Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Instituto de Matemática e Estatística

Gedir Fabiano de Moraes Gonçalves

Uma arquitetura multiagente para sistemas de informação orientados a processos intensivos em conhecimento

Rio de Janeiro
2022

Gedir Fabiano de Moraes Gonçalves

Uma arquitetura multiagente para sistemas de informação orientados a processos intensivos em conhecimento

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Ciências Computacionais, da Universidade do Estado do Rio de Janeiro.

Orientadora: Prof.ª Dra. Vera Maria Benjamim Werneck
Coorientadora: Prof.ª Dra. Flávia Maria Santoro

Rio de Janeiro
2022

CATALOGAÇÃO NA FONTE
UERJ/REDE SIRIUS/BIBLIOTECA CTC/A

G635 Gonçalves, Gedir Fabiano de Moraes.
Uma arquitetura multiagente para sistemas de informação orientados a processos intensivos em conhecimento/ Gedir Fabiano de Moraes Gonçalves. – 2022-
109 f.:il.

Orientadora: Vera Maria Benjamim Werneck
Dissertação (Mestrado em Ciências Computacionais) – Universidade do Estado do Rio de Janeiro, Instituto de Matemática e Estatística.

1. Sistemas multiagentes - Teses. 2. Gerenciamento de sistemas de informação - Teses. I. Werneck, Vera Maria Benjamim. II. Universidade do Estado do Rio de Janeiro. Instituto de Matemática e Estatística. III. Título

CDU 004.78

Patricia Bello Meijinhos CRB7/5217 - Bibliotecária responsável pela elaboração da ficha catalográfica

_____ _____

Assinatura Data

Gedir Fabiano de Moraes Gonçalves

Uma arquitetura multiagente para sistemas de informação orientados a processos intensivos em conhecimento

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Ciências Computacionais, da Universidade do Estado do Rio de Janeiro.

Aprovada em 25 de julho de 2022.
Banca Examinadora:

Prof.ª Dra. Vera Maria Benjamim Werneck (Orientadora)
Instituto de Matemática e Estatística – UERJ

Prof.ª Dra. Flávia Maria Santoro (Coorientadora)
Instituto de Matemática e Estatística – UERJ

Prof. Dr. Eduardo Kinder Almentero
Universidade Federal Rural do Rio de Janeiro

Prof.ª Dra. Vanessa Tavares Nunes
Universidade de Brasília

Rio de Janeiro
2022

DEDICATÓRIA

Durante a minha vida conheci pessoas de carácter, dedicadas, amáveis, de bom coração e outras de indetermináveis adjetivos. Mas uma pessoa teve uma participação especial na minha vida que vou ficar eternamente em débito. Ela não foi só uma avó, foi uma madrinha, uma mãe e uma amiga que em muitos momentos da minha vida estava lá com um sorriso e com uma palavra de carinho a oferecer.

Como era delicioso o macarrão com salsicha e o miolo de macaco, o pote com lascas de chocolate e a procura pelos ovos na Páscoa, as recordações do Natal. São as lembranças e os sabores de uma infância simples e feliz. Uma pena que eu não possa mais dizer pessoalmente para ela essas linhas que estão impressas no papel, mas com certeza ela está dizendo amavelmente que "tudo vai dar certo meu filho". Obrigado Vó!

AGRADECIMENTOS

É difícil achar palavras para agradecer à minha parceira de vida pela paciência e pelo tempo que deixei de compartilhar com ela e com nossa filha para que esse trabalho fosse produzido. Foram horas, dias, meses e anos de expressões como hoje não posso porque tenho aula, prova, pesquisa, escrita, programação, correção de erros, instalação de aplicativo, desinstalação, reinstalação e por aí vai, teve de tudo nessa jornada. Ela estava lá com seu carinho, dedicação, compreensão, motivação, com um lanchinho ou com um chá, com uma palavra amiga. Com ela o caminho foi mais fácil. Não posso dizer que foi a primeira vez, mas com certeza foi uma das mais difíceis. É muito bom compartilhar a vida com uma pessoa que está ao seu lado em todos os momentos independente se são bons ou difíceis.

Gostaria também de agradecer a todos mestres professores que passaram pela minha vida acadêmica, desde a minha alfabetização com a professora Rosita até as doutoras Vera Werneck e Flávia Santoro que orientaram e tiveram muita paciência nesse trabalho. Todos foram importantes e contribuíram para iluminar o caminho. Um muito obrigado por tudo é o mínimo que posso fazer.

RESUMO

GONÇALVES, Gedir Fabiano de. *Uma arquitetura multiagente para sistemas de informação orientados a processos intensivos em conhecimento*. 2022. 109 f. Dissertação (Mestrado em Ciências Computacionais) – Instituto de Matemática e Estatística, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2022.

Este trabalho propõe uma abordagem para a adaptação de processos de negócio utilizando uma arquitetura multiagente e informações de contexto. A abordagem utiliza multiagentes baseados em crenças, desejos e intenções (BDI) com a premissa de monitorar e realizar a adaptação dos processos de negócio em tempo de execução utilizando informações de contexto capturadas do *Process-Aware Information Systems* (PAIS), de sensores externos e de informações oriundas da ontologia do domínio do processo. A proposição da arquitetura adaptação por agentes, nomeada `AAdapt`, é composta de dois agentes que interagem com o ambiente do processo para identificar situações que acarretam problemas na execução padrão do processo, permitindo que soluções de contorno apresentem caminhos alternativos que mitigam riscos e permitem a correta execução do processo.

Palavras-chave: Multiagentes. KiP. BDI. BPM.

ABSTRACT

GONÇALVES, Gedir Fabiano de. *A multi-agent architecture for knowledge-intensive process-oriented information systems.* 2022. 109 f. Dissertação (Mestrado em Ciências Computacionais) – Instituto de Matemática e Estatística, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2022.

This work proposes an approach for adapting business processes using a multi-agent architecture and context information. The approach uses multi-agent based on beliefs, desires and intentions (BDI) with the premise of monitoring and adapting business processes at runtime using context information captured from Process-Aware Information Systems (PAIS), external sensors and information from the ontology of the process domain. The agent adaptation architecture proposition, named AAdapt, is composed of two agents that interact with the process environment to identify situations that cause problems in the standard execution of the process, allowing workarounds solutions to present alternative paths that mitigate risks and allow the correct execution of the process.

Keywords: Multi-agent. KiP. BDI. BPM.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACL	*Agent Communication Language*
ADF	*Agent Definition Files*
API	*Application Programming Interface*
BDI	*Belief, Desire and Intention*
BPA	*Business Process Automation*
BPM	*Business Process Management*
BPMN	*Business Process Management Notation*
BPMS	*Business Process Management Systems*
CPS	*Cyber-Physical System*
DARPA	*Defense Advanced Research Projects Agency*
ERP	*Enterprise Resource Planning*
FIPA	*Foundation for Intelligent Physical Agents*
HTTP	*Hypertext Transfer Protocol*
IEEE	*Institute of Electrical and Electronic Engineers*
IRI	*Internationalized Resource Identifier*
KiP	*Knowledge-Intensive Process*
KiPO	*Knowledge-Intensive Process Ontology*
KQML	*Knowledge Query and Manipulation Language*
MAPE-K	*Monitor-Analyze-Plan-Execute over a shared Knowledge*
MAS	*Multi-Agent Systems*
MaSE	*Multiagent Systems Engineering*
O-MaSE	*Organization-based Multiagent Software Engineering*
OWL	*Web Ontology Language*
PAIS	*Process-Aware Information Systems*
PDG	*Process Dependency Graph*
PRS	*Procedural Reasoning System*
REST	*Representational State Transfer*
SRL	*Systematic Review of Literature*
TCP	*Transmission Control Protocol*
UML	*Unified Modeling Language*
WAR	*Web Application Resource*
W3C	*World Wide Web Consortium*
Web	*World Wide Web* ou WWW
WWW	*World Wide Web*

SUMÁRIO

INTRODUÇÃO

Na busca por melhor desempenho, aumento na produtividade e obtenção de resultados mais expressivos nos negócios, as organizações executam muitas ações para aprimorar os seus processos, tornando-os mais precisos, autônomos e eficientes. Para isso o gestor moderno não deve se preocupar somente com o produto final que é visto pelo cliente, mas com muitas outras atividades que são executadas dentro da empresa para que o produto chegue ao seu destinatário final na forma, tempo e qualidade esperada. Processos são compostos de atividades que permeiam as diversas áreas das empresas como recursos humanos, seleção de fornecedores e de matérias-primas, vendas, marketing, tecnologia da informação e de telecomunicações entre outras. Essas áreas desempenham atividades aglutinadas por dependências, obrigações, funções, produtos ou negócios que, relacionadas, são identificadas como processos do negócio da empresa.

Os processos de negócio da empresa precisam ser mapeados, analisados, melhorados e muitas vezes adaptados. Essa sistematização dos processos é conhecida como Gestão de Processos de Negócio ou *Business Process Management* (BPM). Segundo Dumas et al. (2018), BPM é a arte e ciência de ver como o trabalho na empresa é organizado para garantir resultados consistentes e ter a oportunidade de melhorá-los. Ter et al. (2010) afirmam que BPM recebeu considerável atenção recentemente por seu potencial significativo de aumentar a produtividade e reduzir custos e Braunnagel et al. (2016) argumentam que a gestão de processos de negócio (BPM) propicia uma importante vantagem competitiva na economia globalizada. A disciplina de *Business Process Management* (BPM) define soluções para modelagem, desenvolvimento, análise e melhoria de processos com fluxo de atividades pré-definido (BOISSIER; RYCHKOVA; GRAND, 2019).

BPM, que por muito tempo foi baseado em processos estruturados (SANTORO; BAIAO, 2018), precisa de evoluções constantes para atender os avanços tecnológicos e as necessidades das empresas. Os processos de negócio estão cada vez mais sendo executados em plataformas de mídia social e habilitados por dispositivos móveis (SANTORO; BAIAO, 2018).

A migração da execução de tarefas do computador para os dispositivos móveis facilitou a disseminação da informação e, consequentemente, mais plataformas de gerenciamentos de dados e de informações foram se estabelecendo. Internet, internet das coisas, mídias sociais, relógios e pulseiras inteligentes, *smart* TVs e *smartphones*, impuseram muitos desafios e a necessidade de tratamento de processos por meios não estruturados para atender os desvios que ocorrem com mais frequências no fluxo padrão do processo. Em resumo, há grande quantidade de dados dispersos e muitas vezes é necessária a experiência humana para transformá-los em conhecimento.

Os processos em que o conhecimento é uma característica fundamental são cha-

mados de processos intensivos em conhecimento ou *Knowledge-Intensive Processes* (KiP) (BOISSIER; RYCHKOVA; GRAND, 2019). Santoro e Baiao (2018) ratificam que grande atenção está sendo dedicada aos chamados Processos Intensivos em Conhecimento (KiP), que são favorecidos pelas heterogeneidades das fontes de dados como mídias sociais, dispositivos inteligentes, computação em tempo real e tecnologias para *big data*. Em um KiP, a importância da experiência ou conhecimento tácito dos participantes do processo, também chamados de trabalhadores do conhecimento, é enfatizada (BOISSIER; RYCHKOVA; GRAND, 2019).

Um desafio para a área de BPM é captar as informações que realmente são importantes para apoiar a execução e melhoria dos processos. Em processos intensivos em conhecimento esse desafio se torna ainda maior, pois as informações são provenientes também de fontes não estruturadas. Além disso, um KiP bem-sucedido precisa ser capaz de se adaptar a diferentes situações pois não são processos totalmente estruturados que funcionam segundo um modelo rígido. Por exemplo, é preciso tratar a solicitação de cada cliente como única, em vez de seguir uma sequência predefinida de ações, sendo estas características desafiadoras para automatizar, controlar e testar a sua conformidade (BOISSIER; RYCHKOVA; GRAND, 2019). Essa evolução é necessária nos processos de negócio, permitindo tratamento cada vez mais individualizado, reduzindo tempo, erros e custo dos processos. Por conseguinte o uso de sistemas de suporte aos processos de negócio passou a ser ainda mais importante.

Os sistemas que dão suporte aos processos de negócio são conhecidos como *Process-Aware Information Systems* (PAIS) e visam apoiar os processos operacionais de negócios, combinando avanços em tecnologia da informação com *insights* recentes da ciência da gestão (BOISSIER; RYCHKOVA; GRAND, 2019). Tipos particulares de PAIS muito utilizados nas empresas são os *Business Process Management Systems* (BPMS) e os *Enterprise Resource Planning* (ERP). Porém estes tipos de sistemas apoiam a execução de processos tradicionais, ou seja, processos bastante estruturados.

Os BPMS coordenam o comportamento de pessoas, dispositivos e sistemas produzindo como resultado a automatização dos processos de negócio na empresa. Os sistemas BPMS lidam com a definição, administração, customização e avaliação de tarefas que envolvem processos de negócio, bem como de estruturas organizacionais (KARAGIANNIS, 1995). Choi et al. (2005) argumentam que o BPMS automatiza tarefas rotineiras, gerencia todas as fases do processo e fornece visibilidade e controle ponta a ponta. Bizagi BPM *Suite* (BIZAGI, 2021), Appian BPM *Suite* (APPIAN, 2021), Oracle *Fusion Middleware* (ORACLE, 2021a) e ProcessMaker (PROCESSMAKER, 2021) são exemplos de ferramentas nessa categoria.

Os ERPs são sistemas de planejamento de recursos empresariais usados por empresas como programas centrais para integrar uma informação coordenada em todas as áreas de negócios, ajudando as organizações a gerenciar os processos de negócio em toda a

empresa, usando um banco de dados comum e ferramentas de relatórios de gerenciamento compartilhados (MONK; WAGNER, 2012). Como exemplos de software ERP podem ser citados SAP ERP (SAP, 2021), TOTVS ERP (TOTVS, 2021), Oracle *Fusion Cloud* ERP (COMPANY, 2021), PeopleSoft (PEOPLESOFT, 2021) e Microsoft *Dynamics* 365 *Finance* (MICROSOFT, 2021).

Embora os BPMS e os ERP sejam sistemas maduros e consolidados no mercado, eles foram desenvolvidos para processos de negócio bem estruturados e, em geral, não possuem flexibilidade para tratar exceções e realizar adaptações dinâmicas no processo de negócios que iremos tratar com a pesquisa proposta.

Problema de pesquisa

De forma geral, os PAIS são concebidos para integrar os sistemas das diversas áreas da empresa, automatizar tarefas e apoiar a gestão de processos de negócio. De maneira geral, os fluxos de trabalho são desenhados para seguir um caminho padrão de ações e atividades que são definidas no momento de *design* do processo. Esse mapeamento normalmente não prevê o comportamento de um KiP, cujas instâncias podem sofrer influências externas ou internas, gerando exceções na execução e que, se não tratadas, podem impedir a conclusão do processo, gerar erros, inserir vulnerabilidades em sistemas, acarretar perdas financeiras e de prestígio ou até mesmo danos à imagem da empresa. Uma opção seria criar um fluxo para cada situação identificada como possível, porém, mesmo na hipótese de criação de diversos fluxos, o executor do processo ainda teria que conhecer todo o ambiente de execução para escolher qual fluxo executar e, além disso, não poderiam ocorrer mudanças de contexto das premissas assumidas no início da execução daquela instância do processo. Embora essa solução seja possível tecnicamente de ser construída, ela aumenta o trabalho de *design*, eleva a responsabilidade do executor e não garante que não ocorrerão erros se aparecerem mudanças de valores durante a execução da instância do processo.

Nunes (2014) desenvolveu uma arquitetura que permite a adaptação dinâmica de processos, utilizando o conceito de contexto, e planejamento (*planning*) das tarefas quando identificada uma situação de exceção no processo. Esse trabalho se fundamentou na teoria de sistemas de informação sensíveis a contexto, fazendo dessa forma o replanejamento das tarefas para atender os objetivos definidos em tempo de desenvolvimento. Nomeado de `GCAdapt`, é uma arquitetura que permite a adaptação dinâmica de tarefas de processos com suporte de informações de contexto (*Context-Aware*). O `GCAdapt` possui integração com um motor de processos de sistemas de informação orientados por processos (*Process-Aware Information System* PAIS).

O GCAdapt possui quatro componentes principais na sua arquitetura: *Maintainer,*

Aggregator, Actuator e *Mediator*. O módulo *Maintainer* é responsável por gerenciar as definições de contexto, o *Aggregator* coleta elementos contextuais (*CE - contextual elements*) que estão logicamente relacionados em um repositório. O *Actuator* recebe as atualizações do *Mediator* e envia os comandos de adaptação no processo para o mecanismo de implementação e o *mediator* age na adaptação durante a execução da instância do processo, conforme definido por (NUNES, 2014).

Essa pesquisa pretende implementar as funcionalidades do GCAdapt sob a ótica de sistemas multiagentes inteligentes capazes de captarem informações do sistema PAIS e permitir que exceções do fluxo padrão de trabalho sejam tratadas por meio de planos de agentes com BDI *Belief, Desire and Intention* e utilizar ontologia para obter axiomas e realizar inferências. Por fim, de forma a aumentar a assertividade, em tempo de execução, a proposta permite em algumas situações que a experiência humana decida qual caminho no processo deve ser seguido.

Desta forma, o problema tratado nesta pesquisa é: **como adaptar dinamicamente as tarefas de uma instância de processo de negócio para tratar exceções na execução mitigando a ocorrência de falhas?**

Objetivo geral e específicos

O objetivo principal desse trabalho é propor uma arquitetura multiagente capaz de adaptar processos de negócio em tempo de execução de forma a diminuir os riscos de falhas na execução da instância do processo. Os objetivos específicos são:

1. definir e implementar os agentes participantes dessa arquitetura;
2. especificar o conhecimento sobre o processo por meio de ontologias;
3. definir a forma de interação dos agentes com um sistema PAIS.

Essa pesquisa propõe ampliar a discussão do tratamento de processos de negócio trazendo à análise e discussão uma arquitetura que possibilita o tratamento de exceções numa instância do processo em tempo de execução. Os componentes da arquitetura capturam informações de contexto do ambiente durante a execução da instância para identificar problemas, propor ou tomar caminhos alternativos.

Proposta de solução e metodologia

Essa proposta apresenta uma arquitetura baseada em sistema multiagente que permite acompanhar a execução de uma instância de um processo intensivo em conhecimento

aplicando soluções de contorno a problemas identificados.

A arquitetura também permite que situações identificadas como exceção possam ser decididas por um agente externo, ser humano, que pode utilizar o seu conhecimento para definir a solução a ser tomada. Para aumentar a acurácia da decisão do agente externo a arquitetura apresenta informações de contexto daquela instância do processo e uma proposição de ação.

Uma forma de mitigar os riscos elencados no problema de pesquisa é desenhar um processo com caminhos alternativos e integrar o motor de processos (PAIS) com um mecanismo que permita a análise de contexto da execução daquela instância do processo, permitindo que caminhos diferentes sejam selecionados de acordo com as informações de contexto. Ao realizar o mapeamento dos processos de negócio é importante considerar os aspectos relevantes que envolvem o processo, tipos de agentes e de dados envolvidos, seu contexto de execução e possíveis desvios que possam ocorrer durante a execução das tarefas.

Para capturar todos os aspectos importantes de contexto da execução dos processos que devem ser incorporados na evolução da gestão dos processos de negócio, a solução proposta contempla: agentes inteligentes (inteligência artificial - IA) e o conceito de sensibilidade no tratamento do contexto do processo, *Context-Aware*.

No âmbito da metodologia, de acordo com Recker (2021), métodos qualitativos foram desenvolvidos nas ciências sociais para permitir que os pesquisadores investiguem fenômenos em seus contextos da vida real. Nesta ótica, a pesquisa ora apresentada pode ser categorizada como qualitativa pois foca em um estudo de caso real que permite a atracação de uma embarcação de forma segura utilizando a arquitetura proposta.

A organização restante dessa dissertação está distribuída da seguinte forma:

- Capítulo 1: descreve a fundamentação teórica da dissertação com conceitos de *Business Process Management* (BPM), *Knowledge-Intensive Process* (KiP), *Process-Aware Information Systems* (PAIS), agentes inteligentes, multiagentes e BDI.

- Capítulo 2: nesse capítulo é apresentado o mapeamento sistemático da literatura onde são definidas as questões de pesquisa, o protocolo utilizado, a *string* de busca de artigos, os repositórios de artigos utilizados, o processo de seleção dos artigos, comparações de trabalhos e análise dos dados.

- Capítulo 3: nessa parte do trabalho é descrita a abordagem proposta pelo `AAdapt` e seus agentes, a arquitetura, responsabilidade e comunicação dos agentes, a integração e a modelagem do processo de negócio com o PAIS. Apresenta detalhes da troca de mensagens dos agentes com o PAIS, como foi realizada a modelagem dos agentes e a implementação do BDI com o Jadex.

- Capítulo 4: nesse capítulo são apresentados detalhes de como o `AAdapt` é inicializado

e como o agente monitora os elementos de contexto das tarefas do processo de negócio. Apresenta o cenário no qual os agentes foram aplicados, o estudo de caso e os testes executados, a integração com a ontologia do domínio do processo para inferências e informações de contexto e uma discussão sobre o `AAdapt`.

- Capítulo 5: esse capítulo apresenta um resumo dos trabalhos relacionados e um descritivo das diferenças da proposta com os demais trabalhos analisados no capítulo.
- Capítulo 5: são apresentadas as considerações finais do problema de pesquisa proposto, quais objetivos foram atendidos, quais contribuições foram realizadas, novos problemas que foram identificados e os estudos que podem ser desenvolvidos no futuro.

1 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo são introduzidos os conceitos teóricos que formam a base dessa dissertação. São abordadas concepções na literatura sobre *Business Process Management* (BPM), *Process-aware information systems* (PAIS), *Knowledge-Intensive Process* (KiP), multiagentes e agentes inteligentes, modelagem e ontologia. Tais conceitos são facilitadores e embasadores para o restante da pesquisa.

1.1 BPM - *Business Process Management* - Gestão de Processos de Negócio

Business Process Management (BPM) é uma disciplina que permite modelar, analisar, melhorar e mensurar como as tarefas de trabalho são ou podem ser realizadas num processo de negócio para garantir otimização de recursos, sejam eles computacionais, financeiros, de tempo ou de pessoas. É a arte e a ciência de supervisionar como o trabalho é realizado em uma organização para garantir resultados consistentes e aproveitar as oportunidades de melhoria, conforme (DUMAS et al., 2018). BPM visa melhorar os processos de negócio, possivelmente sem o uso de novas tecnologias (AALST; ROSA; SANTORO, 2016).

1.1.1 Importância dos processos de negócio

Os processos de negócio estão presentes em toda a vida da empresa, seja na prestação de um serviço ou na fabricação de um bem. Viriyasitavat, Xu e Bi (2020) argumentam que o gerenciamento de processos de negócio (BPM) visa otimizar os processos de negócio para obter melhor desempenho do sistema, como maior lucro, resposta mais rápida e melhores serviços. Até nas empresas mais simples temos processos de negócio, porém, em muitas delas não há um formalismo e um mapeamento de como o processo deve ser realizado. A atividade de mapear e visualizar graficamente um processo de negócio abre caminhos para análise, discussão, automatização e melhorias no processo.

A execução do processo e sua eficiência afetam diretamente o produto e o cliente que está recebendo o produto. Por exemplo, se um processo produz um excelente produto mas esse produto não é entregue a contento, o cliente vai ficar insatisfeito e vai dizer que o produto é bom mas a entrega é deficiente. Outros poderiam dizer que o produto é saboroso mas não tem uma boa aparência. Outros poderiam devolver uma bicicleta para o vendedor pelo simples fato de um passador de marchas não ter sido verificado na montagem e ter

vindo com defeito ou o modelo ser incompatível com o cassete. Os processos têm impacto direto na atratividade de produtos e serviços, influenciam as experiências do cliente e, em última instância, a receita no caso de empresas (DUMAS et al., 2018).

Todas as falhas descritas nos exemplos acima poderiam ser mitigadas se a empresa tivesse um processo de negócio mapeado, documentado e maduro que permitisse a monitoração da entrega do produto ao cliente, que tratasse os atrasos, que analisasse a aparência do produto e, por fim, um processo que testasse a bicicleta verificando se as marchas estavam trocando corretamente.

