelo de gerenciamento lista a funcionalidade que pode ser fornecida e a arquitetura de referência apresenta os componentes que fornecem essa funcionalidade. Em seguida, é apresentada uma análise da extensão em que os repositórios de modelos de processos de negócios existentes implementam a funcionalidade do *framework*. A conclusão dos autores é que os repositórios existentes se concentram na funcionalidade tradicional em vez de explorar todo o potencial das ferramentas de gerenciamento de informações, portanto, deduzem que há uma base sólida para futuras pesquisas.

3.2.4 Trabalho 4 - *Challenges and Opportunities of Applying Natural Language Processing in Business Process Management*

O BPM é um campo focado em trabalhos coordenados que são executados em de forma produtos e serviços que sejam entregues de forma apropriada. Ao mesmo tempo, a Linguagem e Processamento Natural (*Natural Language Processsing* - NLP) alcançou um nível de maturidade que permite sua ampla aplicação em muitos contextos. Com foco em mostrar como a NLP tem potencial para aumentar os benefícios das práticas de BPM em diferentes níveis, (AA et al., 2018) mostra os principais desafios para uma aplicação bem-sucedida com técnicas NLP que facilite a automação de tarefas específicas que hoje em dia exigem um esforço significativo para serem realizadas. Os autores dividem a expansão das capacidades do BPM através do NLP em 3 (três) direções de interesse, são elas: (i) Gerenciamento de Multi-processos; (ii) Gerenciamento de Processos; e (iii) Gerenciamento de Instâncias.

As pesquisas de (i) são focadas em gerenciamento de repositórios de processos, construindo técnicas para determinar a similaridade de processos, realizando análises de similaridade entre processos de forma automática, e outras pesquisas envolvem técnicas de queries para buscas de processos. Muitos desses conceitos de processos incluem refatoração automática, incluindo a harmonização de termologias, derivação automática de serviços, busca semântica e união de modelos de processos de negócio. As pesquisa de (ii) se concentram na descoberta/identificação de processos dentro das organizações e correspondências gráficas de processos. As técnicas de NLP são aplicadas em muitas granularidades diferentes para descobertas precisas de modelos de processos. Algumas dessas técnicas são: transformações textuais de descrições em modelos de processos, translação de processos, anotações e inferências, verificação correta e completa de semântica, c cálculo consistente entre modelos e processos e texto, entre outros. A conversação de sistemas é tratada em (iii), que em última instância, é utilizado em bots e chatbots. Os sistemas de conversação são treinados basicamente para darem suporte aos processos de negócio e o objetivo é permitir que as partes interessadas naveguem através de queries e diálogos de sistemas.

3.2.5 Trabalho 5 - *BPMS-RA: A Novel Reference Architecture for Business Process Management Systems*

Baseado na análise da literatura e em implementações comerciais existentes (POURMIRZA et al., 2019) propõem uma arquitetura de referência, chamada BPMS-RA, para o gerenciamento de processos de negócio. Segundo os autores, essa arquitetura de referência visa fornecer um modelo de diretriz para o desenvolvimento dos sistemas modernos de gerenciamento de processos de negócios, especificando funções e interfaces que precisam ser fornecidas por esses sistemas, bem como um conjunto de critérios de qualidade que

eles precisam atender. O framework de classificação proposto categorizou um conjunto de arquiteturas BPMS existentes. No total, foram elencados 438 componentes, obtidos por meio da literatura, indústria e pesquisas em engines de busca. O processo foi dividido em algumas etapas, quais sejam: decomposição de granularidade de componentes, funções de ferramentas de definição de processos, funções dos serviços de ativação de fluxo de trabalho, funções de serviços externos, funções de aplicações e fluxo de trabalho de clientes, funções de ferramentas administrativas e funções de monitoramento e ferramentas de controle. Os autores dividiram a arquitetura em dois níveis de classificação de qualidade de atributos, são eles: atributos de qualidade em design-time e atributos de qualidade em runtime.

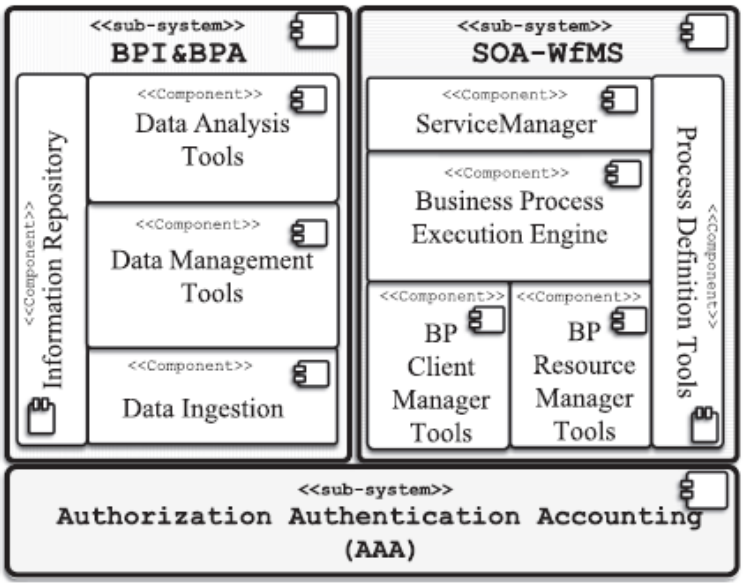

Figura 18 – Uma arquitetura de referência de (POURMIRZA et al., 2019) - BPMS-RA

Uma visão da arquitetura de referência pode ser vista na Figura 18, em que são exibidos alguns componentes de análise de dados, ferramentas para o gerenciamento de dados e comunicação, serviços de gerenciamento, engine de processos, entre outros. Ao término da proposta da arquitetura os autores apresentam um quadro comparativo com alguns BPMSs o mercado como Alfresco Activiti e Bizagi para demonstrar a abrangência da arquitetura.

3.2.6 Trabalho 6 - *Business Process Querying*

No trabalho descrito por (POLYVYANYY, 2019) o autor apresenta uma notação para filtrar, manipular e armazenar modelos descritos a serem observados, bem como os relacionamentos entre modelos, através de um framework de consulta de processos que pode ser visto na Figura 19. O framework é um sistema abstrato de componentes que provê funcionalidades genéricas e podem ser seletivamente reutilizados nos resultados de

novos métodos de processos. Os componentes descritos em retângulo são componentes ativos, enquanto os ovais são componentes passivos do framework.

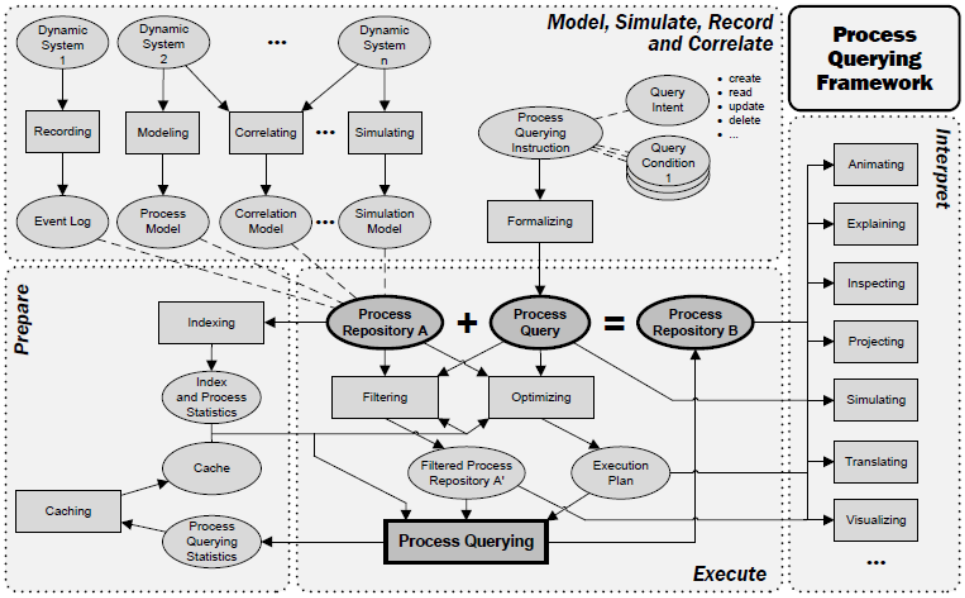

Figura 19 – Visualização semântica do framework de processamento de queries de (POLYVYANYY, 2019)

Os componentes ativos representam como as ações são executadas pelos métodos de consulta de processos, enquanto os componentes passivos representam de coleções de objetos de dados. Essas partes possuem como responsabilidades: (i) o design de repositório de processos e consulta de processos; (ii) preparação; (iii) execução de consulta de processos e (iv) interpretação de resultados de métodos de consulta de processos. Segundo o autor o framework possui 4 (quatro) tipos de comportamentos de modelos: logs de eventos, modelos de processos, simulação de modelos e correlação de modelos. O autor salienta que um método de consulta de processo é uma técnica que fornece um repositório de processo e uma consulta de processo implementa sistematicamente a consulta no repositório, onde uma consulta de processo é uma instrução (formal) para gerenciar um repositório de processos. Por fim, o autor apresenta uma aplicação genérica para a manipulação e o filtro de processos em repositórios, métodos de consultas que possuem como foco a mineração de processos e o gerenciamento de processos de negócio.

3.2.7 Trabalho 7 - *Efficiently querying large process model repositories in smart city cloud workflow systems based on quantitative ordering relations*

Os autores (HUANG et al., 2019) focam em uma abordagem de recuperação em duas fases (filtragem e refino) para consultar grandes repositórios de modelos de processos em sistemas de fluxo de trabalho em nuvem em cidades inteligentes. Na etapa de filtragem, é adotado um índice nomeado de TARIndex para filtrar modelos de processos em repositórios

para obter modelos candidatos; com isso, os modelos candidatos são refinados com base na sua similaridade de comportamento entre modelos candidatos e modelos de consulta. Nesta fase também foi usado o ORTPindex para filtrar uma grande quantidade de modelos de processos em repositórios e pegar o conjunto de modelos candidatos. O índice baseado na relação de ordem quantitativa com restrições de tempo e probabilidade é adotado para reduzir bastante o número de modelos candidatos em grandes repositórios de modelos de processos. Na fase de refino, é proposto um algoritmo de computação de similaridade de comportamento do processo baseado em relações quantitativas de ordenação para refinar o conjunto de modelos candidatos. Para realizar as etapas de filtragem e refino, algumas técnicas foram utilizadas, são elas: ténica para definir o comportamento de similaridade com restrições de probabilidade e tempo; algoritmo para calcular a similaridade do comportamento de processos com restrições de tempo e probabilidade; algoritmo para consultar modelos de processos baseados em relações de ordem quantitativa.

No experimento foi utilizada uma ferramenta para geração de modelos que é usada para modelos de processos sintéticos, usadas em formas de regras. Todos os labels das tarefas de modelos de processos sintéticos foram selecionados por funções de palavras do *WordNet*. Foram gerados 6 (seis) conjuntos de modelos de processos, com 100.000, 200.000, 300.000, 400.000, 500.000 e 600.000 processos sintéticos para experimentação. Como resultado, as experiências ilustraram uma proposta que pode melhorar significativamente a eficiência da consulta de grandes repositórios de modelos de processos em sistemas de fluxo de trabalho em nuvem de cidades inteligentes com base no comportamento.

3.2.8 Discussão

A Tabela 2 apresenta uma comparação dos trabalhos analisados. Na tabela, o símbolo "+" significa que o trabalho atende completamente ao requisito avaliado, o "–" indica que o trabalho atende apenas parcialmente, enquanto que o símbolo "•" representa o não atendimento ao requisito. Os requisitos estão assim identificados: Colaboração entre organizações (R1); Base de Dados Distribuída (R2); Gestão adequada de processos (R3); e Armazenamento de dados de processos (R4).

Tabela 2 – Comparação de trabalhos de repositórios para o gerenciamento de modelos de processos de negócio

Trabalhos	R1	R2	R3	R4
(LA ROSA et al.,2011)	+	+	+	•
(YAN;DIJKMAN;GREFEN;2012)	•	•	+	-
(LAZARTE et al.,2013)	•	+	+	+
(VAN DER Aa et al.,2018)	•	•	+	-
(POURMIRZA et al.,2019)	-	-	+	+
(POLYVYANYY;2019)	•	•	+	+
(HUANG et al.,2019)	•	+	+	+

Conforme mostrado na Tabela 2, a maioria dos trabalhos analisados não atende ao requisito R1 e R2 de forma completa, demonstrando de que existe uma carência de suporte qualificado para o desenvolvimento de aplicações em ambientes que suportam colaboração entre organizações e base de dados distribuídas. No que tange ao requisito R4 pode-se notar que mesmo não estando presente de forma completa em todos os trabalhos, ele é um tema de interesse relevante na maioria dos estudos apresentados. O requisito R3 mostrou-se presente de forma completa em todos os trabalhos analisados. O trabalho de (YAN; DIJKMAN; GREFEN, 2012) armazena apenas dados referentes ao metamodelo do processo, sendo assim, foi qualificado como parcialmente atendiso. Observa-se, ainda, que a pesquisa mostrou que há uma deficiência em relação à parte de colaboração entre organizações e o armazenamento dos dados gerados pelos processos, na maioria dos trabalhos. Isso demonstra que, apesar de ser um dado extremamente importante para as organizações que são movidas basicamente pelos seus dados e estimativas, não são fortemente representados nas arquiteturas conceituais atuais, gerando uma carência de pesquisa nesta área. Neste trabalho a abordagem proposta abrange os requisitos R1, R2, R3 e R4 de forma completa.

3.3 Considerações Finais do Capítulo

Neste capítulo foram apresentados trabalhos da literatura que propõem algum tipo de infraestrutura de integração entre sensibilidade a contexto e gerenciamento de processos de negócio que visam facilitar o desenvolvimento de aplicações para soluções tanto para organizações, como outras áreas do conhecimento. Foram descritos também algumas infraestruturas de repositórios para o gerenciamento de processos de negócio.

No que se refere à provisão de uma arquitetura conceitual de integração entre sensibilidade a situações e gerenciamento de processos de negócio, percebe-se pela análise da literatura da área que existe uma carência de ferramenta de apoio a construção de aplicações em ambientes organizacionais. Do mesmo modo, ainda há poucos trabalhos para apoio a soluções de repositórios de processos de negócio para o armazenamento dos dados gerados pelas organizações.

Por meio da análise realizada, foi possível atestar a importância e comprovar a necessidade de arquiteturas que enfrentem tais desafios, reforçando, portanto, a relevância de arquiteturas conceituais como a proposta nesta dissertação. Esta análise permitiu, ainda, avaliar opções de tecnologias de sensibilidade a contexto, situações, modelos de processos de negócio, entre outras, algumas das quais foram empregadas na implementação desta arquitetura conceitual e da solução para a condução do experimento empírico, os quais são apresentados em detalhes nos próximos capítulos.

Parte II

Proposta

4 Arquitetura Proposta

Este capítulo apresenta o projeto e a implementação da infraestrutura de apoio à construção de aplicações de processos de negócio desenvolvida nesta dissertação. O capítulo se inicia com uma descrição dos requisitos que nortearam a definição da arquitetura conceitual (Seção 4.1). A Seção 4.2 apresenta os componentes da arquitetura proposta. A Seção 4.3 descreve a implementação da arquitetura, ressaltando as escolhas tecnológicas adotadas para a realização do protótipo da arquitetura. A Seção 4.4 discute alguns aspectos importantes de Governança no contexto da arquitetura proposta. A Seção 4.5 conclui o capítulo.

4.1 Princípios e Requisitos

A solução proposta adota como princípio geral de modelagem conceitual o uso de uma abordagem que privilegia a 'separação de interesses' ou 'separação de preocupações' ("*separation of concerns*") (ALMEIDA et al., 2006; MELLOR et al., 2004). A separação de interesses é amplamente conhecida na comunidade de Engenharia de Software, sendo uma estratégia essencial quando se trata de lidar com um ambiente com problemas a serem trabalhados em diferentes níveis de abstração. Isso torna possível o desacoplamento das camadas de solução, evitando que os interessados de nível mais alto precisem se preocupar com os detalhes técnicos de nível inferior. Por exemplo, numa solução BPM coorporativa, sensível à situação e baseada em redes de sensores, os programadores da rede atuariam no tratamento dos aspectos sensoriais, considerando a arquitetura da plataforma de hardware e os protocolos de comunicação e, por outro lado, a definição de regras e eventos do domínio seria tratada pelo especialista do domínio, que possui um conhecimento muito maior da lógica de negócios. No caso particular da arquitetura proposta, esta separação de interesses se reflete nas seguintes escolhas: **P1** uso de uma abordagem de alto nível para a definição de situações, regras e modelos de processos de negócio; e **P2** uso de uma abordagem de baixo nível, orientada a serviços, que opera em tempo de execução e cuja característica é a flexibilidade e promoção do desacoplamento via interfaces padronizadas, por exemplo, REST (VUJOVIĆ et al., 2014) e Publish/Subscribe (EUGSTER et al., 2003).

Descrição completa dos princípios centrais da arquitetura conceitual proposta:

• **P1**. Uso de uma abordagem de alto nível para a definição de situações, regras e modelos de processos de negócio em tempo de *design (design-time)*: A abordagem em alto nível deve possibilitar aos especialistas de processos de negócio a definição de modelos de processos negócio, bem como os serviços a ele realizados, em tempo de modelagem. Esta abordagem deve possibilitar, também, que os programadores definam situações e

regras e suas respectivas ações a serem tomadas. Os modelos de processos de negócio e as ações devem ser providas de forma transparente, tanto para o especialista de processos de negócio, como para os programadores.

• **P2**. Uso de uma abordagem de baixo nível que opera em tempo de execução (*runtime*) para o provimento de soluções organizacionais: Neste nível a arquitetura deve prover, em tempo de execução, a integração entre a sensibilidade a situações e os processos de negócio que são executados pela organização. Tanto as regras quanto as situações e os modelos de processos de negócio podem fazer a utilização de serviços definidos pela organização. Neste nível, a arquitetura deve prover também um repositório de processos de negócio e suas respectivas informações para serem acessados a qualquer tempo por membros da organização. Estes processos devem ser providos pela arquitetura conceitual apenas à funcionários devidamente autorizados, respeitando assim, seus devidos papéis dentro da organização.

Além desses princípios, a arquitetura proposta deve atender aos seguintes requisitos:

• **R1**. Definição de regras e situações em *design-time*: O ambiente de definição de situações e regras deve suportar programadores definindo regras e situações pertinentes ao domínio de aplicação. Deve suportar também a definição de serviços internos e externos da organização e os respectivos processos que serão executados em tempo de execução. Nesta etapa os processos são apenas definidos como gatilhos a serem executados caso situações ou regras ocorram sendo referenciados através dos seus respectivos identificadores para que sejam coletados e executados a partir do repositório de processos.

• **R2**. Definição de modelos de processos de negócio em *design-time*: O ambiente de modelagem deve suportar que especialistas de processos de negócio modelem processos de negócio em alto nível através de modeladores de processos que sejam capazes de gerar modelos em uma sintaxe compreensível ao módulo semântico responsável por preparar os modelos de processos de negócio para que possam ser compreendidos pela Process Engine e ao repositório de processos. A adoção de boas práticas e padrões para a descrição dos modelos de processos de negócio nesta etapa é extremamente importante, uma vez que estes modelos serão executados recorrentemente pelas aplicações que os consumirem. Os processos de negócio salvos nesta etapa devem ser armazenados em um repositório de processos para que seja acessado novamente tanto por especialistas de processos de negócio para realizarem reengenharia dos processos, quanto para as aplicações que precisam destes processos para execução.

• **R3**. Nó de execução de situações, regras e modelos de processos em *runtime*: Um nó de execução deve ser definido. Este nó deve possibilitar que regras e situações executem modelos de processos de negócio como gatilhos definidos a partir das situações e regras descritos em *design-time*. Estas situações e regras também podem executar serviços internos e externos definidos em design-time. O processamento das regras, situações e

modelos de processos de negócio devem ser escaláveis e possuir um mecanismo capaz de invocar serviços e acessar sistemas externos.

• **R4**. Um serviço para o gerenciamento do repositório de processos para modelos de processos de negócio, suas informações importantes e seus serviços: A arquitetura conceitual deve prover um repositório de modelos de processos de negócio, suas informações e serviços para o gerenciamento destas informações por toda a organização. Estes dados, modelos e serviços devem ser acessados apenas por pessoas devidamente autorizadas. Estes processos podem e devem ser consumidos por aplicações sensíveis a situação e de contexto, por meio de chaves de autorização.

• **R5**. Gestão dos serviços: Deve ser possível aos especialistas e envolvidos no projeto o acesso a todos os serviços definidos na organização. Tanto na etapa de modelagem dos processos, quanto na etapa de implementação de aplicações deve ser possível ter acesso aos serviços disponíveis na organização e os acionar como parte da definição das regras de negócio. Os serviços são funcionalidades disponíveis a equipe do projeto com o objetivo de serem utilizados e reutilizados nas ações das regras e modelos de processos de negócio, por exemplo, um serviço de envio de e-mail. O baixo nível de acoplamento entre os serviços é importante, não sendo necessário conhecimento a nível de implementação acerca destes serviços, apenas seus parâmetros de entrada, e o resultado de saída esperado. É necessário que haja um repositório de serviços cadastrados para que os mesmos sejam acessados sempre que necessários por membros da organização. A centralização dos serviços definidos pela organização é de suma importância.

• **R6**. Segurança na comunicação entre aplicações e o serviço de repositório de processos: Deve existir um meio de comunicação segura entre aplicação e o repositório de dados, evitando assim, que outras aplicações não autorizadas se conectem a esta base de dados e se utilizem destes modelos de processos de negócio, assim como seus dados e serviços definidos. Deve ser possível ainda suportar protocolos da rede à qual está vinculada e ser capaz de formatar dados recebidos das redes em um formato que permita uma comunicação entre aplicação e a camada de serviço de dados. Autenticações através de chaves entre estas aplicações e o repositório de processos é necessária, uma vez que abordagens de controle de acesso baseada em papéis tira proveito deste tipo de autenticação.

4.2 Arquitetura Conceitual

A Figura 20 mostra a arquitetura conceitual proposta. São identificados os componentes, os quais são distribuídos em duas camadas, e os atores do processo que são representados nos papéis de especialista de processos de negócio e programadores de aplicações sensíveis a contexto e situações.

A arquitetura conceitual proposta é baseada nos trabalhos (LAZARTE et al., 2013; COSTA et al., 2016; MAIO; SALATINO; ALIVERTI, 2014), que descrevem conceitos de separações de papéis e arquiteturas conceituais para a representação de situações, BPMS e armazenamento de processos distribuídos dentro de organizações. Os trabalhos (SOUZA; FILHO; SPESSIMILLE, 2013; POLPINIJ; GHOSE; DAM, 2015) inspiraram e reforçaram a visão de separação de papéis e a utilização de serviços em nuvem descritos por meio de modelos de processos de negócio. Os componentes assinalados em vermelho foram implementados nesta primeira versão da arquitetura e os componentes em azul serão apresentados em implementações futuras.

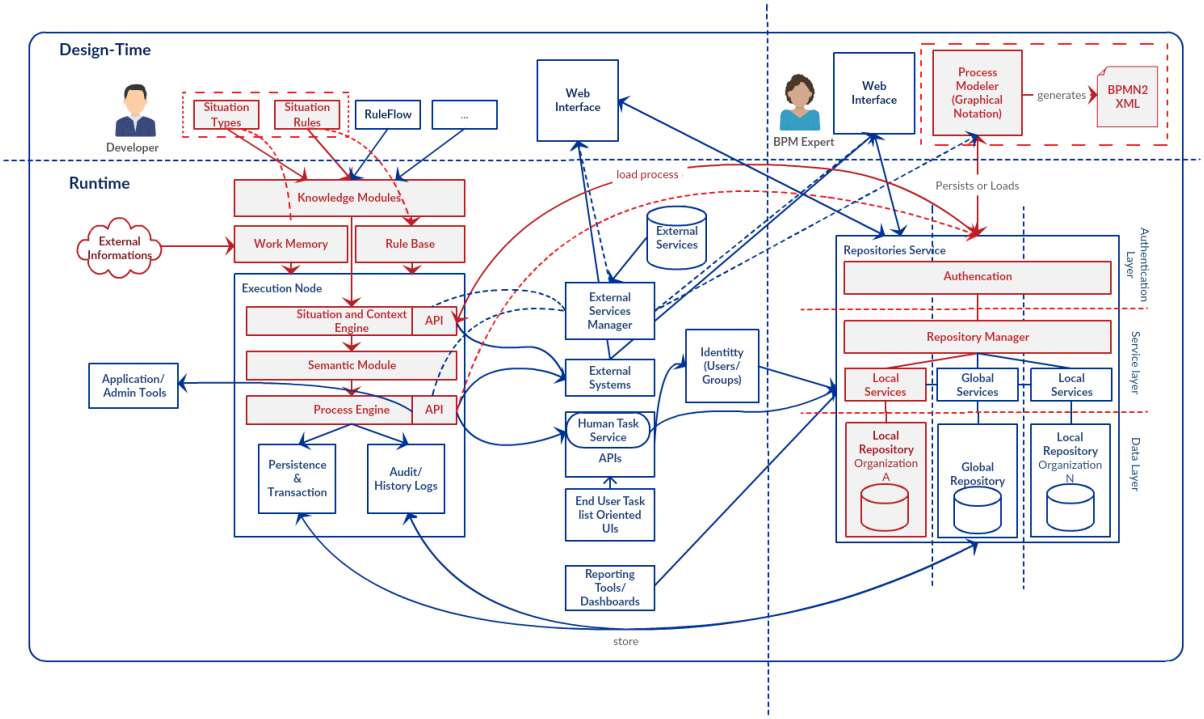

Figura 20 – Arquitetura conceitual proposta

A arquitetura é dividida em dois níveis:

(i) nível de tempo de design ("*Design-Time*"), para programadores e para especialistas de processos de negócio; e

(ii) nível de tempo de execução ("*Runtime*"), que suporta a execução das aplicações da organização e mantém o repositório de processos.

O nível de tempo de design é separado em design-time para desenvolvedores ("*Developers*"); e *design-time* para especialistas de processos de negócio ("*BPM-Experts*"). O nível de tempo de execução é separado em aplicações sensíveis a situações e de runtime para o repositório de processos.

O nível de runtime para repositório de processos pode ser separado em três camadas: (i) uma camada de autenticação; (ii) uma camada de serviços; e (iii) uma camada de dados.

Esta divisão por níveis visa à separação de responsabilidades entre os programadores/-desenvolvedores e os especialistas de processos de negócio. Os desenvolvedores detêm o conhecimento sobre situações, regras e do desenvolvimento de aplicações. Os especialistas de processos de negócio possuem o conhecimento sobre o domínio do processo, técnicas de modelagem, *workflow* do processo, entre outras particularidades referentes ao processo. Esta abordagem permite que cada um dos atores lide com seus respectivos problemas específicos, sem a necessidade de lidar com exigências que fogem da sua área de especialização (RODRIGUES et al., 2010). Tal abordagem possibilita que os desenvolvedores lidem apenas com particularidades acerca do desenvolvimento e especialistas de processos de negócio lidem apenas com domínios envolvendo processos de negócio.

O nível de design-time para BPM Experts provê um ambiente para a modelagem de processos de negócio em alto nível que gera arquivos no formato BPMN2 XML para o armazenamento no repositório de processos. Estes processos são modelados por componentes adequados de modelagem de notação gráfica BPMN. Os *BPM Experts* que atuam nesse nível podem definir serviços a serem executados pelos processos de negócio; entretanto, eles precisam saber apenas quais serviços existem, seus parâmetros de entrada e o seu resultado de saída. Esta abordagem abstrai o nível de implementação dos serviços e possibilita que os *BPM Experts* não necessitem ter conhecimento de baixo nível para a modelagem dos processos de negócio.

O nível de *design-time* para os *Developers*, por sua vez, abstrai o conhecimento de baixo nível no qual as aplicações estão rodando, e permite ao desenvolvedor se preocupar apenas com a definição dos tipos de situações, regras de situações etc. A definição dos *situation types*, *situation rules*, *ruleflows*, dentre outros, são as preocupações nesta etapa. Por exemplo, situation types definem os aspectos estruturais do domínio, ou variáveis a serem analisadas e que, decorrente do contexto na qual estão inseridas, possuem algum significado para o negócio. Situation rules, por sua vez, definem aspectos comportamentais da situação, ou seja, quando um determinado conjunto de parâmetros ao longo do tempo se encaixar nas condições de domínio a serem analisadas, os comportamentos definidos nesta etapa serão seguidos. Adicionalmente, uma *Web Interface* foi definida no nível de design-time devido à necessidade dos desenvolvedores de consultarem rotineiramente identificadores de modelos de processos de negócio, serviços internos e externos, e dados que podem ser acessados no repositório de processos, dentre outros.

O nível de *runtime* para aplicações sensíveis a situação deve prover um ambiente de desenvolvimento adequado, no qual as aplicações que já funcionam na organização possam apenas carregar as definições de situações e contextos, seus respectivos modelos de processos de negócio definidos pelos especialistas de processos de negócio e executar, sem a necessidade de interação humana neste processo. Caso o desenvolvedor vá definir uma nova aplicação para a solução de um problema organizacional, ele definirá apenas as

regras de alto nível descritas em tempo de *design* e os processos que serão executados. Os serviços que serão executados pelos modelos de processos negócio devem ser definidos pelos especialistas de processos de negócio. A ocorrência de situações ou contextos específicos podem disparar serviços internos ou externos que serão carregados a partir do gerenciador de serviços externos ou internos, e uma aplicação base deve ser definida para rodar o conjunto de regras e situações definidas.

O nível de tempo de execução para o repositório de processos é separado em três camadas. A primeira camada é a camada de autenticação. Esta camada deve prover um meio seguro de acesso dos processos, serviços e dados da organização. Uma abordagem de controle de acesso por definição de papéis deve ser adotada, tendo em vista os diversos níveis hierárquicos dentro de uma organização. A segunda camada é a camada de gerenciamento de serviços que provê acesso a modelos de processos de negócio, dados de processos e serviços internos da organização. Esta camada deve prover acesso adequado entre os diversos serviços e processos definidos pela organização. E por fim, a camada de dados que é a camada onde todos os dados são persistidos.

No nível de *design-time* do desenvolvedor temos quatro componentes, a saber: *Situation Types*, *Situation Rules*, *RuleFlow* e *Web Interface*. De acordo com (COSTA et al., 2016) os *situation types* e os *situation rules* são especificados por meio de aspectos estruturais e comportamentais, ou seja, nos situation types são definidos os aspectos estruturais do domínio, ou variáveis a serem analisadas e que, decorrente do contexto na qual está inserida, possuem algum significado para o negócio. Os *situation rules*, por sua vez, definem aspectos comportamentais da situação, ou seja, quando um determinado conjunto de parâmetros ao longo do tempo se encaixar nas condições de domínio a serem analisadas, seguirá os comportamentos definidos nesta etapa. O *ruleflow* é uma abordagem de análise de contexto que pode ser implementada e é suportada pela arquitetura conceitual. Estes componentes são tidos como componentes de conhecimento da arquitetura, que será um pouco mais detalhado no módulo de *Knowledge Modules*. Uma Web Interface foi definida em design-time devido às necessidades de os desenvolvedores consultarem rotineiramente esta interface para tomarem conhecimento dos identificadores de modelos de processos de negócio, serviços internos e externos, dados que podem ser acessados no repositório de processos, entre outros.