Muitos problemas poderiam ser resolvidos antes do produto ou serviço chegar ao cliente, melhorando assim a imagem do produto, a imagem da empresa, a qualidade do produto vista pelo cliente e a redução do custo de produção. Os processos orquestram os recursos corporativos para atender a essas demandas externas e, portanto, são um fator-chave na determinação do custo de serviço e da eficiência operacional (DUMAS et al., 2018).

Não é só custo de produção que pode ser afetado, há também o custo da manutenção do objeto, o custo de logística e logística reversa, administração da empresa, de marketing entre outros que podem ter desperdícios de recursos por uma falha na gestão de processos de negócio. É nesse ponto que muitas vezes algumas empresas se destacam e outras estão fadadas ao fracasso por não terem processos de negócio eficientes. Um processo melhor é, portanto, aquele que melhor contribui para atender aos objetivos estratégicos de uma organização (AALST; ROSA; SANTORO, 2016).

Os processos de negócio envolvem atores que podem ser internos ou externos. Esses atores podem ser pessoas, empresas ou até um sistema que executa uma atividade pelo usuário. Os processos também podem possuir objetos palpáveis (físicos) ou objetos de informação (digitais) como um sensor físico ou um registro em banco de dados.

Para se ter uma ideia da importância do tratamento dos processos de negócio, Ilahi e Ghannouchi (2013) afirmam que uma Gestão de Processos de Negócio eficiente é capaz de garantir o desenvolvimento e competitividade. Pereira e Freitas (2019) arguem que "hoje, as organizações mantêm um foco especial em seus processos de negócio, em um movimento mundial conhecido como BPM (*Business Process Management*)". De acordo com (TER et al., 2010), a área de *Business Process Management* (BPM) recebeu investimentos consideráveis nos últimos anos devido ao seu potencial para aumentar a produtividade e reduzir custos.

Não é por menos que temos mais investimentos nessa área, como a crescente indústria 4.0 requer cada vez mais demandas de produção e produtos mais personalizados, os processos de produção precisam ser cada vez mais dinâmicos e propensos a alterações sem a necessidade de remodelar todo o processo, os sistemas BPM na indústria 4.0 são necessários para digitalizar e automatizar os fluxos de trabalho dos processos de negócio e dar suporte às interoperações transparentes dos fornecedores de serviços (VIRIYASITAVAT;

XU; BI, 2020). Além disso, o tempo no mercado de cada produto está cada vez menor e é necessário que novos produtos entrem no mercado mais rápido e com uma cadeia de produção mais eficiente, o que requer processos mais ágeis. Os processos determinam o potencial e a velocidade de uma organização para se adaptar a novas circunstâncias (DUMAS et al., 2018). E a área de processos de negócio tem um papel importante em toda essa demanda, pois fornece mecanismos que permitem que as informações e os processos das empresas fluam, sejam automatizados, organizados e controlados.

A melhoria dos processos precisa lidar com as restrições organizacionais como pessoas, custos e tempo, com as intempéries que o processo pode sofrer como falhas de equipamentos e de pessoas, mudanças climáticas, mudanças econômicas e ter a capacidade de se adaptar a essas exceções que são apresentadas durante a execução. Além do aspecto organizacional, a melhoria de processos auxilia na interação das pessoas, uma empresa que tem como base a Gestão de Processos de Negócio proporcionará um clima favorável à cooperação de seus colaboradores (ILAHI; GHANNOUCHI, 2013).

A melhoria dos processos começa com o seu mapeamento e pode contemplar desde um fluxo simples como o processo de solicitação de férias de colaborador até um processo complexo como o da produção de um produto ou liberação de um serviço. Essa atividade fica mais complexa à medida que os processos são permeados por tecnologias de acesso e uso heterogêneos como *smartphone*, pulseiras inteligentes, relógios inteligentes, *smartTVs*, *notebooks*, certificação digital, aplicações de propósito específico ou aplicações web.

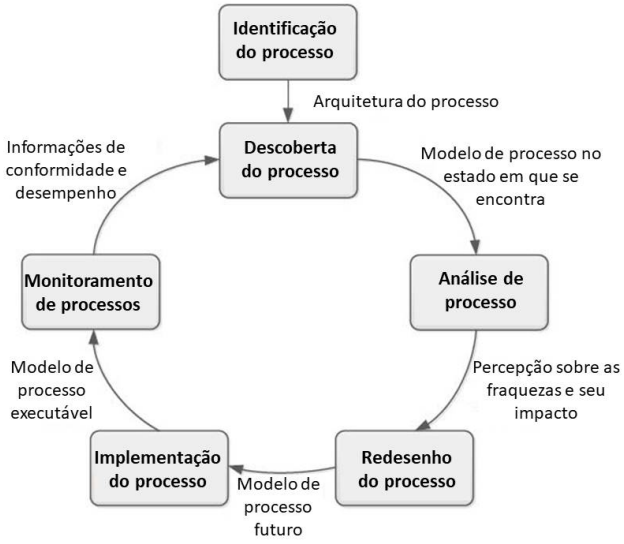
1.1.2 Ciclo de vida dos processos de negócio

Os processos de negócio possuem um ciclo de vida que é uma visão geral esquemática do método de implementação e gerenciamento de processos na organização (SZELAGOWSKI, 2018). A literatura apresenta vários modelos de ciclos de vida de processos na organização, que surgiram dentro da estrutura do conceito tradicional de gerenciamento de processos de negócio (SZELAGOWSKI, 2018). Segundo (DUMAS et al., 2018), um BPM possui um ciclo de vida contínuo de 6 fases compreendido em identificação, descoberta, análise, redesenho, implementação e monitoramento de processos, conforme pode ser observado na Figura 1. As fases são resumidas a seguir, conforme (DUMAS et al., 2018).

A identificação dos processos busca identificar e delimitar o escopo do problema de negócio que será tratado. Durante a fase de descoberta de processos é realizada a documentação de como o processo está sendo realizado na empresa. Na análise dos processos os problemas associados ao processo são identificados, documentados e quantificados e a saída dessa fase apresenta os problemas priorizados conforme sua criticidade e esforço necessários para atingir o seu objetivo.

O redesenho dos processos ou melhoria de processos busca identificar alterações no processo que possibilitem resolver os problemas para atingir os objetivos de medidas de desempenho. A implementação dos processos trata das mudanças necessárias para mudar do estado atual do processo para o estado pretendido pela organização. Um vez que esse processo foi implementado e está em execução entra a fase de monitoração do ciclo de vida na qual as informações são coletadas e analisadas para verificar as medidas e objetivos de desempenho.

Figura 1 - Ciclo de vida de BPM.



Legenda: Ciclo de vida de BPM.
Fonte: Adaptado de (DUMAS et al., 2018)

1.1.3 Situações de exceção e soluções de contorno

O arcabouço tecnológico para apoiar a execução das tarefas de um processo de negócio facilita identificar situações em que as informações de contexto do processo não são as esperadas pelo caminho padrão do processo, as chamadas exceções ou situações de exceção. As situações de exceção são identificadas analisando os valores dos elementos do contexto durante a execução da instância de um processo e compará-los com valores aceitáveis estabelecidos numa ontologia daquele domínio de conhecimento. Na ocorrência de valores anômalos uma situação de exceção é identificada e para cada exceção que pode ser antecipada é possível definir um processo de tratamento de exceção, que inclui primitivas de tratamento de exceção (para remover, suspender, continuar, completar, falhar e reiniciar um item de trabalho) (NUNES et al., 2018b).

Se essas situações de exceção não forem tratadas, podem paralisar o processo e acarretar prejuízos, falhas ou inserir vulnerabilidades no processo. Além disso, mudanças dinâmicas geralmente ocorrem em um ambiente em tempo de execução, e não podemos abortar processos de negócio e reiniciá-los do início após as mudanças (KALIBATIENE; VASILECAS, 2021). Uma forma de mitigar esses problemas é construir mecanismos que permitam tratar essas exceções em processos intensivos em conhecimento propiciando uma solução de contorno.

Os mecanismos que tratam as exceções dos processos de negócios podem ser módulos, agentes ou sistemas multiagentes escritos numa linguagem de programação. Durante a execução do processo de negócio, a cada tarefa, as informações de contexto do processo podem ser analisadas e comparadas com lineares conhecidos para identificar situações indesejadas, o que permite que ações sejam realizadas e caminhos alternativos executados para que o processo seja adaptado durante a sua execução. O processo pode ser alterado em tempo de execução adicionando, substituindo ou removendo fragmentos, ou modificando seus relacionamentos, o que permite que o processo seja adaptado dinamicamente (KALIBATIENE; VASILECAS, 2021).

Essa definição para que um processo seja adequado é uma flexibilidade que acomoda as exceções ocorridas durantes a execução das tarefas. A flexibilidade também é necessária para processos de negócio que estão em evolução, permitindo assim que novos caminhos sejam mapeados e explorados.

1.1.4 Utilização de informações de contexto em processos de negócio

Contexto é um conjunto de informações e circunstâncias de um determinado assunto ou objeto, sendo usadas para definir o status de um ambiente (de Matos et al., 2020). As informações de contexto consistem em vários tipos de subconstruções, como localização, identidade, atividade, tempo e ambiente, etc (LEE; PARK; KIM, 2013). Num contexto marítimo, por exemplo, iremos encontrar elementos contextuais relacionados a navios, marinheiros, água, peixes, velocidade do vento, boreste, bombordo, maré, entre outros. São informações de um ambiente, de um cenário contextualizado.

As informações de contexto auxiliam na execução de uma instância de processo de negócio, principalmente nos processos de conhecimento intensivo por serem mais dinâmicos e flexíveis, irrigando informações do ambiente que permitem antecipar situações que comprometam a execução da instância. A execução de um processo de negócio ocorre dentro de um contexto que pode ser entendido por meio de informações relevantes, que podem ser analisadas e gerenciadas permitindo que decisões sejam tomadas de acordo com informações mais corretas possíveis obtidas previamente ou durante a execução do processo.

O gerenciamento de contexto envolve a definição de padronização e de mecanismos para apoiar a captura e manipulação das informações, o que corrobora com a importância da padronização para identificar e categorizar informações de contexto. Essa padronização pode ser obtida por meio da criação de uma ontologia que defina os domínios, ações, restrições e tipos de dados.

1.2 *Process-Aware Information Systems - PAIS*

Process-Aware Information Systems são ferramentas tecnológicas que apoiam a implementação de execução de processos de negócio numa empresa. São *frameworks* para automação de processos de trabalho. De acordo com Aalst (2009), PAIS é um sistema de software que gerencia e executa processos operacionais envolvendo pessoas, aplicativos e / ou fontes de informação com base em modelos de processo.

Alguns exemplos de sistemas PAIS são sistemas de gestão de fluxo de trabalho, sistemas de gerenciamento de conteúdo, sistemas de arquivamento e controle de versões de arquivos e sistemas ERP e CRM. Essas ferramentas possibilitam a automatização e tratamento de processos de negócio, possibilitando uma compreensão melhor do trabalho a ser realizado. Algumas ferramentas disponíveis no mercado nessas categorias são o Pipefy (PIPEFY, 2021), YAWL (FOUNDATION, 2021) e Camunda (CAMUNDA, 2021) na gestão de fluxo de trabalho, IBM *Content manager* (IBM, 2021a) e Oracle URM (ORACLE, 2021b) para gestão de conteúdo, IBM *Record Manager* (IBM, 2021b) e OpenKM (S.L., 2021) na área de arquivamento e controle de versões de arquivos, TOTVS e SAP na modalidade de ERPs e Pipedrive (PIPEDRIVE, 2021) e OpenText (OPENTEXT, 2021) na categoria de CRMs.

Os sistemas PAIS devem ser capazes de tratar incertezas, situações anormais e mudanças ambientais na execução da instância de um processo de negócio. De acordo com Schwerz et al. (2021) os desvios e a recuperação de erros dos caminhos esperados continuam sendo um desafio significativo porque para cada caminho esperado, existem muitas exceções e falhas potenciais que podem surgir, algumas esperadas e muitas inesperadas. Ainda para Schwerz et al. (2021), esta estrutura do PAIS reflete as aplicações que modelam, que normalmente têm um objetivo bem definido e um caminho bem comportado para atingir esse objetivo o que é confirmado por Reichert, Rinderle-Ma e Dadam (2009) que argumentam que a agilidade de negócios necessária geralmente é prejudicada pela falta de flexibilidade do PAIS existente e uma vez que um processo é implementado, sua lógica não pode mais ser adaptada ou refinada.

1.3 **KiP - *knowledge-intensive process***

Hagen, Ratz e Povalej (2005) classificam os processos de negócio em estruturados, semiestruturados ou não estruturados em relação à estrutura do processo e ao fluxo das atividades.

Os processos estruturados são totalmente pré-definidos, facilmente modelados em uma linguagem específica, como *Business Process Model and Notation* (BPMN), e repetitivos, tendo uma sequência fixa de atividades. Estes processos geralmente se baseiam em um fluxo de controle rígido, definindo-os como uma composição de atividades bem estruturadas ou outros processos (subprocessos) que uma organização realiza para atingir seus objetivos (DUMAS et al., 2018). Cada atividade pode ser composta de tarefas e possui artefatos de entrada e saída, recursos necessários e os procedimentos (métodos, técnicas) a serem seguidos quando são executados.Exemplos de processos estruturados são pedidos de atendimento, entregas, controle de estoque e folha de pagamento.

Por outro lado, os processos não estruturados (ou *ad hoc*) constituem um processo que muda frequentemente, sendo as suas instâncias muito diferentes entre si, tanto em termos de atividades realizadas como de fluxo. Sua natureza traz dificuldade adicional para modelar com um método ou notação tradicional. Finalmente, um processo semiestruturado compartilha partes estruturadas e não estruturadas, compartilhando características de ambos os tipos de processo em diferentes partes de seu fluxo.

Os processos não estruturados são frequentemente associados aos chamados Processos Intensivos em Conhecimento (Knowledge-intensive Process - KiP). Entre as diversas definições encontradas na literatura, Vaculin et al. (2011) afirmam que KiP é um "processo cuja condução e execução dependem fortemente de trabalhadores do conhecimento realizando várias tarefas interconectadas de tomada de decisão intensivas em conhecimento". Os KiPs são genuinamente centrados no conhecimento, nas informações e nos dados e requerem uma flexibilidade substancial tanto no tempo de design quanto na execução.

Expandindo esta breve definição, os estudos de Unger, Leopold e Mendling (2015) e Ciccio, Marrella e Russo (2014) discutem um conjunto de nove características comuns para KiPs: prevalência de conhecimento, colaboração, imprevisibilidade, complexidade, orientação para objetivos, orientação para eventos. Boissier, Rychkova e Grand (2019) argumentam que são características KiP a incorporação de muito conhecimento com tarefas complexas, a requisição de colaboração humana para executar o processo, alto grau de incerteza na execução das atividades e que o KiP requer ideias originais e improvisação. Eventos internos e externos podem afetar a qualidade das informações trocadas durante a execução de um KiP ou exigir que um participante reaja para a realização bem-sucedida do objetivo pretendido do processo KiP. Além disso, o fluxo exato e a ordem das atividades durante a execução de cada instância de um KiP depende de vários fatores situacionais e contextuais, bem como de possíveis eventos externos que afetam seus participantes. KiPs

tendem a ser menos repetíveis do que processos não KiP, portanto, uma repetição exata de uma instância, em termos de fluxo e atividades de um KiP executado anteriormente, parece dificilmente possível devido a vários fatores que afetam cada execução ou instância específica.

As informações necessárias para a execução do KiP geralmente vêm do contexto do processo ou da própria atividade, como dados do aplicativo, dados do processo, dados funcionais, etc. Ao realizar uma tarefa, uma pessoa frequentemente consulta recursos e sua seleção é baseada em vários fatores, tais como suas habilidades pessoais, experiências ou preferências (BRANDER et al., 2011). Além disso a execução de um KiP é causalmente dependente das crenças, desejos e intenções que são inerentes aos participantes do processo, e historicamente dependente das interações comunicativas entre os participantes do KiP.

Santoro e Baiao (2018) descrevem que é necessário integrar fontes heterogêneas de dados para obter informações sobre os processos, podendo tomar ação para identificar e corrigir ações. O conhecimento pode ser diferente em cada caso de execução, portanto, os processos não estruturados requerem um entendimento mais aprimorado dos dados e dos fluxos de controle.

Esse conhecimento gerado durante a execução das tarefas de um processo é muito valioso porque muitas informações que podem ser um diferencial competitivo podem estar dispersas devido a sua heterogeneidade. Assim, novas abordagens se tornaram necessárias para coletar, tratar e interpretar informações oriundas das mais diversas fontes.

Os processos são considerados intensivos em conhecimento quando há uma incerteza no resultado de sua execução, o que muitas vezes depende do seu ambiente e contexto de execução. Essa incerteza pode estar associada a diversos fatores, como fatores ambientais, número de execução de casos, pessoas, desgastes físicos, problemas energéticos. As incertezas levam a necessidade de planejamento de processos que consigam se adaptar durante o processo de execução. Para isso há a necessidade de coletar informações do contexto na execução do processo e construir mecanismos capazes de tratar desvios no processo.

Percebe-se neste contexto que o ambiente de negócio está cada vez mais heterogêneo, saindo das limitações da tradicional aplicação, servidores, banco de dados e clientes de acesso que ficavam restritos aos domínios para um ambiente mais disperso onde muitas vezes os servidores e aplicações estão na nuvem e são dimensionados dinamicamente e com uma diversidade crescente de dispositivos de acessos. Os ativos para o acesso à informação ficaram mais diversificados e cada vez mais longe dos domínios da empresa.

Desta forma, a coleta de informações de contexto em processos intensivos em conhecimento possibilita um amadurecimento constante do processo, com detalhes mais ricos, permitindo mais subsídios para escolha do caminho a ser seguido. A coleta e monitoramento de informações de contexto são mecanismos importantes na evolução dos

processos modernos.

1.4 **Agente e sistemas multiagentes**

Russell e Norvig (2020) definem que um agente é tudo o que pode ser considerado capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por meio de atuadores. Numa definição simples, um agente é um componente de software que opera em um ambiente e é autorizado a executar ações como coleta de informação, comunicação, ou manipular o ambiente. O agente percebe e interage com o ambiente que conhece, podendo identificar situações a partir da coleta de informações e por fim realizar uma ação. São códigos escritos numa linguagem de programação como `Java` ou C++ que tomam decisões baseadas nas restrições e regras que foram informadas durante a sua programação. Muitas dessas regras estão numa base de dados que podem ser manipuladas pelo próprio agente adicionando ou alterando instruções conforme o agente vai aprendendo sobre o ambiente em que ele interage, diz-se então que o agente é inteligente devido a sua capacidade de tomar decisões baseadas nas informações e diretrizes que ele próprio obteve do ambiente.

Os sensores são componentes importantes no trabalho com agentes, eles podem ser tanto dispositivos físicos quanto sensores lógicos, de software. Normalmente cada agente está associado a um sensor porém há diversos casos em que os agentes recebem informações de vários sensores. Normalmente os agentes não possuem o conhecimento de todo o ambiente, ao invés disso, eles costumam conhecer uma parte do ambiente e são responsáveis por ações restritas no ambiente em que operam.

Wooldridge (2009) argumenta que um agente normalmente não tem controle completo do seu ambiente mesmo em domínio de complexidade razoável. A explicação para esse fato é que normalmente os agentes são modelados para atender determinadas operações e, em alguns casos, tratar as exceções que forem identificadas na execução do agente.

Os ambientes de obtenção de informação e de comunicação heterogêneas são mais propícios a apresentar situações que não foram identificadas no momento do *design* e desenvolvimento do agente. Para mitigar essas falhas e tratá-las, uma opção é criar agentes inteligentes que aprendam ações e desvios que podem ser tomados no ambiente.

Para a interação com o ambiente o agente precisa capturar o estado de execução, para isso o agente se comunica com os sensores ou outros mecanismos que captam constantemente informações do estado dos componentes de um objeto. Dependendo da implementação, essa comunicação pode ser tanto acionada ou consultada pelo agente, informada constantemente pelo sensor ou até mesmo acionada a comunicação pelos dois lados.

As ações do agente dependem das informações recebidas e não de informações que os sensores ainda não enviaram para os agentes, embora, em muitos casos os agentes possam antever cenários baseados em tendências comportamentais do ambiente. Outro aspecto a ser considerado é que a ausência de informações perceptivas pelo agente pode disparar ações de alerta ao usuário de que os sensores estão falhando ao coletar dados do ambiente.

Os agentes geram sequências de ações conforme as percepções do ambiente, passando assim por uma sequência de estados. Se essa sequência de estados estiver dentro das medidas de desempenho se entende que o agente teve um bom desempenho. O estado é sempre do ambiente e não do agente, (RUSSELL; NORVIG, 2020). Assim, o estado do ambiente pode mudar acarretando em ações do agente, como por exemplo, um ambiente que ficou mais quente pode acionar um agente de alteração de temperatura, a mudança da velocidade do vento impede a atracação de barcos. Um agente deve selecionar a ação para maximizar sua medida de desempenho de acordo com as percepções do ambiente obtidas pelo conhecimento internalizado do agente e a coleta de informações. Russell e Norvig (2020) definem que um agente racional deve selecionar uma ação que se espera que venha a maximizar sua medida de desempenho.

Com o processo de coleta de informações do ambiente e a sua utilização no aprendizado, o agente modifica a sua percepção do ambiente e toma novas ações de acordo com as novas percepções. Muitos conhecimentos, ou conhecimentos iniciais, do ambiente são inseridos no agente pelo seu projetista, para permitir que o agente comece a operacionalizar no ambiente. Dependendo da construção do agente, ele pode ser utilizado para finalidades semelhantes, mudando apenas alguns parâmetros, maximizando assim a sua utilidade (WOOLDRIDGE, 2009). Nesse ponto integrações com ontologias podem auxiliar na inferência de ações.

Um agente ao possuir a capacidade de aprender com a coleta de informações e as percepções do ambiente, foi construído com princípios de inteligência artificial (IA). Esses agentes podem, por exemplo, acessar registros (*logs*) do sistema para facilitar o aprendizado podendo, inclusive, substituir antigas premissas por novas informações. Corroborando com isso, Russell e Norvig (2020) afirmam sobre outra vantagem do aprendizado, a de permitir ao agente operar em ambientes inicialmente desconhecidos e se tornar mais competente do que seu conhecimento inicial sozinho poderia permitir.

Os problemas aos quais os agentes serão a solução são conhecidos como ambiente de tarefas e eles possuem vastas propriedades quanto a utilização de inteligência artificial. O projetista de agentes com inteligência artificial deve ter o domínio do conhecimento do problema e especificar o ambiente de tarefa o mais claro e completo possível.

Russell e Norvig (2020) detalham que o ambiente de tarefa possui propriedades que auxiliam no entendimento do comportamento e no processo de projetar o agente com IA conforme Tabela 1.

Tabela 1 - Ambiente de tarefa e suas propriedades.

Propriedade	Comportamento
Sensores completamente observáveis	Permitem acesso completo ao ambiente em cada instante
Sensores parcialmente observáveis	Permitem acesso restrito ao ambiente
Agente único	Agente resolve sozinho, sem interação, um determinado problema
Multiagente	Agente interage com outros agentes para obter um resultado
Multiagente competitivo	Agentes disputam entre eles recursos
Multiagente cooperativo	Agentes que cooperam na obtenção do resultado
Ambiente determinístico	O ambiente é completamente determinado
Ambiente estocástico	O ambiente é indeterminado
Ambiente dinâmico	Pode alterar enquanto o agente está deliberando
Ambiente estático	Ambiente não altera na execução do agente
Ambiente episódico	Ambiente onde uma tarefa não influência na outra
Ambiente sequencial	Ambiente onde as tarefas dependem umas das outras
Ambiente conhecido	O agente conhece o ambiente onde irá execução uma ação
Ambiente desconhecido	O agente vai ter que aprender para tomar decisões
Ambiente discreto	Há um número finito de estados distintos para serem tratados
Ambiente contínuo	Quando o agente tem que lidar com grandezas contínuas ligadas ao ambiente

Legenda: Propriedades de comportamento do ambiente de tarefa.
Fonte: Traduzido e adaptado de RUSSELL; NORVIG

Depois de definida a parte comportamental dos agentes há a necessidade de descrever o funcionamento interno dos agentes e, nesse caso, segundo Russell e Norvig (2020), o trabalho da IA é projetar o **programa do agente** que implementa a função do agente – que mapeia percepções em ações.

Nesse entendimento um agente é composto de arquitetura e de programa e há quatro tipos básicos de agentes: reativos simples, reativos baseados em modelos, baseados em objetivos e baseados na utilidade.

Há os agentes que se comunicam com outros agentes e formam um ambiente multiagente (*Multi-Agent Systems* - MAS), que são várias entidades autônomas interagindo entre si e com ambiente. Um sistema de multiagente consiste de recursos compartilhados ou serviços que muitas vezes é implementado como um componente de software externo, conforme (SHEHORY; STURM, 2014). Ertel (2011) também afirma que um sistema multiagente consiste em vários agentes que interagem entre si, geralmente trocando mensagens por meio de alguma infraestrutura de rede de computadores.

Um ambiente de sistema multiagente fornece artefatos ou serviços aos agentes e permitem interação, coordenação e troca de mensagens entre eles para a consecução de suas atividades. Essas mensagens precisam de um padrão para a interoperabilidade entre os agentes, sendo os dois mecanismos mais utilizados de troca de mensagens o *Agent Communication Language* (ACL) da *Foundation for Intelligent Physical Agents* (FIPA) e o *Knowledge Query and Manipulation Language* (KQML).

Os sistemas multiagentes são modelados numa metodologia orientada a agentes, como por exemplo a metodologia *Multiagent Systems Engineering* (MaSE) e a sua evolução *Organization-based Multiagent Software Engineering* (O-MaSE). Essas metodologias de desenvolvimento de software possuem processo de desenvolvimento do software basicamente composto por análise de requisitos, hierarquia de objetivos, casos de uso ou cenários, funções, agentes e suas conversas, (HINDRIKS, 2014).

Cada agente possui o seu próprio objetivo e em um ambiente multiagente, os agentes competem por recursos e seus objetivos podem ser conflitantes. Para tratar esses conflitos são utilizados controles como técnicas de sincronização, modelos organizacionais e sistemas normativos, ficando o ambiente do agente regulado por normas e conceitos organizacionais.

Um ambiente de sistema multiagente é um arcabouço de recursos, artefatos e serviços compartilhados que são disponibilizados para os clientes. Seu gerenciamento utiliza um software, um *framework*, que requer uma linguagem de programação para efetivar as implementações dos conceitos relacionados e as abstrações de recursos, serviços e objetos, permitindo assim a interação, coordenação e troca de mensagens entre agentes.

1.4.1 Agentes Inteligentes

Agentes inteligentes são componentes de software que podem possuir instruções iniciais e a partir das lições aprendidas com as interações aprimorar o seu conhecimento e serem cada vez mais assertivos no seu ambiente. Esse método de aprendizado de máquinas, segundo Russell e Norvig (2020), começou em 1950 quando TURING (1950) considerou a ideia de realmente programar as suas máquinas inteligentes à mão. Os agentes com aprendizagem (IA - inteligência artificial) são considerados o estado da arte por Russell e Norvig (2020). Esses agentes operam em ambientes desconhecidos e com poucas ou sem informações sobre o ambiente vão tomando ações de acordo com premissas e aprendizado do ambiente, se tornando cada vez mais eficazes à medida que ocorrem mais interações.

Wooldridge e Jennings (1995) afirmam que um agente inteligente normalmente tem as seguintes propriedades: reatividade, proatividade e habilidade social. A **reatividade** é a capacidade inerente aos agentes inteligentes de perceber mudanças no ambiente e conseguir responder a essas mudanças em um tempo satisfatório, por exemplo, não adianta detectar um princípio de incêndio após a chegada do corpo de bombeiros. A **proatividade** é uma característica dos agentes inteligentes de conseguir perceber ações e tendências antes que uma situação efetivamente ocorra. Essa capacidade é difícil de ser obtida, requer análise constante da tendência dos dados que chegam para o agente analisar. A **habilidade social** é a capacidade dos agentes inteligentes interagirem com outros agentes e outros ambientes visando troca de mensagens e conhecimento com o objetivo de atender as suas crenças e suas metas. A **autonomia** dos agentes é obtida quando os agentes são capazes de tomar decisões sem intervenção direta de humanos, controlando suas ações somente com os códigos de programação e seu conhecimento do ambiente em que operam. "Quando um agente se baseia no conhecimento anterior de seu projetista e não em suas próprias percepções, dizemos que o agente não tem **autonomia**."(RUSSELL; NORVIG, 2020).

Um agente inteligente é um código escrito em linguagem de computação que possui algumas das propriedades identificadas por Wooldridge e Jennings (1995), podendo apresentar conceitos de *Belief, Desire and Intention* para atingir seus objetivos com o ambiente. Wooldridge e Jennings (1995) afirmam que para muitos pesquisadores, essa ideia de programar sistemas de computador em termos de noções como crenças, desejos e intenções é um componente essencial dos sistemas baseados em agentes.

1.4.2 **BDI -*Belief, Desire and Intention***

Os agentes com BDI são um tipo de agente inteligente com capacidade de tomada de decisão e possuem raciocínio baseado em crenças, desejos e intenções. O paradigma do

agente com *Belief, Desire and Intention* (BDI) fornece uma base poderosa para o desenvolvimento de sistemas complexos baseados em agentes inteligentes autônomos (WATERS; SARDINA, 2015).

Esses agentes utilizam uma base de conhecimento prévia para chegar a conclusões, acionar ou elaborar um plano e para tomar decisões. Esse mecanismo para chegar às conclusões é chamado de mecanismo de inferência por Ertel (2011). A inferência é o mecanismo que permite que os agentes cooperem deliberadamente ou interajam com outros agentes.

Resumidamente, para Norling (2004) um agente que é caracterizado por suas crenças, objetivos (desejos) e intenções terá a intenção de fazer o que acredita que alcançará seus objetivos, dadas suas crenças sobre o mundo. Ainda segundo Norling (2004) um agente BDI é geralmente considerado como tendo uma biblioteca de planos - um conjunto de 'planos como receitas' que pode ser usado para atingir objetivos específicos, dadas condições específicas.

As crenças, objetivos e intenções de um agente são mantidos e gerenciados pelo mecanismo de raciocínio do BDI, permitindo sua manipulação para atualizar as *Belief* e *Goals* e em alguns caso os *Plans* (*Intentions*). As *Beliefs* são as crenças que o agente possui, é o conhecimento detido por ele sobre o ambiente no qual opera, as crenças representam as informações sobre o que o agente conhece do ambiente onde opera. *Belief* representam as informações sobre si mesmo e o ambiente (SANCHEZ-LOPEZ; CEREZO, 2019). Bratman (1999) argumenta que as vezes pensamos na crença como uma questão de tudo ou nada, e às vezes como essencialmente uma questão de grau, mas que ambas as formas de classificar os fenômenos cognitivos têm sua utilidade e que é um erro tentar reduzir uma à outra. Um agente consiste em uma base de crença que codifica o conhecimento do agente sobre o mundo (WATERS; SARDINA, 2015).

Rao e George (1995) distinguem as crenças da noção de conhecimento, uma vez que as crenças do sistema são necessárias apenas para fornecer informações sobre o estado provável do ambiente. Um sistema de agentes precisa ter crenças precisas para que o agente possa acionar os planos de execução mais adequados para a situação identificada. As crenças, dependendo da linguagem de programação, podem ser objetos e podem ser armazenadas em uma base de crença e referenciadas em expressões, bem como acessadas e modificadas a partir de planos usando a interface da base de crença.

Desire está relacionado aos objetivos, metas ou os desejos a serem alcançados e podem conter várias intenções, representam o estado ideal do ambiente para o agente. Os desejos (*desire*) representam o estado motivacional do sistema, uma meta representa um objetivo que um agente deve se esforçar para alcançar. *Desire* representa o estado motivacional (SANCHEZ-LOPEZ; CEREZO, 2019). Plano ou *Intention* é o propósito que um agente possui, é parte de um desejo que pode conter várias intenções. As intenções não podem ir contra suas crenças ou que impeça o agente de alcançar a meta que a intenção

representa. As intenções representam compromissos ou os planos para atingir os desejos e metas e requerem uma certa pró-atividade, característica que não se percebe nas *desires* e *beliefs*. De acordo com Rao e George (1995) um dos pontos importante nos planos é a possibilidade de ações que reajam a situações específicas. Ainda para Rao e George (1995) a construção de planos mais específicos aciona planos baseados nas suas características e sensíveis ao contexto ao serem invocados o que facilita o desenvolvimento modular e incremental.

Rao e George (1995) argumentam que para cada desejo acessível no mundo, existe uma correspondente intenção acessível no mundo que contém apenas o melhor curso de ação. Malle, Moses e Baldwin (2001) afirmam que uma intenção não pode ser reduzida a este par crença-desejo nem são atribuições de intenção apenas abreviações para atribuições de desejos elaborados. Na mesma linha Bratman (1984) entende que este é um modelo redutor que vê as intenções de agir como redutíveis a certos desejos e crenças. E Bratman (1984) afirma que as intenções são estados mentais distintos, que não devem ser reduzidas a grupos de desejos e crenças.

Norling (2004) afirma que uma intenção é formada quando o agente se compromete com um plano particular - uma sequência particular de etapas a serem executadas - desse conjunto para atingir um objetivo. As intenções são traduzidas em planos que são as etapas concretas para se aproximar do cumprimento da meta e são o mecanismo pelo qual o agente age no ambiente. Elas são pré-definidas no desenvolvimento do agente e fazem parte da biblioteca de atividades executáveis. Os agentes podem tentar atender diversos objetivos simultaneamente, podendo em muitos casos ocorrerem conflitos entre eles. Um aspecto do comportamento racional é que os agentes podem perseguir objetivos múltiplos em paralelo (POKAHR; BRAUBACH; LAMERSDORF, 2005). As *goals deliberation* tratam e resolvem esses conflitos. As *goals deliberation* são os desejos que o agente deseja perseguir, tratando do pensamento estratégico decidindo quais desejos devem ser realizados.

O mecanismo de *Means-end-reasoning* é como o agente decide atingir uma meta e é acionado quando uma meta é ativada, o mecanismo continua executando planos até que a meta seja alcançada ou não haja mais planos para serem executados. Para ser coerente entre os meios e os fins em um determinado momento, os planos devem incluir subplanos sobre como atingir os fins pretendidos (BRATMAN, 1981). "O raciocínio por *Means-end-reasoning* está no cerne dos sistemas PRS (*procedural reasoning system*) e representa o processo de como um objetivo é perseguido."(POKAHR et al., 2014).

1.4.3 Comunicação entre os agentes

Em um ambiente com múltiplos agentes há a necessidade de interoperabilidade entre eles. Para permitir essa comunicação vários mecanismos podem ser utilizados, entre eles temos algumas linguagens para a comunicação entre agentes como a *Agent Communication Language* (ACL), proposta pela *Foundation for Intelligent Physical Agents* (FIPA), a *Knowledge Query and Manipulation Language* (KQML) e a comunicação por *sockets* TCP (*Transmission Control Protocol*). De acordo com Lizán e Rizo-Maestre (2017) FIPA é uma organização de padrões da IEEE *Computer Society* que promove a tecnologia baseada em agentes e a interoperabilidade de seus padrões com outras tecnologias. A FIPA é uma coleção de padrões abertos que se destinam a promover a interoperabilidade entre agentes de ambientes heterogêneos por meio de troca de mensagens que seguem um padrão.

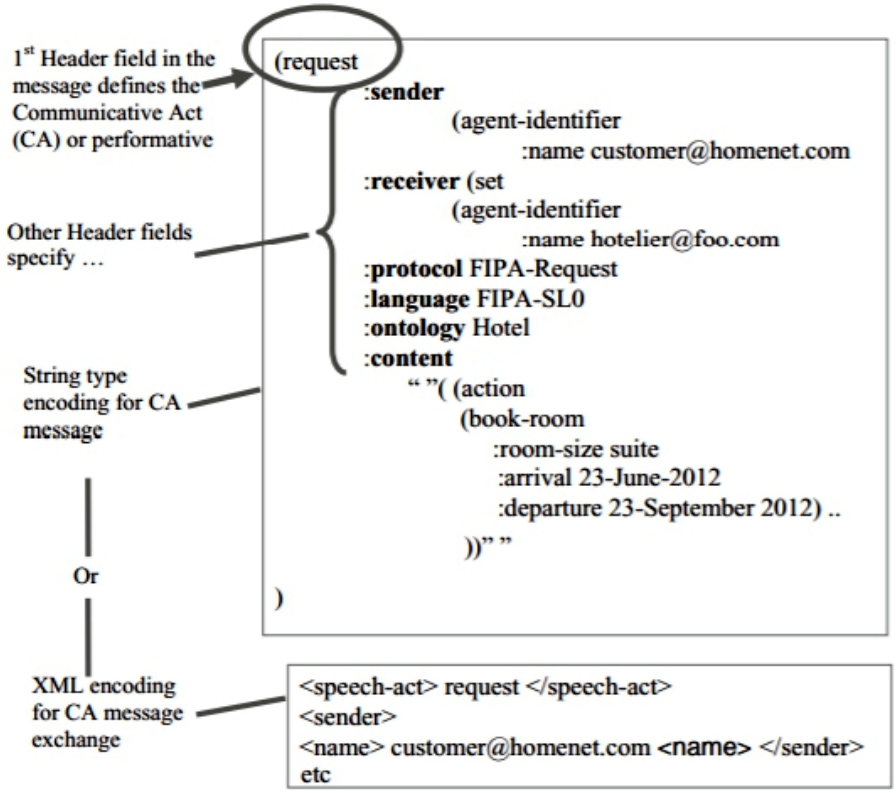
O fórum de padrões FIPA se concentra na especificação para interação externa, serviços de plataforma e comunicação multiagentes, voltado mais para o processo de interoperabilidade externa em detrimento do comportamento do agente interno. O comportamento interno é bem mais fácil de ser observado e controlado que a sua interação externa. A FIPA trabalhou para produzir especificações públicas de MAS, atuando como um habilitador-chave para oferecer suporte a interoperabilidade, interação de serviço aberto e para apoiar o desenvolvimento heterogêneo (POSLAD, 2007).

As especificações do FIPA *Agent Communication* lidam com mensagens numa linguagem de comunicação de agentes *Agent Communication Language* (ACL), definindo os protocolos de interação de troca de mensagens. A FIPA define uma ontologia para a especificação do serviço e a descrição da *Agent Communication Language* (ACL). Uma mensagem ACL FIPA contém um ou mais parâmetros de mensagem como *sender, receiver, reply-to, content, language, encoding, protocol, ontology*, como pode ser visto na Figura 2.

Outra linguagem e protocolo de comunicação entre agentes é a KQML (*Knowledge Query and Manipulation Language*). A linguagem KQML é patrocinada pela DARPA (*Defense Advanced Research Projects Agency)* e é focada no desenvolvimento de técnicas e ferramentas para promover o compartilhamento de conhecimento em sistemas inteligentes (CHALUPSKY et al., 1993). É uma linguagem que facilita a comunicação e a interoperabilidade entre os agentes de software de coordenação (JAGGA; JUNEJA; SINGH, 2015).

A KQML é um formato de mensagem e um protocolo de tratamento de mensagens para oferecer suporte ao compartilhamento de conhecimento em tempo de execução entre os agentes. A KQML pode ser usada como uma linguagem para um programa aplicativo interagir com um sistema inteligente ou para dois ou mais sistemas inteligentes compartilharem conhecimento em apoio à resolução cooperativa de problemas. A linguagem

Figura 2 - Especificação protocolo FIPA.

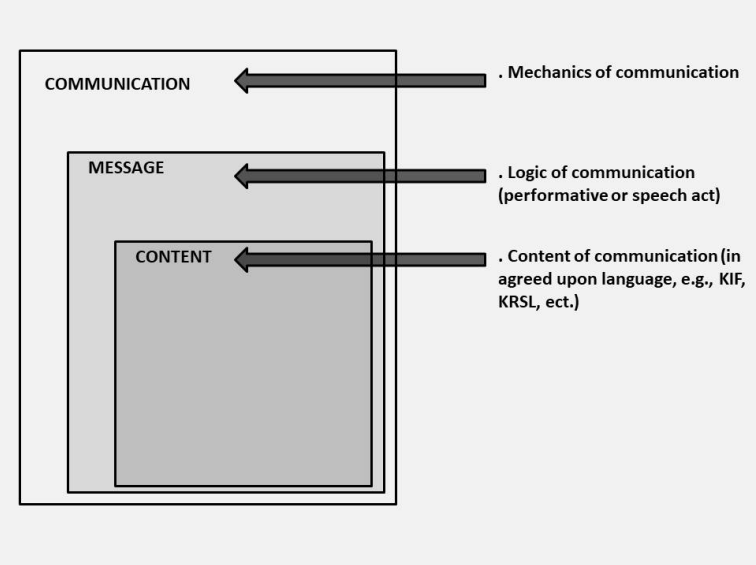


Legenda: Especificação protocolo FIPA.
Fonte: (POSLAD, 2007).

KQML fornece uma estrutura básica para compartilhamento de conhecimento por meio de uma classe especial de agente chamado facilitadores de comunicação que coordenam as interações de outros agentes (FININ et al., 1994). As especificações do KQML se concentram na perspectiva, significado, sintaxe, semântica, cobertura e contexto da comunicação para conduzir ao resultado final derivado da comunicação (JAGGA; JUNEJA; SINGH, 2015).

A linguagem pode ser vista como camadas, Figura 3, tendo uma camada destinada aos mecanismos necessários para comunicação, outra para manipular a lógica da comunicação que define como será a interação com os agentes e a camada de conteúdo da comunicação onde, de acordo com Finin et al. (1994), inclui recursos opcionais que descrevem o conteúdo: seu idioma, a ontologia que ele assume e algum tipo de descrição mais geral, como um descritor que nomeia um tópico dentro da ontologia.

Figura 3 - KQML em camadas.



Legenda: Divisão da linguagem KQML em camadas.
Fonte: Adaptado de (FININ et al., 1994).

1.5 **Modelagem dos agentes**

Modelagem é o processo de desenvolvimento abstrato de um sistema que explica suas características, permitindo a criação de modelos que apresentam perspectivas diferentes do componente que está sendo desenvolvido. Esses modelos ajudam o desenvolvedor na construção dos artefatos pois permitem especificar a estrutura e o comportamento do software que está sendo desenvolvido. Existem várias metodologias para o desenvolvi-

mento de componentes agentes, cada qual com suas especificidades.

1.5.1 *Organization-based Multiagent Software Engineering* O-MaSE

Nesse trabalho a metodologia escolhida para modelar os agentes foi a O-MaSE por possuir clara definição das metas e tarefas e permitir a integração com o Java por *plug-in*. O-MaSE é uma técnica para analisar e desenhar sistemas multiagentes baseada em organizações, com base na metodologia MaSE e adaptado do *OPEN Process Framework* (OPF), que adota uma abordagem de engenharia de método para a construção de processos (LUCK; PADGHAM, 2008).

O objetivo cerne da metodologia O-MaSE é permitir facilidades e padrões no desenvolvimento de agentes. A metodologia O-MaSE possui um conjunto de tecnologias concretas para uma ampla aceitação da indústria, sendo orientada a agente, personalizável com conceitos consistentes e bem definidos suportados por *plug-ins*. O O-MaSE possui um metamodelo, um conjunto de fragmentos de métodos e um conjunto de diretrizes que permitem aos usuários usarem técnicas de engenharia para criar processos personalizados. As principais vantagens encontradas da metodologia O-MaSE foram:

- utilização de um conjunto de fragmentos de métodos baseados em um meta-modelo comum;
- permite a definição explícita de atividades e tarefas;
- permite que as atividades sejam organizadas de maneiras diferentes com base na necessidade do projeto;
- definição personalizada do conjunto de fases e iterações.

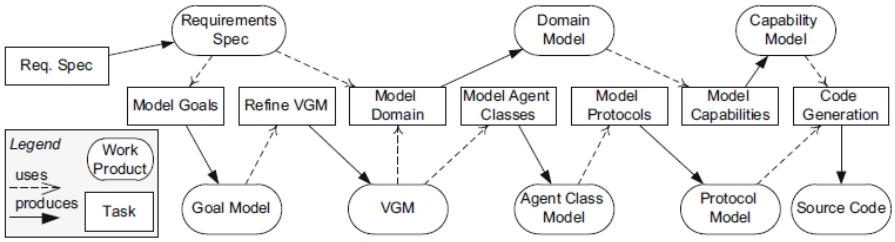
Além disso a metodologia O-MaSE possui um *plug-in* para a IDE (*Integrated development environment*) Eclipse, que é um ambiente de desenvolvimento integrado usado na linguagem de programação orientada a objetos `Java` que facilita a construção dos objetos gráficos. A metodologia O-MaSE permite criar diversos modelos de representação gráfica como *Goal Model, Organizational Model, Role Model, Domain Model, Protocol Model, Agent class Model, Policy Model, Capability Model, Action Model* e *Service Model*. Abaixo são descritos os modelos de metas (*Goal*), funções (*Role*) e classe de agentes (*Agent class*).

O modelo de metas objetiva organizar as metas das diferentes funções do sistema sinalizando as ações que podem ser realizadas pelo sistema. O modelo de metas define um conjunto de metas de nível de sistema na forma de uma árvore de metas (LUCK; PADGHAM, 2008). O modelo começa com a meta principal da organização e vai se decompondo em outras metas que sustentam a meta principal.

O modelo de funções permite identificar e documentar as funções na organização. Para garantir a consistência cada meta folha (*Goal*) deve ser suportada por pelo menos uma função (*role*). O modelo de funções também identifica as interações entre as funções, bem como aquelas com atores externos (SHEHORY; STURM, 2014). O modelo de funções se concentram na definição de funções que cumprem uma ou mais metas (LUCK; PADGHAM, 2008).

O modelo de classes de agentes tem por objetivo identificar os tipos de agentes que participam da organização. No modelo de classes de agentes pode ser identificado um agente sistêmico, como por exemplo um PAIS, que pode interagir com os agentes. O modelo de classes de agentes é um produto da tarefa de modelar classes de agentes e usa a tarefa de modelar protocolos, Figura 4.

Figura 4 - Fluxo da modelagem.



Legenda: Processo compatível com O-MaSE onde setas sólidas denotam linhas tracejadas.
Fonte: (SHEHORY; STURM, 2014)

No capítulo de implementação, 3, podem ser vistos os modelos de metas, funções e classes de agentes, respectivamente as Figuras 17, 18 e 19.

1.6 **Ontologia**

A ontologia tem raízes na filosofia e representa o conhecimento de forma explícita por meio de conceitos e relações que representam o grupo específico de compreensão das pessoas naquele domínio (OLIVEIRA; WERNECK, 2003). Ontologias são os elementos básicos dos sistemas semânticos pois descrevem os aspectos semânticos de qualquer domínio (SALHOFER; STADLHOFER; TRETTER, 2010). Ontologia é o desenvolvimento sistemático, formal e axiomático da lógica de todas as formas e modos de ser (GUARINO, 1995).

Uma ontologia é uma linguagem que permite classificar objetos em categorias dentro de um domínio de conhecimento, uma forma de definir um catálogo sobre um objeto que irá permitir definir conceitos e mitigar dúvidas sobre o seu significado. Assim, uma

bota da classe calçados dificilmente será confundida com bota da terceira pessoa do verbo botar. Uma ontologia explica a conceituação do mundo alvo e nos fornece uma base sólida sobre a qual podemos construir bases de conhecimento compartilháveis para uma usabilidade mais ampla do que a de uma base de conhecimento convencional, segundo (KITAMURA; MIZOGUCHI, 2004).

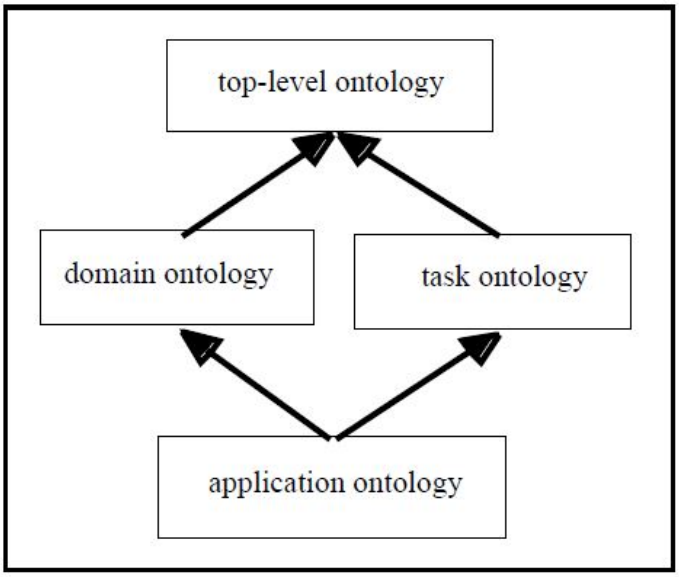
Um das linguagens mais populares para descrever ontologias é a *Web Ontology Language* (OWL) que é uma linguagem lógica de semântica Web que permite representar conhecimento detalhado sobre objetos e permitir a referência e comunicação padronizada entre sistemas de computadores. Na OWL podem ser criadas, publicadas, usadas e referenciadas por outras ontologias. Quando se define uma ontologia estão sendo criados classes e atributos daquele domínio de conhecimento. Uma classe calçados pode conter subclasses como masculino, feminino, tênis, ortopédicos, chinelos e botas e esses podem conter atributos como cor, tamanho e tipo de material. Uma ontologia também pode ser vista como um tipo especial de objeto de informação ou artefato computacional.