No nível de *design-time* do especialista de processos de negócio são descritos três componentes, são eles: *Process Modeler*, BPMN2 XML e a *Web Interface*. A *Web Interface* neste nível possui as mesmas características descritas no parágrafo anterior, no nível de *design-time* do desenvolvedor. O Process Modeler é um componente capaz de modelar em alto nível modelos de processos de negócio, descrevendo suas tarefas, serviços, atividades de início e fim, condicionais, pools, lanes, entre outros elementos, definindo assim o seu chamado workflow do processo. Sendo assim, o especialista de processos de negócio deve

interagir com este componente e descrever o *workflow* dos processos por meio de abordagens de desenho arrastando e soltando (dragging-and-dropping), e gerar um arquivo no formato BPMN2 XML que é capaz de ser interpretado pelas aplicações de baixo nível, pelos próprios componentes de modelagem para reengenharia ou redesenho dos processos e armazenados no repositório de processos. Este componente também deve prover a capacidade de associar serviços aos processos de acordo com as necessidades e a demanda da organização. No módulo semântico será descrito mais detalhadamente a etapa de interpretação destes arquivos BPMN2 XML no baixo nível da arquitetura conceitual.

No nível de *runtime* de aplicações sensíveis a situação são descritos dezesseis componentes. Estes componentes se baseiam na estrutura do Drools (BALI, 2009), jBPM (MAIO; SALATINO; ALIVERTI, 2014; KOENIG, 2004) e Scene (PEREIRA; COSTA; ALMEIDA, 2013), permitindo que abordagens de análise de contexto, sensibilidade a situações e utilização de modelos de processos de negócio sejam suportadas pela arquitetura. Os componentes a saber são:

- *Knowledge Module*: é um componente que possibilita que pequenos módulos de alto nível sejam compreendidos pelo sistema como um todo para que haja interação com o baixo nível. Na arquitetura conceitual foram listados *Situation Types*, *Situation Rules*, *RuleFlow*; entretanto, outros componentes no nível de design-time podem ser suportados pela arquitetura, e é papel deste módulo interpretar essas novas funcionalidades de modelos que são reconhecidos pela arquitetura e gerenciar as funcionalidades adequadas para o bom funcionamento do sistema como um todo. Por exemplo, supondo que seja utilizado o descritor de regras de negócio do Drools, ou um modelo de processos de negócio, é função deste módulo diferenciar que o arquivo de entrada se refere a regras de negócio ou processos de negócio.

- *Work Memory*: As fontes de dados provenientes do contexto são armazenadas neste componente para que posteriormente sejam casadas com as regras de contexto no Execution Node. A arquitetura conceitual descreve essas informações provenientes do contexto como uma nuvem, que representa as informações que são inseridas nesta chamada memória de trabalho.

- *Rule Base*: Este componente armazena as *Situation Types* e *Rules* definidas em algo nível em *design-time*. Na arquitetura conceitual é utilizada uma seta pontilhada para representar a relação direta entre estes dois componentes.

- *Execution Node*: Este nó de execução é basicamente uma caixa preta para o alto nível. Seus detalhes de implementação são descritos abaixo:

  - *Situation and Context engine e API*: É o módulo responsável por realizar o casamento dos fatos da *Work memory* com as regras e situações definidas na

Rule Base. Este casamento é definido como Pattern Matcher ou casamento de padrões. A API deste módulo pode ser definida como o conjunto de funções de baixo nível definidas capazes de interagir diretamente com gatilho de situações e contexto, compreensão de módulos de processos, sinais de eventos, finalização da ocorrência de uma situação ou evento de contexto, realizar consultas de modelos de processos de negócio no repositório de processos, interagir com classes externas, entre outras.

  - O módulo semântico é responsável por preparar os modelos de processos de negócio coletados no repositório de processos. Nos processos definidos no padrão BPMN2 XML um *parser* é realizado, possibilitando que cada tags definida neste modelo passem a ter significado para a *Process Engine*. Ou seja, este módulo realiza a tradução para estruturas internas para que a *Process Engine* possa executar o processo. Este módulo permite ser estendido para suportar a definição de múltiplas linguagens e permite que os usuários modifiquem as linguagens já definidas para a interpretação de casos específicos, permitindo que seja adicionado extensões de interpretações de processos a este módulo.
  - *Process Engine* e sua *API*: Este é o módulo responsável por carregar e executar os modelos de processos de negócio. Ele cria instâncias de processos e mantém seus passos internos. Este módulo possui também uma API para interação com outros componentes externos, interação com classes e serviços externos, capacidade de analisar passo-a-passo do funcionamento do processo, capacidade de executar processos de forma síncrona ou assíncrona, entre outros.
  - *Persistence and Transaction*: Este módulo é extremamente importante para a *Process Engine*. Os sistemas BPM são usualmente integrados com sistemas e pessoas para atingir objetivos comuns decidindo quais os próximos passos a serem tomados em frações de segundos; entretanto, executar cada etapa de um processo poderia levar muito tempo. Este módulo é um mecanismo que permite a liberação de recursos do ambiente da Process Engine, carregamento de distribuições, e mecanismo de recuperação a falhas.
  - *Audit / History Log*: Os logs são responsáveis por armazenar dados relacionados as transações, possibilitando assim, buscas históricas a respeito da execução dos processos. É um componente especifico para armazenar diferentes modificações nas instâncias dos processos, passos executados e variáveis.

- *Application / Admin Tools*: É um componente que se conecta diretamente ao andamento dos processos. Provê também uma metodologia para o gerenciamento dos processos por parte dos administradores a partir de ferramentas de administração. É extremamente importante para organizações que necessitam saber o passo-a-passo do processo, permitindo um monitoramento desses passos a fim de medir indicadores de

qualidade, eficiência, eficácia, etc, em tempo de execução do processo para tomada de decisão estratégica da organização. Por exemplo: caso seja necessária a aprovação de um membro da direção e este membro saiu de férias recentemente, por meio desta ferramenta é possível modificar a atribuição da responsabilidade antes do início do processo ou em tempo de execução do mesmo. Tudo isso é possível através da comunicação direta da API da Process Engine.

- *External Services Manager*: Este é o componente que está vinculado ao componente de serviços externos. Ele é responsável por mapear todos os serviços externos da organização, possibilitando que sejam acessados tanto pelos componentes de modelagem de processos de negócio como interfaces web para a listagem desses serviços tanto para desenvolvedores, quanto para especialistas em processos de negócio.

- *External Services*: Os serviços externos são todos os serviços externos utilizados pela organização. Estes serviços são representados por este componente de forma que os especialistas de processos de negócio tomem ciência de sua existência para implementações atuais ou futuras. Não existe uma ligação direta entre estes serviços e as aplicações desenvolvidas pela organização, uma vez que a execução destes serviços externos poderia ser feita através da execução de modelos de processos de negócio, não limitando a aplicação a este serviço.

- *External Systems*: Este componente representa as relações entre sistemas externos nos quais os processos podem se comunicar, ou até mesmo realizar requisições referentes a alguma aprovação, validação ou algum estado específico relevante para o processo com o objetivo de tomada de decisão. Este componente é responsável por não tornar as aplicações monolíticas gerenciando coisas apenas de um único lado, sendo assim, este componente provê comportamentos de conexões estratégicas para configurar e/ou interagir com códigos externos entre processos e outros sistemas.

- *Human Task Service e APIs*: Responsável por processos referentes a tarefas humanas, ou seja, tarefas de um processo que necessitam de aprovações humanas para que continuem. Ele permite gerenciar particularmente o que está preparado para ser feito a partir de pequenas interações humanas. Um processo de negócio pode necessitar não apenas de uma aprovação para que uma tarefa seja concluída, mas sim de várias, dependendo da necessidade da atividade. Por exemplo, uma atividade pode precisar da interação humana para a aprovação de uma tarefa por parte de cinco diretores de uma empresa para que a mesma seja concluída. Visto isso, torna-se importante que o componente de gerenciamento de tarefas humanas seja especificado como um componente externo, pois o seu gerenciamento é algo de extrema importância por parte de uma organização. Este componente também interage com componentes de notificação para que as pessoas de um determinado domínio tomem conhecimento

sobre as tarefas nas quais elas precisam interagir. Sendo assim, este componente é responsável por garantir de forma segura que todas as tarefas caiam nas mãos corretas para suas respectivas execuções. A *API* é responsável por prover acesso aos dados referentes as tarefas humanas a serem executadas para que medidas pertinentes a elas sejam tomadas.

- *Identitty (Users / Groups)*: Este componente ligado a interação com tarefas humanas vai permitir que o componente de tarefas humanas submeta tarefas para diferentes grupos e usuários dependendo do potencial de usuários, grupos, e administradores definidos em cada tarefa. Estes grupos de usuários são definidos de forma segura com base em algum contexto. Isto permite que seja provido como um entendimento de tarefas humanas com base no domínio a critério de validação de usuários ou grupos.

- *End User Task list Ordened UIs*: Este componente será utilizado para prover por meio da tarefa de serviço uma lista de tarefas relevantes para diferentes grupos e usuários sobre as formas com as quais cada um interagiu com a tarefa. É um componente básico para controle do estado de cada tarefa, ou seja, proverá um caminho para a manipulação das tarefas e possibilitará que a process engine armazene um estado a respeito de cada tarefa que necessite de interação humana, ou seja, registrará se foram aprovadas, desaprovadas, puladas, pendentes, entre outras. Proverá também uma interface para que outros usuários vejam o andamento delas, possibilitando tomadas de decisões com base nas decisões de outros membros da organização.

- *Reporting Tools / Dashboards*: É um componente responsável por gerar um histórico de logs em tempo real para uma ferramenta de monitoramento. É um componente que usualmente é utilizado para mostrar informações relacionados a poucas horas ou meses de execuções no ambiente e apresenta um caminho de assistência aos usuários para ajudar em julgamentos sobre a eficiente do ambiente. São responsáveis por fornecer dados sobre o estado do processo, seu respectivo andamento e os dados gerados por cada processo. Possibilita também que a alta gestão da organização acompanhe a execução dos processos.

- *External Informations*: São as informações representadas por dados que possuem representatividade para um determinado domínio. As informações externas podem fazer parte do conjunto de informações que fazem sentido ou não para o sistema, as informações que fazem sentido para o sistema são conhecidas como o conjunto de dados relevante a ser analisado e permite a ativação de situações caso um determinado evento ocorra, as informações externas que não são relevantes neste momento, podem ser descartadas ou armazenadas pela organização, caso seja necessária para analises futuras, entretanto, este armazenamento fica a cargo da decisão de projeto.

Por fim, temos o nível de runtime para o repositório de processos que é separado em três camadas. As camadas são: *Authentication Layer, Service Layer e Data Layer*, que estão contidas em um serviço nomeado de *Repositories Service*. Este serviço proverá: (i) uma camada de autenticação para acessos seguros para os serviços, processos e conteúdo de processos; (ii) uma camada de serviços responsável por gerenciar os serviços providos pela organização como um todo e os serviços providos por apenas uma organização em específico. Ou seja, será possível acessar os serviços da organização de forma transparente independentemente de onde este serviço estará rodando, desde que a aplicação e o requerente pelo serviço estejam corretamente autorizados a acessar aquele serviço; (ii) uma camada de dados, que será responsável por persistir todos os dados de processos e dos serviços da organização.

Na camada *Authentication Layer* a arquitetura conceitual propõe um método de autenticação por distinção de papéis dentro da organização, conhecido como *Role-Based Access Control* (RBAC), ou Controle de Acesso baseado em Papéis, que possibilita que a organização divida seus funcionários e atribua responsabilidades a cada um baseado em seu papel dentro da organização, o que permite que somente pessoas autorizadas acessem determinados serviços e processos, preservando a integridade e o sigilo de dados importantes para a organização. Esta abordagem separa em papéis e níveis de permissões dentro dos níveis hierárquicos da organização, ou seja, pessoas com nenhum acesso, que apenas podem visualizar, que possuem um controle restrito, possuem controle total e administradores. As aplicações que requerem a autenticação deste serviço são autenticadas via *JSON Web Token* (JWT), que podem ser implementados de maneira isolada ou por meio de autenticações RBAC. A utilização de JWT pode ser unida aos mecanismos RBAC, permitindo que aplicações sejam categorizadas pelos seus respectivos papéis e restringindo o acesso destas aplicações aos dados e serviços internos da organização. Sendo assim, o objetivo desta camada é fornecer uma maior segurança para a organização no nível de acesso de seus dados.

Na camada de *Service Layer* a arquitetura conceitual propõe um método para o gerenciamento e utilização dos serviços. Uma camada chamada *Repository Manager* foi definida para gerenciar o armazenamento dos processos de negócio e serviços com base em cada organização. Este componente possui como objetivo prover um gerenciamento de forma transparente aos processos de negócio e serviços para a organização, não sendo necessário, no nível de aplicação e modelagem, definir onde e de onde os processos estão ou serão armazenados ou coletados. Este nível é importante para criar uma transparência a respeito de todos os serviços e processos da organização, não sendo necessária a definição de processos e serviços já existentes na organização como um todo. Os processos e serviços são utilizados a todo tempo pela organização; sendo assim, o componente de *Local Service* ou *Global Service* será responsável por armazenar e gerenciar os serviços importantes para a organização. Todos eles serão acessados sem a necessidade de se especificar aonde este

serviço está rodando, tornando a utilização dos serviços uma caixa preta quanto a sua utilização. Os serviços locais são definidos no nível de uma organização específica, que entende a falta de um determinado serviço dentro da organização como um todo. Os serviços globais são os serviços que são executados de forma mais rotineiras dentro da organização e que sempre são executados por todas as entidades ligadas a esta organização que a compõem.

Por fim, a camada de *Data Layer* possui como objetivo armazenar e gerenciar todos os dados referentes a modelos de processos de negócio e serviços. Cada entidade organizacional que compõem uma organização pode e deve possuir seu próprio repositório local, possibilitando um armazenamento distribuído com relação a organização como um todo. A disponibilidade dos serviços de armazenamento, prevenção contra falhas, escalabilidade, entre outros aspectos de baixo nível, não foram levados em conta nesta representação, uma vez que isso é particular de cada organização e todas possuem seus próprios mecanismos de gerenciamento destes serviços.

Esta abordagem em si, possibilita ser implementada em diversas plataformas operacionais, como Windows, Linux e Macintosh. Várias engines no mercado de processos e de regras e situações podem ser integradas com tal abordagem e as técnicas utilizadas para a distribuição de dados distribuídos são utilizadas no mercado. Os serviços da camada de Repositories Service pode ser facilmente implementada com abordagens SOA.

4.3 Aspectos de Governança de SOA

Um aspecto fundamental tratado nesta arquitetura conceitual envolve conceitos pertinentes à governança de TI, uma vez que esta está ligada diretamente com o alinhamento estratégico da empresa e deve ser uma preocupação da organização como um todo, partir do alto escalão da organização responsável pelo planejamento estratégico até o nível operacional.

A governança de SOA é uma especialização de governança de TI que coloca as principais decisões de governança de TI dentro do contexto do ciclo de vida dos componentes de serviços, serviços e processos de negócios. É o gerenciamento efetivo desse ciclo de vida que é o principal objetivo da governança de SOA.

A governança de SOA aborda aspectos do ciclo de vida do serviço, como planejamento, publicação, descoberta, controle de versão, gerenciamento e segurança. No SOA, os consumidores de serviços e provedores de serviços são executados em diferentes processos, são desenvolvidos e gerenciados por diferentes departamentos e exigem muita coordenação para trabalhar juntos com êxito. Para que o SOA seja bem-sucedido, vários aplicativos precisam compartilhar serviços comuns, o que significa que eles precisam coordenar para tornar esses serviços comuns e reutilizáveis. Estas são questões de governança, e são muito

mais complexas do que nos dias de aplicações monolíticas ou mesmo nos dias de código e componentes reutilizáveis (MITRA, 2005).

Na prática, a governança SOA orienta o desenvolvimento de serviços reutilizáveis, estabelecendo como os serviços serão projetados e desenvolvidos e como esses serviços serão alterados ao longo do tempo. Estabelece acordos entre os provedores de serviços e os consumidores desses serviços, dizendo aos consumidores o que eles podem esperar e os provedores que eles são obrigados a fornecer. A governança SOA não projeta os serviços, mas orienta como os serviços serão projetados (MITRA, 2005). Isso ajuda a responder a muitas questões espinhosas relacionadas à SOA: Quais serviços estão disponíveis? Quem pode usá-los? Quão confiáveis são eles? Por quanto tempo eles serão suportados? Você pode depender deles para não mudar? E se você quiser que eles mudem, por exemplo, para corrigir um bug? Ou para adicionar um novo recurso? E se dois consumidores quiserem que o mesmo serviço funcione de maneira diferente? Só porque você decide expor um serviço, isso significa que você é obrigado a apoiá-lo para sempre? Se você decidir consumir um serviço, pode ter certeza de que não será encerrado amanhã?

Por fim, vale ressaltar que a governança é mais um problema político do que tecnológico ou empresarial. A tecnologia se concentra em interfaces e protocolos de invocação correspondentes. O negócio se concentra na funcionalidade para atender os clientes. Tecnologia e negócios estão focados em requisitos. Enquanto a governança se envolve nesses aspectos, ela se concentra mais em garantir que todos estejam trabalhando juntos e que esforços separados não se contradigam. A governança não determina quais são os resultados das decisões, mas quais decisões devem ser tomadas e quem as tomará.

4.4 Arquitetura de Implementação

A arquitetura de implementação é mostrada na Figura 21, onde pode se destacar a estrutura tecnológica utilizada para a viabilizar e atender os requisitos da arquitetura conceitual definida na seção anterior.

Segundo a abordagem de desenvolvimento utilizada neste trabalho, o especialista de processos de negócio inicialmente constrói modelos de processos de negócio para solucionar problemas organizacionais, modelos estes que serão armazenados em um repositório de processos de maneira adequada. Esta modelagem será representada por uma linguagem de representação de negócios em alto nível, adequada para o gerenciamento de processos de negócio e alinhamento estratégico da organização. O desenvolvedor, por sua vez, abstrai os detalhes de modelagem de processos de negócio e, por meio de um editor de regras e situações define as regras pertinentes ao negócio em design-time. São definidas nesta etapa as regras e eventos pertinentes à solução do problema de um determinado domínio, por meio dos componentes de *situation type* e *situation rules* definidos na arquitetura.

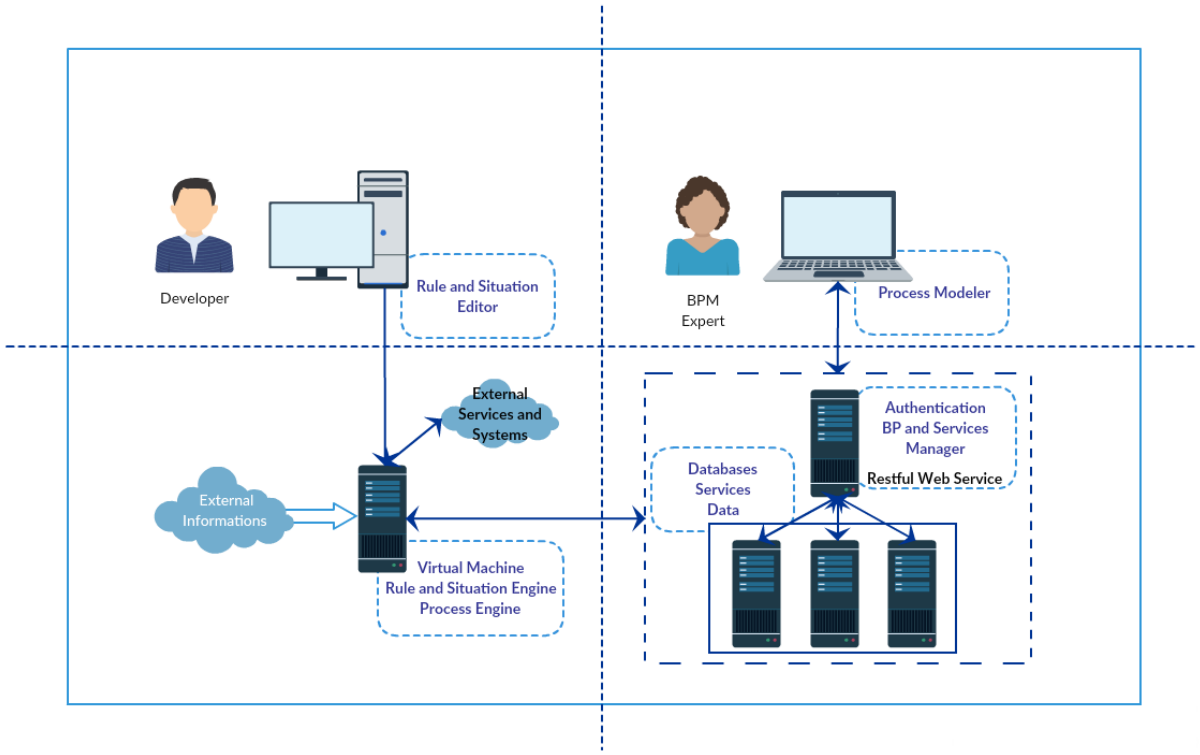

Figura 21 – Arquitetura de Implementação

Os processos poderão ser utilizados à medida que necessários em cada uma das regras e situações a partir da sua referência no repositório de processos, bem como os locais de onde os dados do domínio a serem obtidos serão coletados.

A notação para a definição de modelos de processos de negócio precisa ser adequada para a representação de problemas de maneira a abstrair conceitos de baixo nível, de forma que seja uma linguagem de fácil entendimento por todos os níveis de hierarquia da organização. O entendimento comum acerca do entendimento do modelo de processos de negócio é vital para o engajamento de todas as partes interessadas e objetivos organizacionais. Esta abordagem, possibilita que os desenvolvedores tenham pouco ou nenhum conhecimento no que tange o conhecimento de modelos de processos de negócio, preocupando-se apenas com conceitos de implementação a nível de domínio e recursos necessários por alguns processos.

Para que os especialistas de processos de negócio possam iniciar a modelagem dos processos de negócio, é necessário um entendimento comum entre as partes interessadas do projeto sobre o domínio como um todo, bem como suas necessidades e problemas a solucionar com este projeto. Sendo assim, torna-se necessário realizar levantamento de requisitos e planejamento acerca do modelo a ser desenhado. Um aparato documental deve ser extraído ao final desta etapa, possibilitando que ao ler os documentos os especialistas de processos de negócio possam propor uma solução adequada de acordo com as necessidades das partes interessadas. Após a compreensão do problema como um todo por parte dos especialistas de processos de negócio é feita a documentação de todo o levantamento de

requisitos, planejamento, modelos necessários, etc. É de suma importância que exista um documento de serviços que serão utilizados pelo modelo do processo na documentação do(s) modelo(s) que serão utilizados na etapa de modelagem. Esta documentação dos serviços deve, por sua vez, levar em consideração o grupo de serviços internos e externos gerenciados pela organização, detectando assim os serviços que serão necessários ser implementados e/ou existentes e providos pela própria organização. Esta metodologia possibilitará que os desenvolvedores tenham total conhecimento das necessidades dos modelos de processos de negócio acerca dos serviços que a eles estão atrelados, evitando implementações desnecessárias de serviços dentro da organização.

Após realizar a documentação, os especialistas de processos de negócio realizam suas respectivas modelagens referentes ao domínio do problema por meio de um editor gráfico de modelos de processos de negócio, por exemplo, *BPMN2 Diagram Editor*, que é a ferramenta utilizada neste trabalho. Esta ferramenta deve ser capaz de salvar modelos de processos de negócio em padrões compreensíveis por outros editores de modelos de processos de negócio e a escolha desta ferramenta deve ser uma preocupação da organização como um todo, uma vez que uma má escolha pode acarretar em perdas de modelos de processos de negócio futuros da organização. A ferramenta BPMN2 Diagram Editor permite que modelos de processos de negócio sejam salvos em XML, possibilitando sua importação futuramente em outros editores de modelos de processos de negócio ou até mesmo a possibilidade de serem consumidos por engines de processos. A estratégia de utilização de arquivos XML para a representação de processos é adotada por grandes suítes de BPM, por exemplo: jBPM, Oracle BPM suíte e Activiti. Possui abordagem de modelagem em alto nível por meio de desenhos dragging-and-dropping pelo editor por meio de imagens *Scalable Vector Graphics* (SVG) e também possibilita que serviços sejam definidos em tempo de modelagem em modelos de processos de negócio. Sendo assim, a escolha deste editor se mostrou adequada para a utilização nesta arquitetura de implementação, satisfazendo o R1 proposto anteriormente.

Para que os desenvolvedores comecem a implementação das aplicações pertinentes a um determinado domínio, os especialistas em processos de negócio precisam necessariamente modelar e documentar de forma adequada as diretrizes organizacionais todos os requisitos, planejamentos e modelos necessários acerca das etapas de desenvolvimento de software e gerenciamento de processos de negócio, possibilitando que os desenvolvedores tenham total conhecimento de todas as necessidades do problema.

Esta abordagem permite aos desenvolvedores realizarem escolhas adequadas referente as linguagens e tecnologias que serão adotadas no processo de desenvolvimento, abstraindo assim a necessidade do conhecimento de modelos de processos de negócio dentro da organização e se preocupando apenas com questões de mais baixo nível de implementação quando comparado com modelos de processos de negócio e, aspectos pertinentes a

definição de regras e tipos de situações do domínio do problema. Os desenvolvedores nesta etapa não se preocupam com características de baixo nível referentes ao armazenamento dos dados e serviços armazenados no repositório de processos de negócio. Os desenvolvedores estão preocupados com questões como: definição das *situation types*, *situation rules*, *rules*, serviços que serão utilizados pela aplicação e iniciados por meio de situações e/ou regras, os dados que serão consumidos para alimentar a aplicação de domínio, os tipos de dados que serão recebidos, implementações de baixo nível de processos, quando necessário, tecnologias a serem adotadas.

Um ambiente de definição de situação deverá ser adotado, permitindo que os desenvolvedores definam regras e tipos de situações. O ambiente de edição dessas regras deverá gerar arquivos capazes de serem compreendidos por tecnologias de baixo nível para o processamento de regras e situações de acordo com o modelo de infraestrutura adotado pela organização. Este ambiente deverá também possibilitar integração com métodos definidos na *API Situation and Context Engine* e da *process Engine*, possibilitando aos desenvolvedores um acesso rápido e fácil a todos os métodos nelas disponíveis, provendo assim, um melhor ambiente de desenvolvimento. Apesar dos desenvolvedores não realizarem a modelagem dos processos de negócio, a integração com a *API* da *Process Engine* é fundamental para a utilização dos métodos que envolvem os modelos de processos de negócio, por exemplo: o método que inicializa um modelo de processos de negócio, ou que verifica o estado do processo que está rodando em memória, etc. Além disso, a integração com a *API* da *Situation and Context Engine* é importante para possibilitar uma maior agilidade na implementação das regras e situações definidas no escopo do domínio. Esta *API* deverá conter funções para avaliações de regras, atualização de *work memory*, remoção de dados da *work memory*, funções para análise de estado dos dados, etc. Neste trabalho o ambiente de desenvolvimento escolhido para a definição destas regras e situações foi o *Drools IDE*. Este ambiente de desenvolvimento possibilita criar regras no formato .drl que são utilizados posteriormente pela *Situation and Context Engine*. Esta *IDE* possibilita também a integração com a API da *Process Engine* adotada, bem como a possibilidade de integração com serviços e aplicações internas e externas à organização. Por meio da API da *Process Engine* é possível definir gatilhos de inícios de processos dentro das regras e situações definidas para que processos sejam instanciados tanto de forma síncrona, quanto de forma assíncrona, dependendo da necessidade do domínio na qual está inserida. Como os processos não são armazenados de forma local à aplicação, o acesso ao serviço no qual a base de dados está contida também deve ser possível por meio desta IDE, possibilitando que os modelos de processos de negócios sejam trazidos do repositório de processos para a aplicação, evitando duplicação de modelos de processos e possibilitando a reutilização. Tendo em vista todos os pontos acima destacados, a escolha desta *Drools IDE* tornou-se a mais adequada nesta arquitetura de implementação, por satisfazer os pontos destacados no R2.

Após o especialista de processos de negócio definir os modelos de processos de negócio em alto nível e armazenar no repositório de processos, e os desenvolvedores descreverem o domínio por meio de regras de situações e tipos de situações, é a hora de executar a aplicação instanciada, que tomará como base os arquivos inicialmente descritos pelos desenvolvedores que referenciam seus respectivos modelos de processos de negócio e serviços definidos em alto nível. Inicialmente, os componentes definidos em alto nível como *situation type* e *situation rules* devem ser analisados pelo *Knowledge Module*, que é o módulo de conhecimento da arquitetura. Este módulo deverá direcionar o fluxo por meio do tipo de arquivo que está sendo manipulado para o baixo nível da arquitetura, por exemplo: instanciados arquivos *Situation type* e *Situation rules*, enviar as *Situation rules* para a *Rule base* e os *Situation types* para o nó de execução. A *work memory* carregará todos os dados provenientes da nuvem ou da rede interna da organização para tomada de decisão em tempo de execução da aplicação. Esta informação é proveniente do componente descrito na arquitetura conceitual como informação externa (*External information*). Feito isto, o nó de execução trabalhará com os componentes advindos da rule base e da *Work memory*. Estas informações serão gerenciadas e utilizadas pelo componente *Situation and Context Engine*, que será capaz de realizar casamento de padrões entre as regras e os dados da *Work memory*, agendar um lançamento para a execução destes eventos e efetivar esta execução. O casamento de padrões é conhecido como Pattern Match, o agendamento como Agenda, e a execução efetiva destes eventos é realizada pela *Execution Engine*.

Na arquitetura conceitual os conceitos de *Pattern Match*, Agenda e *Execution Engine* foram todos encapsulados em *Situation and Context Engine* para facilitar o entendimento. O componente *Situation and Context Engine* deverá realizar quase toda a execução da aplicação pertinente ao domínio. Os serviços e sistemas internos e externos à organização serão trabalhados neste nível da aplicação. A *API* deste componente deverá prover uma série de funções que possibilite tal execução. Durante a execução da aplicação neste componente, poderá ser necessário instanciar um determinado modelo de processo de negócio descrito no arquivo de *situation rules*; sendo assim, será necessário ir ao repositório de processos, coletar o processo requisitado, enviar para o módulo semântico para realizar os parsers necessários de identificação dos componentes no processo e enviá-los para a Process Engine. A função do módulo semântico é basicamente identificar componente por componente descrito no arquivo do repositório. Como mencionado anteriormente, nesta arquitetura de implementação foi adotado o *BPMN2 Diagram Editor* e os arquivos são salvos em XML, ou seja, será realizado uma série de *parsers* pela Process Engine nas *tags* XML, quebrando componente a componente deste arquivo de forma a ser compreensível pela *Process Engine*. A *Process Engine*, de posse destes componentes, poderá executar o processo de forma assíncrona ou síncrona, de acordo com a necessidade do projeto. Este componente deverá prover uma *API* para a análise do processo como um todo, desde a sua instanciação, até o seu término. Esta API deverá ser visível ao componente *Situation*

and Context Engine, para que seus métodos possam ser chamados durante a sua execução.