Os processos e sistemas de negócio possuem e geram informações sobre o ambiente em que estão operando e essas informações também precisam ser catalogadas de acordo com a sua classificação para evitar dúvidas sobre o seu significado. Portanto, no meio tecnológico também há a necessidade de se criarem ontologias para representar processos e sistemas. Ontologias computacionais são um meio de modelar formalmente a estrutura de um sistema, ou seja, as entidades e relações relevantes que emergem de sua observação e que são úteis para nossos propósitos (GUARINO; OBERLE; STAAB, 2009).

Para Guarino (1995), as ontologias podem ser classificadas em tipos *Top-level, Domain, Task* e *Application*. A ontologia *Top-level* descreve conceitos gerais, como espaço, tempo, matéria, objeto, evento, ação, etc. Esse tipo de ontologia é independente de problema e domínio e pode ser visto como ponto para unificar outras ontologias. A ontologia de *domain* descreve o vocabulário relacionado a um domínio de conhecimento, por exemplo ambiente marítimo. Uma ontologia de *task* busca uma especialização na tarefa, como por exemplo, atracar uma embarcação num pier. Ontologias de aplicativo descrevem conceitos que dependem de um domínio e tarefa específicos, que geralmente são especializações de ambas as ontologias relacionadas (GUARINO, 1995). As dependências entre as ontologias podem ser vistas na Figura 5.

Uma forma de estender o conjunto de afirmações consideradas verdadeiras (axiomas) do OWL para incluir regras que utilizam lógica matemática e de programação é utilizando a *Semantic Web Rule Language* (SWRL). A SWRL é uma proposta para combinação das sublinguagens OWL com sublinguagens da *Rule Markup Language* e pode ser usada para expressar regras bem como lógica. A proposta estende o conjunto de axiomas OWL para incluir regras semelhantes a Horn, permitindo que sejam combinadas com uma base de conhecimentos OWL e fornecendo uma sintaxe abstrata de alto nível (HORROCKS et al., 2004), expressas em termos de conceitos OWL para raciocinar sobre

Figura 5 - Tipos de Ontologias.



Legenda: Nível de dependência em particular tarefa ou ponto de vista.
Fonte: (GUARINO, 1995)

os indivíduos OWL (O'CONNOR et al., 2005). Uma cláusula de Horn é uma fórmula matemática lógica semelhante a uma regra que facilita a programação lógica. Uma cláusula de Horn é aquela em que a conjunção de zero ou mais condições implica no máximo uma conclusão (LIN et al., 1993)

Um código SWRL que permite atracação num pier somente se a velocidade do vento não estiver acima de 20 kn de velocidade pode ser representado como:

$Pier(?p) \wedge SupportedWindSpeed(?p, ?speed) \wedge swrlb : greaterThanOrEqual($ $?speed, 20) \rightarrow Dock(?p, 0)$

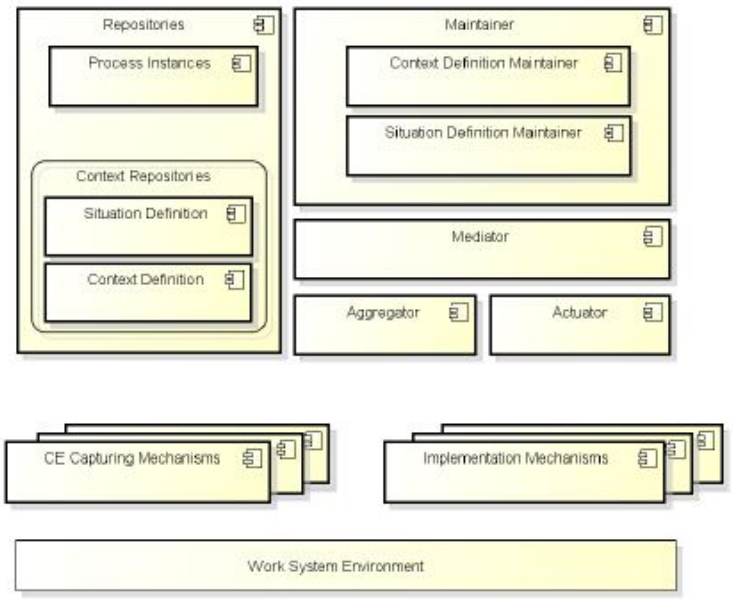
Nessa pesquisa a ontologia é utilizada para capturar informações de contexto do domínio do processo de negócios de atracar uma embarcação para auxiliar os agentes do `AAdapt` na tomada de decisões de exceções identificadas.

1.7 **O GCAdapt**

Nunes (2014) desenvolveu uma arquitetura de gerenciamento de contexto para adaptação dinâmica de processos, utilizando o conceito de contexto e replanejamento (*replanning*) das tarefas quando identificada uma situação de exceção no processo. O sistema fundamentou-se na teoria de sistemas de informação sensíveis a contexto, fazendo dessa forma o replanejamento das tarefas para atender os objetivos definidos em tempo

de desenvolvimento. Nomeado de `GCAdapt`, é uma arquitetura que permite a adaptação dinâmica de tarefas de processos e recursos com suporte a informações de contexto (*Context-Aware*). O `GCAdapt` possui integração com um motor de processos de sistemas de informação orientados por processos (*Process-Aware Information System* PAIS) e possui quatro componentes principais na sua arquitetura: *Maintainer, Aggregator, Actuator* e *Mediator*. O módulo *Maintainer* é responsável por gerenciar as definições de contexto, o *Aggregator* coleta elementos contextuais (*CE - contextual elements*) que estão logicamente relacionados em um repositório, o *mediator* é o módulo principal da arquitetura e age na adaptação durante a execução da instância do processo e *Actuator* recebe as atualizações do *Mediator* e envia os comandos de adaptação no processo para o mecanismo de implementação, conforme definido por (NUNES, 2014). A Figura 6 apresenta a arquitetura do GCAdapt.

Figura 6 - Arquitetura do GCAdapt



Legenda: Arquitetura do GCAdapt
Fonte: (NUNES, 2014).

O GCAdapt foi utilizado como um conceito e não ocorreu um de-para do GCAdapt para o AAdapt, portanto não há um relação direta dos módulos do GCAdapt para os agentes AAdapt. Algumas funcionalidades dos módulos foram incorporadas pelos agentes e outras não foram mais necessárias face aos mecanismos utilizados pelos agentes. As funcionalidades captura de CE's e de aprendizado dos módulos *Aggregator* e *Maintainer* foram implementados no agente `Communicator` que coleta CE's do PAIS, de sensores externos e da ontologia de domínio e identifica situações utilizando informações de crenças e metas do BDI para identificar situações e acionar ações.

As funcionalidades do módulo *Actuator* foram em parte assimiladas pelo agente

`Executor` que possui planos de ações pré-determinados para caminhos alternativos a serem seguidos no processo na identificação de situações de exceção no processo. Por fim o comportamento do módulo *Mediator* pode ser identificado no agente `Communicator` que interage e propõe ao PAIS os caminhos alternativos identificados pelo agente `Executor`.

O GCAdapt utiliza um componente externo de *Planning Domain Definition Language* (PDDL) chamado SAPA que permite o replanejamento das tarefas no processo de negócio que está operando. As possíveis tarefas e caminhos no GCAdapt são previamente mapeados e disponibilizados para que o PDDL possa construir o caminho. O `AAdapt` não utiliza um PDDL e segue uma abordagem diferente. Nela os caminhos alternativos possíveis são desenhados no próprio PAIS e as situações de contorno são implementadas nos *gateways* de decisão do processo.

O objetivo de ambos os sistemas é o mesmo, oferecer caminhos alternativos nos processos para situações de exceção que foram identificadas, porém, utilizando técnicas e estratégias diferentes.

2 MAPEAMENTO SISTEMÁTICO DA LITERATURA

Um mapeamento sistemático é um método para classificar e estruturar um tema de pesquisa e tem por objetivo apresentar respostas para perguntas formuladas utilizando evidências obtidas. James, Randall e Haddaway (2016) argumentam que o mapeamento sistemático segue o mesmo rigor, objetivo e processos transparentes, assim como revisões sistemáticas, para capturar evidências que são relevantes para um tópico específico. Uma revisão sistemática da literatura (*systematic review of literature* - SRL) é uma pesquisa de trabalhos de um determinado domínio de conhecimento que utiliza um protocolo padronizado que pode ser reproduzido facilmente para permitir a qualidade dos resultados apresentados. Segundo James, Randall e Haddaway (2016), o mapeamento sistemático foi desenvolvido nas ciências sociais em resposta à falta de dados empíricos ao responder a perguntas usando métodos de revisão sistemática e à necessidade de um método para descrever a literatura em um amplo assunto de interesse. Tanto as revisões quanto os mapeamentos sistemáticos são essenciais para auxiliar os pesquisadores a se manterem atualizados e dão uma visão geral de uma área de pesquisa através das contribuições realizadas, onde foram publicadas e as informações estatísticas geradas.

Para ampliar o conhecimento sobre a adaptação dinâmica de instâncias de processos em tempo de execução, foi realizado um mapeamento sistemático com o objetivo de identificar pesquisas do domínio estudado, levantar quais contribuições de pesquisas existem nesse segmento e quais lacunas podem ser exploradas. Foi definido o escopo do mapeamento e elaborado o protocolo que definiu a questão de pesquisa principal e as questões secundárias, os repositórios onde seriam realizadas as pesquisas, a *string* de busca nos repositórios e os critérios para seleção dos artigos. Foi elaborada uma visualização gráfica de nuvem com as palavras mais encontradas nos artigos selecionados, uma comparação dos trabalhos e por fim a análise e resultados.

2.1 Escopo e Protocolo do mapeamento sistemático

O escopo do mapeamento contempla artigos que estejam relacionados a processos de negócio adaptados em tempo de execução e que estavam disponíveis para *download* publicamente ou para a conta da Universidade do Estado do Rio de Janeiro - UERJ. Não fazem parte do escopo aspectos que não estejam listados no protocolo definido.

O protocolo contempla a questão de pesquisa principal (QP) e as questões secundárias (QS), os repositórios (fontes) e a *string* de pesquisa de artigos, os critérios para seleção dos artigos e por fim a análise de dados e resultados.

2.2 Questões de pesquisa

A questão principal(QP) que a pesquisa enseja responder é quais estudos realizaram adaptação de processos de negócio em tempo de execução. Com esse levantamento é possível identificar as tecnologias e técnicas que estão sendo utilizadas para permitir adaptação de processos de negócio em tempo de execução.

Além da questão principal(QP) questões secundárias(QS) foram definidas para aprofundar o estudo. As questões de pesquisa para esse mapeamento sistemático da literatura são:

- **QP** - Quais estudos realizaram adaptação de processos de negócio em tempo de execução?
- **QS1**- Quais estudos utilizaram multiagente em adaptação dinâmica de processos em tempo de execução?
- **QS2** - Quais estudos utilizaram IA(AI) em adaptação dinâmica de processos em tempo de execução?
- **QS3** - Quais estudos utilizaram BDI em adaptação dinâmica de processos em tempo de execução?
- **QS4** -Quais estudos utilizaram ontologia em adaptação dinâmica de processos em tempo de execução?
- **QS5** - Quais estudos utilizaram informações de contexto para tomada de decisões em adaptação dinâmica de processos em tempo de execução?
- **QS6**- Quais estudos utilizaram KiP em adaptação dinâmica de processos em tempo de execução?
- **QS7**- Quais estudos integraram com com um PAIS comercial em adaptação dinâmica de processos em tempo de execução?

2.3 Repositórios de artigos utilizados

Para a elaboração desse mapeamento foram utilizados os principais repositórios de artigos acadêmicos por serem reconhecidamente fontes confiáveis. As fontes utilizadas e a quantidade de artigos retornados na pesquisa são apresentadas na Tabela 2.

A pesquisa nos repositórios foi realizada em novembro de 2021 e contemplou todos os artigos publicados por causa do baixo quantitativo de artigos identificados nessa área de pesquisa.

Tabela 2 - Repositórios de artigos utilizados.

Fonte:	**URL para acesso a fonte:**	**Artigos retornados:**
IEEE Xplore	https://ieeexplore.ieee.org/	4
ACM Digital Library	https://dl.acm.org/	23
Scopus	https://www.scopus.com/home.uri	6
Springer	https://link.springer.com/	17
ScienceDirect	http://www.sciencedirect.com/	45
	Total de artigos retornados	**95**

Legenda: Repositórios, URLs para acesso e quantidade de artigos retornados
Fonte: 'O autor:'.

2.4 **Busca por artigos nos repositórios**

A busca por artigos contemplou as expressões nas línguas inglesa e portuguesa nos repositórios e procurou por artigos que tivessem expressões atreladas a processos de negócio que fossem adaptados em tempo de execução. Para realizar a busca de processos de negócios na língua inglesa foi incluída na pesquisa uma junção com os termos *"Business process management", "BPM", "KIP", "Knowledge-Intensive Processes", "Knowledge Intensive Processes"*. Na segunda parte da busca outra junção com os termos *"runtime", "run-time", "real time"* e na última parte a procura por adaptação de processos *"process adaptation"*.

A Tabela 3 apresenta a *string* de pesquisa na língua inglesa que foi executada nos repositórios.

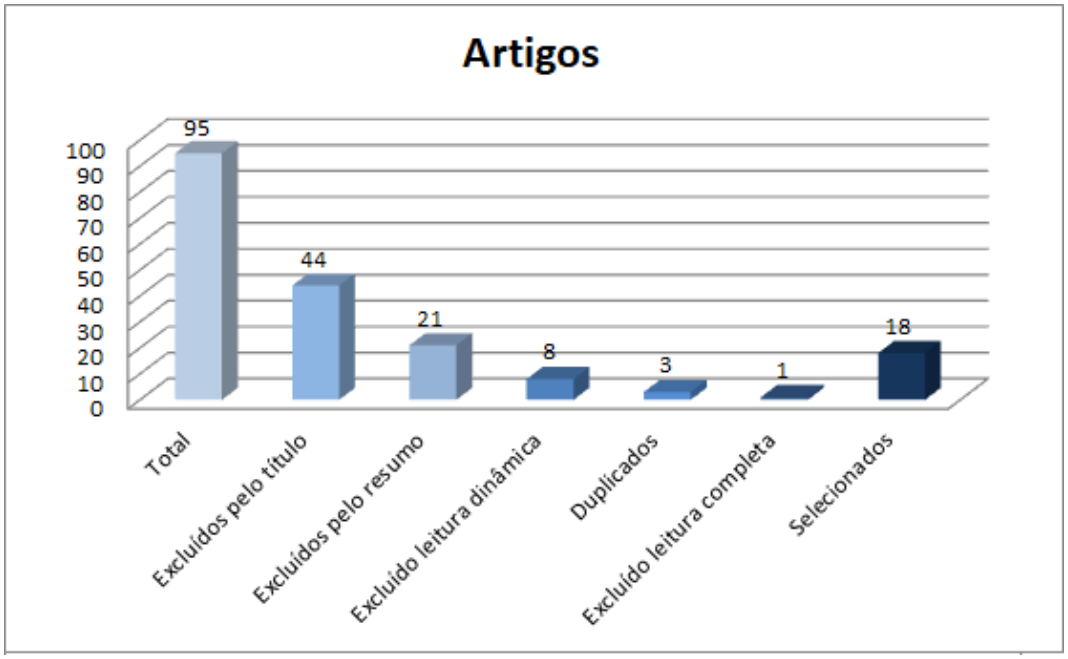
Tabela 3 - Busca nos repositórios com termos em inglês.

***String* de busca em inglês:**
(("Business process management" OR "BPM" OR "KIP" OR "Knowledge-Intensive Processes" OR "Knowledge Intensive Processes") AND ("runtime" OR "run-time" OR "real time") AND ("process adaptation"))

Legenda: *String* de pesquisa utilizada nos repositórios
Fonte: 'O autor:'.

A pequisa com a *string* de busca em inglês retornou 304 (trezentos e quatro) artigos publicados distribuídos pelos repositórios. Após aplicação de filtros de categorias e exclusão de artigos que não foram possíveis baixar, o número de artigos selecionados para verificação caiu para 95. Na análise do título foram excluídos 44 artigos. Foram excluídos ainda 21 artigos na leitura do resumo, 8 na leitura dinâmica do artigo e 1 na leitura completa do artigo. Também foram identificados 3 artigos duplicados no repositório da Scopus que eram artigos da IEEE Xplore ou da ScienceDirect. Ao final foram selecionados 18 artigos para esse mapeamento. A Figura 7 apresenta esses dados.

Figura 7 - Análise de artigos



Legenda: Quantidade de artigos analisados.
Fonte: 'O autor.'.

Também foram pesquisados artigos publicados na língua portuguesa com a *string* de busca levemente alterada para capturar artigos de processos de negócio que adaptam em tempo de execução, conforme Tabela 4. A adaptação da *string* de busca de artigos em português usou junção com termos em inglês visto que alguns desses termos não são comumente traduzidos para o português.

Mesmo usando junção com termos em inglês as pesquisas com *string* em português nos repositórios não foram satisfatórias, retornando apenas um artigo no repositório ACM DL que era uma *Systematic Review of Literature* (SRL) para adaptação em BPMN que a questão principal era *"Quais abordagens propostas na literatura para gerenciar adaptações em* ***tempo de projeto*** *de modelo de processos em BPMN"* (PILLAT; SANTOS; OLIVEIRA, 2019), e que está fora do escopo desse mapeamento por tratar de adaptações em tempo de projeto. Os demais repositórios não retornaram registros. Dessa forma

Tabela 4 - Busca nos repositórios com termos em português / inglês.

***String* de busca em português:**
(("gerenciamento de processos de negócio" OR "BPM" OR "KIP" OR "Knowledge-Intensive Processes" OR "processos de conhecimento intensivo") AND ("tempo de execução") AND ("adaptação de processos"))

Legenda: *String* de pesquisa utilizada nos repositórios
Fonte: 'O autor.'.

esse mapeamento da literatura considerou somente os artigos retornados na pesquisa em inglês, Tabela 3.

2.5 **Seleção de artigos**

Nessa seção é apresentado como foi o processo de seleção dos artigos que compõem esse mapeamento sistemático da literatura. Foram considerados somente artigos que atenderam o escopo, descartando artigos com problemas no link de acesso.

A estratégia de seleção dos artigos que formaram esse mapeamento sistemático da literatura, Figura 8, contou com as seguintes etapas:

- remoção de artigos duplicados;

  Nessa etapa foram removidos os artigos duplicados retornados dos repositórios.

- leitura do título dos artigos;

  Na etapa de leitura do título foi possível identificar vários artigos que não atendiam o escopo da pesquisa.

- leitura dos resumos dos artigos;

  Os artigos que passaram pelas etapas anteriores tiveram seu resumo avaliado para identificar os que não atendiam os requisitos da pesquisa.

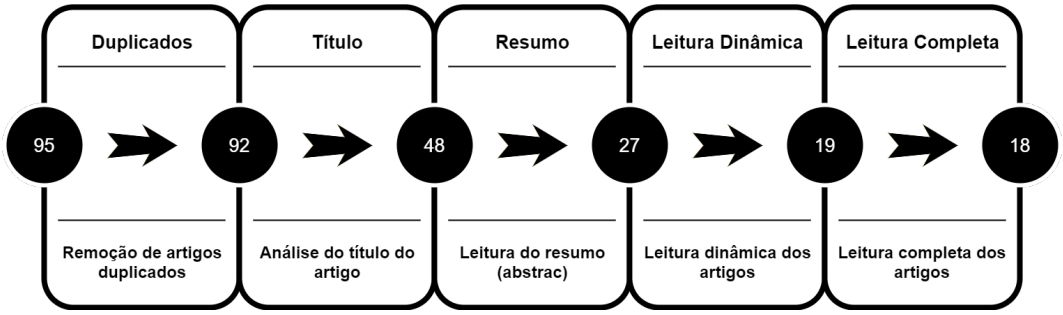
- seleção pela leitura dinâmica dos artigos;

  Nessa etapa foi realizada uma leitura dinâmica dos artigos, uma leitura diagonal. Primeiro foi verificada a estrutura de seções dos artigos e depois foram procurados os seguintes termos *intensive, KiP, agent, IA, BDI, PAIS, process-aware, ontolog[*].* Figuras das propostas / arquiteturas dos trabalhos foram analisadas e por fim as seções da proposta e conclusões foram lidas.

- leitura completa dos artigos.

  Nessa última etapa o artigo inteiro foi analisado.

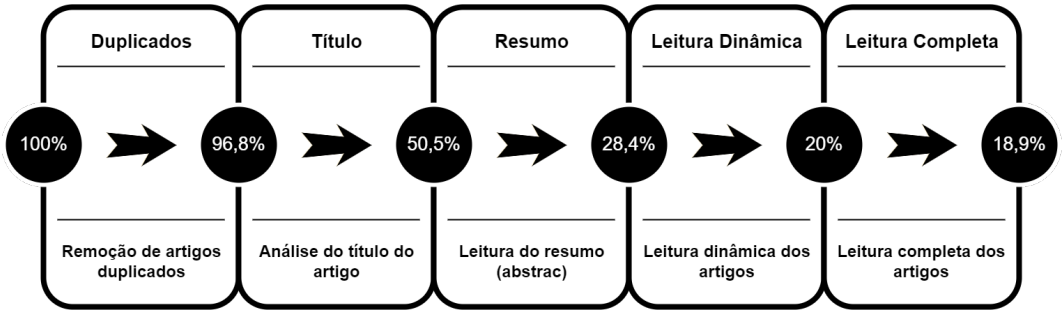
Figura 8 - Etapas de seleção dos artigos (quantidade).



Legenda: Quantidade de artigos retornados nas etapas de seleção.
Fonte: 'O autor.'.

A Figura 9 apresenta uma representação gráfica do percentual de artigos classificados por etapa.

Figura 9 - Etapas de seleção dos artigos (percentual).



Legenda: Percentual de artigos retornados nas etapas de seleção.
Fonte: 'O autor.'.

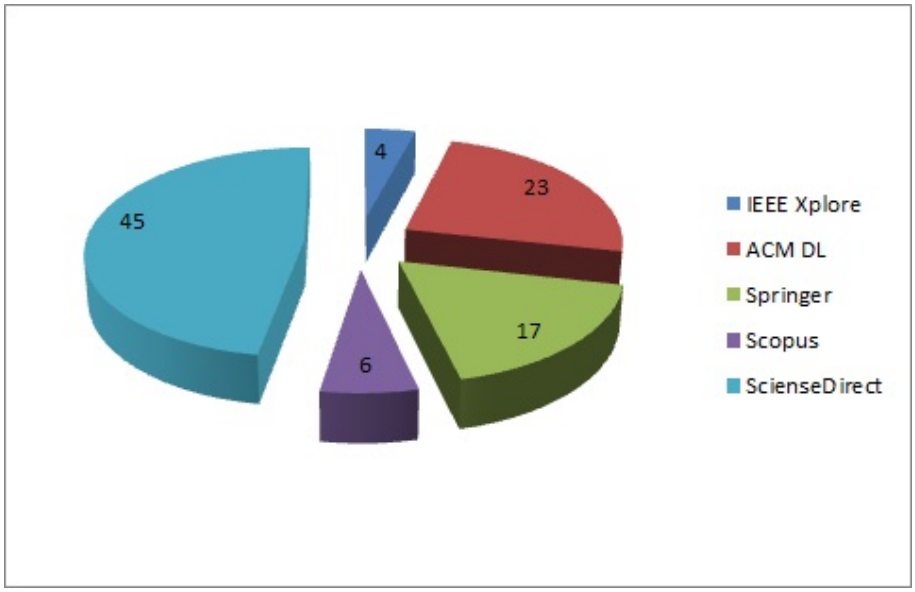
As Figuras 10 e 11 apresentam representações gráficas da quantidade e percentual de artigos retornados na pesquisa por repositório.