A execução dos componentes *Process Engine* e *Situation and Context Engine* poderá ocorrer em paralelo, de acordo com a necessidade do projeto. Por exemplo, caso o processo seja síncrono, a Situation and Context Engine, ao disparar um processo, esperará a execução do processo terminar para que dê prosseguimento à próxima execução interna; entretanto, caso a execução do processo seja assíncrona, a *Process Engine* executará o processo em paralelo e a *Situation and Context Engine* poderá continuar seu fluxo de execução normalmente. Ambos os módulos podem realizar acesso aos serviços internos e externos da organização, bem como os seus sistemas definidos na interface web de acesso comum aos especialistas de processos de negócio e aos desenvolvedores, entretanto, está necessidade de acesso deverá estar descrita na documentação entregue pelo especialista de processos de negócio e descrita como uma necessidade de domínio. Após a execução de toda a aplicação na *Situation and Context Engine* e na *Process Engine*, bem como suas interações com o repositório de processos, a aplicação será finalizada ou, caso necessária, executará por tempo indefinido, até não ser mais necessária. A *Process Engine* interage com vários componentes descritos na seção anterior, por exemplo: *Persistence & Transaction*, *Audit / History Log*, *Human Task Services*, *External Service Manager*, *External Systems*, etc. Nesta arquitetura de implementação foram utilizadas as seguintes ferramentas para satisfazer as necessidades do R3: Drools (BALI, 2009), jBPM (MAIO; SALATINO; ALIVERTI, 2014; KOENIG, 2004) e Scene (PEREIRA; COSTA; ALMEIDA, 2013).

O Drools possibilita a definição de regras de negócio de uma maneira mais natural e declarativa, tornando capaz a criação de aplicações com maior dinamicidade. A implementação a partir do Drools permite o desenvolvimento de sistemas baseados em regras dando suporte a regras simples e complexas. Esta ferramenta provê um ambiente de implementação de regras de negócio dinâmico, possibilitando a manutenção de regras que mudam com frequência nos mais diferentes cenários de uma organização. As regras de negócio definem o comportamento da organização, ou seja, definem as regras e às medidas que devem ser tomadas pela organização a cada ocorrência de um determinado evento. Sendo assim, esta ferramenta possui um motor de regras capaz realizar a implementação de regras de negócio a partir de sua API. Os arquivos .drl são utilizados para criar instâncias de motores de regras capazes de inserir fatos na *work memory* e de disparar regras caso um *pattern match* ocorra na *Execution Node*. A análise se dá início por meio da LHS (*Left Hand Side*) da regra e, caso ocorra um casamento com os fatos da work memory a RHS (*Right Hand Side*) da regra é executada. A *API* do Drools também possibilita fácil integração com o jBPM, o que permite o acesso a API da *Process Engine*, tornando transparente a utilização de ambas as API's para ambas as ferramentas, ou seja, por meio a integração do Drools e jBPM é possível tirar proveito de regras e processos de negócio em ambos os níveis de implementação. O jBPM, por meio de sua API na Process Engine, proverá os métodos necessários a serem acessados no que tange a processos de negócio,

possibilitando acessar todos os estados do processo, bem como determinar as ações a serem tomadas. As soluções de negócios que utilizam ferramentas como o Drools permitem que as organizações obtenham vantagem competitiva entre às organizações, facilitando a automação de processos que antes seriam executados de formas manuais e agora são feitos de forma automatizada, tornando os processos a serem executados mais seguros e auditáveis.

As implementações de baixo nível, tanto no Drools quanto no jBPM podem ser feitas em Java. O jBPM é tida como uma *engine workflow* escrita em java para a execução de processos escritos em BPEL ou seus próprios processos definidos em jPDL. Basicamente, o jBPM tem como entrada um gráfico que descreve um processo de negócio, descrito por atividades, tarefas, representações de início e fim do processo, conectores, entre outros. Este conjunto de componentes descreve um workflow ou fluxo de execução do processo. O diagrama gráfico que define um processo é utilizado como base para a comunicação entre os desenvolvedores, especialistas e usuários não-técnicos da organização. Para que cada processo de negócio seja executado, é necessário que o mesmo seja instanciado, sendo função do jBPM gerenciar essas instâncias. Algumas atividades são consideradas automáticas, como o envio de um e-mail, chamadas a classes internas definidas pelos desenvolvedores, entre algumas outras atividades já definidas internamente pelo próprio jBPM. Outras atividades por sua vez, podem apresentar estado de espera, como: a necessidade da aprovação por um membro da direção da organização, ou a espera para que um determinado cliente invoque um serviço. Tendo em vista isto, é papel do jBPM gerenciar estas esperas, as atividades que estão sendo executadas e as persistências de dados pertinentes a processos de negócio.

O jBPM é baseado na *Process Virtual Machine* (PVM) ou Máquina Virtual de Processos, para suportar múltiplas linguagens de processos de forma nativa. A ideia central na integração entre o Drools e o jBPM é descrever atividades a serem executadas de uma forma mais humana e em mais alto nível, adotando para isto uma linguagem de processos de negócio como o BPMN 2.0 para descrever estas atividades, possibilitando a compreensão pelos mais diversos níveis de gestão da empresa e pessoal envolvido, abstraindo assim, conceitos de implementação de baixo nível e facilitando o entendimento comum. O jBPM tem como função dar suporte à implementação de modelos de processos de negócio, definição de serviços, integração entre serviços e integração com o Drools, permitindo que regras sejam disparadas até mesmo dentro de tarefas definidas em uma instância de processos de negócio. Como proposto na arquitetura conceitual, o jBPM será capaz de realizar os parsers dos modelos de processos de negócios trazidos do repositório de processos e definidos nos arquivos de regras por meio do componente de módulo semântico. O módulo semântico, por sua vez, realizará a separação de todos os componentes descritos no diagrama gráfico que representa o processo, discriminando para a *Process Engine* todos os elementos ali inseridos, e dando sentido a cada um deles. Com todos os elementos

ali discriminados, torna-se função da *Process Engine* realizar a execução dos processos. A *Process Engine* será capaz realizar instâncias de processos, persistências de dados referentes aos processos, bem como o gerenciamento dos processos de negócio. Durante o seu gerenciamento será capaz de realizar atividades automáticas e atividades de espera, por meio de componentes como: *Human Task Service* e *End User Task list Oriented UIs*. As persistências e transações, auditoria e histórico de logs serão armazenados no repositório de processos. A API do jBPM será capaz de interagir com o componente de gerenciamento de serviços externos e sistemas externos. No texto acima foram descritos os ambientes do Drools para a definição de regras de negócio e o jBPM para a definição de processos de negócio por meio de uma *engine workflow*; entretanto, o Drools por si só não é capaz de realizar análise de situações ao longo do tempo, sendo assim, nesta arquitetura de implementação o Scene (PEREIRA; COSTA; ALMEIDA, 2013) foi adotado para suprir esta necessidade, no qual, o Scene é uma extensão do Drools que permite analisar, disparar, gerenciar situações. A adoção do Scene permite que todas as funcionalidades do Drools continuem transparentes para os desenvolvedores.

O Scene é uma plataforma escrita em cima do código fonte do Drools que inclui um arcabouço de funcionalidades para o gerenciamento de situações que suporta o desenvolvimento de aplicações *situation-aware* oferecendo design de artefatos para a especificação de *situation types* e suporte ao gerenciamento do ciclo de vida de situações. O gerenciamento do ciclo de vida de situações inclui detecção de situações, que pode envolver reconhecimento de padrões de situação compostas e desativação de situações. É baseado no Drools Engine e integra a plataforma de *Complex Event Processing* e aprimora sua funcionalidade para suportar nativamente a percepção da situação baseada em regras. Esta plataforma permite a especificação de situação baseada em regras (e ainda mais o gerenciamento do ciclo de vida da situação) por meio de um padrão simples de regras. Os situation types são especificados no Scene dando significado a aspectos estruturais e comportamentais, no qual são realizados pelas *Situation Classes* e *Situation Rules*, respectivamente. *A Situation Class* especializa a classe *SituationType* predefinida, que é uma classe abstrata que aborda as propriedades temporais da situação e as características de composição. Uma *Situation Class* definida pelo usuário deve definir estruturalmente as funções particulares desempenhadas por entidades de domínio Situation Type (COSTA et al., 2016; PEREIRA; COSTA; ALMEIDA, 2013). Por sua implementação ter sido baseada em Drools, as funcionalidades do Drools estão disponíveis no Scene. O Scene necessita do Drools instalado no projeto para que funcione de maneira adequada, possibilitando uma forte integração com o jBPM.

Por meio do Drools IDE é possível acessar métodos da API da Process Engine, realizar a detecção de situações e a instanciação de execução de processos de negócio. Com isso, tais tecnologias preenchem o requisito R3 nos aspectos de execução de situações, regras e compatibilidade com modelos de processos de negócio em runtime, possibilitando

que um nó de execução seja definido. Este nó deverá possibilitar que regras e situações executem modelos de processos de negócio como gatilhos definidos a partir da ocorrência de situações e regras descritas em *design time*. Por meio das *API's* do Drools e jBPM é possível executar soluções de serviços internos e externos definidos em *design time*. A abordagem de solução satisfaz questões de escalabilidade em ambientes organizacionais e possuem mecanismos para a interoperabilidade com serviços e sistemas externos.

Em uma abordagem que se utiliza processos de negócio sendo executado de forma automatizada e recorrente dentro de uma organização, torna-se uma preocupação o local de armazenamento destes processos, informações e os serviços por eles utilizados. Em grandes organizações passa a existir um outro problema, que é a preocupação da colaboração de processos entre diferentes organizações, sendo assim, a colaboração entre organizações exige um gerenciamento adequado destes modelos de processos de negócio. Diante deste cenário surgem diversas preocupações, algumas delas são: adequar e gerenciar os processos das organizações, armazenar os processos de forma adequada, evitar o redesenho desnecessário de modelos de processos de negócio, evitar a implementação desnecessária de serviços utilizados pela organização, entre outros. A abordagem de colaboração entre diversas entidades da organização ou diversas organizações passa a exigir um gerenciamento ainda mais adequado destes recursos entre organizações. Uma solução possível para este problema é um mapeamento feito entre essas organizações dos processos e serviços com o objetivo de tornar estes recursos um só para todas as organizações de maneira colaborativa e sem se preocupar em onde tal recurso está alocado de forma física. Esta abordagem permite o enriquecimento da cultura e o aumentando da maturidade acerca dos processos da organização, entre outros aspectos altamente construtivos para a organização como um todo. Entretanto, gerenciar modelos de processos de negócio torna-se uma tarefa complexa quando as organizações integram redes colaborativas e estabelecem várias colaborações entre organizações.

Esta arquitetura conceitual fornecerá funcionalidades necessárias para se gerenciar modelos de processos de negócio envolvidos em colaboração entre organizações com o objetivo de satisfazer o R4. Como descrito na seção anterior, é proposto uma abordagem orientada a serviços de um repositório distribuído de processos, permitindo que as organizações tenham acesso a repositórios locais e globais. Os repositórios locais possuem uma característica mais privada e os repositórios globais características mais públicas, entretanto, os repositórios locais também podem manter modelos de processos de negócio, que são sincronizados e consistentes com os modelos de processos de negócio colaborativos, preservando aspectos privados. Em execuções de modelos de processos de negócio globais, nada impede que processos locais sejam executados como caixas pretas por modelos de processos de negócios globais. Usando um método de verificação e um método de arquitetura orientada a modelos, o repositório de processos distribuído fornece serviços que suportam a sincronização, consistência e requisitos de interoperabilidade entre processos

locais e processos globais (LAZARTE et al., 2013).

Nesta arquitetura de implementação o Firebase foi adotado como base de dados distribuída, devido a sua capacidade de oferecer um ambiente capaz de implementar funcionalidades de serviços, autenticação e armazenamento de arquivos de forma distribuída. Com esta plataforma é possível implementar soluções organizacionais, possibilitando uma integração com diversas organizações independente da tecnologia utilizada por trás. Esta ferramenta possibilita a criação de repositórios locais, globais ou até mesmo soluções isoladas locais ou globais, esta análise de solução fica a cargo da organização definir.

Outra possível solução seria uma implementação de base de dados feita pela própria organização com base na sua infraestrutura. O Firebase utiliza uma base de dados NoSQL, altamente escalável e com alto poder de armazenamento e processamento dos dados nele armazenados. As soluções com base de dados relacionais como Oracle e IBM DB2 também são viáveis, e podem ser implementadas sem o menor problema, desde que a solução desenvolvida pela organização se atente aos requisitos expostos da arquitetura conceitual. A solução proposta nesta arquitetura de implementação utilizou as funcionalidades do *Firebase Function*, *Firebase Database* e *Firebase Storage*. No Firebase Function foram definidos os serviços disponíveis pela organização, o Firebase Database armazena dados relevantes para os processos e o Firebase Storage armazena os arquivos referentes aos modelos de processos, uma vez que estes são armazenados no formato XML. Todas as 3 (três) abordagens necessitam das suas devidas autenticações para que se obtenha o devido acesso autorizado. Sendo assim, o Firebase atende o requisito R4 proposto, provendo um repositório de processos adequado tanto para modelos de processos de negócio, seus dados e serviços, possibilitando o gerenciamento destes recursos por toda a organização e suas entidades envolvidas. As questões de autorização também são satisfeitas e serão mais bem descritas no requisito R6. Por fim, devido a capacidade do Firebase prover uma *Rest API* para o acesso de seus serviços, o mesmo possibilita que os serviços sejam consumidos por aplicações sensíveis a situação e a contexto, por meio de chaves de autorização devidamente cadastradas.

Como visto, um aspecto fundamental tratado nesta arquitetura conceitual envolve conceitos pertinentes a Governança de SOA. Assim, tomando como preocupação a necessidade do apoio à SOA, a arquitetura conceitual propõe um componente de gerenciamento de serviços externos e internos. Nesta arquitetura de implementação o componente External Services Manager não será contemplado, apenas o Repository Manager, que possui como um de seus papéis o gerenciamento do armazenamento e a disponibilidade de modelos de processos e serviços internos da organização e suas entidades e os Locais Services e Globais Services para gerenciar os serviços internos. Por meio da Web Interface as pessoas envolvidas diretamente no projeto deve ser capaz de analisar a disponibilidade e a viabilidade de utilização de determinados serviços. Como adoção de bons padrões para a

organização é ideal que uma organização madura tenha conhecimento sobre os seus mais diversos serviços oferecidos e seja capaz de realizar um levantamento de serviços necessários envolvidos no projeto, sabendo quais serviços devem ser reutilizados ou implementados. Para organizações que possuem diversas entidades, serviços locais de interesse de outra entidade podem ser disponibilizados globalmente ou de forma a serem utilizados como caixa preta por outras entidades, possibilitando que os mesmos sejam utilizados sem a preocupação em como funcionam.

Com o objetivo de satisfazer o cumprimento do requisito R5 nesta arquitetura de implementação e tendo em vista a adoção da tecnologia Firebase para realizar o mapeamento de serviços possibilitando a listagem de todos os serviços criados pela organização por meio do *Firebase Functions* foi utilizada a sua interface para o mapeamento dos serviços. Esta abordagem deve ser levada em consideração apenas para pequenas organizações e com poucos serviços para se gerenciar. As organizações de médio e grande porte com uma gama gigantesca de serviços a serem gerenciados devem adotas alternativas para o gerenciamento destes serviços, como implementar sua própria infraestrutura para prover estes serviços. Com isso, o Firebase cumpre o requisito R5 proposto na arquitetura conceitual nesta arquitetura de implementação, possibilitando que as partes envolvidas do projeto tomem conhecimento dos serviços disponíveis na organização de forma a tomar decisões importantes no projeto no que tange a reutilização de serviços internos. O componente de serviços externos funciona similarmente da mesma forma, porém necessita de uma implementação baseada em serviços para a listagem dos mesmos e, uma maior maturidade da organização para avaliar a disponibilidade constante destes serviços, a possibilidade de serviços alternativos para o caso de falha também é importante, uma vez que caso um serviço externo pare de funcionar isso não disponibilize a aplicação como um todo.

O requisito R6 não será abordado nesta arquitetura de implementação, tendo em vista que os aspectos adotados nesta arquitetura de implementação como a autenticação utilizando o Firebase Functions para a autenticação de serviços via token satisfaz as necessidades parciais deste requisito, provendo uma abordagem de autenticação segura para os níveis da organização. As autenticações que utilizam padrões RBAC deveriam ser adotadas nesta etapa, bem como todos os padrões de segurança acerca desta abordagem de implementação. No nível de implementação acadêmica a solução disponível no Firebase satisfaz parcialmente este requisito e uma abordagem mais detalhada ficará para trabalhos futuros.

4.5 Considerações Finais do Capítulo

Este capítulo apresentou uma proposta de infraestrutura para integração entre estrutura de apoio à construção de aplicações que utilizam sensibilidade a situações e BPMN, bem como o armazenamento de processos de negócio para o gerenciamento do conhecimento de forma distribuída. Neste ambiente foram destacadas suas respectivas separações de papéis, permitindo uma participação ativa do analista de processos de negócio e dos desenvolvedores de forma independente no desempenho de suas funções.

A implementação da arquitetura consome dados coletados de domínios a fim de possibilitar a análise de situações integradas com modelos de processos de negócio em diferentes aplicações. A plataforma Drools de sistema de regras, Scene para a análise de situações, jBPM para processamento de processos de negócio e o Firebase como base de dados distribuída foram adotadas como base para a implementação do protótipo da infraestrutura proposta.

O Capítulo 5 apresenta um exemplo de aplicação da estrutura proposta, em um cenário na área de estabilidade de fármacos.

Parte III

Parte Final

5 Prova de Conceito

Este capítulo apresenta uma prova de conceito da arquitetura, materializada por meio de um experimento empírico. Inicialmente, na Seção 5.1, é apresentado o cenário de interesse escolhido, dentro do domínio hospitalar, com foco na conservação de medicamentos. Este cenário é tomado como base no decorrer do experimento. A Seção 5.2 apresenta a visão de alto nível da aplicação do ponto de vista do especialista de processos de negócio, demonstrando a modelagem de processos. A Seção 5.3 trata da visão do desenvolvedor no que tange à modelagem de regras e situações contextuais em alto nível de abstração. A Seção 5.4 apresenta a descrição do experimento empírico usado para a validação da arquitetura proposta, incluindo o questionário preenchido pelos colaboradores da pesquisa. A Seção 5.5 mostra os resultados do experimento com base nas respostas e procedimentos descritos na seção anterior. Por fim, a Seção 5.6 apresenta as considerações finais do capítulo.

5.1 Cenário

O cenário de implementação se refere ao domínio de conservação de medicamentos. Dentro do domínio de conservação de medicamentos existem estudos de estabilidades que determinam critérios e o tempo de validade de medicamentos. Os fatores de estabilidade de medicamentos se relacionam com dependências ambientais, tais como: temperatura, umidade e luz, fatores de propriedades químicas e físicas relacionadas ao próprio produto, entre outras. As clínicas e hospitais são cenários típicos de questões relacionadas à estabilidade de medicamentos, sendo necessário locais específicos e apropriados para o armazenamento, tendo em vista que a manipulação e o armazenamento incorreto podem acarretar na perda do produto, causando danos materiais às instituições ou até mesmo a perda de vidas humanas por utilização de materiais vencidos ou sem eficácia.

Os profissionais que atuam no controle de medicamentos nestes ambientes, possuem entre suas atribuições verificar durante determinados intervalos de tempo as variáveis do ambiente onde e como estão armazenados os medicamentos. Alguns medicamentos são mais sensíveis do que outros, sendo assim, medicamentos mais sensíveis devem possuir um rigor maior de controle por possuírem maior sensibilidade a variações de temperatura. A perda de medicamentos dentro desses ambientes pode ser monetariamente muito alta para as instituições, que possuem geralmente um monitoramento de medição e controle das variáveis realizadas de maneira manual, possibilitando que erros humanos sejam cometidos.

Com base nisso, foi desenvolvida uma aplicação para o controle de temperatura de medicamentos tendo em vista o cenário promissor de uso de tecnologias para a auxiliar

no monitoramento de medicamentos. Este cenário possibilita que dados sejam coletados de forma automatizada e sem a necessidade de interação humana. A utilização de análise de situações e modelos de processos de negócios possibilitam um forte apoio para uma possível solução deste problema, uma vez que a análise de situações ocorre ao longo do tempo e os modelos de processos de negócio representam ações a serem tomadas em alto nível, possibilitando que pessoas que não são da área de tecnologia modelem e validem rotinas a serem executadas em alto nível, abstraindo questões de implementação de baixo nível. Tal abordagem possibilita que ações possam ser tomadas caso ocorra elevações de temperatura não esperadas e permite que a instituição gerencie de forma mais eficaz os funcionários que deveriam tomar conta destes medicamentos.

O cenário é constituído por uma clínica que necessita de uma aplicação para controlar medicamentos ou qualquer tipo de produto que precisa ser mantido em freezer ou geladeira sob determinado intervalo de temperatura, pois fora deste intervalo específico o produto pode se deteriorar e não terá mais condições de ser utilizado. O prejuízo pode ser elevado, pois pode ser uma aplicação que controla produtos de alto valor como, por exemplo, córneas ou outros órgãos para transplante ou sêmen de cavalos de raça utilizado por criadores que querem controlar a reprodução de animais. Nesse caso, o produto deve ser mantido a baixas temperaturas. No caso proposto para esse trabalho tratará especificamente do controle de medicamentos de uso médico. A aplicação proposta fará o controle e monitoramento da temperatura de frascos de Botox que são normalmente utilizados por médicos dermatologistas.

O médico especialista em dermatologia reconhecido pela sociedade brasileira de dermatologista é o profissional qualificado para a aplicação do Botox. Normalmente o médico que faz esse tipo de procedimento não compra apenas um frasco. O mais comum é fazer compras mensais ou programadas com aquisição de uma quantidade maior com aproximadamente 20 frascos de 100 unidades cada. Dessa forma, o médico precisa controlar e monitorar a temperatura dos frascos de Botox que normalmente são recebidos por serviços rápidos de entrega de encomendas.

Os produtos são recebidos em um isopor de tamanho pequeno com barras de gelo em gel que podem ficar mais de um dia de transporte sem que o produto sofra algum tipo de dano devido a ser submetido a temperaturas maiores que as permitidas. Atualmente o detentor da marca Botox é a Pfizer, muito conhecida por ser a fabricante do Viagra. Ela comprou a Allergan que era a fabricante do Botox. O Botox é recebido em um frasco ampola de vidro, contendo o produto onabotulinumtoxiA congelada a vácuo estéril, acondicionada individualmente em frascos de 100 unidades cada.

Ao serem recebidas, as caixas com o Botox devem ser mantidas no freezer em uma temperatura menor ou igual a -5º C. Após a diluição com soro para o uso conforme bula e protocolo adotado pelo médico, o produto deve ser mantido somente em geladeira

sob um intervalo de temperatura entre 2º C e 8º C por um período de no máximo de 3 dias. Neste estudo o foco será apenas para o produto com o frasco a vácuo que ainda não foi utilizado ou diluído. Dessa forma, a temperatura deve ser igual ou inferior a -5º C. Normalmente o produto é recebido no consultório do médico dermatologista. Assim que o isopor com os frascos é recebido, a secretária ou o próprio médico deve verificar as condições do produto e, se for possível, fazer a medição da temperatura do interior do isopor com algum medidor manual de temperatura para saber se a temperatura está igual ou inferior a -5ºC. Após a verificação o produto deve ser armazenado no freezer do consultório que deve ser controlado pelo sistema assim que o produto for colocado lá. É muito comum que o médico não deixe todos os frascos que acabaram de chegar no freezer do consultório. O ideal é deixar apenas o que normalmente se usa durante uma semana. Conforme o uso, a reposição é feita por demanda. O restante dos frascos é levado para o freezer de casa que também deve ter um sistema de monitoramento instalado.

Aspectos como defeitos em sensores dentro do freezer ou geladeira também serão tratados neste cenário, tendo em vista que anomalias e discrepâncias em dados provenientes de sensores podem ser detectados neste cenário. A aplicação deverá analisar o conjunto de dados provenientes dos sensores do freezer e analisar o conjunto, caso a discrepância ocorra em apenas um sensor específico o problema deve ser acusado pela aplicação para uma análise e possível substituição do sensor. Na ocorrência de elevação de temperatura ou defeito, o médico e a secretária deverão ser notificados do problema ocorrido. A ação a ser tomada caso este tipo de situação ocorra deverá ser expressa em modelos de processos de negócio, e o domínio definido pelos desenvolvedores.

As próximas seções discutem as diferenças de visões entre especialistas de processos de negócio e desenvolvedores. O cenário apresentado é utilizado em um experimento empírico que toma como base a infraestrutura proposta neste trabalho.

5.2 Visão do Especialista de Processos de Negócio

Com o ambiente simulado criado e tomando como base que eventos como os descritos no ambiente simulados podem ocorrer na vida real, os especialistas de processos de negócio se reunirão com as partes interessadas do projeto dentro da organização para modelarem o problema. Os especialistas modelarão em alto nível ações que deverão ser executadas diante de da ocorrência de um determinado evento. Com base no cenário descrito anteriormente, toma-se como necessário notificar as partes envolvidas no processo de armazenamento de fármacos, sendo eles médicos e secretários. No cenário apresentado existe apenas um médico e um(a) secretário(a); sendo assim, ambos deverão ser notificados caso ocorra a elevação de temperatura. No modelo de processos de negócio, o(a) médico(a) somente será notificado da falha geral no freezer ou caso o freezer seja esquecido aberto

por um longo período. O (A) secretário(a) será notificado(a) da elevação de temperatura, sempre que ela ocorrer e durar mais do que um intervalo de tempo de 5 minutos.

Os modelos de processos de negócio permitem que sejam realizados vários tipos de implementação para a notificação de pessoal; entretanto, neste estudo, serão implementados 3 (três) tipos de notificação diferentes para exemplificar os tipos de notificação, sendo elas: (i) notificação utilizando a atividade de envio de e-mail; a (ii) notificação utilizando uma task de serviço; e por fim, a (iii) notificação utilizando uma solução interna implementada pelo próprio desenvolvedor.

Baseando-se na descrição da seção anterior, caso não haja alterações significativas na elevação de temperatura, não deverá ocorrer notificação; logo, não precisará ser descrito como um modelo de processo de negócios, pois nunca será disparado uma situação nesta ocasião. Quando houver uma pequena discrepância ou divergência na ocorrência nos valores de algum dos sensores ambas as partes devem ser notificadas. Na ocorrência de uma falha geral ambas as partes serão notificadas, como pode ser visto na Figura 22 e, caso o freezer seja esquecido aberto o(a) secretário(a) será notificado(a) se a ocorrência durar mais de 5 minutos; entretanto, se o freezer ficar mais de 30 minutos aberto o(a) médico(a) também deverá ser notificado.

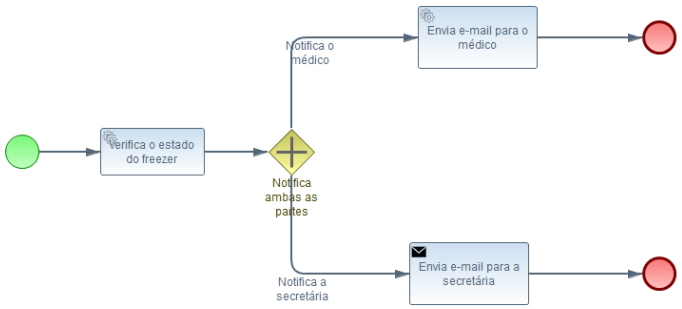

Figura 22 – Realiza a notificação de ambos os interessados caso haja algum defeito nos sensores ou o freezer pare de funcionar

As notificações deverão ocorrer via e-mail e novos eventos de notificação poderão ocorrer. Em outros cenários, caso ocorresse uma falha ou uma possível falha no freezer seria possível que a equipe de manutenção fosse avisada em tempo real sobre possíveis defeitos no equipamento, possibilitando que uma equipe fosse enviada até o local para verificar o possível problema antes mesmo dele ocorrer de fato.

Os arquivos salvos como modelos de processos de negócio foram armazenados com o nome de seus respectivos identificadores e podem ser enviados via post para o

serviço juntamente com o seu e-mail e senha de autenticação, que verificará se o usuário é cadastrado.

Esta abordagem de desenvolvimento unindo regras e processos de negócio simplificam e diminuem a complexidade dos processos, uma vez que ao modelar uma aplicação similar e mais simples do que a descrita acima, criaria um grande modelo de processo de negócio, dificultando até mesmo sua manutenção.

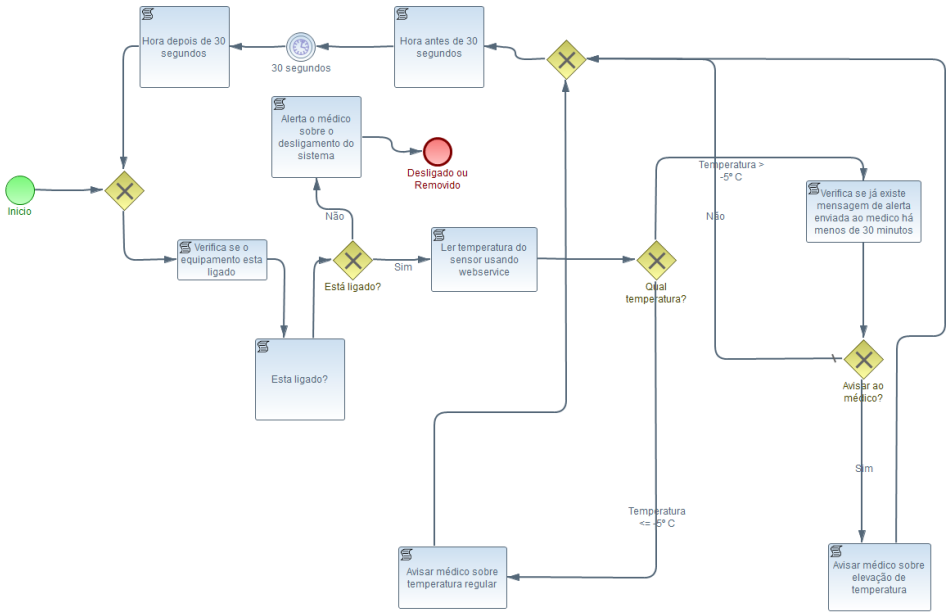

Figura 23 – Modelo de processos de negócio automatizado puramente em BPMN sem a utilização de regras

O modelo de processos de negócio utilizando a notação BPMN 2.0 descrito na Figura 23 descreve um ambiente no qual o médico será notificado caso ocorra uma elevação de temperatura; entretanto, não é analisado qual a duração desta elevação de temperatura, apenas é realizada a notificação do médico e se ele foi avisando nos últimos 30 minutos. Este modelo mostra que aplicações desenvolvidas e automatizadas puramente em BPMN podem ser um tanto quanto complexas. A utilização de abordagens que unem regras e processos de negócio pode diminuir a complexidade dos modelos de processos de negócio e torna-los mais reutilizáveis. Como pôde ser visto no cenário descrito acima, que trata um caso mais complexo contendo processos de negócio descritos de forma mais simples.

5.3 Visão do Desenvolvedor

Tomado conhecimento do domínio, a equipe de desenvolvimento pode preparar o ambiente e implementar a solução para o domínio acima. Os desenvolvedores tomarão a decisão sobre as tecnologias adotadas para solucionar as questões de implementação.

Feito isto, deverão definir os seguintes aspectos principais: os *situation types* e os *situation rules*. Os *situation types* descreverão do que é composto o escopo de cada situação a ser analisada naquele domínio, ou seja, como será estruturado cada dado recebido de cada sensor.

Neste exemplo específico os dados provenientes dos sensores são temperaturas; entretanto, um mesmo nó sensor poderia enviar temperatura, umidade e luminosidade e, todos estes dados estariam presentes no escopo do dado do nó sensor para ser analisado. Os *situation rules* definem as regras que determinam se uma situação deverá ocorrer ou não. Caso um determinado conjunto de dados faça sentido para o sistema, será ativada uma situação.