Após a remoção de artigos duplicados e a aplicação dos critérios de seleção dos artigos, o número de artigos foi reduzido para 18 artigos que estão distribuídos pelos repositórios, conforme Figura 12.

2.6 **Visualização gráfica das palavras mais identificadas nos artigos**

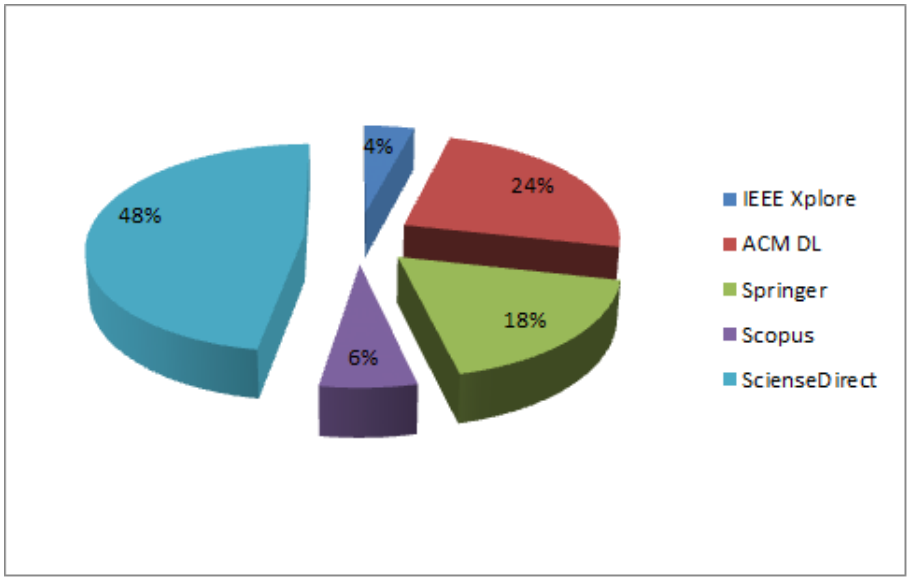
Com o objetivo de ampliar a visão dos dados contidos nos artigos selecionados para esse mapeamento e dar outras possíveis interpretações, uma nuvem de palavras foi criada para dar uma visão gráfica da frequência e do valor das palavras mais encontradas nos artigos selecionados, Figura 13. Para a sua construção foi gerado um arquivo .pdf com

Figura 10 - Artigos retornados na pesquisa



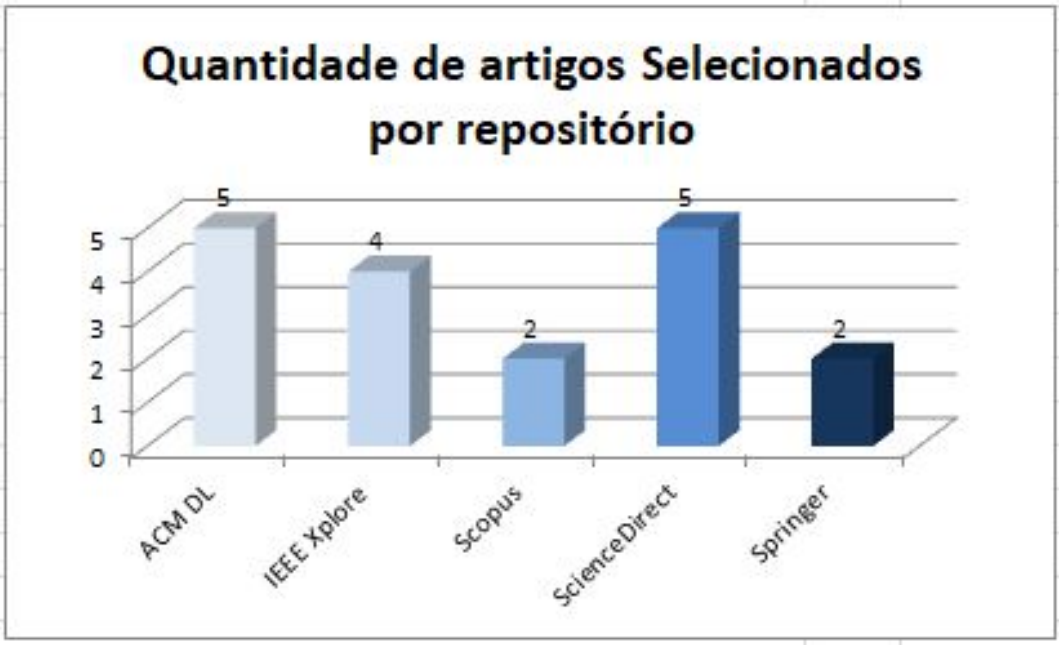
Legenda: Quantidade de artigos retornados na pesquisa aos repositórios.
Fonte: 'O autor.'.

Figura 11 - Percentual de artigos retornados na pesquisa



Legenda: Percentual de artigos retornados na pesquisa por repositório.
Fonte: 'O autor.'.

Figura 12 - Artigos selecionados.



Legenda: Artigos selecionados por repositório.
Fonte: 'O autor.'.

todos os artigos selecionados e utilizada a ferramenta WordItOut[1] para a confecção do gráfico de nuvem. As 5 palavras mais identificadas nos artigos foram *process* com 2548 incidências, *business* com 1066, *data* com 630, *execution* 610 e *adaptation* com 569 que corroboram que os artigos selecionados estão dentro do escopo da pergunta principal.

[1] Permite gerar nuvem de palavras gratuitamente, disponível em https://worditout.com/

Figura 13 - Nuvem de palavras.

Legenda: Nuvem de palavras mais encontradas nos artigos selecionados.
Fonte: 'O autor.'.

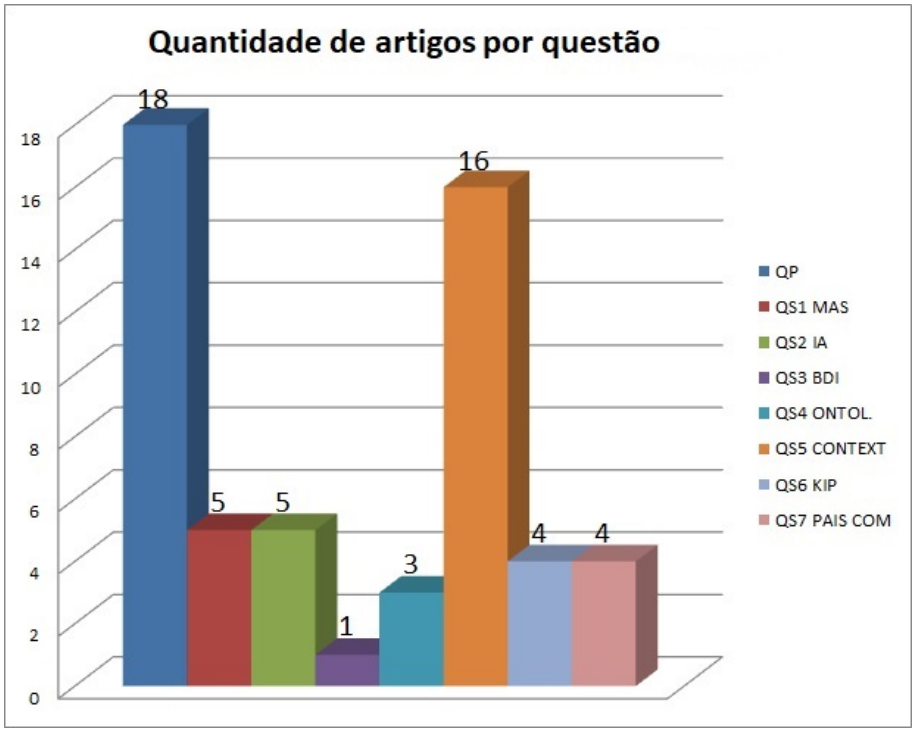
2.7 **Análise dos dados e resultados**

Ao final do processo de seleção de artigos restaram 18 artigos que respondiam a questão principal da pesquisa - **Quais estudos realizaram adaptação de processos de negócio em tempo de execução?** Um mapeamento desses artigos e suas relações com as questões secundárias foram identificadas e apresentadas na Tabela 5. Uma visão gráfica da distribuição quantitativa dos artigos pelas questões de pesquisa é apresentada na Figura 14.

- **QP - Quais estudos realizaram adaptação de processos de negócio em tempo de execução?**

  Na questão primária (QP) foram identificados trabalhos que utilizaram técnicas diferentes como *Machine Learning*, técnicas para deduzir os elementos de processo afetados, fragmento de processo de inserção, exclusão e movimentação foram implementados com REST, XML *Process Definition Language* (XPDL), gatilhos por regras ECA armazenadas em uma base de conhecimento, gatilho por regras ECA e informações de contexto, utilização de algoritmos para utilização de fragmentos de tarefas, motor de processos baseado em micro-serviços, planejador IA, utilização de agentes, decisões por contexto, alterações por *web service* e MAPE-K. As técnicas que foram mais utilizadas nos trabalhos foram as regras ECA, planejador IA e

Figura 14 - Questões do mapeamento sistemático.



Legenda: Quantidade de artigos que atenderam questões da pesquisa.
Fonte: 'O autor.'.

decisões baseadas em contexto.

- **QS1 - Quais estudos utilizam multiagente em adaptação dinâmica de processos em tempo de execução?**

  Quase um terço dos trabalhos utilizaram sistemas multiagentes, 5 trabalhos. A linguagem mais mencionada para os agentes foi Java e o Jadex foi mencionado em um trabalho. A flexibilidade de uso de agentes deve ter contribuído para a sua escolha nas implementações.

- **QS2 - Quais estudos utilizaram IA(AI) em adaptação dinâmica de processos em tempo de execução?**

  Cinco trabalhos utilizaram IA para tratar adaptação de processos, em sua maioria utilizando planejador IA. Essa é uma técnica madura com várias opções de implementação e bastante utilizada.

- **QS3 - Quais estudos utilizaram BDI em adaptação dinâmica de processos em tempo de execução?**

  Foi identificado apenas um trabalho com BDI. A maioria preferiu utilizar regras ECA ou planejamento IA para a tomada de decisões. Com o BDI os *belief* e *desire* podem ser utilizados com a ontologia para a tomada de decisões das adaptações.

- **QS4 - Quais estudos utilizaram ontologia em adaptação dinâmica de processos em tempo de execução?**

  Três trabalhos sinalizaram a utilização de ontologia. Um número considerado pequeno visto que a ontologia pode ser utilizada para realizar inferências e carregar informações de contexto para os processos.

- **QS5 - Quais estudos utilizaram informações de contexto para tomada de decisões em adaptação dinâmica de processos em tempo de execução?**

  A maioria dos trabalhos pesquisados, 16, utilizaram de alguma forma informações de contexto para a tomada de decisões. Percebe-se que essa é uma técnica bastante utilizada para decisão na adaptação das tarefas.

- **QS6 - Quais estudos utilizaram KIP em adaptação dinâmica de processos em tempo de execução?**

  Foram identificados 4 trabalhos que utilizavam processos de conhecimento intensivo, alguns com interação humana.

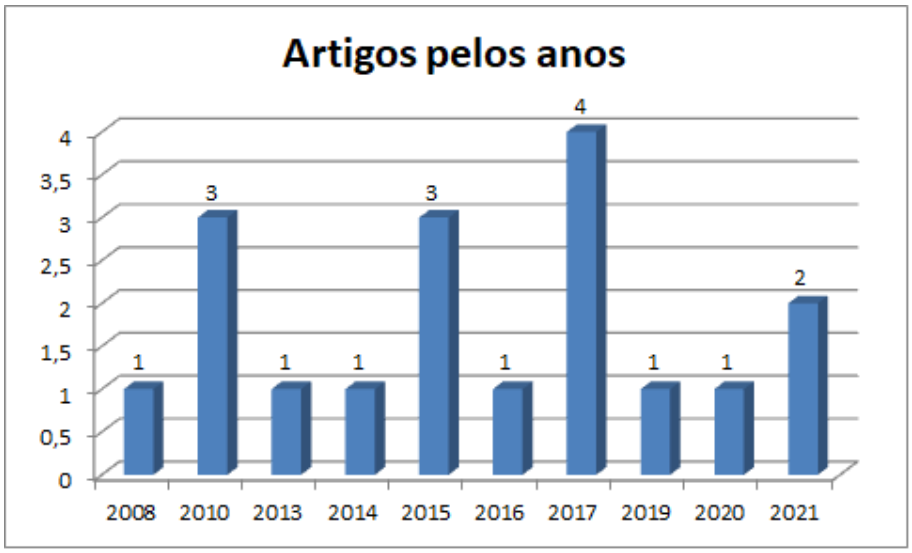
- **QS7 - Quais estudos integraram com com um PAIS comercial em adaptação dinâmica de processos em tempo de execução?**

Nem todos os trabalhos sinalizaram claramente sua integração com PAIS comerciais. Em 4 trabalhos, pela descrição, percebeu-se que os trabalhos poderiam ser integrados com PAIS comerciais.

Pelos dados apresentados na Figura 14 identifica-se que a maioria dos trabalhos utiliza informações de contexto para a tomada de decisões. Uma lacuna que pode ser explorada em trabalhos futuros é a utilização de BDI em adaptação de processos de negócio, visto que só foi identificada uma incidência de uso. Ontologias e KiPs também poderiam ser mais utilizados em adaptações de processos de negócio. Pela dispersão dos artigos pelos anos mostrada na Figura 15 se percebe que não há muitos trabalhos em adaptação de processos de negócio e que esta é uma área que carece de mais estudos.

Como as pesquisas nos repositórios retornaram poucos artigos não foi limitado o período de tempo da pesquisa. Dessa forma todos os artigos retornados foram considerados, o que incluiu artigos entre os anos de 2008 e 2021. A distribuição dos artigos pelos anos de publicação pode ser observada na Figura 15.

Figura 15 - Artigos publicados.



Legenda: Distribuição da publicação dos artigos por ano.
Fonte: 'O autor.'.

Todos os trabalhos listados são boas propostas e que, de alguma forma, propõem a adaptação de processos de negócio em tempo de execução ou de adaptação na implementação. Um dos principais diferenciais do `AAdapt` para os demais trabalhos foi a utilização de multiagentes com BDI e inferências com ontologia de domínio do processo. Na análise desses artigos foi verificado o atendimento a QP (questão principal) e QSs(questões secundárias) da pesquisa. A consolidação dos dados pode ser vista na Tabela 5. Na tabela foram utilizadas as legendas S para sim, N para não e N/D para não detectado / identificado.

Tabela 5 - Artigos e as relações com as questões de pesquisa

Artigo	**Referência**	**QP**	**QS 1**	**QS 2**	**QS 3**	**QS 4**	**QS 5**	**QS 6**	**QS 7**
Context-driven Business Process Adaptation for ad hoc changes	Xia e Wei (2008)	S	N	N	N	N	S	N	N
A Hypermedia-Driven Approach for Adapting Processes via Adaptation Processess	Oberhauser (2016)	S	S	N	N	N	S	N	S
Impact Analysis of Process Change at Run-time	Hajmoosaei et al. (2015)	S	N	N	N	N	S	N	N
Prediction of Incompliance With Business GoalsWith Business-Related Data and Context Data	Zhang e Liu (2020)	S	N	S	N	N	S	N	N
Integration of RESTfulBP with BDIM Decision Making	Lu et al. (2010)	S	N	N	N	N	N	N	N
Dynamic Context-Aware Business Process: A Rule-Based Approach Supported by Pattern Identification	Bernal et al. (2010)	S	N	N	N	N	S	N	N
ADIGE: Semantic Business Process Management forSmart Retail Environments	Kahl et al. (2015)	S	S	N	N	S	S	N	N/D

Continua na próxima página

Tabela 5 – *Continuação da página anterior*

Artigo	**Referência**	**QP**	**QS1**	**QS2**	**QS3**	**QS4**	**QS5**	**QS6**	**QS7**
Enabling run time flexibility in data-centric and data-driven process execution engines	Andrews, Steinau e Reichert (2021)	S	N	N	N	N	N	N	N
Automated runtime repair of business processes	Beest et al. (2014)	S	N	S	N	N	S	N	S
A refinement-based approach for verifying dynamic changes on time-aware processes	Fakhfakh (2019)	S	N	N	N	N	S	N	N
A knowledge-intensive adaptive business process management framework	Kir e Erdogan (2021)	S	S	S	S	S	S	S	N
Approach based on web services for business process adaptation	Awadid e Gnannouchi (2015)	S	N	N	N	N	S	N	N
Assessing flexibility in event-driven process adaptation	Patiniotakis et al. (2019)	S	N	N	N	N	S	N	S
Dynamic Adaptation of Business Process Based on Context Changes: A Rule-Oriented Approach	Hu, Wu e Chen (2013)	S	N	N	N	N	S	S	N/D
A context-aware framework for dynamic composition of process fragments in the internet of services	Bucchiarone et al. (2017)	S	N	S	N	N	S	S	N

Continua na próxima página

Tabela 5 – *Continuação da página anterior*

Artigo	**Referência**	**QP**	**QS1**	**QS2**	**QS3**	**QS4**	**QS5**	**QS6**	**QS7**
Toward a framework for self-adaptive workflows in cyber-physical systems	Seiger et al. (2019)	S	S	N	N	S	S	N	N
Intelligent Process Adaptation in the SmartPM System	Marrella, Mecella e Sardina (2017)	S	S	S	N	N	S	S	S
Autonomic Internet-Scale Workflows	Tretola e Zimeo (2010)	S	N	N	N	N	S	N	N

Legenda: Tabela da relação dos artigos com as questões do mapeamento. S para sim, N para não e N/D para não detectado / identificado.

Fonte: 'O autor.'.

3 AADAPT

O `AAdapt` (Adaptação de processos intensivos em conhecimento por Agentes) propõe uma abordagem para a adaptação de processos intensivos em conhecimento utilizando agentes e informações de contexto. Nesta abordagem, um sistema multiagente baseado em crenças, desejos e intenções (BDI) é usado com a premissa de monitorar e adaptar processos de negócio em tempo de execução usando informações de contexto capturadas de *Process-Aware Information Systems* (PAIS) de sensores e informações da ontologia do domínio do processo. A arquitetura é composta por dois agentes que interagem com o ambiente do processo para identificar situações que causam problemas na execução padrão do processo, permitindo caminhos alternativos que mitiguem riscos e permitam a correta execução do processo.

3.1 Agentes do `AAdapt`

O `AAdapt` é uma arquitetura de sistema multiagente inteligente que propõe um mecanismo para trabalhar em conjunto com PAIS. Ele monitora elementos de contexto em tempo real do processo obtidos do PAIS, dos sensores externos e do carregamento da ontologia do domínio do processo para identificar situações de exceção e executar planos para contorná-las. O `AAdapt` utiliza o mecanismo de BDI para atingir seus objetivos propondo ao PAIS uma solução de contorno sempre que uma situação de exceção ao fluxo padrão do processo seja identificada.

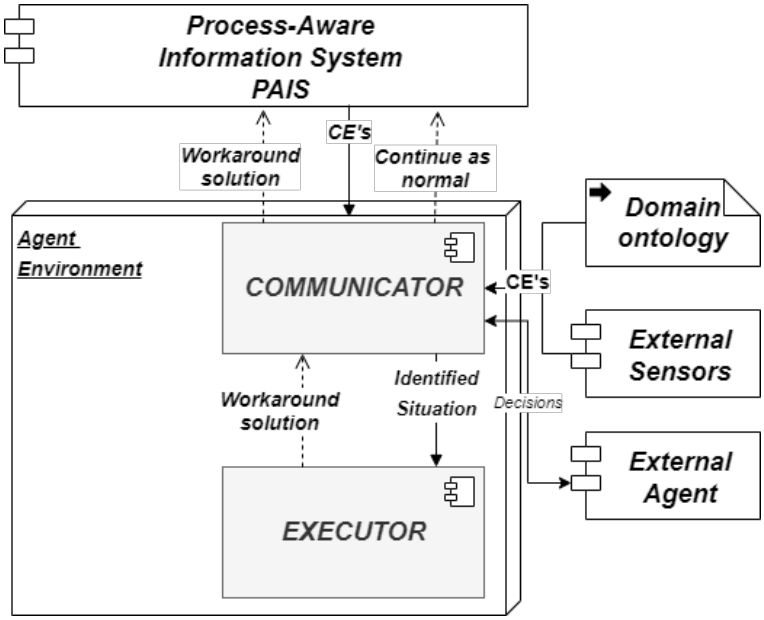
3.1.1 Arquitetura

A arquitetura do `AAdapt` possui dois agentes, o `Communicator` e o `Executor`, cada um com suas funcionalidades e responsabilidades para integrar com o PAIS. Além dos agentes, também compõem a arquitetura os módulos *Domain ontology, External Sensors* e *External Agent*. A Figura 16 apresenta a arquitetura do `AAdapt`.

O agente `Communicator` é responsável por interagir com o sistema PAIS, com a ontologia de domínio, com os sensores e com os agentes externos. Também é responsável por identificar situações de exceção no processo e repassá-las para o agente `Executor`. Toda a interação com o PAIS é realizada por meio desse agente.

O agente `Executor` é responsável por identificar uma solução de contorno e encaminhá-la ao agente `Communicator`. Esse agente interage com a base de soluções para propor uma alternativa para exceção identificada.

Figura 16 - Arquitetura `AAdapt`.



Legenda: Interação com o software PAIS e sensores externos.
Fonte: 'O autor.'.

O componente *Domain ontology* é a ontologia de domínio. Esse componente deve ser criado para cada processo de negócio a ser atendido pelo `AAdapt` permitindo carregar para o agente `Communicator` as informações de contexto (CE's) necessárias para a identificação de situações de exceção.

Outro componente da arquitetura é o *External Sensors* cuja responsabilidade é captar informações de contexto ambientais durante a execução das tarefas e disponibilizá-las para o agente `Communicator`.

O último componente da arquitetura é o *External Agent*, que é uma integração com um agente externo (recurso humano) que objetiva ratificar decisões em casos que os elementos de contexto apontem condições ambientais extremas.

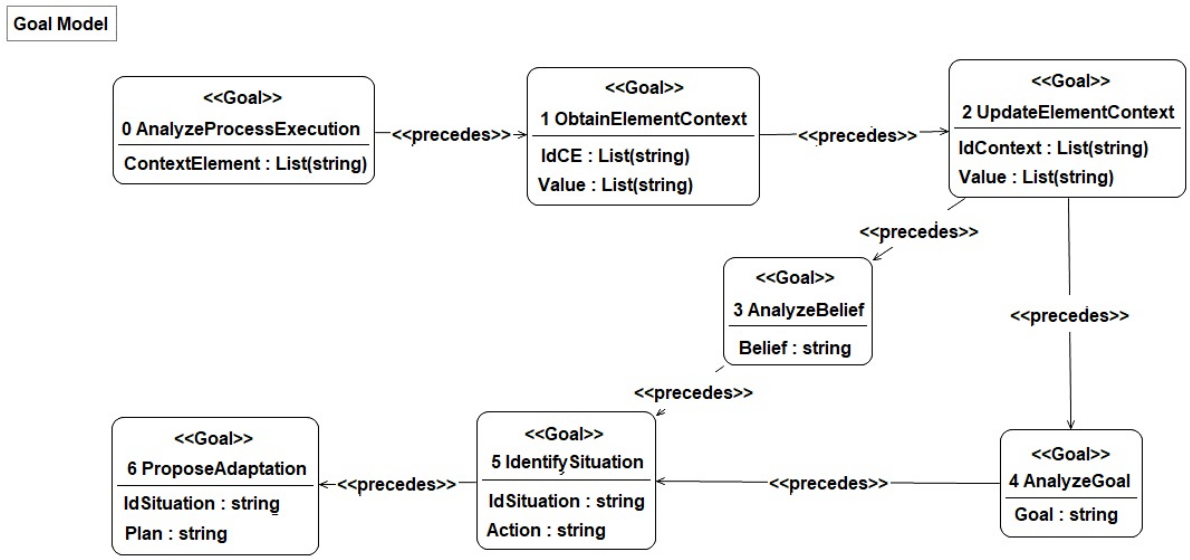
3.2 **Modelagem dos agentes**

A metodologia escolhida para modelar o `AAdapt` foi a O-MaSE *(Organization-based Multiagent Software Engineering)*, que é uma estrutura baseada na organização, que integra um conjunto de tecnologias concretas destinadas a facilitar a aceitação industrial (GARCíA-OJEDA; DELOACH, 2014). Os modelos de representação gráfica utilizados para a modelagem dos agentes do *AAdapt* foram o *Goal Model, Role Model* e *Agent Class Model*, respectivamente os modelos de metas, funções e classe de agentes. Nas Figuras

17, 18 e 19 são apresentados os modelos de metas, de funções e de classes de agentes que representam a modelagem conceitual da solução proposta.