Os *situation types* definem os dados que fazem sentido para o domínio e que serão casados e analisados pelas regras de negócio. A Figura 24 mostra a declaração do *Situation Type* da temperatura, que é associado a um sensor que contém uma lista de transactions. Essas transactions representam uma lista de temperaturas contidas naquele sensor ao longo do tempo. A cada instante em que um novo dado chega, esta lista precisa ser atualizada.

```
package co.pextra.botox.scenary;

import java.util.List;

import br.ufes.inf.lprm.scene.model.impl.Situation;
import br.ufes.inf.lprm.scene.util.SituationHelper;
import br.ufes.inf.lprm.situation.annotations.part;
import br.ufes.inf.lprm.situation.model.events.*;
import org.kie.api.runtime.process.ProcessInstance;

declare TemperatureSituation extends Situation
    sensor: Sensor @part
    transactions: List @part
end
```

Figura 24 – Definição do *Situation Type*

Na definição das regras, os desenvolvedores deverão consultar os serviços que serão consumidos pela organização. Os serviços estão descritos na *Web Interface* e, como boa prática, até mesmo na documentação. Os desenvolvedores referenciaram o modelo de processo de negócio, utilizando um método adequado, junto com o seu identificador (ID). Após esta referência, será possível que a *Rule Engine* faça uma requisição ao repositório de processos e pegue o processo requisitado. O processo sofrerá um parser pelo módulo semântico e enviará para a *Process Engine*, que executará o processo de forma síncrona ou assíncrona. Neste estudo a execução será assíncrona, pois a aplicação não precisará esperar a notificação das partes interessadas para que continue funcionando.

```
rule "More than 6 high temperatures in  thirty minutes for a sensor of temperature"
@role(situation)
@type(TemperatureSituation)
    when
       sensor : Sensor();
       transactions: List(size > 6) from accumulate(
          transaction: TemperatureEvent(sensorId == sensor.sensorId, temperature > -5) over window:time (30m),
          collectList(transaction)
       )
     then
       SituationHelper.situationDetected(drools);
end

rule "More than 3 high temperatures in five minutes for a sensor of temperature"
@role(situation)
@type(TemperatureSituation)
    when
       sensor : Sensor();
       transactions: List(size > 3) from accumulate(
          transaction: TemperatureEvent(sensorId == sensor.sensorId, temperature > -5) over window:time (5m),
          collectList(transaction)
       )
     then
       SituationHelper.situationDetected(drools);
end

rule "start botox situation when have a one case suspicious"
    when
       temperatureSituation: List(size > 2) from accumulate (
         Activation($sit: situation, $sit isA TemperatureSituation.class) over window:time (10m);
         collectList($sit)
       )
    then
        ProcessInstance processInstance = kcontext.getKieRuntime().startProcess(ws.getProcess('0100123'));
        System.out.println("uma situação de temperatura ocorreu");
end
```

Figura 25 – *Situation Rules* e a chamada do processo no Web Service por meio do identificador

```
rule "Ocorreram 3 alterações de temperatura com o mesmo sensor em 2 minutos"
@role(situation)
@type(TemperatureSituation)
     when
        sensor : Sensor();
        transactions: List(size > 3) from accumulate(
           transaction: TemperatureEvent(sensorId == sensor.sensorId)
                 over window:time (2m),
           collectList(transaction)
        )
      then
        SituationHelper.situationDetected(drools);
end
rule "Mais de 5 elevações de temperaturas em um intervalo de 2 minutos para um único sensor"
@role(situation)
@type(TemperatureSituation)
     when
        sensor : Sensor();
        transactions: List(size > 5) from accumulate(
           transaction: TemperatureEvent(sensorId == sensor.sensorId, temperature > -5)
                 over window:time (2m),
           collectList(transaction)
        )
      then
        SituationHelper.situationDetected(drools);
end
rule "Mais de 10 elevações de temperaturas em um intervalo de 2 minutos para um único sensor"
@role(situation)
@type(TemperatureSituation)
     when
        sensor : Sensor();
        transactions: List(size > 10) from accumulate(
           transaction: TemperatureEvent(sensorId == sensor.sensorId, temperature > 0)
                 over window:time (2m),
           collectList(transaction)
        )
      then
        SituationHelper.situationDetected(drools);
end
rule "inicia uma situação quando houver um caso suspeito"
     when
        temperatureSituation: List(size > 2) from accumulate (
          Activation($sit: situation, $sit isA TemperatureSituation.class)
                 over window:time (2m);
          collectList($sit)
        )
     then
         //Para 2 (duas) ativações de regras que ocorreram num intervalo de 2 minutos,
         //um processo é iniciado.
         ProcessInstance processInstance = kcontext.getKieRuntime()
                                        .startProcess("co.pextra.botox.notification");
end
```

Figura 26 – Definição das *Situation Rules*

As regras de negócio serão definidas pelos desenvolvedores satisfazendo as condições definidas por meio dos dados contidos nos *situation types*. A chamada do processo será

realizada quando uma ativação de situação ocorrida foi concretizada, como pode ser visto na Figura 25 acima.

A aplicação base que usará os *situation types* e os *situation rules* também será implementada pela equipe de desenvolvimento. Neste trabalho o ambiente dos dados simulados está inserido juntamente com esta aplicação base, e o anexo do código está disponível no Apêndice B.

A Figura 26 aborda outro método de utilização de processos em meio a regras e situações. Por meio do identificador do processo que está carregado no projeto local, o processo passa a ser identificável no Drools e pode ser arrancado a qualquer momento. Esta abordagem entra em contraste com a abordagem descrita na Figura 25, que realiza a chamada direta ao repositório de processos durante a execução da regra, enquanto a Figura 26 permite a referência a um processo que foi carregado previamente na aplicação. Esta situação pode ocorrer caso a abordagem de carregamento seja realizada de uma forma diferenciada na qual o desenvolvedor opta por carregar os processos em disco antes de inserir os dados de domínio na *Work Memory*. Ambas as abordagens geram resultados consistentes e variam de metodologia de acordo com as necessidades do desenvolvedor sendo, assim, mais uma possibilidade em seu arcabouço de conhecimento.

5.4 Experimento Empírico

Esta seção descreve como o experimento foi conduzido. Os tópicos levantados são apresentados ao longo da Seção 5.4, e incluem os seguintes pontos: Apresentação, Estruturação do Projeto, Procedimento do *BPM Expert*, Procedimento do Desenvolvedor, e Passos a serem seguidos no experimento. A Seção 5.4.2 apresenta as perguntas a serem respondidas ao término do experimento;

5.4.1 Apresentação

O seguinte texto de apresentação foi passado para os participantes do experimento:

Esta pesquisa tem como foco a validação de uma arquitetura conceitual através de um experimento empírico. Você provavelmente chegou até este formulário para colaborar com este experimento por meio de um link enviado por pessoas do seu meio, instituições, empresas, etc. Caso não tenha recebido o link por qualquer meio mencionado e queira participar do experimento, fique à vontade. Nas próximas seções serão descritos conceitos da arquitetura conceitual proposta, pontuado pontos relevantes acerca do experimento e ao término da explicação será conduzido um experimento empírico para a validação da arquitetura conceitual. O experimento consiste em propor o uso da arquitetura através de uma máquina virtual, e ao fim do experimento pedimos apenas que preencham o questionário que objetiva entender a experiência de uso da solução proposta.

O objetivo do experimento é verificar (avaliar) a visão do *BPM Expert* e a do Desenvolvedor de Sistemas na criação de uma possível solução para o problema apresentado no cenário descrito anteriormente, usando a infraestrutura proposta. Para este estudo foi criado um ambiente de simulação que fornece dados de variação de temperatura para que o Scene analise se houve a ocorrência de alguma situação de interesse. Aos colaboradores do experimento foi entregue um resumo da arquitetura, dividido em 2 (duas) seções, *introdução* e *arquitetura conceitual*, que detalham o trabalho proposto, listando as suas principais características e componentes, conforme descrito na Seção 4.2. Adicionalmente, um resumo baseado na seção 5.1 foi apresentado aos participantes, com o objetivo de elucidar a *contextualização do domínio de interesse*, cujo foco são as temperaturas limites do medicamento Botox, bem como algumas peculiaridades do domínio de interesse e das partes interessadas. Um link de acesso à pesquisa completa realizada com ambos os perfis de interesse é disponibilizado ao final da Seção 5.4.8.

5.4.2 Estrutura do Projeto

A estrutura do projeto é dividida em um pacote de classes Java, um pacote de testes e um pacote de recursos para as classes Java. O pacote de recursos dos testes não foi utilizado. Na estrutura do pacote de classes, é definido a classe que representa a entidade sensor e a estrutura que representa a entidade evento de temperatura.

```java
package co.pextra.botox;

public class Sensor {
    private int sensor_id;
    private String description;

    public Sensor(int sensor_id, String description){
        this.sensor_id = sensor_id;
        this.description = description;
    }

    public int getSensorId() {
        return sensor_id;
    }

    public void setSensorId(int sensor_id) {
        this.sensor_id = sensor_id;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }
}
```

Figura 27 – Entidade Sensor

A entidade sensor é basicamente definida como pode ser visto na Figura 27 por um identificador e uma descrição para rotular o sensor em específico. Os métodos implementados são: construtor do sensor, um método pegador e atribuidor de Id's, e um método pegador e atribuidor de rótulos.

A implementação da entidade sensor é utilizada para representar os sensores alocados no domínio físico. Estes sensores são instânciados na aplicação e podem ser rotulados de sensores virtuais, que por sua vez, passam a representar o comportamento do meio físico no meio virtual. Caso a natureza do sensor seja alterada ou seja necessário expandir a capacidade de captação de dados no sensor físico, esta classe deve ser alterada para aumentar a sua representatividade acerca da entidade sensor que está no meio físico.

A entidade de evento de temperatura é constituída por um executionTime que possui o tipo de data ("Timestamp"), este executionTime é utilizado pelo Scene para realizar o controle da situação ao longo do tempo, um atributo temperatura e o identificador do sensor. Esta classe possui todos os seus métodos pegadores e atribuidores para cada atributo.

```java
package co.pextra.botox;
import java.io.Serializable;

@Role(Role.Type.EVENT)
@Timestamp("executionTime")
@Expires("2m")
public class TemperatureEvent implements Serializable {
    private static final long serialVersionUID = 1L;
    private Date executionTime;
    private Double temperature;
    private int sensorId;

    public TemperatureEvent(Double temperature, int sensorId) {
        super();
        this.executionTime = new Date();
        this.temperature = temperature;
        this.sensorId = sensorId;
    }

    public Date getExecutionTime() {
        return executionTime;
    }

    public void setExecutionTime(Date executionTime) {
        this.executionTime = executionTime;
    }

    public Double getTemperature() {
        return temperature;
    }

    public void setTemperature(Double temp) {
        this.temperature = temp;
    }

    public int getSensorId() {
        return sensorId;
    }

    public void setSensorId(int sensorId) {
        this.sensorId = sensorId;
    }
}
```

Figura 28 – Representação do Evento de Temperatura

A Figura 28 destaca a classe que possui como responsabilidade representar uma entidade de evento dentro do Scene, com base nela o Scene é capaz de instanciar seus eventos internos. O @Role, torna a classe como sendo do tipo Evento, o @Timestamp cria a representatividade de tempo para cada instancia de evento que estiver executando e @expires cria a janela de tempo que o evento ficará vivo. Este tempo pode ser definido como 2(dois) minutos para simulação e utilizar o pseudoclock do Drools para representar 2(dois) minutos como sendo por exemplo 30 (trinta) minutos. No pacote de testes, é criado uma classe que representa uma thread e nesta thread é manipulado o contexto do Drools por meio da ksession e o método run como arranque das regras definidas na ksession. Por fim, no pacote de recursos para as classes Java, é definido um kmodule.xml que é utilizado para a configuração das ksessions, os arquivos de regras e situações e o(s) processo(s) de negócio(s). A Figura 29 representa a estrutura do projeto descrito.

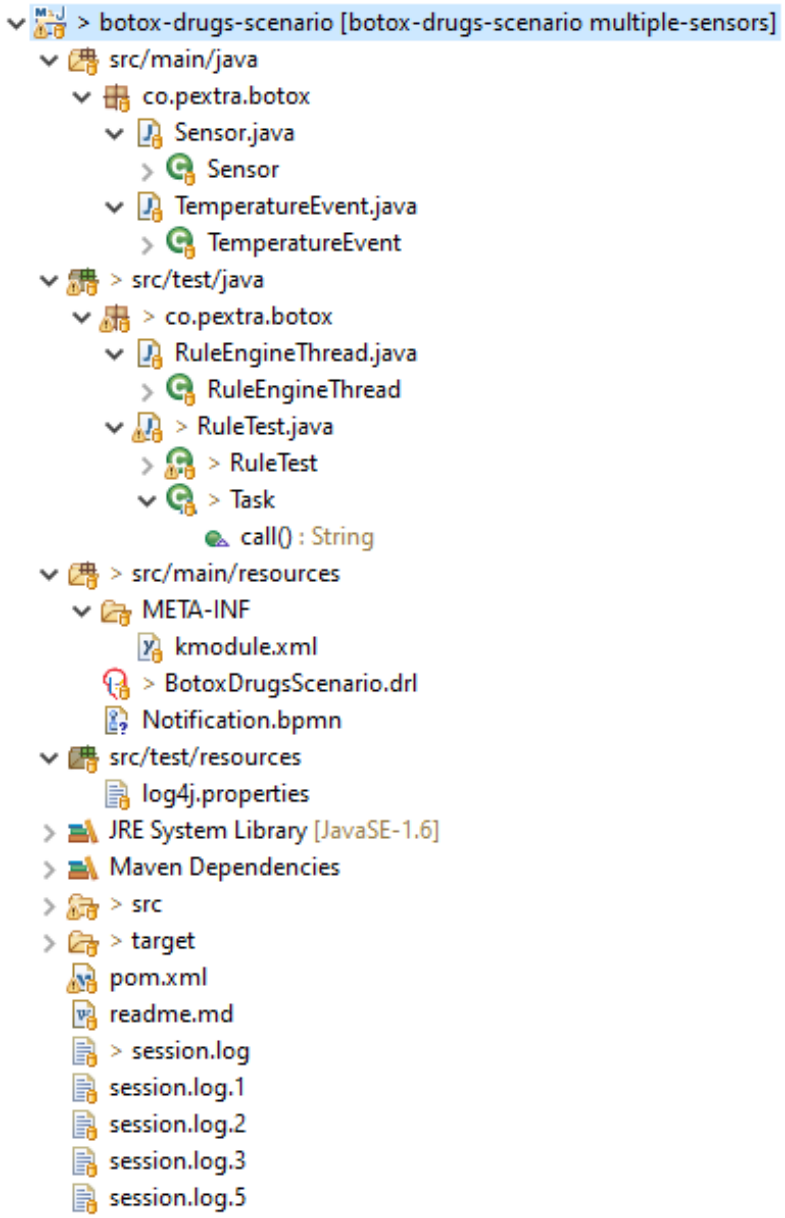

Figura 29 – Estrutura do Projeto

A seguir, uma descrição acerca dos arquivos de regras e situações será realizada com base em cada perfil, sendo estes *BPM Experts* e *Developers*. Os arquivos arquivo de RuleTest.java é o arquivo principal para rodar a aplicação. Para dar início na aplicação é necessário ir em RuleTest.java, clicar com o botão direito > ir em Run As > JUnit Test, como pode ser visto na Figura 30, em seguida o console de saída aparecerá na tela por meio da barra lateral localizada no canto direito. As mensanges exibidas no console podem ser definidas tanto na aplicação base, nas regras e situações e nos próprios processos de negócio. As mensagens exibidas no console servem para dar feedbacks visuais do que está acontecendo naquele momento.

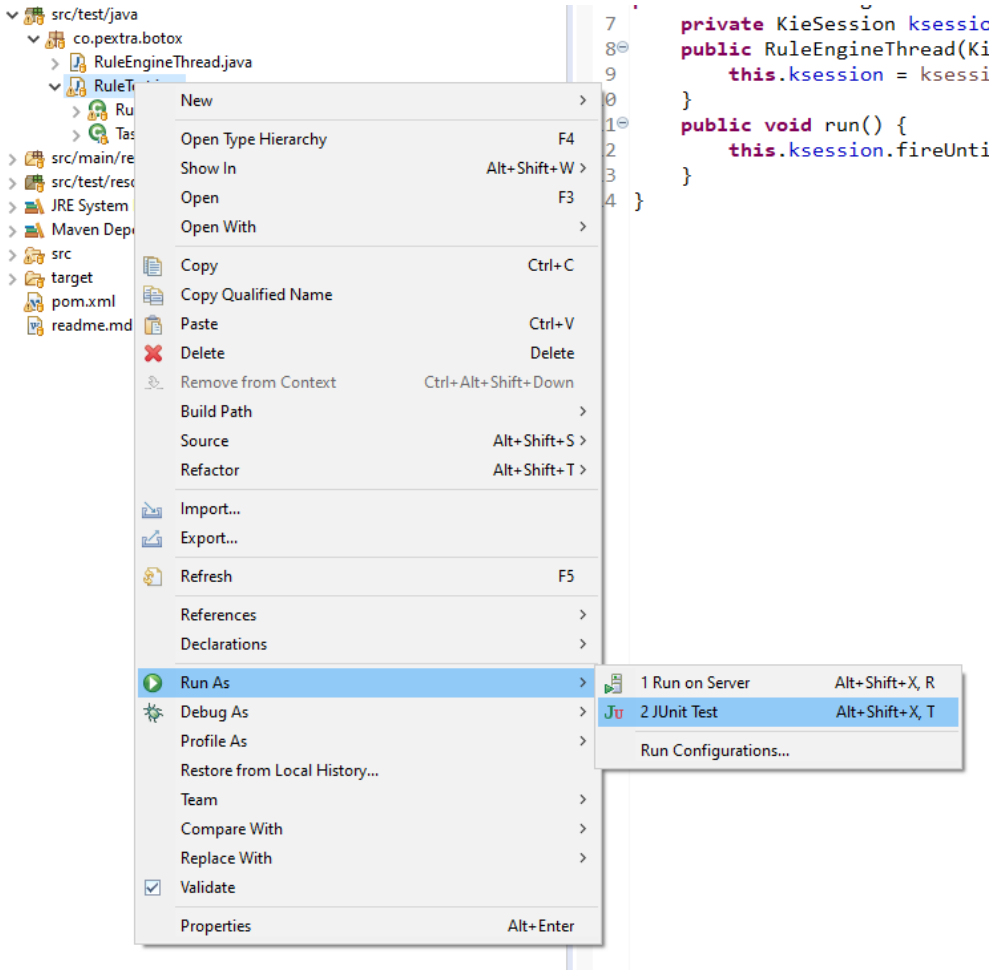

Figura 30 – Iniciando a aplicação do projeto

Devido a extensão deste arquivo, será citado apenas dois fragmentos de código que representa a simulação das geladeiras e a geração dos dados sensoriados, como parâmetro inicial foi passado o identificador de cada sensor, neste caso de 1 (um) a 5 (cinco). O rótulo dado foi o identificador da geladeira no qual o sensor está atribuído, ou seja, geladeira 1 (um), conforme pode ser visto na Figura 31.

```
LOG.info("Now running data");

// Criar loops para simular dados em 5 sensores diferentes! São necessários 5
// sensores.
Sensor sensor1 = new Sensor(1, "Geladeira 1");
session.insert(sensor1);
Sensor sensor2 = new Sensor(2, "Geladeira 1");
session.insert(sensor2);
Sensor sensor3 = new Sensor(3, "Geladeira 1");
session.insert(sensor3);
Sensor sensor4 = new Sensor(4, "Geladeira 1");
session.insert(sensor4);
Sensor sensor5 = new Sensor(5, "Geladeira 1");
session.insert(sensor5);

// Cada sensor possui uma lista de temperaturas
insertTemperatures(sensor1, -6, -8, 7, 5, session);
insertTemperatures(sensor2, -8, -6, 5, 3, session);
insertTemperatures(sensor3, -7, -9, 7, 5, session);
insertTemperatures(sensor4, -10, -6, 4, 2, session);
insertTemperatures(sensor5, -15, -10, 8, 6, session);

LOG.info("Final checks");
```

Figura 31 – Instanciação dos Sensores

Após a instância dos sensores o método para a geração de temperaturas chamado insertTemperatures foi chamado, e será descrito a seguir na Figura 32.

```
public static void insertTemperatures(Sensor s, int temp_start, int temp_end, int temp_mid, int temp_invert,
        KieSession session) {

    for (int temp = temp_start, invert = 0; (temp <= temp_mid && invert == 0)
            || (temp >= temp_end && invert == 1);) {
        if (temp > temp_invert) {
            invert = 1;
        }

        if (temp <= temp_invert && invert == 0)
            temp++;
        else
            temp--;
        session.insert(new TemperatureEvent((double) temp, s.getSensorId()));
    }

}
```

Figura 32 – Geração de dados de temperatura

Este método obtêm as temperaturas iniciais e finais do intervalo de temperaturas a serem geradas, a temperatura de pico e o ponto de inversão ou inflexão da temperatura. Nas seções seguintes será descrito procedimentos pertinentes a cada área de conhecimento a serem de entendimento para o experimento.

5.4.3 Procedimento do *BPM Expert*

Esta seção é focada na área de conhecimento dos especialistas de processos. A título de contextualização, considere um processo de negócio que executa um cenário similar ao descrito anteriormente, apresentado na Figura 23. Este processo inicialmente analisa se o freezer está ligado e, após isso, realiza uma leitura do dado de temperatura no serviço que se comunica com os sensores de temperatura. Caso a temperatura esteja acima de -5 graus

Celsius, as partes interessadas são notificadas sobre esta elevação. Caso a temperatura persista acima da prevista, ocorre uma nova notificação após 30 minutos.

Na Figura 22, é descrito um modelo de processos de negócio para notificar aos responsáveis pelo produto a ocorrência de algum problema. O mecanismo de notificação é disparado mediante a detecção de uma particular situação de interesse previamente especificada. O estado do freezer define se há sensores defeituosos ou se o mesmo parou de funcionar. Para que o e-mail seja enviado, é necessário modelar o processo e associar uma classe de envio de e-mails a tarefa adicionada, esta tarefa pode ser do tipo serviços ou envio de e-mail.

Na abordagem proposta, tanto serviços internos da implementação da *engine* de processos quanto serviços externos de e-mail podem ser utilizados para realizar a notificação. De forma comparativa, pode-se perceber que há uma diminuição considerável no tamanho do processo apresentado na Figura 22 quando comparado ao processo da Figura 23. Como estrutura inicial de processos, será dado apenas um processo simples, com nó de início do processo, uma tarefa de serviços com um "hello world" que será apresentado no console, e um nó de fim de processo. Um exemplo deste nó pode ser visto na Figura 33 abaixo, juntamente com o método para a impressão de uma mensagem no console escrito em Java.

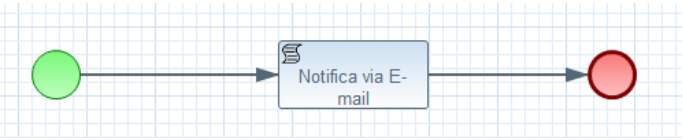

Figura 33 – Processo default do projeto

Na Figura 34 é possível editar características importantes do processo, como o Id do processo que será utilizado como identificação para se dar início no processo pelos desenvolvedores. O identificador default é co.petrax.botox.notification.

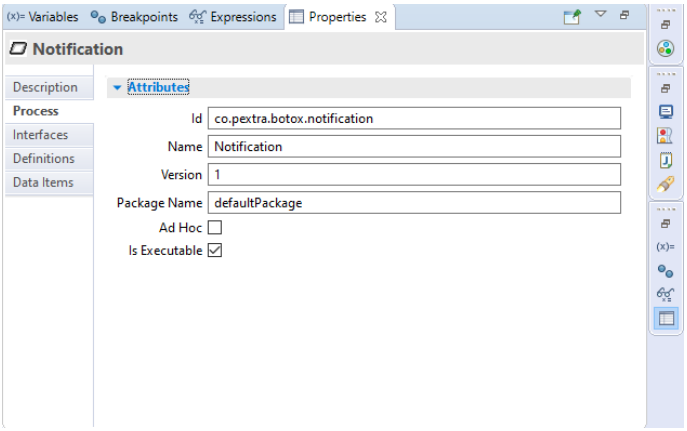

Figura 34 – Definições do Processo

Na aba de Definitions é possível importar as classes que serão utilizadas para

executarem rotinas programáticas dentro dos processos. Sendo assim, as classes podem ser procuradas a partir dos nomes definidos em seus pacotes (para classes externas) e pelos seus próprios nomes para classes padrões do Java, como pode ser visto na Figura 35. Ao adicionarem as classes aos processos, estas serão carregadas juntamente ao processo toda vez que o processo for carregado.

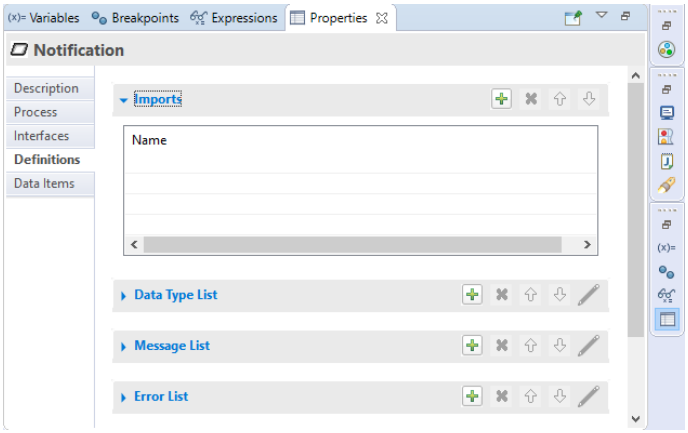

Figura 35 – Importação de classes

Em alguns processos ou tarefas, são necessários definir atributos ou variáveis de estado inicial, por exemplo: uma flag de controle para a manipulação de um estado dentro do processo, ou até mesmo, o incremento da quantidade de notificações que foram disparadas, o local de configuração das variáveis globais pode ser na Figura 36.

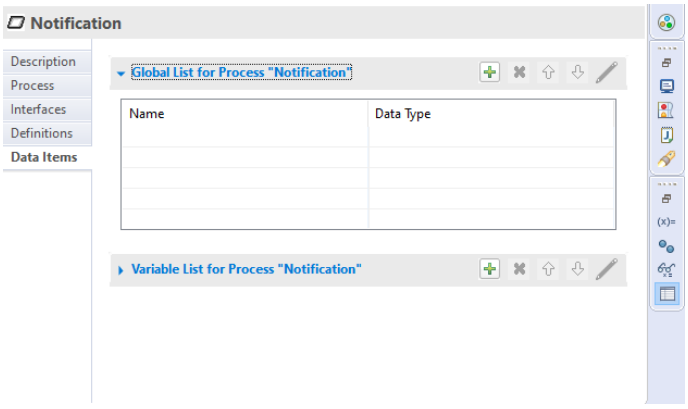

Figura 36 – Definições de variáveis globais

As variáveis globais muitas vezes podem ser utilizadas para comunicação e tratamento interno de estados ou dados dentro dos processos, ou para a comunicação entre processo e regra/situação, podendo funcionar como uma passagem de parâmetro.

5.4.4 Procedimento do Desenvolvedor

Esta seção é focada na área de conhecimento dos desenvolvedores. A perspectiva do desenvolvedor envolve a modelagem das regras de negócio por meio regras e situações e a ativação do processo de negócios. Uma definição do *Situation Type* pode ser vista na Figura 24, que descreve quais dados são pertinentes ao domínio em questão. Os dados definidos no *Situation Type* devem ser escolhidos com base na natureza e relevância da informação para o domínio para que seja utilizado no momento de detecção das situações. Neste exemplo, o tipo de situação a ser definida representa dados a serem colhidos por sensores. Os sensores por sua vez estão associados a uma classe sensor, exemplificada na seção anterior e uma lista que conterá os dados de temperatura coletados.

A classe definir o tipo de situação pode ter vários parâmetros, no exemplo da Figura 24 é definido uma classe que estende de uma classe pai *Situation* definida pelo Scene. Em seguinte, os atributos pertinentes a essa classe são definidos e poderão ser acessados nas *Situation Rules*. A Figura 24 define uma classe *TemperatureSituation* que representa uma entidade com atributo sensor e um atributo de lista. Ambos os tipos serão manipulados pelo Scene durante a detecção das situações, no qual o sensor é o objeto a ser observado, e as transactions são a lista é o conjunto de dados que precisam ser analisados.

Após a definição do *Situation Type*, é necessário descrever as regras que serão casadas para a detecção de situações. Com isso, caso um determinado conjunto de eventos ocorra, uma situação é detectada e um processo de negócio é iniciado. A Figura 26 mostra um fragmento de código de definição de *Situation Rules* no cenário descrito.

A Figura 26 descreve 4 (quatro) regras que tomam como base o *Situation Type* definido como *TemperatureSituation*. A primeira regra analisa se houve mais de 3 (três) elevações de temperatura como mesmo sensor em um intervalo de 2 (dois) minutos; A segunda regra analisa se houve mais de 5 (cinco) picos de temperatura acima de -5 graus em um intervalo de 2 minutos e a terceira regra analisa se houve mais de 10 (dez) picos de temperatura acima de 0 (zero) grau em um intervalo de 2 (dois) minutos. Por fim, a quarta regra analisa se em um intervalo de tempo de 2 (dois) minutos houve pelo menos 2 (duas) ativações do *TemperatureSituation*; caso isso ocorra, um processo de negócio de notificação às partes interessadas é disparado.

Os *Situation Types* e os *Situation Rules* são interpretados pelo *Knowledge Module*, os *Situation Rules* são armazenados na *Rule Base*, os dados de temperatura vindos do freezer são armazenados na *Work Memory* e o componente *Situation and Context Engine* realiza o processamento das informações armazenadas tanto na *Work Memory* quanto na *Rule Base*, com o objetivo de realizar um *pattern match*. Ao ocorrer o casamento de padrão e devido à ocorrência um conjunto de eventos de temperatura em um intervalo de tempo relevante para as regras de negócio, o processo de negócio é coletado no repositório

de processos, passa pelo Semantic Module, que realiza o parser de todo o processo XML e o entrega para a *Process Engine* para que esta execute o processo forma assíncrona e automatizada.

Caso seja necessário configurar ou importar novos pacotes Java ao projeto, estas modificações são feitas através do gerenciados de pacotes Maven, e pode ser acessado através do pom.xml. A Figura 37 mostra as dependências importadas no projeto base.

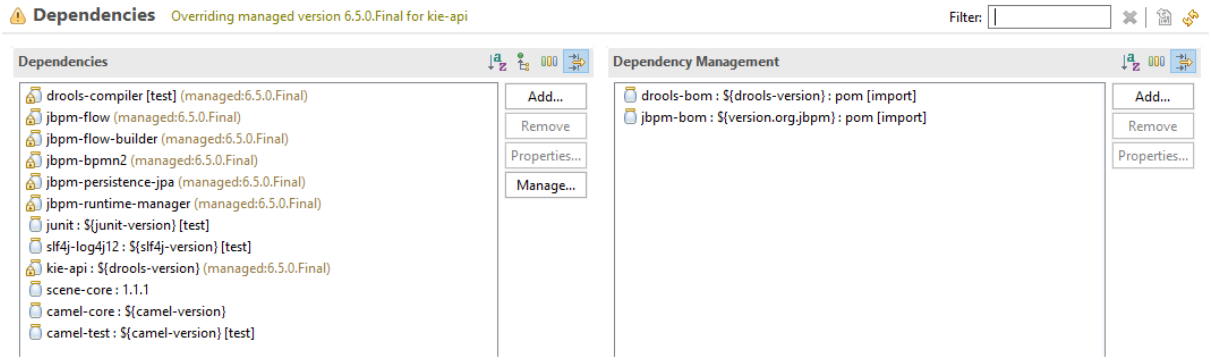

Figura 37 – Arquivo de configuração pom.xml

Essas dependências são necessárias para importar novas funcionalidades implementadas previamente, por exemplo: bibliotecas de envio de e-mail ou de construção de API's.

5.4.5 Passo-a-passo do experimento

Nesta seção será descrito como conduzir o experimento. O experimento será conduzido em um ambiente preparado tomando como base a arquitetura conceitual proposta, com o objetivo de validar a eficácia da solução proposta. A condução do experimento será realizada em duas vertentes, com o objetivo de receber feedbacks tanto dos *BPM Experts* quanto dos Desenvolvedores. Esta seção será descrita em 4 (quatro) subitens que trataram dos seguintes aspectos: Download da Máquina Virtual para a experimentação, Acesso ao Ambiente de Desenvolvimento e Modelagem, Revisão do Ambiente e Passo-a-passo dos *BPM Experts*/Desenvolvedores.

- Download da máquina virtual: O download da máquina virtual desenvolvida utilizando o VMware está disponível no https://bit.ly/33CAHdP. O VMware é uma ferramenta paga desenvolvida pela VMware Inc., entretanto, caso seja necessário, é possível executar a Máquina Virtual utilizando o Virtual Box. O procedimento é realizar o download da máquina virtual disponível no link acima, e seguir a descrição do tutorial detalhado que está disponível em: https://bit.ly/33pkUij.
- Acesso ao Ambiente de Desenvolvimento e Modelagem: A ferramenta adotada como ambientes de modelagem dos desenvolvedores e *BPM Experts* é o Eclipse. Nesta

ferramenta é possível realizar a modelagem tanto das regras e situações, quanto os processos de negócios. Para a modelagem das regras e situações é utilizado o arquivo .drl, e os processos nos arquivos .bpmn.

- Passo-a-passo dos *BPM Experts*: O objetivo deste experimento para os *BPM Experts* é o desenvolvimento de um processo para a notificação dos stakeholders. Nas seções anteriores foram mostrados alguns exemplos de processos de notificação. Os *BPM Experts* terão que criar um processo para a notificação dos stakeholders, recomendado importar uma classe Java e utilizar uma task de serviços ou de e-mail para realizar o disparo da notificação. É aconselhável aos usuários que possuem uma maior experiência com o jBPM a implementação do processo utilizando a task de envio de e-mail. Para os usuários menos experientes com o jBPM é aconselhável a implementação da task de serviço. Antes da implementação, é recomendado que se rode a aplicação como mostrado na Figura 30. Após isso,o usuário deve implementar seu processo de notificação e rodar a aplicação novamente, e observar o resultado. Caso haja maiores dificuldades na implementação, está disponível um tutorial para a configuração de uma classe de envio de e-mail em Java, utilizando um Mail API: https://bit.ly/2wbSBb4.

- Passo-a-passo dos Desenvolvedores: O objetivo deste experimento para os Desenvolvedores é basicamente configurar e modelar as regras e situações acerca do domínio de interesse. Nas seções anteriores foram mostrados alguns exemplos de situações para a captura de dados de sensores em um dado domínio. O objetivo deste experimento é a criação de novas regras de análise de situações, com o objetivo de compor um cenário ainda mais à prova de falhas. No que tange à regra de ativação da situação, caso haja novos critérios de comparação para a satisfação das condições de ativação do processo, a mesma pode ser modificada sem qualquer problema.

5.4.6 Perguntas

As perguntas do experimento foram divididas entre, perguntas comuns para as duas áreas, perguntas específicas para os especialistas de processos e perguntas específicas para os desenvolvedores. As perguntas em comuns para as duas áreas foram:

- A – Qual seu nome completo? (Não obrigatória);
- B – Qual sua idade? (Obrigatória);
- C – Qual sua formação acadêmica? (Obrigatória);
  - 1 Não graduado - Profissional da área
  - 2 Graduando/Graduado

  - 3 Graduado - Profissional da área
  - 4 Pós-graduando/Pós-graduado
  - 5 Pós-graduando/Pós-graduado - Profissional da área

- D – Quanto tempo levou para realizar o experimento? (Obrigatória);
  - 1 Menos de 30 minutos
  - 2 Aproximadamente 1 hora
  - 3 Aproximadamente 2 horas
  - 4 Mais de 2 horas

- E – Levando em conta sua experiência profissional, o problema apresentado no cenário do aplicativo é considerado semelhante ou equivalente a um problema do mundo real? (Obrigatória);

- F – A arquitetura foi útil para resolver o problema proposto? (Obrigatória);

- G – Supondo que você precise usar uma abordagem de análise de situações que mediante a ocorrência de uma situação dispare BP, como o apresentado anteriormente, você utilizaria a arquitetura proposta para resolver o problema? (Obrigatória);

As demais perguntas foram específicas para cada área de interesse. A seguir são descrito as perguntas da área do Especialista de Processos:

- H – É complexo iniciar serviços a partir dos modelos de processos de negócios definidos dentro da arquitetura? (Obrigatória);

- I.1 – Quantos anos de experiência como Especialista em processos? (Obrigatória);
  - 1 Menos de 1 ano
  - 2 De 1 - 3 anos
  - 3 De 4 - 6 anos
  - 4 Mais de 6 anos

- J.1 – Possui certificações CBPP? (Não obrigatória);
  - 1 Sim
  - 2 Não

- K.1 – É fácil configurar e/ou reaproveitar o BP com o uso da arquitetura para resolver o problema proposto por meio de um fluxo de trabalho de BPM? (Obrigatória);

- L.1 – É fácil armazenar os processos nos repositórios e mapear os processos criados? (Obrigatória);

As perguntas realizadas aos Desenvolvedores foram:

- I.2 – Quantos anos de experiência com desenvolvimento? (Obrigatória);
  - 1 Menos de 1 ano
  - 2 De 1 - 3 anos
  - 3 De 4 - 6 anos
  - 4 Mais de 6 anos
- J.2 – Possui certificação Java? (Não obrigatória);
  - 1 Sim
  - 2 Não
- K.2 – É fácil configurar e/ou reaproveitar o aplicativo que você desenvolveu com o uso da arquitetura para resolver o problema proposto? (Obrigatória);
- L.2 – É complexo associar os identificadores de processos ao gatilho de início dos processos mediante a ocorrência de uma situação? (Obrigatória);

Por fim, um campo de sugestão para ambas as áreas adicionarem sugestões ao trabalho ou ao experimento apresentado.

- M – Observações complementares? (Sugestões) (Não obrigatória)

As questões A e M possuem respostas não obrigatórias e de texto livre. A questão B é obrigatória e de texto livre. As perguntas E, F, G, H, K e L possuem as seguintes opções: 1 – discordo totalmente; 2 – discordo; 3 – neutro (nem concordo nem discordo); 4 – concordo; 5 – concordo plenamente;

De posse dos questionários, os públicos-alvo foram abordados por meio de convites e mídias sociais e convidados a participarem do experimento. Os resultados da pesquisa podem ser acessados por meio das seguintes URL's. Para os *BPM Experts*: https://bit.ly/3dmpH8U e para os Desenvolvedores: https://bit.ly/2xgO8Eh.

Caso o usuário não possuisse muita experiências com o Drools, o participante era aconselhado a visualizar o seguinte endereço para uma explicação mais aprofundada: https://red.ht/2IPDhDF.

5.5 Resultados

Como resultado ao experimento proposto, houveram 15 avaliações de desenvolvedores e 12 avaliações de especialistas de processos de negócio, tendo um conjunto total de 27 colaborações. O público de desenvolvedores com perfil acadêmico foi de aproximadamente 40%, ou seja, 6 avaliações, enquanto os outros 60%, ou seja, 9 dos avaliadores estão no mercado de trabalho e atuam nas mais diversas áreas. O público de especialistas de processos obteve um percentual de aproximadamente 91,7%, ou seja, 11 profissionais formados e atuantes no mercado de trabalho e aproximadamente 8,3% de acadêmicos.

Para facilitar o entendimento das questões respondidas por desenvolvedores e especialistas de processos, as perguntas listadas na seção anterior foram classificadas da seguinte forma: respostas dos desenvolvedores (R1:E, R2:F, R3:G, R4:K.2, R5:L.2) e respostas dos especialistas de processos (R1:E, R2:F, R3:G, R4:H, R5:K.2, R6:L.2).

As Tabelas 3 e 4 apresentam os resultados obtidos no experimento realizado, mostrando a visão e satisfação dos desenvolvedores e especialistas de processos ao experimento apresentado.

Tabela 3 – Respostas dos Desenvolvedores

Níveis	R1	R2	R3	R4	R5
discordo totalmente	0	0	0	0	14
discordo	0	0	0	0	1
neutro	1	0	1	0	0
concordo	5	4	4	4	0
concordo plenamente	9	11	10	11	0

Tabela 4 – Respostas dos Especialistas de Processos

Níveis	R1	R2	R3	R4	R5	R6
discordo totalmente	0	0	0	0	0	7
discordo	0	0	0	0	0	4
neutro	0	0	0	0	1	1
concordo	2	2	3	1	3	0
concordo plenamente	10	10	9	11	8	0

Como feedbacks textuais, foram levantados possíveis melhorias de integração entre o Scene e o Jbpm, uma vez que algumas funções do Drools ao serem utilizadas com o Scene geram um certo retrabalho. Além disso, foram também sugeridas melhorias na facilidade de subir os processos para o repositório de processos; e questões pertinentes à segurança (ligadas ao requisito R6), mas que não foram abordados nesse trabalho.

A avaliação conduzida mostra que o método empírico utilizado trouxe resultados expressivos e relevantes, uma vez que os níveis das respostas de concordância com as

perguntas propostas ficaram em média entre 91,66% e 93,34% de positividade aos questionamentos levantados. Tendo em vista a dificuldade de se validar modelos conceituais, a adoção da abordagem utilizada, de mobilizar profissionais do mercado e acadêmicos para o fornecimento de feedbacks, mostrou-se eficaz para fomentar a contribuição científica e validar o trabalho proposto.

5.6 Considerações Finais do Capítulo

Este capítulo apresentou um experimento empírico para validar a abordagem proposta neste trabalho, focando na separação de preocupações entre desenvolvedores e analistas de processos de negócio. O cenário do estudo escolhido é um problema real, que provê uma ampla gama de possibilidades de situações contextuais que podem ser integradas com processos de negócio.

Com a abordagem proposta nesta dissertação, os serviços que comumente são definidos por meio de código, passam a ser definidos por especialistas de processos de negócio, possibilitando que as partes interessadas nestes processos façam suas próprias alterações nos modelos, sem a necessidade de equipes de desenvolvimento, uma vez que as aplicações bases já estão implementadas. Técnicas AS-IS e TO-BE ("como o processo será" e "como ele deveria ser") podem ser aplicadas com este tipo de solução, sem a necessidade de reimplementação de código por parte dos desenvolvedores, possibilitando melhorias continuadas nos processos. Todos os códigos utilizados nesta implementação estarão disponíveis no Apêndice B.

O próximo capítulo apresenta as conclusões deste trabalho, destacando as suas considerações finais e perspectivas de trabalhos futuros.

6 Conclusões e Trabalhos Futuros

6.1 Conclusões

Este trabalho apresentou uma arquitetura de suporte ao desenvolvimento de aplicações organizacionais apoiados em modelos de processos de negócio, que promove a separação de papéis entre especialistas de domínio e desenvolvedores. Na arquitetura proposta, a definição das regras de negócio e o tratamento de eventos associados a estas regras são realizadas pelos desenvolvedores, enquanto que os modelos de processos de negócio são descritos pelos especialistas de processos de negócio. Esta abordagem de divisão de responsabilidade é interessante no ambiente coorporativo considerando que a complexidade de implementação de algumas tarefas são retiradas dos desenvolvedores e executadas por meio de modelos de processos de negócio que podem ser automatizados, diminuindo a dependência departamental dentro das organizações. Em outras palavras, a estratégia de separação de papéis permite que os desenvolvedores abstraiam detalhes de modelagem de processos de negócio e que os especialistas de processos de negócio abstraiam conceitos de implementação de baixo nível de *software*. Assim, os desenvolvedores passam a interagir minimamente com os especialistas de processos de negócio, uma vez que eles passam a se preocupar apenas com os aspectos de implementação, referenciando processos por meio de chamadas *HTTP* em uma *API RESTful* usando seus identificadores e serviços por meio de seus nomes.

A arquitetura proposta segue a abordagem de orientação a serviços (SOA) e implementa um serviço de repositório de processos de negócio para o armazenamento de dados dos processos, dos modelos de processos de negócio e dos serviços que são mapeados pela organização. Isso permite que dados importantes para a organização passem a ser armazenados de maneira adequada, proporcionando que colaborações entre organizações também ocorram. A implementação da arquitetura utilizou a plataforma Drools como sistema de regras, o ambiente Scene para prover o suporte de análise de situações, jBPM como *engine workflow* de processos de negócio, o *BPMN Diagram Editor* para a modelagem dos processos de negócio, e o Firebase para a definição de um serviço distribuído de armazenamento de processos de negócio, dados e serviços.

O jBPM, Drools e Scene possuem fácil integração devido ao Scene ter sido desenvolvido no topo da plataforma Drools e o jBPM e o Drools pertencerem ao mesmo grupo de desenvolvimento. Os processos de negócio modelados com o *BPMN Diagram Editor* são compreensíveis pelo jBPM e são estruturados em linguagem XML. O Firebase, por sua vez, possui inúmeras facilidades de interesse do projeto, sendo algumas delas a possibilidade de definição de serviços, autenticação, armazenamento de dados utilizando No-SQL e

armazenamento via *Storage* – o que permitiu uma implementação leve no nível de prova de conceito. O Firebase também permite que Web Services RESTful sejam implementados e acessados facilmente por aplicações externas. Para a geração de chaves de autenticação foi utilizada a linguagem Node.js e uma implementação de tokens JWT. Por fim, um ambiente de simulação para analisar o funcionamento da aplicação foi desenvolvido, com o objetivo de simular o processo de coleta de informações de baixo nível, o que é constantemente implementado em grandes organizações para o monitoramento de ambientes internos e/ou externos.

A arquitetura proposta, com tal conjunto de funcionalidades, foi concebida com base em um mapeamento sistemático da literatura da área. O trabalho junta-se, portanto, ao movimento recente de concepção de aplicações sensíveis a situação integradas aos modelos BPMN em diferentes domínios de aplicações. Observa-se que a integração destas aplicações com um repositório de processos de negócio é uma tendência atual, particularmente a possibilitar de melhoria continuada dos processos envolvidos na organização, permitindo que as aplicações e sistemas sempre rodem modelos atualizados com uma infraestrutura altamente escalável, adaptável, alta capacidade de armazenamento e tráfego de rede.

6.2 Trabalhos Futuros

Algumas questões importantes não foram tratadas nesta versão da arquitetura, constituindo, portanto, em temas naturais de trabalhos futuros. Dentre estes temas, destacamos:

- **Autenticação.**

Implementação do componente de autenticação da arquitetura conceitual pertinente ao serviço de repositório de processos de negócio, descrito no R6;

- **Otimização automática de processos de negócio.**

Estudos para a melhoria continuada dos processos de negócio em meios organizacionais podem ser conduzidos com o objetivo otimizar cada vez mais a arquitetura de acordo com as necessidades da organização, atendendo aos modelos de melhoria AS-IS e TO-BE. Sendo assim, mecanismos para uma melhoria contínua destes processos podem ser incorporados na arquitetura conceitual objetivando atender melhor as demandas da organização, e agregando mais conhecimento em sua cultura de processos de negócio.

- **Implementar os demais módulos descritos em azul na arquitetura conceitual.**

A arquitetura conceitual dividiu componentes em azuis e vermelhos. Esta distinção de cores foi realizada devido à complexidade de implementação de todos os componentes na prova de conceito e na arquitetura de implementação; sendo assim, apenas os compo-

nentes em vermelho foram implementados nesta versão. Em trabalhos futuros, pretende-se complementar a implementação, desenvolvendo uma versão com todos os módulos da arquitetura conceitual proposta.

Referências

AA, H. Van der et al. Challenges and opportunities of applying natural language processing in business process management. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *COLING 2018: The 27th International Conference on Computational Linguistics: Proceedings of the Conference: August 20-26, 2018 Santa Fe, New Mexico, USA*. [S.l.], 2018. p. 2791–2801. Citado 2 vezes nas páginas 37 e 41.

ABBASI, A. Z.; SHAIKH, Z. A. A conceptual framework for smart workflow management. In: IEEE. *Information Management and Engineering, 2009. ICIME'09. International Conference on*. [S.l.], 2009. p. 574–578. Citado 2 vezes nas páginas 15 e 26.

ADI, A.; BOTZER, D.; ETZION, O. Semantic event model and its implication on situation detection. *ECIS 2000 Proceedings*, p. 2, 2000. Citado 2 vezes nas páginas 3 e 22.

ALMEIDA, J. P. A. et al. Model-driven development of context-aware services. In: SPRINGER. *IFIP International Conference on Distributed Applications and Interoperable Systems*. [S.l.], 2006. p. 213–227. Citado na página 49.

ANTHES, G. *Eyes everywhere: business activity monitoring offers a constant watch on business processes, computerworld*. [S.l.]: November, 2003. Citado na página 2.

BAJEC, M.; KRISPER, M. A methodology and tool support for managing business rules in organisations. *Information Systems*, Elsevier, v. 30, n. 6, p. 423–443, 2005. Citado 5 vezes nas páginas 15, 26, 30, 31 e 32.

BALCER, M. J. *BPM Implementation with SOA and MDA*. 2006. Disponível em: <https://www.omg.org/news/meetings/workshops/soa-bpm-mda-2006/06-1_Balcer_Revised.pdf>. Citado 3 vezes nas páginas 4, 5 e 18.

BALDAM, R. et al. Gerenciamento de processos de negócios-bpm. *Business Process Management*, Elsevier, v. 1, 2007. Citado 2 vezes nas páginas 1 e 11.

BALI, M. *Drools JBoss Rules 5.0 Developer's Guide*. [S.l.]: Packt Publishing Ltd, 2009. Citado 6 vezes nas páginas 2, 19, 20, 21, 55 e 66.

BENITEZ, M. *Business Process Management (BPM) For The Masses*. 2006. Disponível em: <https://www.oracle.com/technetwork/articles/entarch/bpm-for-the-masses-090863.html>. Citado na página 12.

BOSCH, J. *Design and Use of Software Architectures: Adopting and Evolving a ProductLine Approach*. [S.l.]: Harlow: Pearson Education Ltd, 2000. v. 1. Citado na página 17.

BRERETON, P. et al. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of systems and software*, Elsevier, v. 80, n. 4, p. 571–583, 2007. Citado na página 25.

BUCCHIARONE, A. et al. Dynamic adaptation of fragment-based and context-aware business processes. In: IEEE. *Web Services (ICWS), 2012 IEEE 19th International Conference on*. [S.l.], 2012. p. 33–41. Citado na página 27.

BUCCHIARONE, A.; MEZZINA, C.; PISTORE, M. Captlang: a language for context-aware and adaptable business processes. In: ACM. *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*. [S.l.], 2013. p. 12. Citado 2 vezes nas páginas 26 e 27.

CBOK, B. Guia para o gerenciamento de processos de negócio corpo comum de conhecimento. *Association of Business Process Management Professionals. ABPMP BPM CBOK*, v. 3, 2013. Citado 3 vezes nas páginas 13, 14 e 15.

CELESTRINI, J. R. et al. An architecture and its tools for integrating iot and bpmn in agriculture scenarios. In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. [S.l.: s.n.], 2019. p. 824–831. Citado 4 vezes nas páginas 15, 26, 34 e 35.

CHOI, I.; KIM, K.; JANG, M. An xml-based process repository and process query language for integrated process management. *Knowledge and Process Management*, Wiley Online Library, v. 14, n. 4, p. 303–316, 2007. Citado 2 vezes nas páginas 3 e 4.

CONNER, P.; ROBINSON, S. *Service-oriented architecture*. [S.l.]: Google Patents, 2007. US Patent App. 11/388,624. Citado na página 17.

COSTA, P. D. Architectural support for context-aware applications: from context models to services platforms. 2007. Citado na página 21.

COSTA, P. D. et al. Rule-based support for situation management. In: *Fusion Methodologies in Crisis Management*. [S.l.]: Springer, 2016. p. 341–364. Citado 4 vezes nas páginas 3, 52, 54 e 68.

CRUZ, T. *Sistemas, métodos & processos: administrando organizações por meio de processos de negócios* . [S.l.]: Editora Atlas SA, 2003. Citado na página 11.

DAVENPORT, T. H. *Process innovation: reengineering work through information technology*. [S.l.]: Harvard Business Press, 1993. Citado na página 12.

ERL, T. *Service-oriented architecture*. [S.l.]: Pearson India, 2005. v. 8. Citado na página 17.

EUGSTER, P. T. et al. The many faces of publish/subscribe. *ACM computing surveys (CSUR)*, ACM, v. 35, n. 2, p. 114–131, 2003. Citado na página 49.

FALBO, R. de A. Mapeamento sistemático. *Retrieved October*, v. 7, 2018. Citado na página 109.

FIORINI, S. T.; LEITE, J. C. S. do P.; LUCENA, C. J. P. de. Process reuse architecture. In: SPRINGER. *International Conference on Advanced Information Systems Engineering*. [S.l.], 2001. p. 284–298. Citado 3 vezes nas páginas 1, 3 e 4.

GATTI, L. A. d. C.; SANTORO, F. M.; NUNES, V. T. An agent-based architecture for knowledge management in context-aware business processes. In: IEEE. *The 2010 14th International Conference on Computer Supported Cooperative Work in Design*. [S.l.], 2010. p. 318–323. Citado na página 5.

HAYES-ROTH, F. Rule-based systems. *Communications of the ACM*, ACM, v. 28, n. 9, p. 921–932, 1985. Citado na página 20.

HERBST, H. Business rules in systems analysis: a meta-model and repository system. *Information Systems*, Elsevier, v. 21, n. 2, p. 147–166, 1996. Citado na página 2.

HILL, E. F. *Jess in action: Java rule-based systems*. [S.l.]: Manning Publications Co., 2003. Citado 2 vezes nas páginas 19 e 20.

HUANG, H. et al. Efficiently querying large process model repositories in smart city cloud workflow systems based on quantitative ordering relations. *Information Sciences*, Elsevier, v. 495, p. 100–115, 2019. Citado 2 vezes nas páginas 37 e 43.

HUANG, W.; TAO, T. Adding context-awareness to knowledge management in modern enterprises. In: IEEE. *Intelligent Systems, 2004. Proceedings. 2004 2nd International IEEE Conference*. [S.l.], 2004. v. 2, p. 393–398. Citado na página 5.

KAMBONA, K.; BOIX, E. G.; MEUTER, W. D. Serena: scalable middleware for real-time web applications. In: ACM. *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. [S.l.], 2015. p. 802–805. Citado 3 vezes nas páginas 19, 20 e 21.

KITCHENHAM, B. Procedures for performing systematic reviews. *Keele, UK, Keele University*, v. 33, n. 2004, p. 1–26, 2004. Citado na página 25.

KITCHENHAM, B. A.; BUDGEN, D.; BRERETON, O. P. Using mapping studies as the basis for further research–a participant-observer case study. *Information and Software Technology*, Elsevier, v. 53, n. 6, p. 638–651, 2011. Citado na página 25.

KOENIG, J. Jboss jbpm. *White Paper, JBoss Labs, Atlanta, GA*, 2004. Citado 2 vezes nas páginas 55 e 66.

LAURINDO, F. J.; ROTONDARO, R. G. Gestão integrada de processos e da tecnologia da informação. *São Paulo: Atlas*, v. 1, 2006. Citado na página 12.

LAZARTE, I. M. et al. A distributed repository for managing business process models in cross-organizational collaborations. *Computers in Industry*, Elsevier, v. 64, n. 3, p. 252–267, 2013. Citado 7 vezes nas páginas 15, 5, 17, 37, 38, 52 e 70.

MAIO, M. N. D.; SALATINO, M.; ALIVERTI, E. *jBPM6 Developer Guide*. [S.l.]: Packt Publishing Ltd, 2014. Citado 5 vezes nas páginas 1, 5, 52, 55 e 66.

MARY, S.; DAVID, G. Software architecture: Perspectives on an emerging discipline. *Prentice-Hall*, 1996. Citado na página 18.

MATHEW, A. *Benchmarking of complex event processing engine-esper*. [S.l.], 2014. Citado na página 21.

MATTOS, T. da C. et al. A formal representation for context-aware business processes. *Computers in Industry*, Elsevier, v. 65, n. 8, p. 1193–1214, 2014. Citado na página 5.

MELLOR, S. J. et al. *MDA distilled: principles of model-driven architecture*. [S.l.]: Addison-Wesley Professional, 2004. Citado na página 49.

MITCHELL, R. J. *Managing complexity in software engineering*. [S.l.]: IET, 1990. Citado na página 4.

MITRA, T. A case for soa governance. *IBM developerworks August*, 2005. Citado na página 61.

MOREIRA, J. L. et al. Developing situation-aware applications for disaster management with a distributed rule-based platform. In: *Challenge+ DC@ RuleML*. [S.l.: s.n.], 2015. Citado na página 29.

MOREIRA, J. L. et al. Towards ontology-driven situation-aware disaster management. *Applied ontology*, IOS Press, v. 10, n. 3-4, p. 339–353, 2015. Citado 6 vezes nas páginas 15, 3, 5, 26, 29 e 30.

MORIARTY, T. Business rule management facility: System architect 2001. *Intelligent Enterprise*, v. 3, n. 12, p. 14–18, 2000. Citado na página 2.

NARDI, J. C.; FALBO, R. de A.; ALMEIDA, J. P. A. Foundational ontologies for semantic integration in eai: a systematic literature review. In: SPRINGER. *Conference on e-Business, e-Services and e-Society*. [S.l.], 2013. p. 238–249. Citado na página 25.

NEUMANN, J. et al. Extending bpmn 2.0 for intraoperative workflow modeling with ieee 11073 sdc for description and orchestration of interoperable, networked medical devices. *International journal of computer assisted radiology and surgery*, Springer, v. 14, n. 8, p. 1403–1413, 2019. Citado 2 vezes nas páginas 26 e 33.

OCA, I. M.-M. de et al. A systematic literature review of studies on business process modeling quality. *Information and Software Technology*, Elsevier, v. 58, p. 187–205, 2015. Citado 2 vezes nas páginas 17 e 37.

PEREIRA, I. S.; COSTA, P. D.; ALMEIDA, J. P. A. A rule-based platform for situation management. In: IEEE. *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2013 IEEE International Multi-Disciplinary Conference on*. [S.l.], 2013. p. 83–90. Citado 6 vezes nas páginas 3, 21, 22, 55, 66 e 68.

PERREY, R.; LYCETT, M. Service-oriented architecture. In: IEEE. *Applications and the Internet Workshops, 2003. Proceedings. 2003 Symposium on*. [S.l.], 2003. p. 116–119. Citado 2 vezes nas páginas 15 e 18.

POLPINIJ, J.; GHOSE, A.; DAM, H. K. Mining business rules from business process model repositories. *Business Process Management Journal*, Emerald Group Publishing Limited, v. 21, n. 4, p. 820–836, 2015. Citado 4 vezes nas páginas 1, 5, 37 e 52.

POLYVYANYY, A. *Business Process Querying*. 2019. Citado 4 vezes nas páginas 15, 37, 42 e 43.

POURMIRZA, S. et al. Bpms-ra: A novel reference architecture for business process management systems. *ACM Transactions on Internet Technology (TOIT)*, ACM New York, NY, USA, v. 19, n. 1, p. 1–23, 2019. Citado 4 vezes nas páginas 15, 37, 41 e 42.

RAIK, H. et al. Astro-captevo: Dynamic context-aware adaptation for service-based systems. In: IEEE. *Services (SERVICES), 2012 IEEE Eighth World Congress on*. [S.l.], 2012. p. 385–392. Citado 3 vezes nas páginas 15, 27 e 28.

RODRIGUES, T. et al. A model-based approach for building ubiquitous applications based on wireless sensor network. In: SPRINGER. *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*. [S.l.], 2010. p. 350–352. Citado na página 53.

ROSA, M. L. et al. Apromore: An advanced process model repository. *Expert Systems with Applications*, Elsevier, v. 38, n. 6, p. 7029–7040, 2011. Citado 6 vezes nas páginas 15, 5, 16, 17, 37 e 39.

ROSEMANN, M. Potential pitfalls of process modeling: part a. *Business Process Management Journal*, Emerald Group Publishing Limited, v. 12, n. 2, p. 249–254, 2006. Citado 2 vezes nas páginas 1 e 2.

ROSEN, M. et al. *Applied SOA: service-oriented architecture and design strategies*. [S.l.]: John Wiley & Sons, 2012. Citado na página 5.

RUMMLER, G.; BRACHE, A. *Improving performance: Managing the white space in organizations*. [S.l.]: San Francisco: Jossey-Bass, 1995. Citado na página 12.

SANTRA, D.; CHOUDHURY, S. C-bpmn: A context aware bpmn for modeling complex business process. *arXiv preprint arXiv:1806.01333*, 2018. Citado 2 vezes nas páginas 26 e 32.

SCHILIT, B. N.; THEIMER, M. M. Disseminating active map information to mobile hosts. *IEEE network*, IEEE, v. 8, n. 5, p. 22–32, 1994. Citado na página 15.

SMITH, H.; FINGAR, P. *Business process management: the third wave*. [S.l.]: Meghan-Kiffer Press Tampa, 2003. v. 1. Citado na página 14.

SOUZA, S. C. de; FILHO, J. G. P.; SPESSIMILLE, E. F. *A Rule-Base Approach for WSN Application Development in a Cloud Environment*. [S.l.]: JACN, 2013. Citado na página 52.

STRUCK, D. L. *Business rule continuous requirements environment*. [S.l.]: Colorado Technical University, 1999. Citado na página 2.

TEMPORA final review. *Technical report, TEMPORA consortium*, 1994. Citado na página 2.

TENG, J. C. *Template based workflow definition*. [S.l.]: Google Patents, 2009. US Patent 7,581,011. Citado 2 vezes nas páginas 15 e 16.

VASILECAS, O.; KALIBATIENE, D.; LAVBIČ, D. Rule-and context-based dynamic business process modelling and simulation. *Journal of Systems and Software*, Elsevier, v. 122, p. 1–15, 2016. Citado 6 vezes nas páginas 15, 1, 5, 26, 28 e 29.

VUJOVIĆ, V. et al. A graphical editor for restful sensor web networks modeling. In: IEEE. *2014 IEEE 9th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI)*. [S.l.], 2014. p. 61–66. Citado na página 49.

WANG, W.; INDULSKA, M.; SADIQ, S. Integrated modelling of business process models and business rules: a research agenda. In: ACIS. [S.l.], 2014. Citado na página 3.

WEBER, B. et al. Refactoring large process model repositories. *Computers in Industry*, Elsevier, v. 62, n. 5, p. 467–486, 2011. Citado na página 17.

YAN, Z.; DIJKMAN, R.; GREFEN, P. Business process model repositories–framework and survey. *Information and Software Technology*, Elsevier, v. 54, n. 4, p. 380–395, 2012. Citado 8 vezes nas páginas 15, 5, 16, 17, 37, 39, 40 e 45.