O modelo de metas, Figura 17, tem como objetivo organizar as metas das diferentes funções do sistema sinalizando as ações que podem ser realizadas pelo sistema. As metas

Figura 17 - Modelo de metas do `AAdapt`.



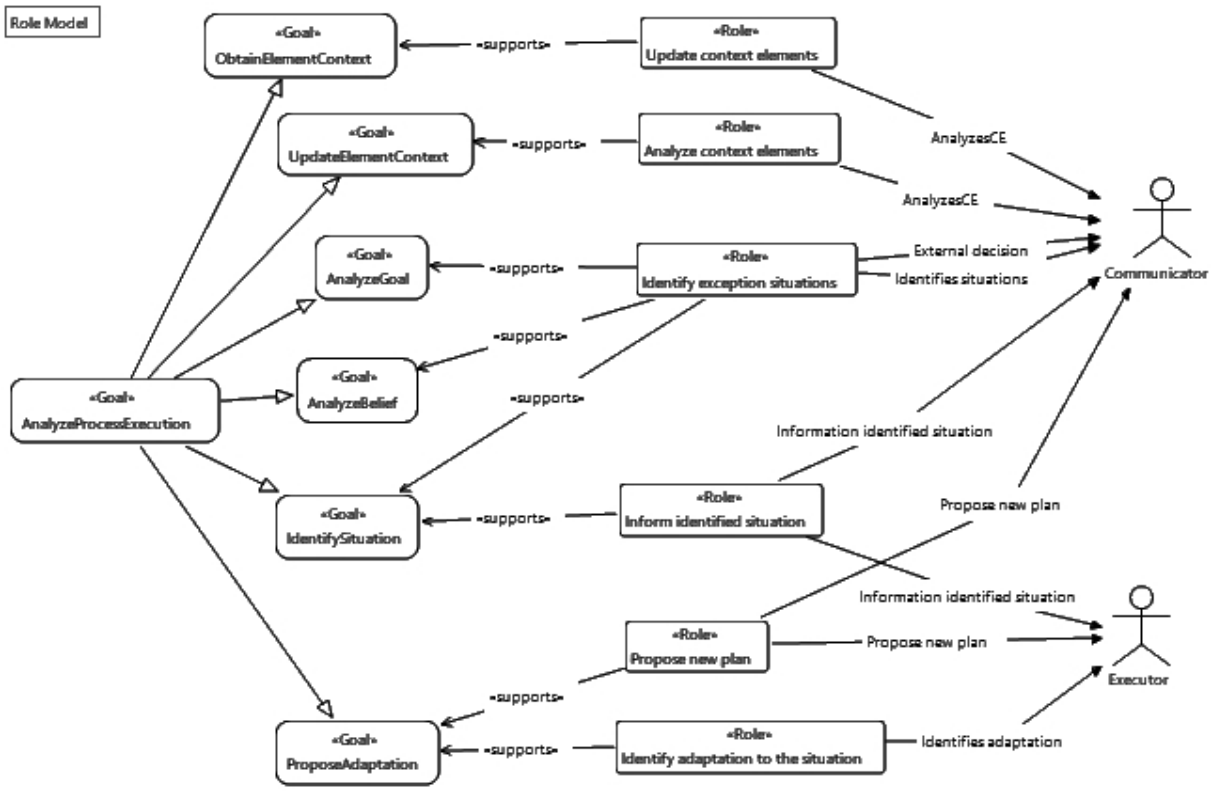
Legenda: Modelo de metas do `AAdapt`.
Fonte: 'O autor'.

ObtainElementContext, UpdateElementContext, AnalyzeBelief, AnalyzeGoal e *IdentifySituation* apresentadas na Figura 17 estão associadas a funções do agente `Communicator` e funcionam como um processo de monitoração do ambiente no qual o sistema multiagente está inserido, obtendo e atualizando elementos de contexto e analisando as metas e crenças dos agentes para identificar situações em que o sistema precise tomar uma ação. A meta *ProposeAdaptation* é o alvo das funções do agente `Executor`, é o momento que uma situação é identificada e o agente propõe uma adaptação ao processo. É a ação para a escolha e proposição de um caminho alternativo no processo quando uma situação indesejada é identificada.

O modelo de funções, Figura 18, permite identificar e documentar as funções na organização. As funções (*Role*) são a base para que o objetivo de uma meta seja atingida. O modelo de funções permite identificar e documentar as funções na organização que os agentes desempenham e assumem responsabilidade. Para garantir a consistência, cada meta folha (*Goal*) deve ser suportada por pelo menos uma função (*Role*). Por exemplo, a *Goal ProposeAdaptation* é suportada pela *Role* `ProposeNewPlan`, conforme é apresentado na Figura 18.

O modelo de classes de agentes, Figura 19, tem por objetivo identificar os tipos de agentes que participam da organização. No modelo de classes de agentes foi identificado

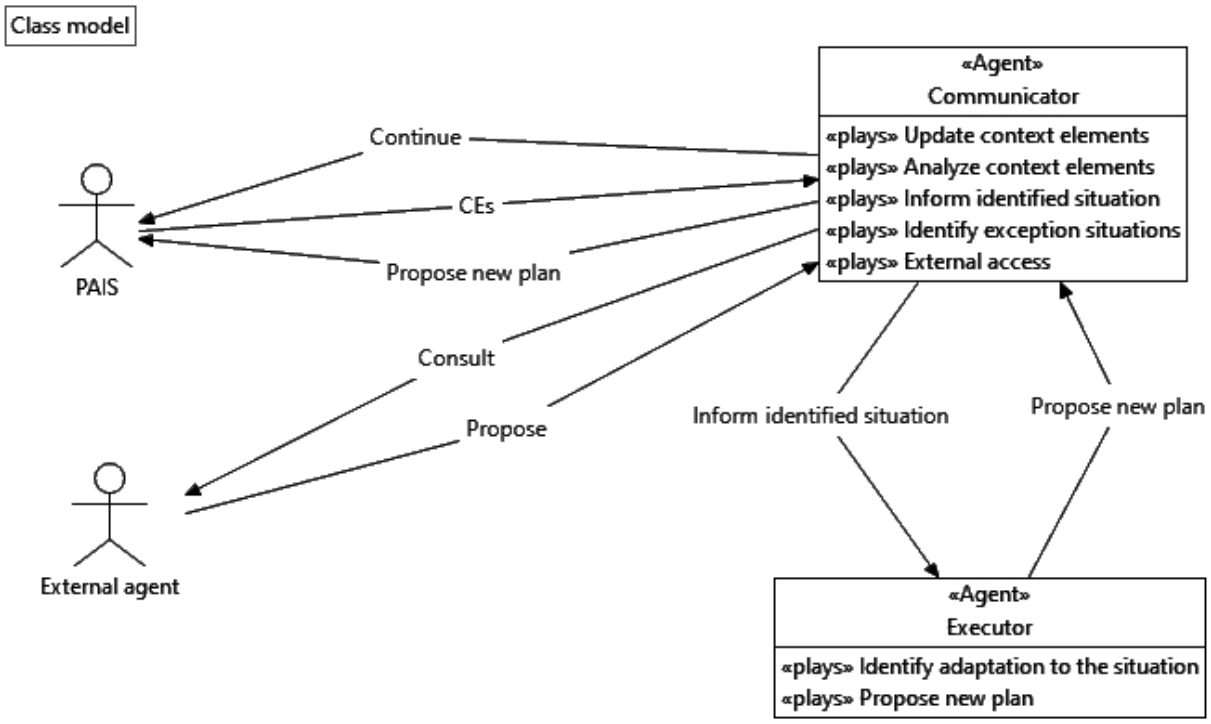
Figura 18 - Modelo de funções do `AAdapt`.



Legenda: Modelo de funções do `AAdapt`.
Fonte: 'O autor'.

um agente sistêmico, PAIS, que interage com o agente `Communicator` para a consecução do objetivo de fornecer um processo alternativo na eventualidade de uma exceção no processo padrão.

Figura 19 - Modelo de classes de agentes.



Legenda: Modelo de classes de agentes `AAdapt`.
Fonte: 'O autor'.

Uma ferramenta de modelagem para o O-MaSE é a agentTool3 (aT3) que possui integração com o *framework* de programação Eclipse Java fornecendo ferramentas para a criação de modelos de classe, funções, metas, entre outros, que permite a análise, projeto e implementação de sistemas multiagentes seguindo a metodologia O-MaSE. O `agentTool3` é um *plug-in* para ambiente de desenvolvimento gráfico baseado em Java e foi utilizado para a criação gráfica da modelagem do modelo de metas, de funções e de classe de agentes, Figuras 17, 18 e 19, respectivamente.

3.3 Ontologia do domínio do processo

Para cada processo de negócio que seja atendido pelo `AAdapt` deve ser criada uma ontologia de domínio do processo para estabelecer para aos agentes as informações de contexto, os limites para as *Belief* e *Desire* possibilitando as inferências no processo de negócio. No processo de negócio de atracar embarcação foi criada uma ontologia do domínio do processo que permite o carregamento das informações de contexto, definição de limites de velocidade do vento e da corrente marítima e a manipulação e inferência da ontologia de domínio do processo. Para essa implementação foi utilizada a API OWL Java em um projeto Maven, o carregamento da ontologia pelo agente pode ser visto no *algoritmo* 3.1.

```
1 public static void main( String[] args )
2     {
3       OWLOntologyManager man = OWLManager.createOWLOntologyManager();
4       File file = new
            File("C:\\Maven_src\\ontologia\\onto1\\DockNavyProcess.owl");
5       OWLOntology o = null;
6       try {
7          o = man.loadOntologyFromOntologyDocument(file);
8       }
9    }
```

Algorithm 3.1 - Carregando a ontologia

Os agentes do `AAdapt` obtém parte das informações de contexto da ontologia do domínio do processo. No caso do processo de atracar uma embarcação dois grandes impeditivos são a velocidade da corrente marítima acima de 2kn nós (3,704Km/h) e o vento forte acima dos 20kn (37,04Km/h), como pode ser visto na definição de valores máximos aceitáveis, *DataMaxCardinality cardinality*, dos atributos *VelocityCurrent* e *VelocityWind*, *algoritmo* 3.2.

```
<ObjectPropertyRange>
        <ObjectProperty IRI="#Dock"/>
        <DataMaxCardinality cardinality="2">
            <DataProperty IRI="#VelocityCurrent"/>
            <Datatype abbreviatedIRI="xsd:double"/>
        </DataMaxCardinality>
    </ObjectPropertyRange>
    <ObjectPropertyRange>
        <ObjectProperty IRI="#Dock"/>
        <DataMaxCardinality cardinality="20">
            <DataProperty IRI="#VelocityWind"/>
            <Datatype abbreviatedIRI="xsd:double"/>
        </DataMaxCardinality>
    </ObjectPropertyRange>
```

Algorithm 3.2 - Restrições na ontologia

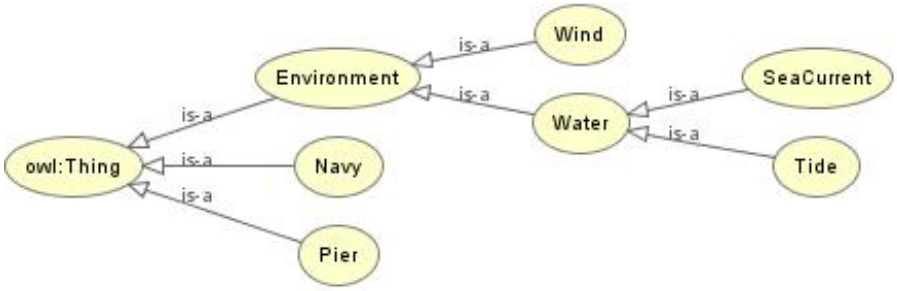
A Ontologia do domínio do processo é utilizada pelo `AAdapt` para realizar inferências utilizando os atributos dos objetos da ontologia, *algoritmo* 3.3.

```
public static HashMap<Object, Object> readData() {
      OWLOntologyManager man = OWLManager.createOWLOntologyManager();

      HashMap<Object, Object> map = new HashMap<Object, Object>();

      File file = new File("C:\\WorkspaceAdapt\\Ontologia\\navy5.owl");
      OWLOntology ontology;

      try {
         ontology = man.loadOntologyFromOntologyDocument(file);
         Set<SWRLRule> rules = ontology.getAxioms(AxiomType.SWRL_RULE);
         System.out.print("rules: ");
         System.out.println(rules);

         OWLReasonerFactory reasonerFactory = new ReasonerFactory();
         OWLReasoner reasoner = reasonerFactory.createReasoner(ontology);
         OWLDataFactory df = man.getOWLDataFactory();
         ...
       }
   }
```

Algorithm 3.3 - Inferência com a ontologia

A Figura 20 apresenta a hierarquia de classes da ontologia OWL do processo de atracar embarcação com as classes *Environment, Navy* e *Pier* no primeiro nível e suas respectivas classes filhas. A ontologia do domínio do processo de atracar embarcação foi

Figura 20 - Visão gráfica da ontologia do domínio do processo



Legenda: Ontologia do domínio do processo de atracar embarcação.
Fonte: 'O autor.'.

criada utilizando a linguagem *Web Ontology Language* (OWL), com o software Protégé. [2]

3.4 **Informações de Contexto**

As informações de contexto são essenciais para o funcionamento dos agentes do AAdapt. Elas são obtidas a partir da ontologia do domínio do processo de negócios e dos sensores externos (*external sensors*). Da ontologia são obtidas informações de contexto mais estruturais como dados do atracadouro e condições ambientais aceitáveis, já os sensores externos capturam informações de contexto ambientais que são dinâmicas como velocidade do vento e corrente marítima.

Com as informações de contexto os agentes podem identificar exceções e propor ações de contorno ao PAIS que impeçam falhas na execução da instância do processo.

3.5 **Funcionamento dos agentes**

Quando uma instância do processo é iniciada no PAIS a arquitetura `AAdapt` começa o monitoramento e coleta de informações do ambiente da instância do processo. O agente `Communicator` é o responsável por receber os elementos de contexto (CE's) e compará-los

[2] Pode ser obtido gratuitamente em https://protege.stanford.edu/

ante uma base de dados de crenças e objetivos estabelecidos. Os elementos de contexto são recebidos do PAIS a partir das tarefas da instância do processo e da ontologia do domínio para identificar situações de exceção na instância do processo. Somente quando o `Communicator` identifica uma situação de exceção, o agente `Executor` é acionado para tratá-la. Em situações de exceção específicas o `Communicator` consulta um agente externo (humano) para auxiliar com a sua experiência na tomada da decisão, a Figura 16 apresenta a arquitetura do `AAdapt` e a possibilidade de uso de um agente externo (*External Agent*).

Caso o agente `Communicator` não identifique uma situação de exceção, ele orienta o PAIS a avançar para a sua próxima tarefa do caminho padrão da instância do processo. Nesse caso não há interação com o agente `Executor`. Há vários ciclos de coleta de informações de contexto, análise do agente `Communicator` e avanço pelas tarefas da instância do processo até o encerramento daquela instância, podendo existir simultaneamente várias instâncias do processo e várias instâncias (papéis) dos agentes.

Para identificar uma situação de exceção, são analisadas crenças e objetivos do agente, os *@Belief* e *@Goal*. Na ocorrência de uma situação de exceção o agente `Communicator` aciona o agente `Executor` que procura por um caminho alternativo para o processo. Esses caminhos alternativos são modelados na fase de *design* do processo adicionando *gateways* de decisão e tarefas que possibilitam as soluções de contorno. Essas decisões de qual caminho seguir são retornados pelo `Executor` para o agente `Communicator` que interage com o PAIS. As tabelas 6 e 7 apresentam as responsabilidades dos agentes dentro do ambiente multiagente.

Tabela 6 - Responsabilidades do agente `Executor`.

Atividade realizada
Recebe do agente `Communicator` uma situação de exceção identificada.
Identifica um plano para a situação de exceção identificada.
Encaminha para o agente `Communicator` a solução de contorno.

Legenda: Atividades realizadas pelo agente `Executor`.
Fonte: 'O autor.'.

O código dos agentes do `AAdapt` foi desenvolvido na linguagem Java e com o *framework* Jadex. Além das classes dos agentes, ExecutorBDI e CommunicatorBDI, foram necessárias as classes auxiliares *HandleDataGoal, OWLParser* e *Utils* as quais são brevemente descritas.

***HandleDataGoal*:** é uma classe auxiliar para tratar informações recebidas do PAIS Camunda.

***OWLParser*:** é uma classe auxiliar que lida com funções relativas ao arquivo de ontologia para utilização de *DataProperty*, *ObjectProperty*, busca de objetos pelo IRI (*Internationalized Resource Identifier*), *parse* de valores em *string* ou em inteiros para

Tabela 7 - Responsabilidades do agente `Communicator`.

Atividade realizada
Recebe informações (elementos) de contexto do PAIS.
Recebe informações (elementos) de contexto dos sensores externos.
Carrega informações (elementos) da ontologia.
Verifica se suas crenças e objetivos não foram violados.
Identifica situações de exceção no processo.
Consulta agente externo.
Encaminha situação de exceção identificada para o agente `Executor`.
Encaminha para o PAIS a solução de contorno identificada pelo agente `Executor`.
Informa ao PAIS continuar o processo normalmente.

Legenda: Atividades realizadas pelo agente `Communicator`.
Fonte: 'O autor.'.

referenciação numérica, como a velocidade máxima suportada pelo píer. A OWLParser analisa a ontologia e adiciona os axiomas no objeto OWLOntology passado para os métodos de análise.

***Utils*:** é uma classe auxiliar com funções relativas ao envio de requisições web POST/GET, utilizado para comunicação com o PAIS Camunda.

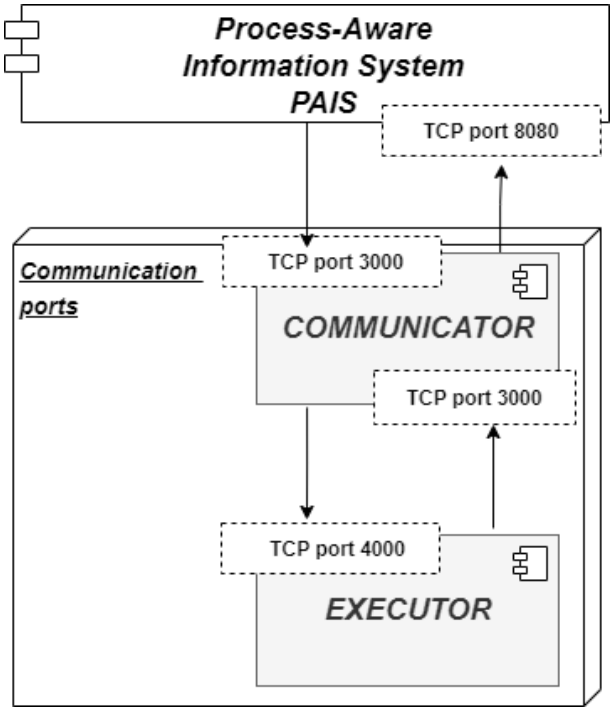
3.5.1 Comunicação entre o AAdapt e o PAIS

A comunicação entre o sistema PAIS e os agentes é realizada através de *server sockets* pelo protocolo HTTP utilizando as portas de serviço TCP 3000 do lado do agente e na porta TCP 8080 no lado do servidor web do motor de processos do Camunda, como apresentado na Figura 21. O PAIS Camunda envia informações de contexto da execução do processo constantemente para o agente, essas informações são analisadas pelo `AAdapt` e, se não modificarem *Belief* ou *Goals*, o agente devolve ao PAIS a informação para seguir o caminho normal (padrão) do processo. Caso as informações de contexto alterem *Belief* ou *Goals* o agente devolve para o PAIS o caminho a ser seguido na próxima tarefa. A devolução para o PAIS é realizada pela porta TCP 8080 e o tipo de dado retornado depende do estágio em que o processo se encontra e o plano que foi acionado.

As portas de comunicação entre o PAIS e o agente podem ser alteradas conforme a preferência, somente se tomando o cuidado para não utilizar as portas TCP baixas, até 1024, que são portas destinadas a serviços conhecidos como 22 do SSH (*Secure Shell Protocol*), 53 do DNS (*Domain Name System*) e 389 do serviço de diretório. A troca da porta utilizada pelo agente é realizada dentro do código do agente, já a porta de

comunicação do PAIS pode ser realizada na instalação do servidor Web do motor de gestão de processos. Caso o PAIS utilizado seja a ferramenta Camunda, a configuração de porta TCP utilizada pode ser alterada nas configurações do Apache Tomcat.

Figura 21 - Comunicação



Legenda: Comunicação TCP entre os agentes e o PAIS.
Fonte: 'O autor.'.

A comunicação interna entre os agentes `Communicator` e `Executor` ocorre pela porta TCP 4000. Nessa comunicação o `Communicator` envia situações identificadas ao `Executor` que devolve a solução de contorno. O agente `Executor` não contata diretamente o PAIS, ele sempre se comunica com o outro agente. As classes Java ServerSocket foram usadas para programação de socket orientada à conexão na comunicação do PAIS e o `AAdapt` e internamente entre os agentes.

O agente recebe informações de contexto do PAIS pela porta TCP 3000, para isso é criado um receptor que fica ativo até o fim da execução do agente, conforme código em *algoritmo* 3.4.

```
1  public void run() {
2                 try {
3                   receiver = new ServerSocket(3000);
4
5                     while (true) {
6
7                         final Socket client = receiver.accept();
8
9                         execFeature.scheduleStep(new IComponentStep<Void>() {
10                            @Override
11                            @Classname("receiveData")
12                            public IFuture<Void> execute(IInternalAccess ia) {
13                                bdiFeature.dispatchTopLevelGoal(new
                                      ReceiveData(client));
14                                return IFuture.DONE;
15                            }
16                        });
17                    }
18                } catch (IOException e) {
19                    throw new RuntimeException(e.getMessage());
20                }
21            }
```

Algorithm 3.4 - Recebendo informações do PAIS

Em seguida é criada uma entidade HTTP para enviar dados para o PAIS, identificando o ID da tarefa no processo que precisa receber as informações do agente. Após a identificação da tarefa é enviada a requisição HTTP com os dados de resposta do agente.

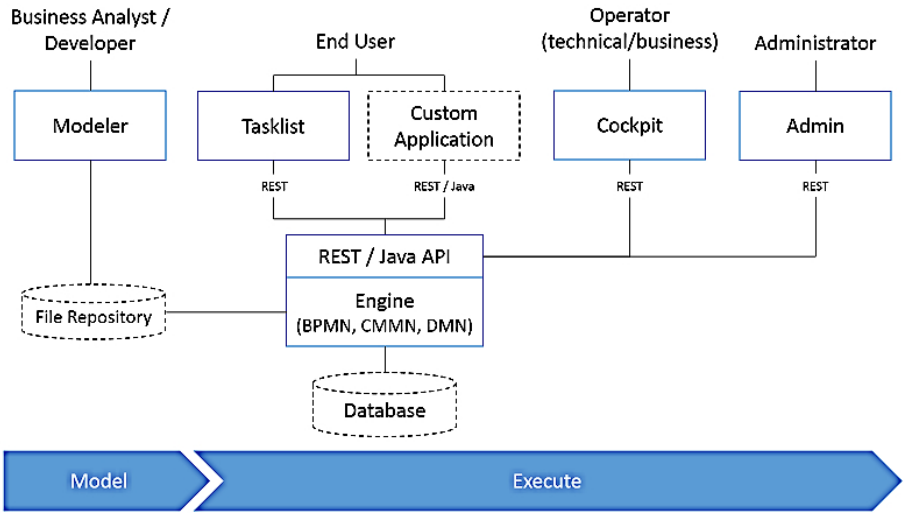
3.5.2 Integração com o PAIS Camunda

O *Process-Aware Information Systems* (PAIS) Camunda foi escolhido para a implementação do processo de negócios por ser um padrão aberto e possuir integração com a linguagem JAVA. Outros sistemas de gerenciamento de processos de negócios também podem ser utilizados desde que respeitados os seus padrões de integração e de interface.

O Camunda é um sistema de padrões abertos e escalonável para automação de processos que permite projetar e automatizar as tarefas de um processo. Utiliza os pa-

drões BPMN e DMN e permite a integração de diferentes tecnologias como, por exemplo, Java. Ele pode ser visto como um motor de gestão de fluxo de tarefas de um processo, permitindo a definição, execução e acompanhamento de instâncias de processos. No Camunda, é possível acessar o *Workflow Engine* através de REST *(Representational state transfer)* para iniciar instâncias de processo e completar tarefas. Também pode adicionar o *Workflow Engine* como uma dependência simples do Maven e usar uma API Java em seu próprio aplicativo, chamando o código Java de dentro de um fluxo de trabalho BPMN. As duas tecnologias para acesso a *engine* do BPMN são REST e Java API (*application programming interface*), conforme pode ser visto na Figura 22.

Figura 22 - Visão geral da arquitetura do Camunda



Legenda: Módulos da arquitetura do PAIS Camunda.
Fonte: https://docs.camunda.org/ manual/ 7.14/introduction /img/ architecture-overview.jpg.

3.5.2.1 Integrando com o Camunda

A integração do Camunda com os agentes pode ser realizada por serviço remoto REST ou por API Java. REST é um padrão de arquitetura para comunicação entre aplicações e a API Java permite a incorporação no código para a integração entre o `AAdapt` e o PAIS Camunda. No caso da integração por API, algumas classes para a interoperabilidade com o Java são disponibilizadas como "org.camunda.bpm.engine.delegate.DelegateExecution"e "org.camunda.bpm.engine.delegate.JavaDelegate".

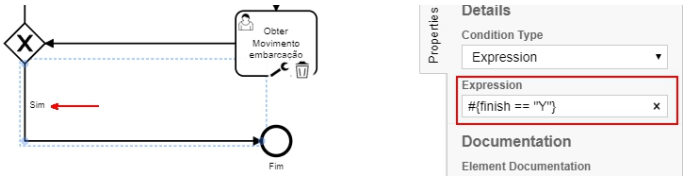
Para operacionalizar esse ambiente conceitual de agentes e multiagentes é necessária uma linguagem de programação e um *framework*. Um exemplo desse ambiente é um ambiente JAVA com as bibliotecas do Jadex para permitir a implementação dos agentes.

Shehory e Sturm (2014) afirmam que essa estrutura requer linguagens de programação dedicadas e estruturas de desenvolvimento que permitem implementações diretas e eficazes de conceitos e abstrações relacionados, como recursos, serviços e objetos.

3.5.2.2 Configurando o Camunda para receber os caminhos alternativos do agente

O agente `Communicator` devolve ao PAIS o caminho (tarefa) a ser seguido. Dentro do processo esse retorno é capturado e tratado no fluxo da mensagem que sai de um *gateway* de decisão (representação gráfica de um losango). Conforme pode ser visto na Figura 23, há um fluxo saindo do *gateway* de decisão e vai para o fim do processo, com a representação gráfica de um círculo. No *design* do processo no Camunda pode ser definida a expressão que identifica quando seguir por esse fluxo. Essa expressão é o retorno que o agente `Communicator` devolve para o PAIS permitindo-o decidir entre os diversos caminhos existentes.

Figura 23 - Retorno do agente



Legenda: O PAIS seleciona o caminho a ser seguido pelo valor retornado do agente.
Fonte: 'O autor.'.

3.5.2.3 Enviando o caminho a ser seguindo pelo Camunda

No agente `Communicator`, a classe sendPAISData é responsável por devolver ao Camunda o caminho a ser seguido, seja ele o caminho padrão ou uma solução de contorno. Antes de enviar o resultado para o PAIS, o agente busca o ID da tarefa atual. Isso é necessário para identificar a tarefa do Camunda que está aguardando a resposta do agente com o caminho que o Camunda vai seguir no *gateway* de decisão, conforme *algoritmo* 3.5:

```
1  public String sendPAISData(String data) {
2      try {
3          String url = "http://localhost:8080/engine-rest/task/" +
               this.getProcessID() + "/complete";
4              return Utils.sendPOST(url, new JSONObject(data));
```

```
5
6       } catch (Exception e) {
7           e.printStackTrace();
8         return null;
9       }
10    }
```

Algorithm 3.5 - Devolvendo dados para o PAIS

3.5.2.4 Visualizando a execução das instâncias do processo no Camunda

O Camunda permite a visualização gráfica no portal do andamento de uma instância do processo, Figura 24, ou de duas ou mais instâncias do processo em execução simultaneamente, Figura 25. O Camunda apresenta cada instância que está sendo executada com uma representação gráfica circular com fundo azul. O número dentro da representação gráfica informa a quantidade de instâncias do processo naquela tarefa.

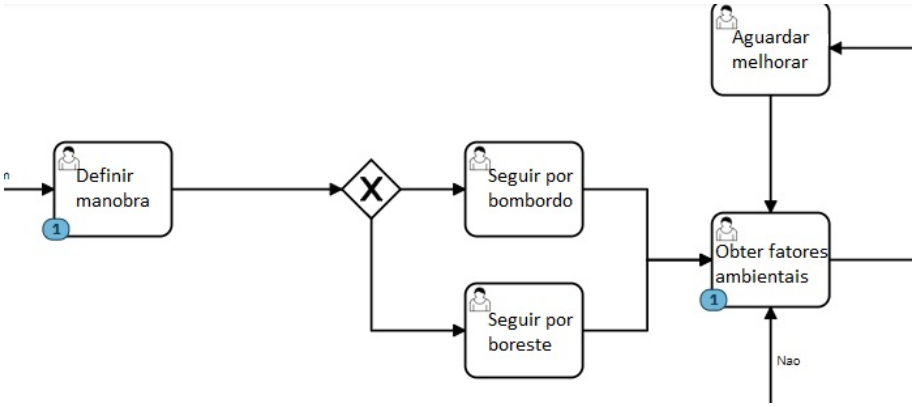
Figura 24 - Visão gráfica da execução do processo

Legenda: Uma representação gráfica azul de um círculo apresenta uma instância do processo.
Fonte: 'O autor.'.

3.5.2.5 Criando um projeto Maven para a integração dos agentes com o Camunda

Para a integração do `AAdapt` com o Camunda foi necessário adicionar nas preferências *Archetypes* do Eclipse o catálogo remoto do Camunda *Add Remote Catalog*. Após a configuração do *Remote Catalog* foi criado um projeto Maven no Eclipse com o *Dropdown Catalog* Camunda e a opção camunda-archetype-servlet-war. Em *Group ID*, foi configurado com.camunda.demo e em *Artifact ID*. Com o projeto Maven criado, foi editado o *workflow* associado a esse projeto, que estava na pasta src/main/resources/ para conter as tarefas do processo de negócio com suas particularidades.

Figura 25 - Visão gráfica de duas instâncias do processo



Legenda: Duas instâncias do processo sendo executadas.
Fonte: 'O autor.'.

3.5.2.6 Gerando o .war e implantando o projeto

O próximo passo foi gerar um arquivo .war (*Web Application Resource*) que é um arquivo usado para distribuir uma coleção de arquivos JAR, *Java Servlets*, *JavaServer Pages*, classes Java e outros objetos de um projeto. Foi necessário ajustar também o arquivo pom.xml (arquivo de configuração utilizado no Maven) *tag plugin*, conforme *algoritmo* 3.6:

```
...
<plugin>
   <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>2.5</version>
        <configuration>
            <skipTests>false</skipTests>
            <testFailureIgnore>true</testFailureIgnore>
            <forkMode>once</forkMode>
        </configuration>
</plugin>
```

Algorithm 3.6 - Ajustando arquivo de configuração

Depois de criado o .war foi necessário iniciar a *engine* do Camunda acionando o *script* disponibilizado pelo fornecedor do Camunda, conforme *algoritmo* 3.7:

```
@echo off

set "CATALINA_HOME=%CD%\server\apache-tomcat-9.0.36"
```

```
4
5  echo "starting camunda BPM platform 7.14.0 on Apache Tomcat 9.0.36"
6
7  cd server\apache-tomcat-9.0.36\bin\
8  start startup.bat
9
10 ping -n 5 localhost > NULL
11 start http://localhost:8080/camunda-welcome/index.html
```

Algorithm 3.7 - Iniciando a *engine*

O último passo foi o *deploy* do projeto na pasta do Camunda. Para isso foi copiado o arquivo .war para o diretório /DiretoriodeInstalacaoCamunda/server/ apache-tomcat-9.0.36/webapps. Imediatamente após a cópia do arquivo .war um novo diretório é criado contendo o projeto Maven.

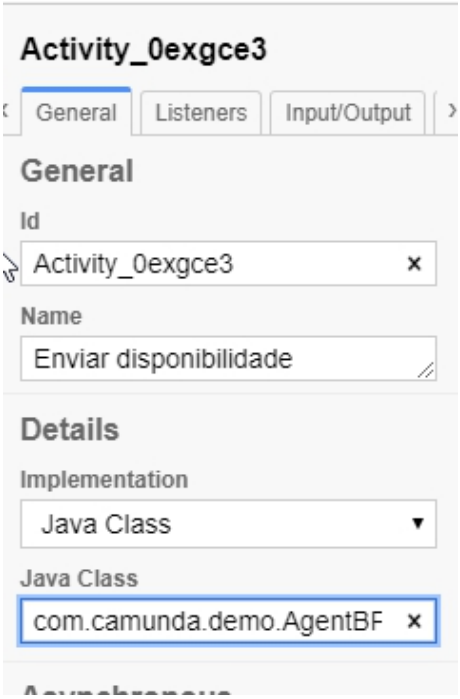
3.5.2.7 Criando o processo de negócio no Camunda

A construção do processo de negócio foi realizada no Camunda Modeller. Sua integração com o `AAdapt` é definida nas tarefas que enviam elementos de contexto para o agente `Communicator`, por meio de uma classe Java. No exemplo, a classe "com. camunda. demo. AgentBPMN. SendAvaiableToAgent"envia o elemento de contexto espaço disponível no pier para o agente, Figura 26.

Para a implementação da classe com. camunda. demo. AgentBPMN. SendAvaiableToAgent no Java foram necessários *imports* específicos do Camunda, configuração de porta e outras definições, conforme *algoritmo* 3.8:

```
1
2  public class SendAvaiableToAgent implements JavaDelegate {
3     @Override
4     public void execute(DelegateExecution execution) throws Exception {
5        HttpClient httpclient = HttpClients.createDefault();
6        HttpPost httppost = new HttpPost("http://localhost:3000");
7        String json = "{\"available\":\""+
             execution.getVariable("available").toString() + "\"}";
8         StringEntity entity = new StringEntity(json);
9         httppost.setEntity(entity);
10        httppost.setHeader("Accept", "application/json");
11        httppost.setHeader("Content-type", "application/json");
12       httpclient.execute(httppost);
13    }
```

Figura 26 - Configuração da conexão com Java Class



Legenda: Utilização da classe para conexão e envio de CE.

Fonte: 'O autor.'.

```
14 }
```

Algorithm 3.8 - Implementando o SendAvaiableToAgent

3.5.2.8 Encerrando o Camunda

Após o encerramento do processo, a *engine* do Camunda pode ser encerrada por *script*, conforme *algoritmo* 3.9:

```
1 @echo off
2 set "CATALINA\_HOME=%CD%\server\apache-tomcat-9.0.36"
3 cd server\apache-tomcat-9.0.36\bin\start shutdown.bat
```

Algorithm 3.9 - Encerrando o PAIS Camunda

3.6 **BDI**

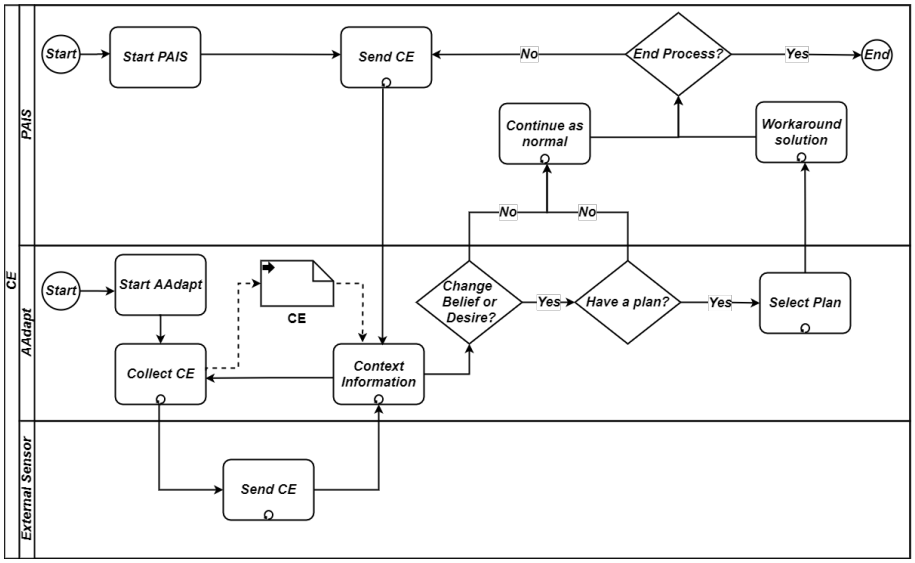
Os agentes reagem a eventos executando um plano e os eventos acontecem em razão de mudanças nas *Beliefs* ou *Goals*. Os agentes do `AAdapt` possuem raciocínio e tomada de decisão baseado em suas crenças, desejos e intenções. Esses agentes inteligentes se utilizam de uma base de conhecimento prévia ou adquirida durante as execuções para chegar a conclusões, acionar ou elaborar um plano e para tomar decisões.

As informações de contexto do ambiente são obtidas constantemente dos sensores externos e do PAIS e no carregamento de informações da ontologia do domínio do processo. Os agentes precisam analisar essas informações e verificar se *beliefs and desires* não foram violadas. Caso as *belief* ou *desire* sejam violadas o `AAdapt` procura por um plano (*intention*) que proponha uma solução de contorno para o processo em execução no PAIS, Fig. 27.

Caso não sejam violadas as *belief* ou *desire* o agente informa ao PAIS para continuar com o fluxo padrão do processo, situação identificada como "*Continue as normal*".

Os agentes com BDI são criados em ambientes de programação como `Jason, Jadex, 2APL` e `Jack`. Nesses códigos são detalhadas as instruções a serem executadas no plano, as informações do ambiente (crenças) e os eventos ligados aos desejos e às crenças que acionam a execução dos planos.

Figura 27 - Funcionamento interno do `AAdapt`



Legenda: Fluxo interno de verificação de CE, *Belief* e *Desire* e execução de planos (*Intention*).
Fonte: 'O autor'.

3.6.1 Jadex - raciocínio baseado em BDI

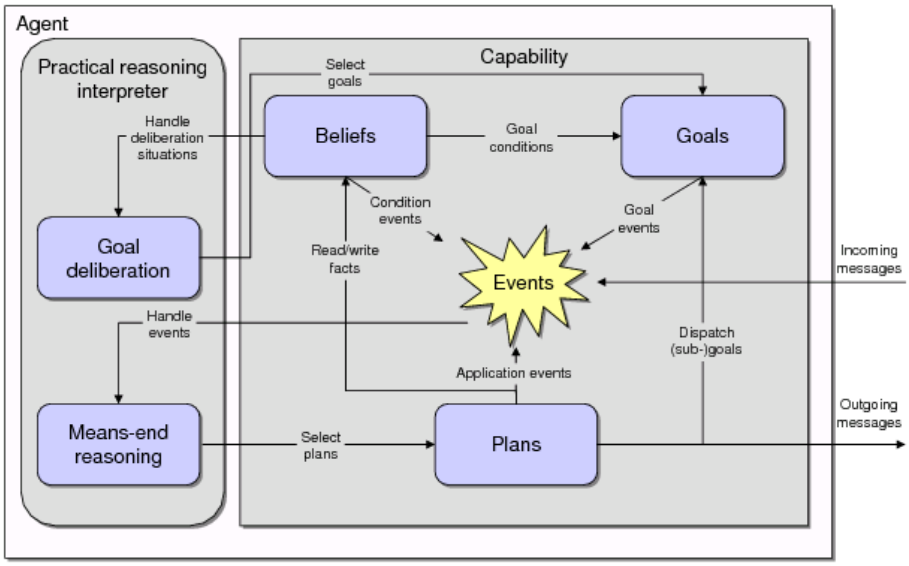
O Jadex é um mecanismo de raciocínio baseado em crenças, desejos e intenções que permite desenvolver agentes inteligentes distribuídos em Java e XML (*Extensible Markup Language*) por meio de uma técnica de componentes ativos (*Active Components*), combinando uma arquitetura de componente de serviço hierárquica com lógica de negócios baseada em agentes BDI, BPMN *(Business Process Modelling Notation)*, *Workflows*, micro agentes e componentes XML. Ele permite utilizar as técnicas de componentes ativos para desenvolver aplicações distribuídas em linguagem imperativa de programação. É baseado no Jade (Java *Agent Development framework*) e o estende com construções de programação para implementar conceitos de BDI.

Um componente ativo é uma unidade de software autônoma, gerenciada e hierárquica que utiliza funcionalidades através de serviços e tem o comportamento moldado na arquitetura interna, Figura 28.

Os agentes do `AAdapt` foram desenvolvidos utilizando o Jadex porque esse *framework* permite criar agentes orientados a objetivos com o modelo BDI e por ter uma relação direta das *annotations Belief, Goal and Plan* e os conceitos BDI, conforme pode ser visto na Figura 29, o que foi um facilitador.

O modelo de comunicação entre os componentes é baseado em serviço e sua *interface* pode ser implementada como uma classe Java. O conjunto desses serviços é referido como plataforma de serviços Jadex e permite em tempo de execução a prestação de servi-

Figura 28 - Arquitetura Jadex.



Legenda: Arquitetura Jadex.
Fonte: https://download.actoron.com/docs/releases/latest/jadex-mkdocs/guides/bdiv3/jadexabstractarchitecture.png.

Figura 29 - BDI e Java *Annotations*.

BDI	*Annotation in* JAVA / JADEX
Belief	*Belief*
Desire	*Goal*
Intention	*Plan*

Legenda: Mapeamento dos conceitos BDI para as anotações do Java
Fonte: 'O autor.'.

ços, o gerenciamento do ciclo de vida, o acionamento de serviços remoto e o gerenciamento de componentes. A arquitetura de agentes BDI descreve claramente o comportamento com objetivos, planos e crenças, distinguindo seus planos e como alcançá-los, suas metas. Dessa forma, facilita o projeto de comportamentos complexos tornando o comportamento do agente mais compreensível à medida que sua meta é alcançada e a execução do plano é realizada.

3.6.1.1 Funcionamento do BDI no Jadex

O BDI é uma das arquiteturas para desenvolvimento de agentes inteligentes simples e robustos, fornecendo a implementação por padrão de várias classes já pré-definidas. Um agente BDI no Java/Jadex é uma classe do Java com uma anotação, que é um mecanismo para inclusão de informações de metadado, sinalizando que se trata de agente. Uma anotação em Java também pode ser aplicada a métodos, parâmetros e pacotes Java. Alguns exemplos de anotações de códigos dos agentes BDI são *@Agent, @AgentBody, @AgentCreated, @AgentFeature, @Belief, @Description, @Goal, @GoalParameter, @GoalResult* e *@Plan*, conforme *algoritmo* 3.10:

```
1  @Agent
2  @Description("Agente ExecutorBDI")
3  @Plans(@Plan(body=@Body(classePlano.class)))
4  public class ExecutorBDI {
5      @AgentFeature
6      protected IBDIAgentFeature bdiFeature;
7      @AgentBody
8      public void body()
9     {
10      bdiFeature.adoptPlan(new classePlano());
11     ...    }
12 }
```

Algorithm 3.10 - Anotações no agente ExecutorBDI

Outro exemplo de uso das anotações no Java/Jadex é a criação e configuração do agente com a anotação *@AgentCreated* que cria o agente e a anotação *@AgentBody* onde são definidas as variáveis e serviços disponíveis, *algoritmo* 3.10.

O propósito do BDI é munir os agentes como se eles possuíssem estado mental composto do que o agente sabe sobre o ambiente, os possíveis estados dos objetos com os quais o agente interage e as ações que o agente realiza. No Java/Jadex é possível utilizar componentes ativos para desenvolver aplicativos distribuídos, combinando a arquitetura

de componente de serviço hierárquico com a implementação de lógica de negócios, podendo integrar fluxos de trabalho (motor de processos) com agentes inteligentes. Um agente é composto de *beliefs*, *plans*, *goals*, *goal deliberation* e *means-end-reasoning*, Figura 28. As crenças podem ser definidas em campos ou em métodos como *getter* e *setter* do Java e são identificadas pela anotação *Belief*. Como o *algoritmo* 3.11.

```
1 @Belief(updaterate=1000)
2 protected long time = System.currentTimeMillis();
3
4 @Belief
5 protected Map<String, String> wordtable;
6
7 @Belief
8 public long getException()
```

Algorithm 3.11 - Crenças

A programação orientada a objetivos está relacionada a agentes que utilizam técnicas para atender suas crenças e objetivos tentando todos os planos existentes em sua codificação para atingir as suas metas. Há algumas metas que são essenciais para o agente, ele vai tentar executar todos os planos até atingi-la. Muitas vezes as metas dependem que outras metas sejam atendidas para ela ser possível. As metas podem ser implementadas como uma classe separada ou interna do Java e podem ser atendidas por diferentes planos. As metas podem ter submetas que podem, por suas vez, terem outras submetas e são representadas pela anotação *@Goal*. Um exemplo de metas por ser visto no *algoritmo* 3.12. Uma meta é uma definição mais macro e é diferente de um plano porque descreve um objetivo sem indicar como deve ser alcançado.

```
1 @GoalMaintainCondition(beliefs="distanciaEmbarcacoes")
2
3 @GoalTargetCondition(beliefs="atracarEmbarcacao")
```

Algorithm 3.12 - Metas

Plano é o mecanismo que permite executar códigos dentro do agente, ele é composto de duas partes, o cabeçalho que contém meta informações sobre o plano e como ele é acionado e o corpo onde é descrito o que deve ser executado. Os planos de um agente contêm as etapas concretas para se aproximar do cumprimento da meta (POKAHR et al., 2014). Um plano pode ser utilizado como uma classe separada do Java, como um método ou como uma classe interna. Cada tipo de utilização tem suas vantagens,como por exemplo, uma classe separada que facilita a reutilização do código. Os eventos e objetivos de disparo (*trigger*) são especificados no cabeçalho do plano (*plan*) permitindo ao agente conhecer quando deve ser acionado.

A identificação de um plano e seu objetivo são realizados pela inserção da anotação *@Plan*, tendo como exemplos as anotações *@PlanPassed, @PlanAborted, @PlanFailed, @Plan(trigger = @Trigger(factchangeds = "alarm"))* e *@Plan(trigger = @Trigger(goals = NomeClasse.class))*. As anotações de métodos de plano *@PlanPassed, @PlanBody, @PlanFailed* e *@PlanAborted* são usadas respectivamente quando o plano é executado com sucesso, na definição do corpo do plano, quando ocorre uma falha com exceção ou quando o plano falha e é abortado.

Além das anotações de planos citadas, há outras que também podem ser utilizadas de acordo com a finalidade de seu acionamento *@PlanAPI, @PlanbaseWrapper, @PlanCapability, @PlanContextCondition, @PlanFinishedTaskCondition, @PlanPrecondition* e *@PlanReason*. Diferentes planos podem ser utilizados para atender uma meta. Já as *trigger* podem assumir a adição de uma nova *belief* com *factadded*, a alteração com *factchanged*, a remoção com *factremoved*, a finalização de uma meta por meio de *goalfinisheds*, reação a uma meta utilizando *goals* ou ativação de um serviço com *service*.