YE, J.; DOBSON, S.; MCKEEVER, S. Situation identification techniques in pervasive computing: A review. *Pervasive and mobile computing*, Elsevier, v. 8, n. 1, p. 36–66, 2012. Citado 2 vezes nas páginas 3 e 22.

ZOET, M. et al. Alignment of business process management and business rules. In: *ECIS*. [S.l.: s.n.], 2011. p. 34. Citado na página 1.

Apêndices

APÊNDICE A – Mapeamento Sistemático

Este Apêndice apresenta de forma resumida a metodologia e as strings de busca para a obtenção de trabalhos relacionados e os resultados encontrados com o mapeamento sistemático realizado. Baseando-se nos estudos realizados por (FALBO, 2018), o mapeamento consiste em 4 (quatro) etapas de refinamento, sendo estas realizadas de forma gradual a cada fase do processo.

A Etapa 1 inicia a atividade de pesquisa por trabalhos correlatos. Nesta etapa, as strings de buscas são formadas com o objetivo de capturar os artigos de interesse com os temas relacionados. É necessário utilizar engines de busca tais como IEEE, Compedex, ACM Library, entre outras, para a captura de trabalhos com temas relacionados. A Etapa 2 consiste na leitura dos abstracts com o objetivo de minerar possíveis trabalhos pertinentes ao tema de interesse. Ainda nesta etapa, após a leitura dos abstracts, é válido verificar, de forma superficial, se o corpo do trabalho atende às necessidades inerentes ao tema de interesse, uma vez que os abstracts podem não dar detalhes suficientes acerca de um critério de descarte. A Etapa 3 refere-se à leitura superficial nos artigos resultantes da etapa anterior. Nesta etapa, questões como estruturação do trabalho, imagens, tabelas, etc, são levadas em consideração para a tomada de decisão. Na Etapa 4 é realizada a leitura completa dos trabalhos restantes, com o objetivo de entender a fundo sobre o arcabouço de tecnologias utilizadas, modelos e metodologias que estão sendo adotadas no meio científico.

As pesquisas foram realizadas nas seguintes engines de busca: IEEE, ACM Library, Scopus, Science Direct, Web of Knowledge e Compedex. A pesquisa obteve o seguinte resultado: IEEE (32), ACM Library (3), Scopus (42), Science Direct (43), Web of Knowledge (10) e Compedex (10). Como forma de detalhamento, seguem as strings de busca utilizadas:

• **IEEE:** (("conceptual architecture"OR "concept model"OR "conceptual model"OR "knowledge management"OR "architecture design") AND ("situation awareness"OR "rule based"OR "context aware"OR "situation identification"OR "complex events") AND ("business"AND ("process"OR "processes") OR "BPM"))

• **ACM:** +("conceptual architectureconceptual modelconcept modelconcept architectureknowledge managementarchitecture design") + +("situation awarenessrule based context awaresituation identificationcomplex events") + +("business processbusiness processes"BPM)

• **Scopus:** ALL((conceptual architecture OR conceptual model OR concept model) AND (situation awareness OR rule-based OR context aware OR situation identification OR complex events) AND (intelligent AND (business process management OR BPM)))

• **Science Direct:** (conceptual architecture OR conceptual model OR concept model) AND (situation awareness OR rule-based OR context aware OR situation identification OR complex events) AND (intelligent AND (business process management OR BPM))

• **Web of Knowledge:** TS=(("conceptual architecture"OR "conceptual model"OR "concept model"OR "conceptual models"OR "concept models"OR "knowledge management"OR "architecture design") AND ("situation awareness"OR "rule based"OR "context aware"OR "situation identification"OR "complex events") AND ("business process"OR "business processes"OR "BPM"))

• **Compedex:** ((conceptual architecture OR conceptual model OR concept model) AND (situation awareness OR rule-based OR context aware OR situation identification OR complex events) AND (business process OR BPM))

A Etapa 1 resultou em 140 (cento e quarenta) trabalhos relacionados aos temas de interesse. Após a leitura dos abtracts na Etapa 2 apenas 44 (quarenta e quatro) trabalhos se mostraram relevantes e foram encaminhados para a etapa seguinte. A Etapa 3 resultou em 14 (quatorze) artigos aderentes ao tema de interesse. Por fim, na Etapa 4, foram elencados requisitos relevantes pertinentes às arquiteturas conceituais que abrangem conceitos de situações contextuais e processos de negócio, e definidos os artigos finais de interesse. Uma discussão dos trabalhos analisados é realizada no Capítulo 3 desta dissertação.

Decorrente da necesssidade de armazenamento de modelos de processos de negócio durante a modelagem de processos, foi realizada uma pesquisa que relacionava conceitos de sistemas inteligentes e repositórios de processos de negócio, não sendo possível encontrar trabalhos que relacionassem o tema a modelos, arquiteturas ou frameworks de armazenamento e gerenciamento de processos de negócio. Sendo assim, foi feita uma segunda busca no Google Scholar para verificar trabalhos que tratassem especificamente do tema de repositórios de processos de negócio.

Como forma complementar a captura de trabalhos relacionados, uma terceira busca foi realizada no Google Scholar com o objetivo de encontrar trabalhos mais recentes (particularmente, 2019) que relacionassem os temas de sensibilidade a situações e bpmn, bem como repositórios de processos de negócio. Estes trabalhos foram incorporados à discussão realizada nas Seções 3.1 e 3.2.

APÊNDICE B – Implementação

Neste Apêndice são apresentados os códigos utilizados na implementação da arquitetura referente ao ambiente de modelagem de processos e situações e o mecanismo de autenticação de tokens de API's utilizado no Firebase.

B.1 Entidade Sensor

```
1
2  package co.pextra.botox.entities;
3
4  public class Sensor {
5      private int sensor_id;
6      private String description;
7
8      public Sensor(int sensor_id, String description){
9          this.sensor_id = sensor_id;
10         this.description = description;
11     }
12
13     public int getSensorId() {
14         return sensor_id;
15     }
16
17     public void setSensorId(int sensor_id) {
18         this.sensor_id = sensor_id;
19     }
20
21     public String getDescription() {
22         return description;
23     }
24
25     public void setDescription(String description) {
26         this.description = description;
27     }
28 }
```

B.2 Estrutura de dados associativa Temperatura x Sensor

```
1  package co.pextra.botox.entities;
2
3  import java.io.Serializable;
4  import java.util.Date;
5
6  import org.kie.api.definition.type.Expires;
7  import org.kie.api.definition.type.Role;
8  import org.kie.api.definition.type.Timestamp;
9
10 @Role(Role.Type.EVENT)
11 @Timestamp("executionTime")
12 @Expires("2m")
13 public class TemperatureEvent implements Serializable {
14
15     private static final long serialVersionUID = 1L;
16     private Date executionTime;
17     private Double temperature;
18     private int sensorId;
19
20     public TemperatureEvent(Double temperature, int sensorId) {
21         super();
22         this.executionTime = new Date();
23         this.temperature = temperature;
24         this.sensorId = sensorId;
25     }
26
27     public TemperatureEvent(Double temperature, int sensorId, Date
           executionTime) {
28         super();
29         this.executionTime = executionTime;
30         this.temperature = temperature;
31         this.sensorId = sensorId;
32     }
33
34
35     public Date getExecutionTime() {
36         return executionTime;
37     }
```

```
38
39      public void setExecutionTime(Date executionTime) {
40          this.executionTime = executionTime;
41      }
42
43      public Double getTemperature() {
44          return temperature;
45      }
46
47      public void setTemperature(Double temp) {
48          this.temperature = temp;
49      }
50
51      public int getSensorId() {
52          return sensorId;
53      }
54
55      public void setSensorId(int sensorId) {
56          this.sensorId = sensorId;
57      }
58  }
```

B.3 Rule Engine Thread

```
1
2   package co.pextra.botox.scenary;
3
4
5   import org.kie.api.runtime.KieSession;
6
7
8   public class RuleEngineThread extends Thread {
9       private KieSession ksession;
10      public RuleEngineThread(KieSession ksession) {
11          this.ksession = ksession;
12      }
13      public void run() {
14          this.ksession.fireUntilHalt();
15      }
16  }
```

B.4 Definição das Regras de Análise de Situações

```
1  package co.pextra.botox.scenary;
2
3  import java.util.List;
4
5  import br.ufes.inf.lprm.scene.model.impl.Situation;
6  import br.ufes.inf.lprm.scene.util.SituationHelper;
7  import br.ufes.inf.lprm.situation.annotations.part;
8  import br.ufes.inf.lprm.situation.model.events.*;
9  import org.kie.api.runtime.process.ProcessInstance;
10
11 declare TemperatureSituation extends Situation
12     sensor: Sensor @part
13     transactions: List @part
14 end
15
16 /*
17 Verifica se em uma lista de 3 ou mais temperaturas existem temperaturas
       acima de -3 graus em uma janela de tempo de 2 minutos.
18 */
19 rule "Ocorreram 3 mudanas de temperatura com o mesmo sensor em um
       intervalo de 2 minutos"
20 @role(situation)
21 @type(TemperatureSituation)
22     when
23        sensor : Sensor();
24        transactions: List(size > 5) from accumulate(
25           transaction: TemperatureEvent(sensorId == sensor.sensorId) over
                 window:time (5m),
26           collectList(transaction)
27        )
28      then
29        SituationHelper.situationDetected(drools);
30 end
31
32 rule "More than 6 high temperatures in thirty minutes for a sensor of
       temperature"
33 @role(situation)
34 @type(TemperatureSituation)
```

```
35      when
36          sensor : Sensor();
37          transactions: List(size > 6) from accumulate(
38              transaction: TemperatureEvent(sensorId == sensor.sensorId,
                    temperature > -5) over window:time (30m),
39              collectList(transaction)
40          )
41       then
42          SituationHelper.situationDetected(drools);
43  end
44
45  rule "More than 3 high temperatures in five minutes for a sensor of
        temperature"
46  @role(situation)
47  @type(TemperatureSituation)
48      when
49          sensor : Sensor();
50          transactions: List(size > 3) from accumulate(
51              transaction: TemperatureEvent(sensorId == sensor.sensorId,
                    temperature > -5) over window:time (5m),
52              collectList(transaction)
53          )
54       then
55          SituationHelper.situationDetected(drools);
56  end
57
58  rule "start botox situation when have a one case suspicious"
59      when
60          temperatureSituation: List(size > 2) from accumulate (
61            Activation($sit: situation, $sit isA TemperatureSituation.class)
                    over window:time (10m);
62            collectList($sit)
63          )
64      then
65           ProcessInstance processInstance = kcontext.getKieRuntime().
                startProcess(ws.getProcess('0100123'));
66           System.out.println("uma situao de temperatura ocorreu");
67  end
```

B.5 RuleMain

```
1  package co.pextra.botox.scenary;
2
3  import br.ufes.inf.lprm.scene.SceneApplication;
4  import br.ufes.inf.lprm.scene.base.listeners.SCENESessionListener;
5  import co.pextra.botox.entities.Sensor;
6  import co.pextra.botox.entities.TemperatureEvent;
7
8  import java.util.concurrent.Callable;
9  import java.util.concurrent.ExecutionException;
10 import java.util.concurrent.ExecutorService;
11 import java.util.concurrent.Executors;
12 import java.util.concurrent.Future;
13 import java.util.concurrent.TimeUnit;
14 import java.util.concurrent.TimeoutException;
15
16 import org.kie.api.KieBase;
17 import org.kie.api.KieBaseConfiguration;
18 import org.kie.api.KieServices;
19 import org.kie.api.builder.Message;
20 import org.kie.api.builder.Results;
21 import org.kie.api.conf.EventProcessingOption;
22 import org.kie.api.definition.KiePackage;
23 import org.kie.api.definition.rule.Rule;
24 import org.kie.api.runtime.KieContainer;
25 import org.kie.api.runtime.KieSession;
26 import org.slf4j.Logger;
27 import org.slf4j.LoggerFactory;
28
29
30 public class RuleMain {
31     static final Logger LOG = LoggerFactory.getLogger(RuleMain.class);
32
33     public static void insertTemperatures(Sensor s, int temp_start, int
        temp_end, int temp_mid, int temp_invert, KieSession session){
34
35         for(int instante = 0, temp = temp_start, invert = 0; (temp <=
            temp_mid && invert == 0) || (temp >= temp_end && invert == 1);
            instante++){
```

```
36          if(temp > temp_invert){
37              invert = 1;
38          }
39
40          if (temp <= temp_invert && invert == 0)
41              temp++;
42          else
43              temp--;
44
45          System.out.println(s.getSensorId() + ";" + temp + ";" +
                instante);
46          session.insert(new TemperatureEvent((double)temp, s.
                getSensorId()));
47      }
48
49  }
50
51  public static final void main(String[] args) throws
        InterruptedException, ExecutionException {
52
53
54      try{
55          System.out.println("Starting Project");
56          KieServices kieServices = KieServices.Factory.get();
57
58          KieContainer kContainer = kieServices.getKieClasspathContainer
                ();
59
60          //ReleaseId releaseId = kieServices.newReleaseId("br.ufes.inf.
                lprm", "botox-drugs-scenario", "0.0.1-SNAPSHOT");
61          //KieContainer kContainer = kieServices.newKieContainer(
                releaseId);
62
63          Results verifyResults = kContainer.verify();
64          for (Message m : verifyResults.getMessages()) {
65              LOG.info("{}", m);
66          }
67
68          LOG.info("Creating kieBase");
```

```
69
70          KieBaseConfiguration config = KieServices.Factory.get().
               newKieBaseConfiguration();
71          config.setOption(EventProcessingOption.STREAM);
72          KieBase kieBase = kContainer.newKieBase(config);
73
74          LOG.info("There should be rules: ");
75          for ( KiePackage kp : kieBase.getKiePackages() ) {
76              for (Rule rule : kp.getRules()) {
77                  LOG.info("kp " + kp + " rule " + rule.getName());
78              }
79          }
80
81          LOG.info("Creating kieSession");
82          KieSession session = kieBase.newKieSession();
83
84          SceneApplication app = new SceneApplication("botox-drugs-
               scenario", session);
85
86          session.addEventListener(new SCENESessionListener());
87
88
89          final RuleEngineThread ruleEngineThread = new RuleEngineThread
               (session);
90          ruleEngineThread.start();
91
92          LOG.info("Now running data");
93
94          //Criar loops para simular dados em 5 sensores diferentes! So
               necessrios 5 sensores.
95          Sensor sensor1 = new Sensor(1, "Geladeira 1");
96          session.insert(sensor1);
97          Sensor sensor2 = new Sensor(2, "Geladeira 1");
98          session.insert(sensor2);
99          Sensor sensor3 = new Sensor(3, "Geladeira 1");
100         session.insert(sensor3);
101         Sensor sensor4 = new Sensor(4, "Geladeira 1");
102         session.insert(sensor4);
103         Sensor sensor5 = new Sensor(5, "Geladeira 1");
```

```
104             session.insert(sensor5);
105 
106             //Cada sensor possui uma lista de temperaturas
107             insertTemperatures(sensor1, /*insero dos intervalos de
                    temperatura da simulacao*/, session);
108             insertTemperatures(sensor2, /*insero dos intervalos de
                    temperatura da simulacao*/, session);
109             insertTemperatures(sensor3, /*insero dos intervalos de
                    temperatura da simulacao*/, session);
110             insertTemperatures(sensor4, /*insero dos intervalos de
                    temperatura da simulacao*/, session);
111             insertTemperatures(sensor5, /*insero dos intervalos de
                    temperatura da simulacao*/, session);
112 
113             LOG.info("Final checks");
114 
115             ExecutorService executor = Executors.newSingleThreadExecutor();
116             Future<String> future = executor.submit(new Task());
117 
118             //Deixa a thread principal rodando por 30 segundos, aps esse
                    tempo, ela morre!
119             try {
120                 System.out.println("Started..");
121                 System.out.println(future.get(30, TimeUnit.SECONDS));
122                 System.out.println("Finished!");
123             } catch (TimeoutException e) {
124                 future.cancel(true);
125                 System.out.println("Terminated!");
126             }
127 
128             executor.shutdownNow();
129 
130         }
131         catch (Throwable t) {
132             t.printStackTrace();
133         }
134 
135     }
```

```
136 }
137
138 class Task implements Callable<String> {
139     @Override
140     public String call() throws Exception {
141         while (!Thread.interrupted()) {
142         }
143         return "Ready";
144     }
145 }
```

B.6 Modelos de Processos de Negócio - Notifica ambas as partes

```
1
2 <?xml version="1.0" encoding="UTF-8"?>
3 <!-- origin at X=0.0 Y=0.0 -->
4 <bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:bpmn2="http://www.omg.org/spec/BPMN/20100524/MODEL" xmlns:bpmndi
    ="http://www.omg.org/spec/BPMN/20100524/DI" xmlns:dc="http://www.omg.
    org/spec/DD/20100524/DC" xmlns:di="http://www.omg.org/spec/DD
    /20100524/DI" xmlns:java="http://www.java.com/javaTypes" xmlns:tns="
    http://www.jboss.org/drools" xmlns="http://www.jboss.org/drools" xsi:
    schemaLocation="http://www.omg.org/spec/BPMN/20100524/MODEL BPMN20.xsd
     http://www.jboss.org/drools drools.xsd http://www.bpsim.org/schemas
    /1.0 bpsim.xsd" id="Definition" exporter="org.eclipse.bpmn2.modeler.
    core" exporterVersion="1.3.0.Final-v20160602-2145-B47"
    expressionLanguage="http://www.mvel.org/2.0" targetNamespace="http://
    www.jboss.org/drools" typeLanguage="http://www.java.com/javaTypes">
5   <bpmn2:process id="co.pextra.botox.notification" tns:packageName="
      defaultPackage" name="Notification" isExecutable="true" processType=
      "Private">
6     <bpmn2:startEvent id="_1">
7       <bpmn2:extensionElements>
8         <tns:metaData name="elementname">
9           <tns:metaValue><![CDATA[]]></tns:metaValue>
10        </tns:metaData>
11      </bpmn2:extensionElements>
12      <bpmn2:outgoing>SequenceFlow_1</bpmn2:outgoing>
13    </bpmn2:startEvent>
14    <bpmn2:inclusiveGateway id="InclusiveGateway_1" name="Temperatura >= 5
```

```
        a menos de 30 minutos" gatewayDirection="Diverging">
15      <bpmn2:incoming>SequenceFlow_3</bpmn2:incoming>
16      <bpmn2:outgoing>SequenceFlow_5</bpmn2:outgoing>
17      <bpmn2:outgoing>SequenceFlow_7</bpmn2:outgoing>
18    </bpmn2:inclusiveGateway>
19    <bpmn2:serviceTask id="ServiceTask_2" name="O freezer est aberto">
20      <bpmn2:extensionElements>
21        <tns:metaData name="elementname">
22          <tns:metaValue><![CDATA[O freezer est aberto]]></tns:metaValue>
23        </tns:metaData>
24      </bpmn2:extensionElements>
25      <bpmn2:incoming>SequenceFlow_1</bpmn2:incoming>
26      <bpmn2:outgoing>SequenceFlow_3</bpmn2:outgoing>
27    </bpmn2:serviceTask>
28    <bpmn2:sequenceFlow id="SequenceFlow_1" tns:priority="1" sourceRef="_1
        " targetRef="ServiceTask_2"/>
29    <bpmn2:sequenceFlow id="SequenceFlow_3" tns:priority="1" sourceRef="
        ServiceTask_2" targetRef="InclusiveGateway_1"/>
30    <bpmn2:sendTask id="SendTask_1" name="Envia e-mail para a secretria">
31      <bpmn2:extensionElements>
32        <tns:metaData name="elementname">
33          <tns:metaValue><![CDATA[Envia e-mail para a secretria]]></tns:
              metaValue>
34        </tns:metaData>
35      </bpmn2:extensionElements>
36      <bpmn2:incoming>SequenceFlow_5</bpmn2:incoming>
37      <bpmn2:outgoing>SequenceFlow_9</bpmn2:outgoing>
38    </bpmn2:sendTask>
39    <bpmn2:sequenceFlow id="SequenceFlow_5" tns:priority="1" name="
        Notifica a secretria" sourceRef="InclusiveGateway_1" targetRef="
        SendTask_1"/>
40    <bpmn2:serviceTask id="ServiceTask_3" name="Envia e-mail para o mdico"
        >
41      <bpmn2:extensionElements>
42        <tns:metaData name="elementname">
43          <tns:metaValue><![CDATA[Envia e-mail para o mdico]]></tns:
              metaValue>
44        </tns:metaData>
45      </bpmn2:extensionElements>
```

```
46      <bpmn2:incoming>SequenceFlow_7</bpmn2:incoming>
47      <bpmn2:outgoing>SequenceFlow_10</bpmn2:outgoing>
48    </bpmn2:serviceTask>
49    <bpmn2:sequenceFlow id="SequenceFlow_7" tns:priority="1" name="
        Notifica o mdico" sourceRef="InclusiveGateway_1" targetRef="
        ServiceTask_3"/>
50    <bpmn2:inclusiveGateway id="InclusiveGateway_2" name="Envio realizado"
         gatewayDirection="Converging">
51      <bpmn2:incoming>SequenceFlow_9</bpmn2:incoming>
52      <bpmn2:incoming>SequenceFlow_10</bpmn2:incoming>
53      <bpmn2:outgoing>SequenceFlow_11</bpmn2:outgoing>
54    </bpmn2:inclusiveGateway>
55    <bpmn2:sequenceFlow id="SequenceFlow_9" tns:priority="1" name="E-mail
        enviado para a secretria" sourceRef="SendTask_1" targetRef="
        InclusiveGateway_2"/>
56    <bpmn2:sequenceFlow id="SequenceFlow_10" tns:priority="1" name="E-mail
         enviado para o mdico" sourceRef="ServiceTask_3" targetRef="
        InclusiveGateway_2"/>
57    <bpmn2:endEvent id="EndEvent_2">
58      <bpmn2:extensionElements>
59        <tns:metaData name="elementname">
60          <tns:metaValue><![CDATA[]]></tns:metaValue>
61        </tns:metaData>
62      </bpmn2:extensionElements>
63      <bpmn2:incoming>SequenceFlow_11</bpmn2:incoming>
64    </bpmn2:endEvent>
65    <bpmn2:sequenceFlow id="SequenceFlow_11" tns:priority="1" sourceRef="
        InclusiveGateway_2" targetRef="EndEvent_2"/>
66  </bpmn2:process>
67  <bpmndi:BPMNDiagram id="BPMNDiagram_1">
68    <bpmndi:BPMNPlane id="BPMNPlane_Process_1" bpmnElement="co.pextra.
        botox.notification">
69      <bpmndi:BPMNShape id="BPMNShape_StartEvent_1" bpmnElement="_1">
70        <dc:Bounds height="36.0" width="36.0" x="70.0" y="265.0"/>
71        <bpmndi:BPMNLabel id="BPMNLabel_1"/>
72      </bpmndi:BPMNShape>
73      <bpmndi:BPMNShape id="BPMNShape_InclusiveGateway_1" bpmnElement="
          InclusiveGateway_1" isMarkerVisible="true">
74        <dc:Bounds height="50.0" width="50.0" x="367.0" y="258.0"/>
```

```
75        <bpmndi:BPMNLabel>
76          <dc:Bounds height="45.0" width="80.0" x="352.0" y="308.0"/>
77        </bpmndi:BPMNLabel>
78      </bpmndi:BPMNShape>
79      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_2" bpmnElement="
          ServiceTask_2" isExpanded="true">
80        <dc:Bounds height="56.0" width="131.0" x="150.0" y="255.0"/>
81        <bpmndi:BPMNLabel>
82          <dc:Bounds height="15.0" width="114.0" x="158.0" y="275.0"/>
83        </bpmndi:BPMNLabel>
84      </bpmndi:BPMNShape>
85      <bpmndi:BPMNShape id="BPMNShape_SendTask_1" bpmnElement="SendTask_1"
           isExpanded="true">
86        <dc:Bounds height="65.0" width="125.0" x="470.0" y="150.0"/>
87        <bpmndi:BPMNLabel>
88          <dc:Bounds height="30.0" width="108.0" x="478.0" y="167.0"/>
89        </bpmndi:BPMNLabel>
90      </bpmndi:BPMNShape>
91      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_3" bpmnElement="
          ServiceTask_3" isExpanded="true">
92        <dc:Bounds height="66.0" width="146.0" x="465.0" y="345.0"/>
93        <bpmndi:BPMNLabel>
94          <dc:Bounds height="30.0" width="110.0" x="483.0" y="363.0"/>
95        </bpmndi:BPMNLabel>
96      </bpmndi:BPMNShape>
97      <bpmndi:BPMNShape id="BPMNShape_InclusiveGateway_2" bpmnElement="
          InclusiveGateway_2" isMarkerVisible="true">
98        <dc:Bounds height="50.0" width="50.0" x="680.0" y="258.0"/>
99        <bpmndi:BPMNLabel>
100         <dc:Bounds height="30.0" width="55.0" x="678.0" y="308.0"/>
101       </bpmndi:BPMNLabel>
102     </bpmndi:BPMNShape>
103     <bpmndi:BPMNShape id="BPMNShape_EndEvent_2" bpmnElement="EndEvent_2"
          >
104       <dc:Bounds height="36.0" width="36.0" x="807.0" y="265.0"/>
105       <bpmndi:BPMNLabel/>
106     </bpmndi:BPMNShape>
107     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_1" bpmnElement="
          SequenceFlow_1" sourceElement="BPMNShape_StartEvent_1"
```

```
        targetElement="BPMNShape_ServiceTask_2">
108       <di:waypoint xsi:type="dc:Point" x="106.0" y="283.0"/>
109       <di:waypoint xsi:type="dc:Point" x="128.0" y="283.0"/>
110       <di:waypoint xsi:type="dc:Point" x="150.0" y="283.0"/>
111       <bpmndi:BPMNLabel/>
112     </bpmndi:BPMNEdge>
113     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_3" bpmnElement="
          SequenceFlow_3" sourceElement="BPMNShape_ServiceTask_2"
          targetElement="BPMNShape_InclusiveGateway_1">
114       <di:waypoint xsi:type="dc:Point" x="281.0" y="283.0"/>
115       <di:waypoint xsi:type="dc:Point" x="324.0" y="283.0"/>
116       <di:waypoint xsi:type="dc:Point" x="367.0" y="283.0"/>
117       <bpmndi:BPMNLabel/>
118     </bpmndi:BPMNEdge>
119     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_5" bpmnElement="
          SequenceFlow_5" sourceElement="BPMNShape_InclusiveGateway_1"
          targetElement="BPMNShape_SendTask_1">
120       <di:waypoint xsi:type="dc:Point" x="392.0" y="258.0"/>
121       <di:waypoint xsi:type="dc:Point" x="392.0" y="182.0"/>
122       <di:waypoint xsi:type="dc:Point" x="470.0" y="182.0"/>
123       <bpmndi:BPMNLabel>
124         <dc:Bounds height="30.0" width="56.0" x="366.0" y="183.0"/>
125       </bpmndi:BPMNLabel>
126     </bpmndi:BPMNEdge>
127     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_7" bpmnElement="
          SequenceFlow_7" sourceElement="BPMNShape_InclusiveGateway_1"
          targetElement="BPMNShape_ServiceTask_3">
128       <di:waypoint xsi:type="dc:Point" x="392.0" y="308.0"/>
129       <di:waypoint xsi:type="dc:Point" x="392.0" y="378.0"/>
130       <di:waypoint xsi:type="dc:Point" x="465.0" y="378.0"/>
131       <bpmndi:BPMNLabel>
132         <dc:Bounds height="30.0" width="56.0" x="367.0" y="379.0"/>
133       </bpmndi:BPMNLabel>
134     </bpmndi:BPMNEdge>
135     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_9" bpmnElement="
          SequenceFlow_9" sourceElement="BPMNShape_SendTask_1"
          targetElement="BPMNShape_InclusiveGateway_2">
136       <di:waypoint xsi:type="dc:Point" x="595.0" y="182.0"/>
137       <di:waypoint xsi:type="dc:Point" x="705.0" y="182.0"/>
```

```
138         <di:waypoint xsi:type="dc:Point" x="705.0" y="258.0"/>
139         <bpmndi:BPMNLabel>
140           <dc:Bounds height="45.0" width="76.0" x="651.0" y="183.0"/>
141         </bpmndi:BPMNLabel>
142       </bpmndi:BPMNEdge>
143       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_10" bpmnElement="
            SequenceFlow_10" sourceElement="BPMNShape_ServiceTask_3"
            targetElement="BPMNShape_InclusiveGateway_2">
144         <di:waypoint xsi:type="dc:Point" x="611.0" y="378.0"/>
145         <di:waypoint xsi:type="dc:Point" x="705.0" y="378.0"/>
146         <di:waypoint xsi:type="dc:Point" x="705.0" y="308.0"/>
147         <bpmndi:BPMNLabel>
148           <dc:Bounds height="45.0" width="76.0" x="656.0" y="379.0"/>
149         </bpmndi:BPMNLabel>
150       </bpmndi:BPMNEdge>
151       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_11" bpmnElement="
            SequenceFlow_11" sourceElement="BPMNShape_InclusiveGateway_2"
            targetElement="BPMNShape_EndEvent_2">
152         <di:waypoint xsi:type="dc:Point" x="730.0" y="283.0"/>
153         <di:waypoint xsi:type="dc:Point" x="768.0" y="283.0"/>
154         <di:waypoint xsi:type="dc:Point" x="807.0" y="283.0"/>
155         <bpmndi:BPMNLabel/>
156       </bpmndi:BPMNEdge>
157     </bpmndi:BPMNPlane>
158   </bpmndi:BPMNDiagram>
159 </bpmn2:definitions>
```

B.7 Modelos de Processos de Negócio - Notifica ambas as partes após um intervalo de tempo

```
1
2 <?xml version="1.0" encoding="UTF-8"?>
3 <!-- origin at X=0.0 Y=0.0 -->
4 <bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:bpmn2="http://www.omg.org/spec/BPMN/20100524/MODEL" xmlns:bpmndi
    ="http://www.omg.org/spec/BPMN/20100524/DI" xmlns:dc="http://www.omg.
    org/spec/DD/20100524/DC" xmlns:di="http://www.omg.org/spec/DD
    /20100524/DI" xmlns:java="http://www.java.com/javaTypes" xmlns:tns="
    http://www.jboss.org/drools" xmlns="http://www.jboss.org/drools" xsi:
```

```
  schemaLocation="http://www.omg.org/spec/BPMN/20100524/MODEL BPMN20.xsd
   http://www.jboss.org/drools drools.xsd http://www.bpsim.org/schemas
  /1.0 bpsim.xsd" id="Definition" exporter="org.eclipse.bpmn2.modeler.
  core" exporterVersion="1.3.0.Final-v20160602-2145-B47"
  expressionLanguage="http://www.mvel.org/2.0" targetNamespace="http://
  www.jboss.org/drools" typeLanguage="http://www.java.com/javaTypes">
<bpmn2:process id="co.pextra.botox.notification" tns:packageName="
    defaultPackage" name="Notification" isExecutable="true" processType=
    "Private">
  <bpmn2:startEvent id="_1">
    <bpmn2:extensionElements>
      <tns:metaData name="elementname">
        <tns:metaValue><![CDATA[]]></tns:metaValue>
      </tns:metaData>
    </bpmn2:extensionElements>
    <bpmn2:outgoing>SequenceFlow_1</bpmn2:outgoing>
  </bpmn2:startEvent>
  <bpmn2:endEvent id="EndEvent_1">
    <bpmn2:extensionElements>
      <tns:metaData name="elementname">
        <tns:metaValue><![CDATA[]]></tns:metaValue>
      </tns:metaData>
    </bpmn2:extensionElements>
    <bpmn2:incoming>SequenceFlow_8</bpmn2:incoming>
  </bpmn2:endEvent>
  <bpmn2:serviceTask id="ServiceTask_1" name="Verifica o estado do
     freezer">
    <bpmn2:extensionElements>
      <tns:metaData name="elementname">
        <tns:metaValue><![CDATA[Verifica o estado do freezer]]></tns:
           metaValue>
      </tns:metaData>
    </bpmn2:extensionElements>
    <bpmn2:incoming>SequenceFlow_1</bpmn2:incoming>
    <bpmn2:outgoing>SequenceFlow_3</bpmn2:outgoing>
  </bpmn2:serviceTask>
  <bpmn2:sequenceFlow id="SequenceFlow_1" tns:priority="1" sourceRef="_1
     " targetRef="ServiceTask_1"/>
  <bpmn2:parallelGateway id="ParallelGateway_1" name="Notifica ambas as
```

```
      partes" gatewayDirection="Diverging">
33      <bpmn2:incoming>SequenceFlow_3</bpmn2:incoming>
34      <bpmn2:outgoing>SequenceFlow_5</bpmn2:outgoing>
35      <bpmn2:outgoing>SequenceFlow_6</bpmn2:outgoing>
36    </bpmn2:parallelGateway>
37    <bpmn2:sequenceFlow id="SequenceFlow_3" tns:priority="1" sourceRef="
        ServiceTask_1" targetRef="ParallelGateway_1"/>
38    <bpmn2:serviceTask id="ServiceTask_2" name="Envia e-mail para o mdico"
        >
39      <bpmn2:extensionElements>
40        <tns:metaData name="elementname">
41          <tns:metaValue><![CDATA[Envia e-mail para o mdico]]></tns:
              metaValue>
42        </tns:metaData>
43      </bpmn2:extensionElements>
44      <bpmn2:incoming>SequenceFlow_5</bpmn2:incoming>
45      <bpmn2:outgoing>SequenceFlow_8</bpmn2:outgoing>
46    </bpmn2:serviceTask>
47    <bpmn2:sendTask id="SendTask_1" name="Envia e-mail para a secretria">
48      <bpmn2:extensionElements>
49        <tns:metaData name="elementname">
50          <tns:metaValue><![CDATA[Envia e-mail para a secretria]]></tns:
              metaValue>
51        </tns:metaData>
52      </bpmn2:extensionElements>
53      <bpmn2:incoming>SequenceFlow_6</bpmn2:incoming>
54      <bpmn2:outgoing>SequenceFlow_7</bpmn2:outgoing>
55    </bpmn2:sendTask>
56    <bpmn2:sequenceFlow id="SequenceFlow_5" tns:priority="1" name="
        Notifica o mdico" sourceRef="ParallelGateway_1" targetRef="
        ServiceTask_2"/>
57    <bpmn2:sequenceFlow id="SequenceFlow_6" tns:priority="1" name="
        Notifica a secretria" sourceRef="ParallelGateway_1" targetRef="
        SendTask_1"/>
58    <bpmn2:endEvent id="EndEvent_2">
59      <bpmn2:extensionElements>
60        <tns:metaData name="elementname">
61          <tns:metaValue><![CDATA[]]></tns:metaValue>
62        </tns:metaData>
```

```
63      </bpmn2:extensionElements>
64      <bpmn2:incoming>SequenceFlow_7</bpmn2:incoming>
65    </bpmn2:endEvent>
66    <bpmn2:sequenceFlow id="SequenceFlow_7" tns:priority="1" sourceRef="
        SendTask_1" targetRef="EndEvent_2"/>
67    <bpmn2:sequenceFlow id="SequenceFlow_8" tns:priority="1" sourceRef="
        ServiceTask_2" targetRef="EndEvent_1"/>
68  </bpmn2:process>
69  <bpmndi:BPMNDiagram id="BPMNDiagram_1">
70    <bpmndi:BPMNPlane id="BPMNPlane_Process_1" bpmnElement="co.pextra.
        botox.notification">
71      <bpmndi:BPMNShape id="BPMNShape_StartEvent_1" bpmnElement="_1">
72        <dc:Bounds height="36.0" width="36.0" x="120.0" y="262.0"/>
73        <bpmndi:BPMNLabel id="BPMNLabel_1"/>
74      </bpmndi:BPMNShape>
75      <bpmndi:BPMNShape id="BPMNShape_EndEvent_1" bpmnElement="EndEvent_1"
          >
76        <dc:Bounds height="36.0" width="36.0" x="790.0" y="142.0"/>
77        <bpmndi:BPMNLabel id="BPMNLabel_2"/>
78      </bpmndi:BPMNShape>
79      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_1" bpmnElement="
          ServiceTask_1" isExpanded="true">
80        <dc:Bounds height="50.0" width="110.0" x="220.0" y="255.0"/>
81        <bpmndi:BPMNLabel>
82          <dc:Bounds height="30.0" width="97.0" x="226.0" y="265.0"/>
83        </bpmndi:BPMNLabel>
84      </bpmndi:BPMNShape>
85      <bpmndi:BPMNShape id="BPMNShape_ParallelGateway_1" bpmnElement="
          ParallelGateway_1" isMarkerVisible="true">
86        <dc:Bounds height="50.0" width="50.0" x="396.0" y="255.0"/>
87        <bpmndi:BPMNLabel>
88          <dc:Bounds height="45.0" width="59.0" x="392.0" y="305.0"/>
89        </bpmndi:BPMNLabel>
90      </bpmndi:BPMNShape>
91      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_2" bpmnElement="
          ServiceTask_2" isExpanded="true">
92        <dc:Bounds height="66.0" width="126.0" x="554.0" y="127.0"/>
93        <bpmndi:BPMNLabel>
94          <dc:Bounds height="30.0" width="110.0" x="562.0" y="145.0"/>
```

```
95        </bpmndi:BPMNLabel>
96      </bpmndi:BPMNShape>
97      <bpmndi:BPMNShape id="BPMNShape_SendTask_1" bpmnElement="SendTask_1"
            isExpanded="true">
98        <dc:Bounds height="66.0" width="126.0" x="545.0" y="375.0"/>
99        <bpmndi:BPMNLabel>
100         <dc:Bounds height="30.0" width="108.0" x="554.0" y="393.0"/>
101       </bpmndi:BPMNLabel>
102     </bpmndi:BPMNShape>
103     <bpmndi:BPMNShape id="BPMNShape_EndEvent_2" bpmnElement="EndEvent_2"
            >
104       <dc:Bounds height="36.0" width="36.0" x="790.0" y="390.0"/>
105       <bpmndi:BPMNLabel/>
106     </bpmndi:BPMNShape>
107     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_1" bpmnElement="
            SequenceFlow_1" sourceElement="BPMNShape_StartEvent_1"
            targetElement="BPMNShape_ServiceTask_1">
108       <di:waypoint xsi:type="dc:Point" x="156.0" y="280.0"/>
109       <di:waypoint xsi:type="dc:Point" x="188.0" y="280.0"/>
110       <di:waypoint xsi:type="dc:Point" x="220.0" y="280.0"/>
111       <bpmndi:BPMNLabel/>
112     </bpmndi:BPMNEdge>
113     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_3" bpmnElement="
            SequenceFlow_3" sourceElement="BPMNShape_ServiceTask_1"
            targetElement="BPMNShape_ParallelGateway_1">
114       <di:waypoint xsi:type="dc:Point" x="330.0" y="280.0"/>
115       <di:waypoint xsi:type="dc:Point" x="363.0" y="280.0"/>
116       <di:waypoint xsi:type="dc:Point" x="396.0" y="280.0"/>
117       <bpmndi:BPMNLabel/>
118     </bpmndi:BPMNEdge>
119     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_5" bpmnElement="
            SequenceFlow_5" sourceElement="BPMNShape_ParallelGateway_1"
            targetElement="BPMNShape_ServiceTask_2">
120       <di:waypoint xsi:type="dc:Point" x="421.0" y="255.0"/>
121       <di:waypoint xsi:type="dc:Point" x="421.0" y="160.0"/>
122       <di:waypoint xsi:type="dc:Point" x="554.0" y="160.0"/>
123       <bpmndi:BPMNLabel>
124         <dc:Bounds height="30.0" width="56.0" x="413.0" y="161.0"/>
125       </bpmndi:BPMNLabel>
```

```
126       </bpmndi:BPMNEdge>
127       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_6" bpmnElement="
              SequenceFlow_6" sourceElement="BPMNShape_ParallelGateway_1"
              targetElement="BPMNShape_SendTask_1">
128         <di:waypoint xsi:type="dc:Point" x="421.0" y="305.0"/>
129         <di:waypoint xsi:type="dc:Point" x="421.0" y="408.0"/>
130         <di:waypoint xsi:type="dc:Point" x="545.0" y="408.0"/>
131         <bpmndi:BPMNLabel>
132           <dc:Bounds height="30.0" width="56.0" x="405.0" y="409.0"/>
133         </bpmndi:BPMNLabel>
134       </bpmndi:BPMNEdge>
135       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_7" bpmnElement="
              SequenceFlow_7" sourceElement="BPMNShape_SendTask_1"
              targetElement="BPMNShape_EndEvent_2">
136         <di:waypoint xsi:type="dc:Point" x="671.0" y="408.0"/>
137         <di:waypoint xsi:type="dc:Point" x="730.0" y="408.0"/>
138         <di:waypoint xsi:type="dc:Point" x="790.0" y="408.0"/>
139         <bpmndi:BPMNLabel/>
140       </bpmndi:BPMNEdge>
141       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_8" bpmnElement="
              SequenceFlow_8" sourceElement="BPMNShape_ServiceTask_2"
              targetElement="BPMNShape_EndEvent_1">
142         <di:waypoint xsi:type="dc:Point" x="680.0" y="160.0"/>
143         <di:waypoint xsi:type="dc:Point" x="735.0" y="160.0"/>
144         <di:waypoint xsi:type="dc:Point" x="790.0" y="160.0"/>
145         <bpmndi:BPMNLabel/>
146       </bpmndi:BPMNEdge>
147     </bpmndi:BPMNPlane>
148   </bpmndi:BPMNDiagram>
149 </bpmn2:definitions>
```

B.8 Modelos de Processos de Negócio - Modelo Complexo

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- origin at X=0.0 Y=0.0 -->
3 <bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:bpmn2="http://www.omg.org/spec/BPMN/20100524/MODEL" xmlns:bpmndi
    ="http://www.omg.org/spec/BPMN/20100524/DI" xmlns:dc="http://www.omg.
    org/spec/DD/20100524/DC" xmlns:di="http://www.omg.org/spec/DD
    /20100524/DI" xmlns:java="http://www.java.com/javaTypes" xmlns:tns="
```

```
  http://www.jboss.org/drools" xmlns="http://www.jboss.org/drools" xsi:
  schemaLocation="http://www.omg.org/spec/BPMN/20100524/MODEL BPMN20.xsd
   http://www.jboss.org/drools drools.xsd http://www.bpsim.org/schemas
  /1.0 bpsim.xsd" id="Definition" exporter="org.eclipse.bpmn2.modeler.
  core" exporterVersion="1.3.0.Final-v20160602-2145-B47"
  expressionLanguage="http://www.mvel.org/2.0" targetNamespace="http://
  www.jboss.org/drools" typeLanguage="http://www.java.com/javaTypes">
4 <bpmn2:itemDefinition id="ItemDefinition_23" isCollection="false"
    structureRef="Integer"/>
5 <bpmn2:itemDefinition id="ItemDefinition_99" isCollection="false"
    structureRef="Boolean"/>
6 <bpmn2:itemDefinition id="ItemDefinition_45" isCollection="false"
    structureRef="com.medicine.control.handler.MailWithAuthentication"/>
7 <bpmn2:itemDefinition id="ItemDefinition_67" isCollection="false"
    structureRef="com.medicine.control.handler.CallRestfulWebService"/>
8 <bpmn2:itemDefinition id="ItemDefinition_61" isCollection="false"
    structureRef="com.medicine.control.handler.VerificationCurrentTime"
    />
9 <bpmn2:itemDefinition id="ItemDefinition_348" isCollection="false"
    structureRef="Object"/>
10 <bpmn2:itemDefinition id="ItemDefinition_505" isCollection="false"
    structureRef="java.lang.Long">
11   <bpmn2:documentation id="Documentation_196">long</bpmn2:documentation>
12 </bpmn2:itemDefinition>
13 <bpmn2:signal id="Signal_1" name="Signal 1"/>
14 <bpmn2:process id="medicine.control.ProcessBotox" tns:packageName="
    defaultPackage" name="ProcessBotox" isExecutable="true" processType=
    "Private">
15   <bpmn2:extensionElements>
16     <tns:import name="com.medicine.control.handler.
        MailWithAuthentication"/>
17     <tns:import name="com.medicine.control.handler.CallRestfulWebService
        "/>
18     <tns:import name="com.medicine.control.handler.
        VerificationCurrentTime"/>
19     <tns:import name="java.lang.Long"/>
20   </bpmn2:extensionElements>
21   <bpmn2:property id="count" itemSubjectRef="ItemDefinition_23" name="
      count"/>
```

```
22    <bpmn2:property id="itsON" itemSubjectRef="ItemDefinition_99" name="
         itsON"/>
23    <bpmn2:property id="temperature" itemSubjectRef="ItemDefinition_23"
         name="temperature"/>
24    <bpmn2:property id="sendMessageTemp" itemSubjectRef="ItemDefinition_99
         " name="sendMessageTemp"/>
25    <bpmn2:property id="alertTemp" itemSubjectRef="ItemDefinition_99" name
         ="alertTemp"/>
26    <bpmn2:property id="notification" itemSubjectRef="ItemDefinition_45"
         name="notification"/>
27    <bpmn2:property id="ws" itemSubjectRef="ItemDefinition_67" name="ws"/>
28    <bpmn2:property id="CurrentTime" itemSubjectRef="ItemDefinition_61"
         name="CurrentTime"/>
29    <bpmn2:property id="CurrentTimestamp" itemSubjectRef="
         ItemDefinition_505" name="CurrentTimestamp"/>
30    <bpmn2:scriptTask id="ScriptTask_2" name="Verifica se o equipamento
         esta ligado" scriptFormat="http://www.java.com/java">
31      <bpmn2:extensionElements>
32        <tns:metaData name="elementname">
33          <tns:metaValue><![CDATA[Verifica se o equipamento esta ligado
               ]]></tns:metaValue>
34        </tns:metaData>
35      </bpmn2:extensionElements>
36      <bpmn2:incoming>SequenceFlow_22</bpmn2:incoming>
37      <bpmn2:outgoing>SequenceFlow_9</bpmn2:outgoing>
38      <bpmn2:script>kcontext.setVariable(&amp;quot;itsON&amp;quot;, ws.getItsON(&
           amp;quot;http://localhost:3000/itsON&amp;quot;));</bpmn2:script>
39    </bpmn2:scriptTask>
40    <bpmn2:exclusiveGateway id="ExclusiveGateway_10" name="Qual
         temperatura?" gatewayDirection="Diverging">
41      <bpmn2:incoming>SequenceFlow_8</bpmn2:incoming>
42      <bpmn2:outgoing>SequenceFlow_23</bpmn2:outgoing>
43      <bpmn2:outgoing>SequenceFlow_15</bpmn2:outgoing>
44    </bpmn2:exclusiveGateway>
45    <bpmn2:scriptTask id="ScriptTask_11" name="Verifica se j existe
         mensagem de alerta enviada ao medico h menos de 30 minutos"
         scriptFormat="http://www.java.com/java">
46      <bpmn2:extensionElements>
47        <tns:metaData name="elementname">
```

```
48          <tns:metaValue><![CDATA[Verifica se j existe mensagem de alerta
               enviada ao medico h menos de 30 minutos]]></tns:metaValue>
49        </tns:metaData>
50      </bpmn2:extensionElements>
51      <bpmn2:incoming>SequenceFlow_23</bpmn2:incoming>
52      <bpmn2:outgoing>SequenceFlow_7</bpmn2:outgoing>
53      <bpmn2:script>System.out.println(&quot;Temperatura: &quot; +
           temperature + &quot; alertTemp: &quot; + alertTemp);&#xD;
54 System.out.println(&quot;Verifica se j existe mensagem de alerta enviada
      ao medico h menos de 30 minutos&quot;);&#xD;
55 &#xD;
56 if(alertTemp == false){&#xD;
57     kcontext.setVariable(&quot;alertTemp&quot;, true);&#xD;
58     kcontext.setVariable(&quot;sendMessageTemp&quot;, true);&#xD;
59 }&#xD;
60 else{//alertTemp == true&#xD;
61     //verifica se a mensagem foi enviada a menos de 30 minutos.&#xD;
62     if(CurrentTime.isThirtyMinutes(CurrentTimestamp) == true){&#xD;
63         kcontext.setVariable(&quot;sendMessageTemp&quot;, true);&#xD;
64     }&#xD;
65     else{&#xD;
66         kcontext.setVariable(&quot;sendMessageTemp&quot;, false);&#xD;
67     }&#xD;
68 }&#xD;
69 </bpmn2:script>
70     </bpmn2:scriptTask>
71     <bpmn2:sequenceFlow id="SequenceFlow_23" tns:priority="1" name="
          Temperatura > -5 C" sourceRef="ExclusiveGateway_10" targetRef="
          ScriptTask_11">
72       <bpmn2:conditionExpression xsi:type="bpmn2:tFormalExpression" id="
            FormalExpression_8" language="http://www.java.com/java">return
            temperature > -5;</bpmn2:conditionExpression>
73     </bpmn2:sequenceFlow>
74     <bpmn2:scriptTask id="ScriptTask_3" name="Avisar mdico sobre
          temperatura regular" scriptFormat="http://www.java.com/java">
75       <bpmn2:extensionElements>
76         <tns:metaData name="elementname">
77           <tns:metaValue><![CDATA[Avisar mdico sobre temperatura regular
               ]]></tns:metaValue>
```

```
78          </tns:metaData>
79        </bpmn2:extensionElements>
80        <bpmn2:incoming>SequenceFlow_15</bpmn2:incoming>
81        <bpmn2:outgoing>SequenceFlow_14</bpmn2:outgoing>
82        <bpmn2:script>System.out.println(&quot;Temperatura: &quot; +
              temperature);&#xD;
83  System.out.println(&quot;Avisar mdico sobre temperatura regular?&quot;)
      ;&#xD;
84  if(alertTemp == true){&#xD;
85      kcontext.setVariable(&quot;alertTemp&quot;, false);&#xD;
86      notification.sendEmail(&quot;Temperatura estabilizada&quot;, &quot;A
          temperatura do freezer voltou ao normal!&quot;);&#xD;
87      System.out.println(&quot;Sim&quot;);&#xD;
88  }&#xD;
89  else{&#xD;
90      System.out.println(&quot;No&quot;);&#xD;
91  }</bpmn2:script>
92      </bpmn2:scriptTask>
93      <bpmn2:exclusiveGateway id="ExclusiveGateway_5" name="Avisar ao mdico?
          " gatewayDirection="Diverging" default="SequenceFlow_12">
94        <bpmn2:incoming>SequenceFlow_7</bpmn2:incoming>
95        <bpmn2:outgoing>SequenceFlow_6</bpmn2:outgoing>
96        <bpmn2:outgoing>SequenceFlow_12</bpmn2:outgoing>
97      </bpmn2:exclusiveGateway>
98      <bpmn2:scriptTask id="ScriptTask_6" name="Avisar mdico sobre elevao de
           temperatura" scriptFormat="http://www.java.com/java">
99        <bpmn2:extensionElements>
100         <tns:metaData name="elementname">
101           <tns:metaValue><![CDATA[Avisar mdico sobre elevao de temperatura
                ]]></tns:metaValue>
102         </tns:metaData>
103       </bpmn2:extensionElements>
104       <bpmn2:incoming>SequenceFlow_6</bpmn2:incoming>
105       <bpmn2:outgoing>SequenceFlow_13</bpmn2:outgoing>
106       <bpmn2:script>System.out.println(&quot;mdico avisado sobre elevao de
            temperatura&quot;);&#xD;
107 &#xD;
108 notification.sendEmail(&quot;Elevao de temperatura [URGENTE]&quot;, &quot
      ;Foi detectada a elevao de temperatura do freezer, recomendado
```

```
   verificar!&amp;quot;);&amp;#xD;
109 kcontext.setVariable(&amp;quot;CurrentTimestamp&amp;quot;, System.
   currentTimeMillis());</bpmn2:script>
110    </bpmn2:scriptTask>
111    <bpmn2:sequenceFlow id="SequenceFlow_6" tns:priority="1" name="Sim"
          sourceRef="ExclusiveGateway_5" targetRef="ScriptTask_6">
112      <bpmn2:conditionExpression xsi:type="bpmn2:tFormalExpression" id="
            FormalExpression_7" language="http://www.java.com/java">return (
            alertTemp == true &amp;amp;&amp;amp; sendMessageTemp == true);</bpmn2:
            conditionExpression>
113    </bpmn2:sequenceFlow>
114    <bpmn2:sequenceFlow id="SequenceFlow_7" tns:priority="1" sourceRef="
          ScriptTask_11" targetRef="ExclusiveGateway_5"/>
115    <bpmn2:exclusiveGateway id="ExclusiveGateway_7" gatewayDirection="
          Converging">
116      <bpmn2:incoming>SequenceFlow_12</bpmn2:incoming>
117      <bpmn2:incoming>SequenceFlow_13</bpmn2:incoming>
118      <bpmn2:incoming>SequenceFlow_14</bpmn2:incoming>
119      <bpmn2:outgoing>SequenceFlow_1</bpmn2:outgoing>
120    </bpmn2:exclusiveGateway>
121    <bpmn2:sequenceFlow id="SequenceFlow_12" tns:priority="1" name="No"
          sourceRef="ExclusiveGateway_5" targetRef="ExclusiveGateway_7"/>
122    <bpmn2:sequenceFlow id="SequenceFlow_13" tns:priority="1" sourceRef="
          ScriptTask_6" targetRef="ExclusiveGateway_7"/>
123    <bpmn2:sequenceFlow id="SequenceFlow_14" tns:priority="1" sourceRef="
          ScriptTask_3" targetRef="ExclusiveGateway_7"/>
124    <bpmn2:exclusiveGateway id="ExclusiveGateway_9" gatewayDirection="
          Converging">
125      <bpmn2:incoming>SequenceFlow_4</bpmn2:incoming>
126      <bpmn2:incoming>SequenceFlow_17</bpmn2:incoming>
127      <bpmn2:outgoing>SequenceFlow_22</bpmn2:outgoing>
128    </bpmn2:exclusiveGateway>
129    <bpmn2:sequenceFlow id="SequenceFlow_22" tns:priority="1" sourceRef="
          ExclusiveGateway_9" targetRef="ScriptTask_2"/>
130    <bpmn2:endEvent id="EndEvent_1" name="Desligado ou Removido">
131      <bpmn2:extensionElements>
132        <tns:metaData name="elementname">
133          <tns:metaValue><![CDATA[Desligado ou Removido]]></tns:metaValue>
134        </tns:metaData>
```

```
135       </bpmn2:extensionElements>
136       <bpmn2:incoming>SequenceFlow_18</bpmn2:incoming>
137     </bpmn2:endEvent>
138     <bpmn2:startEvent id="StartEvent_1" name="Inicio">
139       <bpmn2:extensionElements>
140         <tns:metaData name="elementname">
141           <tns:metaValue><![CDATA[Inicio]]></tns:metaValue>
142         </tns:metaData>
143       </bpmn2:extensionElements>
144       <bpmn2:outgoing>SequenceFlow_4</bpmn2:outgoing>
145     </bpmn2:startEvent>
146     <bpmn2:sequenceFlow id="SequenceFlow_4" tns:priority="1" sourceRef="
          StartEvent_1" targetRef="ExclusiveGateway_9"/>
147     <bpmn2:scriptTask id="ScriptTask_1" name="Ler temperatura do sensor
          usando webservice" scriptFormat="http://www.java.com/java">
148       <bpmn2:extensionElements>
149         <tns:metaData name="elementname">
150           <tns:metaValue><![CDATA[Ler temperatura do sensor usando
                webservice]]></tns:metaValue>
151         </tns:metaData>
152       </bpmn2:extensionElements>
153       <bpmn2:incoming>SequenceFlow_11</bpmn2:incoming>
154       <bpmn2:outgoing>SequenceFlow_8</bpmn2:outgoing>
155       <bpmn2:script>kcontext.setVariable(&quot;temperature&quot;, ws.
            getTemperature(&quot;http://localhost:3000/temperature&quot;));</
            bpmn2:script>
156     </bpmn2:scriptTask>
157     <bpmn2:sequenceFlow id="SequenceFlow_8" tns:priority="1" sourceRef="
          ScriptTask_1" targetRef="ExclusiveGateway_10"/>
158     <bpmn2:exclusiveGateway id="ExclusiveGateway_1" name="Est ligado?"
          gatewayDirection="Diverging">
159       <bpmn2:incoming>SequenceFlow_16</bpmn2:incoming>
160       <bpmn2:outgoing>SequenceFlow_10</bpmn2:outgoing>
161       <bpmn2:outgoing>SequenceFlow_11</bpmn2:outgoing>
162     </bpmn2:exclusiveGateway>
163     <bpmn2:sequenceFlow id="SequenceFlow_9" tns:priority="1" sourceRef="
          ScriptTask_2" targetRef="ScriptTask_4"/>
164     <bpmn2:sequenceFlow id="SequenceFlow_10" tns:priority="1" name="No"
          sourceRef="ExclusiveGateway_1" targetRef="ScriptTask_8">
```

```
165     <bpmn2:conditionExpression xsi:type="bpmn2:tFormalExpression" id="
            FormalExpression_9" language="http://www.java.com/java">return
            itsON == false;</bpmn2:conditionExpression>
166   </bpmn2:sequenceFlow>
167   <bpmn2:sequenceFlow id="SequenceFlow_11" tns:priority="1" name="Sim"
        sourceRef="ExclusiveGateway_1" targetRef="ScriptTask_1">
168     <bpmn2:conditionExpression xsi:type="bpmn2:tFormalExpression" id="
            FormalExpression_12" language="http://www.java.com/java">return
            itsON == true;</bpmn2:conditionExpression>
169   </bpmn2:sequenceFlow>
170   <bpmn2:sequenceFlow id="SequenceFlow_15" tns:priority="1" name="
        Temperatura &lt;= -5 C" sourceRef="ExclusiveGateway_10" targetRef=
        "ScriptTask_3">
171     <bpmn2:conditionExpression xsi:type="bpmn2:tFormalExpression" id="
            FormalExpression_1" language="http://www.java.com/java">return
            temperature &lt;= -5;</bpmn2:conditionExpression>
172   </bpmn2:sequenceFlow>
173   <bpmn2:scriptTask id="ScriptTask_4" name="Esta ligado?" scriptFormat="
        http://www.java.com/java">
174     <bpmn2:extensionElements>
175       <tns:metaData name="elementname">
176         <tns:metaValue><![CDATA[Esta ligado?]]></tns:metaValue>
177       </tns:metaData>
178     </bpmn2:extensionElements>
179     <bpmn2:incoming>SequenceFlow_9</bpmn2:incoming>
180     <bpmn2:outgoing>SequenceFlow_16</bpmn2:outgoing>
181     <bpmn2:script>System.out.println(&quot;\nEsta ligado?: &quot; +
            itsON);</bpmn2:script>
182   </bpmn2:scriptTask>
183   <bpmn2:sequenceFlow id="SequenceFlow_16" tns:priority="1" sourceRef="
        ScriptTask_4" targetRef="ExclusiveGateway_1"/>
184   <bpmn2:scriptTask id="ScriptTask_5" name="Hora antes de 30 segundos"
        scriptFormat="http://www.java.com/java">
185     <bpmn2:extensionElements>
186       <tns:metaData name="elementname">
187         <tns:metaValue><![CDATA[Hora antes de 30 segundos]]></tns:
              metaValue>
188       </tns:metaData>
189     </bpmn2:extensionElements>
```

```
190       <bpmn2:incoming>SequenceFlow_1</bpmn2:incoming>
191       <bpmn2:outgoing>SequenceFlow_2</bpmn2:outgoing>
192       <bpmn2:script>System.out.println(&quot;Antes de 30 segundos: &quot;
              +new java.util.Date(System.currentTimeMillis()));</bpmn2:script>
193     </bpmn2:scriptTask>
194     <bpmn2:scriptTask id="ScriptTask_7" name="Hora depois de 30 segundos"
            scriptFormat="http://www.java.com/java">
195       <bpmn2:extensionElements>
196         <tns:metaData name="elementname">
197           <tns:metaValue><![CDATA[Hora depois de 30 segundos]]></tns:
                metaValue>
198         </tns:metaData>
199       </bpmn2:extensionElements>
200       <bpmn2:incoming>SequenceFlow_5</bpmn2:incoming>
201       <bpmn2:outgoing>SequenceFlow_17</bpmn2:outgoing>
202       <bpmn2:script>System.out.println(&quot;Depois de 30 segundos: &quot;
              +new java.util.Date(System.currentTimeMillis()));</bpmn2:script>

203     </bpmn2:scriptTask>
204     <bpmn2:sequenceFlow id="SequenceFlow_1" tns:priority="1" sourceRef="
            ExclusiveGateway_7" targetRef="ScriptTask_5"/>
205     <bpmn2:intermediateCatchEvent id="IntermediateCatchEvent_1" name="30
            segundos">
206       <bpmn2:extensionElements>
207         <tns:metaData name="elementname">
208           <tns:metaValue><![CDATA[30 segundos]]></tns:metaValue>
209         </tns:metaData>
210       </bpmn2:extensionElements>
211       <bpmn2:incoming>SequenceFlow_2</bpmn2:incoming>
212       <bpmn2:outgoing>SequenceFlow_5</bpmn2:outgoing>
213       <bpmn2:timerEventDefinition id="TimerEventDefinition_1">
214         <bpmn2:timeDuration xsi:type="bpmn2:tFormalExpression" id="
                FormalExpression_2">30s</bpmn2:timeDuration>
215       </bpmn2:timerEventDefinition>
216     </bpmn2:intermediateCatchEvent>
217     <bpmn2:sequenceFlow id="SequenceFlow_2" tns:priority="1" sourceRef="
            ScriptTask_5" targetRef="IntermediateCatchEvent_1"/>
218     <bpmn2:sequenceFlow id="SequenceFlow_5" tns:priority="1" sourceRef="
            IntermediateCatchEvent_1" targetRef="ScriptTask_7"/>
```

```
219     <bpmn2:sequenceFlow id="SequenceFlow_17" tns:priority="1" sourceRef="
           ScriptTask_7" targetRef="ExclusiveGateway_9"/>
220     <bpmn2:scriptTask id="ScriptTask_8" name="Alerta o mdico sobre o
           desligamento do sistema" scriptFormat="http://www.java.com/java">
221       <bpmn2:extensionElements>
222         <tns:metaData name="elementname">
223           <tns:metaValue><![CDATA[Alerta o mdico sobre o desligamento do
                sistema]]></tns:metaValue>
224         </tns:metaData>
225       </bpmn2:extensionElements>
226       <bpmn2:incoming>SequenceFlow_10</bpmn2:incoming>
227       <bpmn2:outgoing>SequenceFlow_18</bpmn2:outgoing>
228       <bpmn2:script>notification.sendEmail(&amp;quot;Sensor Desligado&amp;quot;, &
             quot;No foi possvel receber dados do sensor. Existe algum
             problema de comunicao ou falha no equipamento.&amp;quot;);&amp;#xD;
229 System.out.println(&amp;quot;No foi possvel receber dados do sensor. Existe
    algum problema de comunicao ou falha no equipamento.&amp;quot;);</bpmn2:
    script>
230     </bpmn2:scriptTask>
231     <bpmn2:sequenceFlow id="SequenceFlow_18" tns:priority="1" sourceRef="
           ScriptTask_8" targetRef="EndEvent_1"/>
232   </bpmn2:process>
233   <bpmndi:BPMNDiagram id="BPMNDiagram_1">
234     <bpmndi:BPMNPlane id="BPMNPlane_Process_1" bpmnElement="medicine.
           control.ProcessBotox">
235       <bpmndi:BPMNShape id="BPMNShape_ScriptTask_1" bpmnElement="
             ScriptTask_2">
236         <dc:Bounds height="50.0" width="110.0" x="230.0" y="285.0"/>
237         <bpmndi:BPMNLabel id="BPMNLabel_2">
238           <dc:Bounds height="45.0" width="105.0" x="232.0" y="287.0"/>
239         </bpmndi:BPMNLabel>
240       </bpmndi:BPMNShape>
241       <bpmndi:BPMNShape id="BPMNShape_ExclusiveGateway_6" bpmnElement="
             ExclusiveGateway_10" isMarkerVisible="true">
242         <dc:Bounds height="50.0" width="50.0" x="790.0" y="285.0"/>
243         <bpmndi:BPMNLabel id="BPMNLabel_8">
244           <dc:Bounds height="30.0" width="74.0" x="778.0" y="335.0"/>
245         </bpmndi:BPMNLabel>
246       </bpmndi:BPMNShape>
```

```
247     <bpmndi:BPMNShape id="BPMNShape_ScriptTask_7" bpmnElement="
            ScriptTask_11">
248       <dc:Bounds height="76.0" width="141.0" x="998.0" y="213.0"/>
249       <bpmndi:BPMNLabel id="BPMNLabel_9">
250         <dc:Bounds height="60.0" width="125.0" x="1006.0" y="221.0"/>
251       </bpmndi:BPMNLabel>
252     </bpmndi:BPMNShape>
253     <bpmndi:BPMNShape id="BPMNShape_ScriptTask_2" bpmnElement="
            ScriptTask_3">
254       <dc:Bounds height="76.0" width="133.0" x="500.0" y="660.0"/>
255       <bpmndi:BPMNLabel id="BPMNLabel_6">
256         <dc:Bounds height="30.0" width="118.0" x="507.0" y="683.0"/>
257       </bpmndi:BPMNLabel>
258     </bpmndi:BPMNShape>
259     <bpmndi:BPMNShape id="BPMNShape_ExclusiveGateway_2" bpmnElement="
            ExclusiveGateway_5" isMarkerVisible="true">
260       <dc:Bounds height="50.0" width="50.0" x="1043.0" y="406.0"/>
261       <bpmndi:BPMNLabel id="BPMNLabel_10">
262         <dc:Bounds height="30.0" width="53.0" x="1042.0" y="456.0"/>
263       </bpmndi:BPMNLabel>
264     </bpmndi:BPMNShape>
265     <bpmndi:BPMNShape id="BPMNShape_ScriptTask_4" bpmnElement="
            ScriptTask_6">
266       <dc:Bounds height="83.0" width="126.0" x="1025.0" y="656.0"/>
267       <bpmndi:BPMNLabel id="BPMNLabel_11">
268         <dc:Bounds height="45.0" width="115.0" x="1030.0" y="675.0"/>
269       </bpmndi:BPMNLabel>
270     </bpmndi:BPMNShape>
271     <bpmndi:BPMNShape id="BPMNShape_ExclusiveGateway_5" bpmnElement="
            ExclusiveGateway_7" isMarkerVisible="true">
272       <dc:Bounds height="50.0" width="50.0" x="680.0" y="40.0"/>
273       <bpmndi:BPMNLabel id="BPMNLabel_12"/>
274     </bpmndi:BPMNShape>
275     <bpmndi:BPMNShape id="BPMNShape_ExclusiveGateway_8" bpmnElement="
            ExclusiveGateway_9" isMarkerVisible="true">
276       <dc:Bounds height="50.0" width="50.0" x="140.0" y="200.0"/>
277       <bpmndi:BPMNLabel id="BPMNLabel_14"/>
278     </bpmndi:BPMNShape>
279     <bpmndi:BPMNShape id="BPMNShape_EndEvent_1" bpmnElement="EndEvent_1"
```

```
        >
280       <dc:Bounds height="36.0" width="36.0" x="530.0" y="160.0"/>
281       <bpmndi:BPMNLabel id="BPMNLabel_1">
282         <dc:Bounds height="30.0" width="79.0" x="509.0" y="196.0"/>
283       </bpmndi:BPMNLabel>
284     </bpmndi:BPMNShape>
285     <bpmndi:BPMNShape id="BPMNShape_StartEvent_1" bpmnElement="
          StartEvent_1">
286       <dc:Bounds height="36.0" width="36.0" x="20.0" y="207.0"/>
287       <bpmndi:BPMNLabel id="BPMNLabel_3">
288         <dc:Bounds height="15.0" width="29.0" x="23.0" y="243.0"/>
289       </bpmndi:BPMNLabel>
290     </bpmndi:BPMNShape>
291     <bpmndi:BPMNShape id="BPMNShape_ScriptTask_5" bpmnElement="
          ScriptTask_1">
292       <dc:Bounds height="79.0" width="130.0" x="540.0" y="270.0"/>
293       <bpmndi:BPMNLabel id="BPMNLabel_5">
294         <dc:Bounds height="45.0" width="114.0" x="548.0" y="287.0"/>
295       </bpmndi:BPMNLabel>
296     </bpmndi:BPMNShape>
297     <bpmndi:BPMNShape id="BPMNShape_ExclusiveGateway_4" bpmnElement="
          ExclusiveGateway_1" isMarkerVisible="true">
298       <dc:Bounds height="50.0" width="50.0" x="419.0" y="285.0"/>
299       <bpmndi:BPMNLabel id="BPMNLabel_15">
300         <dc:Bounds height="15.0" width="69.0" x="410.0" y="335.0"/>
301       </bpmndi:BPMNLabel>
302     </bpmndi:BPMNShape>
303     <bpmndi:BPMNShape id="BPMNShape_ScriptTask_3" bpmnElement="
          ScriptTask_4">
304       <dc:Bounds height="100.0" width="110.0" x="325.0" y="406.0"/>
305       <bpmndi:BPMNLabel id="BPMNLabel_4">
306         <dc:Bounds height="15.0" width="69.0" x="345.0" y="448.0"/>
307       </bpmndi:BPMNLabel>
308     </bpmndi:BPMNShape>
309     <bpmndi:BPMNShape id="BPMNShape_ScriptTask_6" bpmnElement="
          ScriptTask_5">
310       <dc:Bounds height="100.0" width="110.0" x="495.0" y="0.0"/>
311       <bpmndi:BPMNLabel id="BPMNLabel_7">
312         <dc:Bounds height="30.0" width="100.0" x="500.0" y="35.0"/>
```

```
313         </bpmndi:BPMNLabel>
314       </bpmndi:BPMNShape>
315       <bpmndi:BPMNShape id="BPMNShape_ScriptTask_8" bpmnElement="
              ScriptTask_7">
316         <dc:Bounds height="100.0" width="110.0" x="184.0" y="0.0"/>
317         <bpmndi:BPMNLabel id="BPMNLabel_13">
318           <dc:Bounds height="30.0" width="109.0" x="184.0" y="35.0"/>
319         </bpmndi:BPMNLabel>
320       </bpmndi:BPMNShape>
321       <bpmndi:BPMNShape id="BPMNShape_IntermediateCatchEvent_1"
              bpmnElement="IntermediateCatchEvent_1">
322         <dc:Bounds height="36.0" width="36.0" x="363.0" y="32.0"/>
323         <bpmndi:BPMNLabel id="BPMNLabel_16">
324           <dc:Bounds height="15.0" width="73.0" x="345.0" y="68.0"/>
325         </bpmndi:BPMNLabel>
326       </bpmndi:BPMNShape>
327       <bpmndi:BPMNShape id="BPMNShape_ScriptTask_9" bpmnElement="
              ScriptTask_8">
328         <dc:Bounds height="100.0" width="110.0" x="345.0" y="110.0"/>
329         <bpmndi:BPMNLabel id="BPMNLabel_17">
330           <dc:Bounds height="60.0" width="100.0" x="350.0" y="130.0"/>
331         </bpmndi:BPMNLabel>
332       </bpmndi:BPMNShape>
333       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_19" bpmnElement="
              SequenceFlow_23" sourceElement="BPMNShape_ExclusiveGateway_6"
              targetElement="BPMNShape_ScriptTask_7">
334         <di:waypoint xsi:type="dc:Point" x="815.0" y="285.0"/>
335         <di:waypoint xsi:type="dc:Point" x="815.0" y="185.0"/>
336         <di:waypoint xsi:type="dc:Point" x="1068.0" y="185.0"/>
337         <di:waypoint xsi:type="dc:Point" x="1068.0" y="213.0"/>
338         <bpmndi:BPMNLabel id="BPMNLabel_16">
339           <dc:Bounds height="30.0" width="80.0" x="867.0" y="186.0"/>
340         </bpmndi:BPMNLabel>
341       </bpmndi:BPMNEdge>
342       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_4" bpmnElement="
              SequenceFlow_6" sourceElement="BPMNShape_ExclusiveGateway_2"
              targetElement="BPMNShape_ScriptTask_4">
343         <di:waypoint xsi:type="dc:Point" x="1068.0" y="456.0"/>
344         <di:waypoint xsi:type="dc:Point" x="1067.0" y="656.0"/>
```

```
345         <bpmndi:BPMNLabel id="BPMNLabel_20">
346           <dc:Bounds height="15.0" width="22.0" x="1058.0" y="557.0"/>
347         </bpmndi:BPMNLabel>
348       </bpmndi:BPMNEdge>
349       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_5" bpmnElement="
            SequenceFlow_7" sourceElement="BPMNShape_ScriptTask_7"
            targetElement="BPMNShape_ExclusiveGateway_2">
350         <di:waypoint xsi:type="dc:Point" x="1068.0" y="289.0"/>
351         <di:waypoint xsi:type="dc:Point" x="1068.0" y="406.0"/>
352         <bpmndi:BPMNLabel id="BPMNLabel_21"/>
353       </bpmndi:BPMNEdge>
354       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_10" bpmnElement="
            SequenceFlow_12" sourceElement="BPMNShape_ExclusiveGateway_2"
            targetElement="BPMNShape_ExclusiveGateway_5">
355         <di:waypoint xsi:type="dc:Point" x="1043.0" y="431.0"/>
356         <di:waypoint xsi:type="dc:Point" x="887.0" y="431.0"/>
357         <di:waypoint xsi:type="dc:Point" x="887.0" y="65.0"/>
358         <di:waypoint xsi:type="dc:Point" x="730.0" y="65.0"/>
359         <bpmndi:BPMNLabel id="BPMNLabel_22">
360           <dc:Bounds height="15.0" width="23.0" x="876.0" y="249.0"/>
361         </bpmndi:BPMNLabel>
362       </bpmndi:BPMNEdge>
363       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_11" bpmnElement="
            SequenceFlow_13" sourceElement="BPMNShape_ScriptTask_4"
            targetElement="BPMNShape_ExclusiveGateway_5">
364         <di:waypoint xsi:type="dc:Point" x="1109.0" y="656.0"/>
365         <di:waypoint xsi:type="dc:Point" x="1109.0" y="403.0"/>
366         <di:waypoint xsi:type="dc:Point" x="1168.0" y="403.0"/>
367         <di:waypoint xsi:type="dc:Point" x="1168.0" y="65.0"/>
368         <di:waypoint xsi:type="dc:Point" x="730.0" y="65.0"/>
369         <bpmndi:BPMNLabel id="BPMNLabel_23"/>
370       </bpmndi:BPMNEdge>
371       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_12" bpmnElement="
            SequenceFlow_14" sourceElement="BPMNShape_ScriptTask_2"
            targetElement="BPMNShape_ExclusiveGateway_5">
372         <di:waypoint xsi:type="dc:Point" x="544.0" y="660.0"/>
373         <di:waypoint xsi:type="dc:Point" x="544.0" y="496.0"/>
374         <di:waypoint xsi:type="dc:Point" x="705.0" y="496.0"/>
375         <di:waypoint xsi:type="dc:Point" x="705.0" y="90.0"/>
```

```
376         <bpmndi:BPMNLabel id="BPMNLabel_24"/>
377       </bpmndi:BPMNEdge>
378       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_24" bpmnElement="
              SequenceFlow_22" sourceElement="BPMNShape_ExclusiveGateway_8"
              targetElement="BPMNShape_ScriptTask_1">
379         <di:waypoint xsi:type="dc:Point" x="165.0" y="250.0"/>
380         <di:waypoint xsi:type="dc:Point" x="165.0" y="310.0"/>
381         <di:waypoint xsi:type="dc:Point" x="230.0" y="310.0"/>
382         <bpmndi:BPMNLabel id="BPMNLabel_27"/>
383       </bpmndi:BPMNEdge>
384       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_6" bpmnElement="
              SequenceFlow_4" sourceElement="BPMNShape_StartEvent_1"
              targetElement="BPMNShape_ExclusiveGateway_8">
385         <di:waypoint xsi:type="dc:Point" x="56.0" y="225.0"/>
386         <di:waypoint xsi:type="dc:Point" x="98.0" y="225.0"/>
387         <di:waypoint xsi:type="dc:Point" x="140.0" y="225.0"/>
388         <bpmndi:BPMNLabel id="BPMNLabel_17"/>
389       </bpmndi:BPMNEdge>
390       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_8" bpmnElement="
              SequenceFlow_8" sourceElement="BPMNShape_ScriptTask_5"
              targetElement="BPMNShape_ExclusiveGateway_6">
391         <di:waypoint xsi:type="dc:Point" x="670.0" y="309.0"/>
392         <di:waypoint xsi:type="dc:Point" x="730.0" y="309.0"/>
393         <di:waypoint xsi:type="dc:Point" x="790.0" y="310.0"/>
394         <bpmndi:BPMNLabel id="BPMNLabel_29"/>
395       </bpmndi:BPMNEdge>
396       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_9" bpmnElement="
              SequenceFlow_9" sourceElement="BPMNShape_ScriptTask_1"
              targetElement="BPMNShape_ScriptTask_3">
397         <di:waypoint xsi:type="dc:Point" x="340.0" y="310.0"/>
398         <di:waypoint xsi:type="dc:Point" x="361.0" y="310.0"/>
399         <di:waypoint xsi:type="dc:Point" x="361.0" y="406.0"/>
400         <bpmndi:BPMNLabel id="BPMNLabel_30"/>
401       </bpmndi:BPMNEdge>
402       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_15" bpmnElement="
              SequenceFlow_10" sourceElement="BPMNShape_ExclusiveGateway_4"
              targetElement="BPMNShape_ScriptTask_9">
403         <di:waypoint xsi:type="dc:Point" x="444.0" y="285.0"/>
404         <di:waypoint xsi:type="dc:Point" x="444.0" y="248.0"/>
```

```
405         <di:waypoint xsi:type="dc:Point" x="400.0" y="248.0"/>
406         <di:waypoint xsi:type="dc:Point" x="400.0" y="210.0"/>
407         <bpmndi:BPMNLabel id="BPMNLabel_31">
408           <dc:Bounds height="15.0" width="23.0" x="411.0" y="249.0"/>
409         </bpmndi:BPMNLabel>
410       </bpmndi:BPMNEdge>
411       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_21" bpmnElement="
            SequenceFlow_11" sourceElement="BPMNShape_ExclusiveGateway_4"
            targetElement="BPMNShape_ScriptTask_5">
412         <di:waypoint xsi:type="dc:Point" x="469.0" y="310.0"/>
413         <di:waypoint xsi:type="dc:Point" x="504.0" y="310.0"/>
414         <di:waypoint xsi:type="dc:Point" x="540.0" y="309.0"/>
415         <bpmndi:BPMNLabel id="BPMNLabel_32">
416           <dc:Bounds height="15.0" width="22.0" x="495.0" y="311.0"/>
417         </bpmndi:BPMNLabel>
418       </bpmndi:BPMNEdge>
419       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_13" bpmnElement="
            SequenceFlow_15" sourceElement="BPMNShape_ExclusiveGateway_6"
            targetElement="BPMNShape_ScriptTask_2">
420         <di:waypoint xsi:type="dc:Point" x="815.0" y="335.0"/>
421         <di:waypoint xsi:type="dc:Point" x="815.0" y="698.0"/>
422         <di:waypoint xsi:type="dc:Point" x="633.0" y="698.0"/>
423         <bpmndi:BPMNLabel id="BPMNLabel_7">
424           <dc:Bounds height="30.0" width="80.0" x="776.0" y="609.0"/>
425         </bpmndi:BPMNLabel>
426       </bpmndi:BPMNEdge>
427       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_14" bpmnElement="
            SequenceFlow_16" sourceElement="BPMNShape_ScriptTask_3"
            targetElement="BPMNShape_ExclusiveGateway_4">
428         <di:waypoint xsi:type="dc:Point" x="397.0" y="406.0"/>
429         <di:waypoint xsi:type="dc:Point" x="397.0" y="310.0"/>
430         <di:waypoint xsi:type="dc:Point" x="419.0" y="310.0"/>
431         <bpmndi:BPMNLabel id="BPMNLabel_13"/>
432       </bpmndi:BPMNEdge>
433       <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_1" bpmnElement="
            SequenceFlow_1" sourceElement="BPMNShape_ExclusiveGateway_5"
            targetElement="BPMNShape_ScriptTask_6">
434         <di:waypoint xsi:type="dc:Point" x="680.0" y="65.0"/>
435         <di:waypoint xsi:type="dc:Point" x="643.0" y="65.0"/>
```

```
436       <di:waypoint xsi:type="dc:Point" x="643.0" y="50.0"/>
437       <di:waypoint xsi:type="dc:Point" x="605.0" y="50.0"/>
438       <bpmndi:BPMNLabel id="BPMNLabel_18"/>
439     </bpmndi:BPMNEdge>
440     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_2" bpmnElement="
          SequenceFlow_2" sourceElement="BPMNShape_ScriptTask_6"
          targetElement="BPMNShape_IntermediateCatchEvent_1">
441       <di:waypoint xsi:type="dc:Point" x="495.0" y="50.0"/>
442       <di:waypoint xsi:type="dc:Point" x="447.0" y="50.0"/>
443       <di:waypoint xsi:type="dc:Point" x="399.0" y="50.0"/>
444       <bpmndi:BPMNLabel id="BPMNLabel_19"/>
445     </bpmndi:BPMNEdge>
446     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_7" bpmnElement="
          SequenceFlow_5" sourceElement="BPMNShape_IntermediateCatchEvent_1
          " targetElement="BPMNShape_ScriptTask_8">
447       <di:waypoint xsi:type="dc:Point" x="363.0" y="50.0"/>
448       <di:waypoint xsi:type="dc:Point" x="329.0" y="50.0"/>
449       <di:waypoint xsi:type="dc:Point" x="294.0" y="50.0"/>
450       <bpmndi:BPMNLabel id="BPMNLabel_25"/>
451     </bpmndi:BPMNEdge>
452     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_16" bpmnElement="
          SequenceFlow_17" sourceElement="BPMNShape_ScriptTask_8"
          targetElement="BPMNShape_ExclusiveGateway_8">
453       <di:waypoint xsi:type="dc:Point" x="184.0" y="50.0"/>
454       <di:waypoint xsi:type="dc:Point" x="164.0" y="50.0"/>
455       <di:waypoint xsi:type="dc:Point" x="164.0" y="125.0"/>
456       <di:waypoint xsi:type="dc:Point" x="165.0" y="200.0"/>
457       <bpmndi:BPMNLabel id="BPMNLabel_26"/>
458     </bpmndi:BPMNEdge>
459     <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_17" bpmnElement="
          SequenceFlow_18" sourceElement="BPMNShape_ScriptTask_9"
          targetElement="BPMNShape_EndEvent_1">
460       <di:waypoint xsi:type="dc:Point" x="455.0" y="160.0"/>
461       <di:waypoint xsi:type="dc:Point" x="492.0" y="160.0"/>
462       <di:waypoint xsi:type="dc:Point" x="492.0" y="178.0"/>
463       <di:waypoint xsi:type="dc:Point" x="530.0" y="178.0"/>
464       <bpmndi:BPMNLabel id="BPMNLabel_28"/>
465     </bpmndi:BPMNEdge>
466   </bpmndi:BPMNPlane>
```

```
467   </bpmndi:BPMNDiagram>
468 </bpmn2:definitions>
```

B.9 POM.xml

```
1  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.
       w3.org/2001/XMLSchema-instance"
2    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.
       apache.org/xsd/maven-4.0.0.xsd">
3    <modelVersion>4.0.0</modelVersion>
4
5    <groupId>br.ufes.inf.lprm</groupId>
6    <artifactId>botox-drugs-scenario</artifactId>
7    <version>0.0.1-SNAPSHOT</version>
8    <packaging>jar</packaging>
9
10   <name>co.pextra</name>
11   <url>http://maven.apache.org</url>
12
13   <properties>
14     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15     <drools-version>6.5.0.Final</drools-version>
16     <version.org.jbpm>6.5.0.Final</version.org.jbpm>
17     <scene-version>1.1.1</scene-version>
18     <slf4j-version>1.7.2</slf4j-version>
19     <junit-version>4.12</junit-version>
20   </properties>
21
22   <dependencyManagement>
23     <dependencies>
24       <dependency>
25         <groupId>org.drools</groupId>
26         <artifactId>drools-bom</artifactId>
27         <type>pom</type>
28         <version>${drools-version}</version>
29         <scope>import</scope>
30       </dependency>
31       <dependency>
32         <groupId>org.jbpm</groupId>
33         <artifactId>jbpm-bom</artifactId>
```

```
34        <type>pom</type>
35        <version>${version.org.jbpm}</version>
36        <scope>import</scope>
37      </dependency>
38    </dependencies>
39  </dependencyManagement>
40
41  <dependencies>
42
43    <dependency>
44        <groupId>org.drools</groupId>
45        <artifactId>drools-compiler</artifactId>
46    </dependency>
47    <dependency>
48          <groupId>org.jbpm</groupId>
49          <artifactId>jbpm-flow</artifactId>
50    </dependency>
51    <dependency>
52          <groupId>org.jbpm</groupId>
53          <artifactId>jbpm-flow-builder</artifactId>
54    </dependency>
55    <dependency>
56          <groupId>org.jbpm</groupId>
57          <artifactId>jbpm-bpmn2</artifactId>
58    </dependency>
59    <dependency>
60          <groupId>org.jbpm</groupId>
61          <artifactId>jbpm-persistence-jpa</artifactId>
62    </dependency>
63    <dependency>
64          <groupId>org.jbpm</groupId>
65          <artifactId>jbpm-runtime-manager</artifactId>
66    </dependency>
67    <dependency>
68          <groupId>junit</groupId>
69          <artifactId>junit</artifactId>
70          <version>${junit-version}</version>
71    </dependency>
72    <dependency>
```

```
73          <groupId>org.slf4j</groupId>
74          <artifactId>slf4j-log4j12</artifactId>
75          <version>${slf4j-version}</version>
76    </dependency>
77    <dependency>
78          <groupId>org.kie</groupId>
79          <artifactId>kie-api</artifactId>
80          <version>${drools-version}</version>
81    </dependency>
82    <dependency>
83          <groupId>br.ufes.inf.lprm</groupId>
84          <artifactId>scene-core</artifactId>
85          <version>${scene-version}</version>
86    </dependency>
87  </dependencies>
88
89  <repositories>
90    <repository>
91      <id>Scene repo</id>
92      <url>https://mymavenrepo.com/repo/BG5Za6vz3CyY7SaQjMOa/</url>
93    </repository>
94
95  </repositories>
96
97  <build>
98    <plugins>
99      <plugin>
100       <artifactId>maven-compiler-plugin</artifactId>
101       <version>3.1</version>
102       <configuration>
103         <source>1.6</source>
104         <target>1.6</target>
105       </configuration>
106     </plugin>
107     <plugin>
108       <groupId>org.kie</groupId>
109       <artifactId>kie-maven-plugin</artifactId>
110       <version>${drools-version}</version>
111       <extensions>true</extensions>
```

```
112       </plugin>
113     </plugins>
114
115   </build>
116 </project>
```

B.10 Serviço de Envio de e-mail

```
1  package com.medicine.control.handler;
2
3
4  //import java.util.ArrayList;
5  //import java.util.List;
6
7  import javax.mail.*;
8  import javax.mail.internet.*;
9
10 import java.util.Date;
11 import java.util.Properties;
12
13 import com.medicine.control.handler.SMTPAuthenticator;
14
15 public class MailWithAuthentication {
16
17     public static void alert() throws Exception{
18         String subject = "Elevao de temperatura [URGENTE]";
19         String body = "Foi detectada a elevao de temperatura do freezer,
               recomendado verificar!";
20         sendEmail(subject, body);
21     }
22
23     public static void announcement() throws Exception{
24         String subject = "Temperatura estabilizada";
25         String body = "A temperatura do freezer voltou ao normal!";
26         sendEmail(subject, body);
27     }
28
29     public static void sendEmail(String subject, String body) throws
           Exception{
30         Properties props = new Properties();
```

```
31          props.put("mail.smtp.user", SMTPAuthenticator.SMTP_AUTH_USER);
32          props.put("mail.smtp.host", SMTPAuthenticator.SMTP_HOST_NAME);
33          props.put("mail.smtp.port", SMTPAuthenticator.SMTP_HOST_PORT);
34          props.put("mail.smtp.starttls.enable","true");
35          props.put("mail.smtp.debug", "true");
36          props.put("mail.smtp.auth", "true");
37          props.put("mail.smtp.socketFactory.port", SMTPAuthenticator.
                SMTP_HOST_PORT);
38          props.put("mail.smtp.socketFactory.class", "javax.net.ssl.
                SSLSocketFactory");
39          props.put("mail.smtp.socketFactory.fallback", "false");
40
41          SMTPAuthenticator auth = new SMTPAuthenticator();
42          Session session = Session.getInstance(props, auth);
43
44          // creates a new e-mail message
45          Message msg = new MimeMessage(session);
46
47          msg.setFrom(new InternetAddress(SMTPAuthenticator.SMTP_AUTH_USER))
                ;
48          //InternetAddress[] toAddresses = { new InternetAddress(
                SMTPAuthenticator.SMTP_AUTH_USER) };
49          //msg.setRecipients(Message.RecipientType.TO, toAddresses);
50          msg.addRecipient(Message.RecipientType.TO, new InternetAddress(
                SMTPAuthenticator.SEND_TO));
51          msg.setSubject(subject);
52          msg.setSentDate(new Date());
53
54          // creates message part
55          MimeBodyPart messageBodyPart = new MimeBodyPart();
56          messageBodyPart.setContent(body, "text/html");
57
58          // creates multi-part
59          Multipart multipart = new MimeMultipart();
60          multipart.addBodyPart(messageBodyPart);
61
62          // sets the multi-part as e-mail's content
63          msg.setContent(multipart);
64
```

```
65          try{
66              Transport transport = session.getTransport("smtps");
67                  transport.connect(SMTPAuthenticator.SMTP_HOST_NAME, Integer
                        .valueOf(SMTPAuthenticator.SMTP_HOST_PORT),
                        SMTPAuthenticator.SMTP_AUTH_USER, SMTPAuthenticator.
                        SMTP_AUTH_PWD);
68                  transport.sendMessage(msg, msg.getAllRecipients());
69                  transport.close();
70
71              } catch (AddressException e) {
72                  e.printStackTrace();
73              } catch (MessagingException e) {
74                  e.printStackTrace();
75              }
76      }
77
78
79  }
```

B.11 SMTP Authenticator

```
1   package com.medicine.control.handler;
2
3   import javax.mail.PasswordAuthentication;
4
5   public class SMTPAuthenticator extends javax.mail.Authenticator {
6
7       public static final String SMTP_HOST_NAME = "smtp.gmail.com";
8       public static final String SMTP_HOST_PORT = "465";
9       public static final String SMTP_AUTH_USER = "credenciais";
10      public static final String SMTP_AUTH_PWD = "credenciais";
11      public static final String SEND_TO = "destinatrio";
12         public PasswordAuthentication getPasswordAuthentication() {
13            String username = SMTP_AUTH_USER;
14            String password = SMTP_AUTH_PWD;
15            return new PasswordAuthentication(username, password);
16         }
17  }
```

B.12 Controladores do Serviço de Autenticação JWT do Repositório de Processos(Node)

```
1  const jwt = require('jsonwebtoken') // used to create, sign, and verify
       tokens
2  const express = require('express')
3  const apiRoutes = express.Router()
4
5  function init (app, User) {
6    apiRoutes.post('/authenticate', function (req, res) {
7      User.findOne({
8      // find the user
9        name: req.body.name
10     }, function (err, user) {
11       if (err) {
12         throw err
13       }
14
15       if (!user) {
16         res.json({ success: false, message: 'Authentication failed. User
             not found.' })
17       } else if (user) {
18       // check if password matches
19         if (user.password !== req.body.password) {
20           res.json({ success: false, message: 'Authentication failed.
               Wrong password.' })
21         } else {
22         // if user is found and password is right create a token
23           var token = jwt.sign(user, app.get('superSecret'))
24
25           res.json({
26             success: true,
27             message: 'Enjoy your token!',
28             token: token
29           })
30         }
31       }
32     })
33   })
```

```
34
35 // ---------------------------------------------------------
36 // route middleware to authenticate and check token
37 // ---------------------------------------------------------
38   apiRoutes.use(function (req, res, next) {
39   // check header or url parameters or post parameters for token
40     var token = req.body.token || req.query['token'] || req.headers['x-
         access-token']
41
42   // decode token
43     if (token) {
44     // verifies secret and checks exp
45       jwt.verify(token, app.get('superSecret'), function (err, decoded) {
46         if (err) {
47           return res.json({ success: false, message: 'Failed to
               authenticate token.' })
48         } else {
49         // if everything is good, save to request for use in other routes
50           req.decoded = decoded
51           next()
52         }
53       })
54     } else {
55     // if there is no token
56     // return an error
57       return res.status(403).send({success: false, message: 'No token
           provided.:)'})
58     }
59   })
60
61 // ---------------------------------------------------------
62 // authenticated routes
63 // ---------------------------------------------------------
64 // Send a GET to ex: localhost:8080/api/check?token=
     eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyIkX18iOnsic3RyaWN0TW9kZSI6dHJ1ZSwic2VsZWN0ZWQiOnt9LCJnZXR0ZXJzIjp7fSwiX2lkIjoiNTk5
     .BEFNKwRyC1zYVjeIfsXHEnRsBQjMSnqnqDKEryM1tuE
65 // or localhost:8080/api?token=... or localhost:8080/check?token=...
66   apiRoutes.get('/', function (req, res) {
```

```
67      res.json({ message: 'Welcome to API!' })
68    })
69
70    apiRoutes.get('/users', function (req, res) {
71      User.find({}, function (err, users) {
72        if (!err) { res.json(users) } else { res.json(err) }
73      })
74    })
75
76    apiRoutes.get('/check', function (req, res) {
77      res.json(req.decoded)
78    })
79
80    app.use('/api', apiRoutes)
81  }
82  // ---------------------------------------------------------
83  // authentication (no middleware necessary since this isnt authenticated)
84  // ---------------------------------------------------------
85  // http://localhost:8080/api/authenticate
86  // POST -> Body -> x-www-form-unlencoded
87
88  module.exports = {
89    init
90  }
```