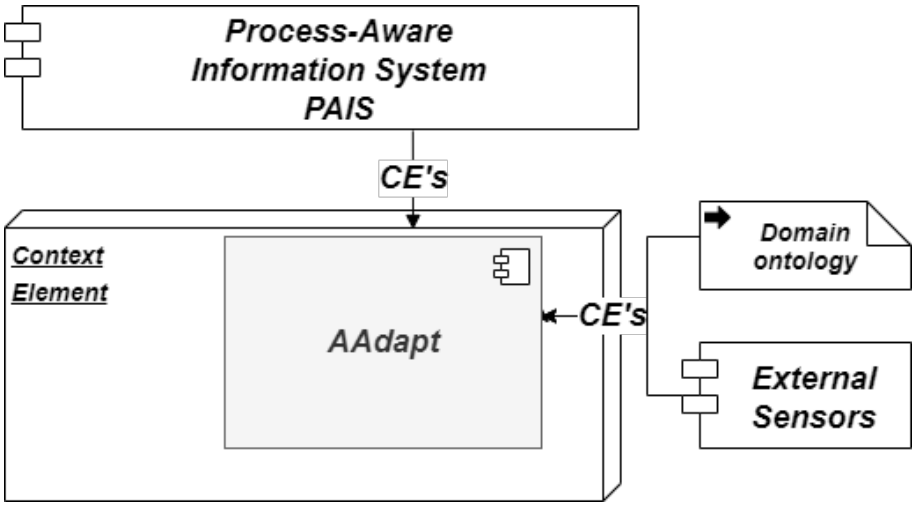
Há uma associação entre *Belief, Desire* e *Intention* do BDI com as anotações Java *@Belief*, *@Goal* e *@Plan*. O *Belief* do BDI e a anotação *@Belief* do Java possuem a incumbência de definir o que o agente conhece sobre o ambiente, *@Desire* e *@Goal* constituem o desejo de alcançar uma meta e por fim *Intention* e *@Plan* que são os planos específicos para atingir um objetivo.

O JADEX possui três formas de realizar comunicação entre agentes, através de mensagens como FIPA e outros formatos de mensagem, por meio de chamadas de serviços ou definindo o módulo estaticamente. Na comunicação por mensagens é utilizada a *interface IMessageFeature* para envio e tratamento das mensagens. Na comunicação por serviços os agentes utilizam as anotações *@RequiredServices* e *@ProvidedServices* para requerer ou fornecer um serviço a outro agente. Os serviços no Jadex consistem em uma especificação de interface e uma implementação de serviço. As interfaces de serviço são definidas como interfaces Java e implementações como classes Java e permitem a comunicação entre agentes pela implementação de classes de serviços providos e serviços requeridos. Por fim uma opção de comunicação por *socket* TCP no Java. Um *socket* é um *endpoint* numa comunicação entre dois programas de rede, nesse caso entre dois agentes, que utilizam uma ligação por porta TCP para realizar o processo de comunicação. Por exemplo, o agente `Executor` escuta e aceita tráfego pelo porta TCP 4000, *receiver = new ServerSocket(4000);*, o agente `Communicator` escuta o tráfego pela porta 3000, *receiver = new ServerSocket(3000);*. Nos códigos dos agentes há outras instruções para tratamento do tráfego, porém, a classe do Java *ServerSocket* é a responsável por definir o número da porta TCP pela qual será esperada a conexão do cliente PAIS ou de outro agente.

A Figura 30 mostra a obtenção de informações de contexto de três fontes, da ontologia de domínio, dos sensores externos e do PAIS. As informações de contexto são utilizadas para identificar o estado atual do processo e dos artefatos envolvidos, permitindo

que o agente valide suas crenças e desejos com os *Beliefs* e *Goals*.

Figura 30 - Obtendo informações de contexto (CE's)



Legenda: Informações de contexto obtidade de sensores, PAIS e ontologia.
Fonte: 'O autor'.

4 CENÁRIO DE APLICAÇÃO

Durante a execução de uma instância de processo diversos problemas podem ocorrer, entre eles estão a mudança de contexto durante a execução e a dependência da experiência humana para a tomada de decisões. O `AAdapt` apresentou um sistema multiagente BDI de adaptação dinâmica em tempo de execução das instâncias dos processos, baseado em contexto e que permite a integração com agente externo para tratar processos de conhecimento intensivo.

O cenário escolhido para a avaliação da proposta foi um estudo de caso de um processo intensivo em conhecimento (KiP) de atracação de uma embarcação num pier. Esse processo intensivo em conhecimento foi escolhido porque apresenta uma grande quantidade de informações de contexto dinâmicas em tempo real oriundas de fontes heterogêneas que estão dispersas pelas tarefas do processo de negócio e que, em muitos casos, requerem a intervenção e experiência de um agente externo (humano) para auxiliar na tomada de decisão.

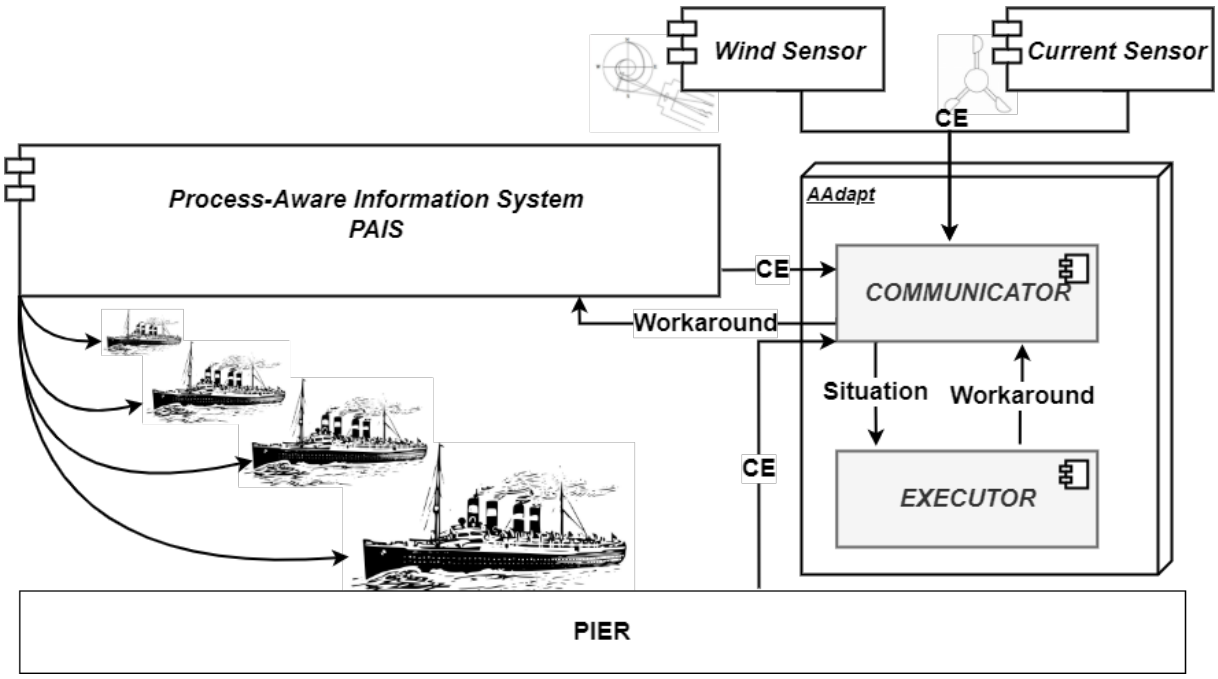
A pesquisa de estudo de caso é o método qualitativo mais popular na pesquisa de sistemas de informação, (RECKER, 2021), envolvendo pesquisa intensiva sobre um fenômeno (um caso) em seu ambiente natural. Para subsidiar as tarefas, premissas, decisões e riscos desse estudo de caso foram realizadas duas entrevistas informais. A primeira entrevista foi realizada com um engenheiro naval formado em engenharia Naval e Oceânica na Universidade Federal do Rio de Janeiro, pós-graduado em Engenharia de Dutos na PUC Rio de Janeiro e mestrado no Departamento de Engenharia Naval e Oceânica – COPPE/ UFRJ. A segunda entrevista foi realizada com sub-oficial aposentado da Marinha do Brasil com mais de 30 anos de experiência na operação naval tendo passagem por diversas situações de condições climática e processos de atracações.

Ambas as entrevistas objetivaram obter informações sobre o processo de negócio, problemas que podem ocorrer durante o processo e fatores estruturais e ambientais a serem considerados para a consecução correta da atracação.

O processo intensivo em conhecimento (KiP) de atracação de uma embarcação contemplou tarefas de um processo real que foram suficientes para ratificar os conceitos que são abordados na pesquisa. Nessas tarefas os *Belief* foram utilizados em situações com atributos sem muita variância de valor, como, se o pier comporta a embarcação, e os *desire* abarcaram atributos com espectro de maior variação, como a velocidade da corrente marítima. Os dados dos sensores externos foram produzidos de forma aleatória numa escala de mar calmo com aspecto espelhado até furacões para garantir que todas as combinações de condições climáticas fossem testadas pelos agentes. Quando os sensores detectavam condições climáticas extremas o `AAdapt` acionou o agente externo (recurso humano) para ratificar a tomada da decisão.

A Figura 31 apresenta uma visão gráfica da integração do `AAdapt` com o PAIS para monitorar e tomar decisões durante o processo de atracação da embarcação.

Figura 31 - Atracando uma embarcação



Legenda: Agentes do `AAdapt` auxiliando na atracação de uma embarcação.
Fonte: 'O autor.'.

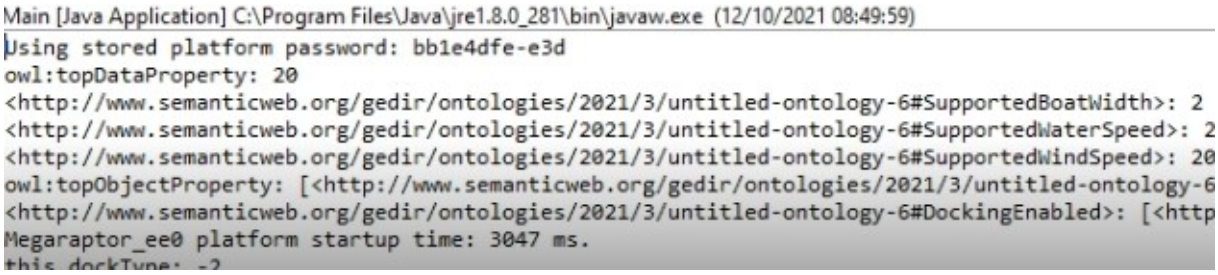
No estudo de caso no sistema `AAdapt` os agentes utilizaram elementos de contexto coletados em tempo de execução oriundos do pier, do PAIS, do *Wind Sensor* e do *Current Sensor* para conduzir o processo de atracação da embarcação com segurança.

4.1 **Iniciando o `AAdapt`**

O `AAdapt` é iniciado dentro do *framework* Eclipse que é um ambiente integrado de desenvolvimento. O pré-requisito para o seu funcionamento é que o PAIS Camunda tenha sido iniciado e tenha carregado o processo de negócio. Ao iniciar o módulo principal (*main*) do `AAdapt` os agentes BDI *Communicator* e *Executor* são instanciados. A Figura 32 mostra o início da execução instanciando os agentes BDI e o carregamento de informações da ontologia do domínio do processo.

A ontologia é utilizada para fazer inferências como, por exemplo, se a velocidade do vento permite realizar a atração da embarcação com segurança. Informações de contexto também são carregadas a partir da ontologia.

Figura 32 - Iniciando o agente



Legenda: Agente iniciando e carregando informações da ontologia do domínio.
Fonte: 'O autor.'.

4.2 Estudo de caso: Processo de atracar a embarcação

Após ser iniciado, o agente *Communicator* monitora os elementos de contexto das tarefas do processo de negócio. Cada tarefa da instância do processo tem um identificador (Id) que é utilizado para a comunicação. O *Communicator* captura esse Id no método *getProcessID* e utiliza-o para enviar respostas ao PAIS utilizando o método *sendPAISData*.

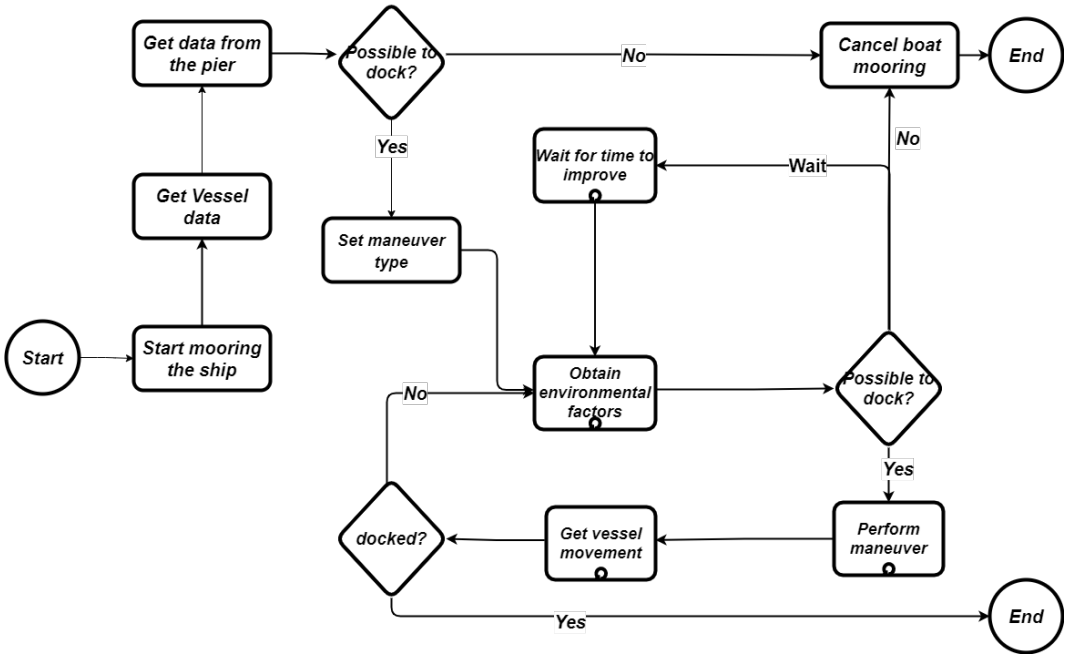
```
1  public String getProcessID() {
2      try {
3          String url =
               "http://localhost:8080/engine-rest/task?processDefinitionKey=AgentBPMN";
4          final JSONObject obj = new
               JSONArray(Utils.sendGET(url)).getJSONObject(0);
5
6            if (obj.has("id"))
7                return obj.getString("id");
8
9        } catch (Exception e) {
10           e.printStackTrace();
11       }
12       return "";
13   }
```

Algorithm 4.1 - Identificando as tarefas

Na aproximação da embarcação ao pier, Figura 33, a tarefa do PAIS ***Start mooring the ship*** coleta elementos de contexto da embarcação como, por exemplo, seu tamanho. Os dados coletados são enviados para o agente `Communicator`. Em seguida, a tarefa ***Get vessel data*** obtém informações do pier, como espaço disponível e tipos possíveis de atracação ***Get data from the pier***. Nesse momento, se o agente `Communicator`

identificar uma situação adversa pode acionar o agente `Executor` para fornecer uma solução de contorno.

Figura 33 - PAIS atracando uma embarcação



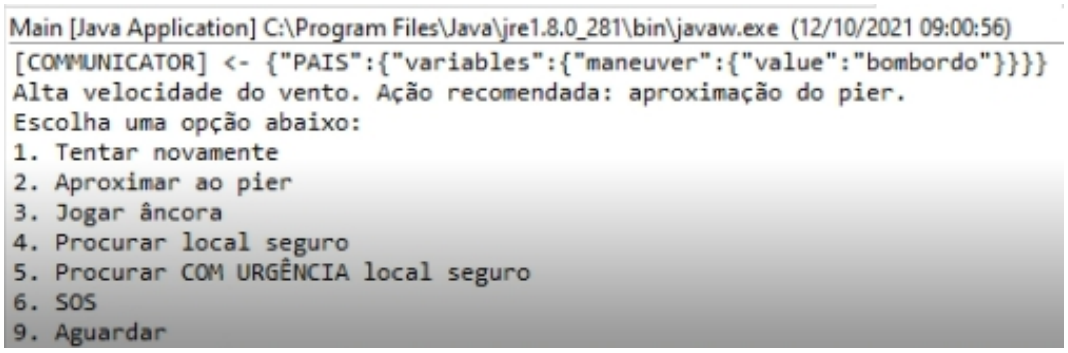
Legenda: Fluxo do processo de atracação de uma embarcação.
Fonte: 'O autor.'.

Caso a atracação não seja cancelada, na próxima tarefa do processo é definido o tipo de manobra a ser realizada na tarefa do processo ***Set maneuver type***, como manobrar de boreste (lado direito) ou bombordo (lado esquerdo). Na tarefa seguinte, dados meteorológicos são coletados, ***Obtain environmental factors***, como velocidade do vento e corrente marítima. Esses elementos de contexto são enviados para o agente `Communicator` que pode identificar uma situação adversa e identificar, por exemplo, risco de abalroamento. Nos casos que forem identificadas situações indesejadas o agente `Executor` é acionado para localizar um plano para contornar a situação.

A fase que obtém os fatores ambientais é o momento mais complexo do processo de atracação devido à possibilidade de mudanças bruscas nas condições climáticas e o impacto que elas exercem sobre o processo de atracação da embarcação. Nesse ponto o `AAdapt` pode solicitar a intervenção de um agente externo (ser humano) que pode utilizar a sua experiência para propor uma solução de contorno. Na Figura 34 uma situação de risco foi identificada pelo *Communicator* e o agente pediu auxílio de um agente externo.

Esse processo de coleta de elementos de contexto continua constante durante as próximas tarefas ***Perform maneuver*** e ***Get vessel movement***, obtendo valores de elementos do contexto e voltando para as condições climáticas, identificando situações e executando plano para contorná-las. Essas tarefas formam um *loop* que vai aproximando

Figura 34 - `AAdapt` coletando experiência de agente externo



Legenda: Processo intensivo de conhecimento obtido de agente externo.
Fonte: 'O autor.'.

a embarcação do pier até que ela esteja completamente atracada, Figura 31.

Casos de teste para verificar informações do tamanho da embarcação e disponibilidade do pier e para verificar a existência de condições climáticas impróprias que impeçam a embarcação de atracar com segurança, conforme pode ser visto com mais detalhes nas tabelas 8 e 9, foram executados para situações identificadas executando planos de contorno quando necessários.

Os casos de teste que foram executados continham situações nas quais:

- o pier não comportava a embarcação;
- a corrente marítima não permitia o atracamento seguro da embarcação;
- a velocidade do vento estava acima do tolerado;
- apresentavam condições desfavoráveis tanto da corrente marítima quanto da velocidade do vento;
- ocorreram mudanças climáticas durante o atracamento;
- a embarcação demorava mais tempo para atracar;
- as condições favoráveis permitiam o atracamento da embarcação sem percalços;
- situações de interação com agente humano externo.

Tabela 8 - Caso de teste Espaço para atracar.

Identificação:	Espaço para atracar.
Objeto de teste:	Fluxo do processo verificando espaço para atracação.
Pré-condição:	Dados da embarcação.
Procedimento:	Obter dados do pier para verificar se a embarcação pode atracar.
Resultado esperado:	Dados do pier obtidos e verificado se o barco pode atracar.

Legenda: Caso de teste do sistema `AAdapt`.
Fonte: 'O autor.'.

Tabela 9 - Caso de teste Condições climáticas impróprias para atracar .

Identificação:	Condições climáticas impróprias para atracar.
Objeto de teste:	Fluxo do processo cancelando a atracação.
Pré-condição:	Espaço disponível no Pier.
Procedimento:	Obter dos sensores externos dados de velocidade do vento ou da corrente marítima fora do intervalo que permite a atracação do barco com segurança.
Resultado esperado:	Dados dos sensores obtidos fora do intervalo e cancelamento pelo PAIS da atracação.

Legenda: Caso de teste do sistema `AAdapt`.
Fonte: 'O autor.'.

A escala de vento Beaufort foi utilizada nos testes do `AAdapt` para o processo de atracar a embarcação. A escala de vento de Beaufort é a mais conhecida pela maioria dos meteorologistas como uma lista de números de código para transmitir observações da velocidade do vento em mensagens meteorológicas sinóticas (HULER, 2005). A descrição do ambiente utilizado foi o do centro de hidrografia da Marinha do Brasil na escala Beafourt (BRASIL, 2021), cujas descrições dos ambientes podem ser vistas na tabela 10.

Tabela 10 - Condições climáticas consideradas.

Velocidade do vento	**Ação recomendada**	**Ambiente**
<= 20	Atracação permitida	Moderado/ Fresco
> 20 e <= 22	Aproximar do pier	Muito Fresco
> 22 e <= 25	Jogar âncora	Muito Fresco
> 25 e <= 33	Procurar local seguro	Forte
> 33 e <= 55	Verificar urgente local seguro	Muito Forte/ Duro/ Muito Duro
> 55	Situação de emergência climática	Tempestade / Furacão

Legenda: Tabela Beaufort adaptada - `AAdapt`.
Fonte: 'O autor.'.

Várias execuções de instâncias de processos exercitaram as situações propostas nas tabelas 8 e 9 aplicando as diversas escalas de condições climáticas definidas na tabela 10. Uma dessas execuções que capta situações extremas e dinâmicas foi documentada em vídeo disponível em Goncalves (2022) com 1 minuto e 47 segundos de duração. Essa execução mostra o início do `AAdapt` no *framework* Eclipse (aos 10s do vídeo) e a inicialização do PAIS Camunda (19s). A primeira visualização gráfica da instância do processo é apresentada aos 23s e, como a execução da instância do processo é muito rápida, já é apresentada tarefa de decisão da atracação - "Atracar?". Nesse ponto a ontologia do domínio do processo foi carregada, as informações de contexto da embarcação e do pier foram analisadas e o `AAdapt` precisa decidir sobre continuar o processo de atracação ou cancelá-lo. A representação iconográfica de um losango (*gateway* de decisão) mostra um ponto de decisão onde um caminho deve ser escolhido e um pequeno círculo preenchido em azul no canto esquerdo inferior sinaliza a tarefa ativa. Nesse ponto o `AAdapt` analisando as informações de contexto obtidas decide por continuar o processo de atracação e definir o tipo de manobra a ser realizada(aos 43s). As próximas tarefas são o cerne do processo de atracação e começam aos 45s. Nelas há um *loop* de análise das condições climáticas e aproximação da embarcação ao pier até que a embarcação seja atracada ou que o processo de atracação seja cancelado. Na execução dessa instância o início da atracação é com condições climáticas favoráveis, o que permitiu a aproximação da embarcação ao pier, mas logo a seguir as condições climáticas degradaram rapidamente o que fez com que o `AAdapt` consultasse um recurso humano (1min25s) para auxiliar com sua experiência que ação deveria ser realizada. O recurso humano orienta a aguardar (1min28s) e o `AAdapt` segue sua orientação. Porém, o `AAdapt` percebe que as condições climáticas pioram (1min40s) e decide por cancelar o processo de atracação.

Nos casos de teste executados o `AAdapt` atendeu aos objetivos propostos em várias condições climáticas. O cenário mais complexo do processo de atracação da embarcação, apresentado no vídeo (GONCALVES, 2022) e descrito no parágrafo anterior, é quando as condições climáticas da corrente marítima e velocidade do vento mudam desfavoravelmente durante o processo de atracação, com valores acima do tolerado. Mesmo nesse ambiente com condições climáticas extremas, o `AAdapt` se comportou como planejado na sua implementação.

4.3 **Discussão**

O problema tratado nessa pesquisa foi **adaptar dinamicamente as tarefas de uma instância de processo de negócio para tratar exceções na execução mitigando a ocorrência de falhas.** Para isso foi desenvolvida uma arquitetura multiagente nomeada `AAdapt` capaz de adaptar processos de negócio em tempo de execução de forma a

diminuir os riscos de falhas na execução da instância do processo. A arquitetura proposta foi capaz de definir e implementar os agentes participantes, especificar o conhecimento sobre o processo por meio de ontologias e definir a forma de interação dos agentes com um sistema PAIS.

Um dos primeiros objetivos tratados na pesquisa foi a definição e a implementação dos agentes participantes da arquitetura. Para a consecução desse objetivo foi modelado e desenvolvido um sistema multiagente composto de dois agentes baseados em crenças, desejos e intenções (BDI) que possibilitaram o acompanhamento e auxílio na execução de um processo intensivo em conhecimento de atracação de embarcação. Para o acompanhamento e auxílio na execução do processo foi realizada a integração do sistema multiagente com um sistema PAIS (Camunda). Para obter o conhecimento do domínio do processo foi modelada uma ontologia do domínio do processo a qual o sistema multiagente foi integrado para obter axiomas e realizar inferências. A construção e integração com a ontologia do domínio do processo também possibilitou que implementações da arquitetura em outros domínios de processos seja realizada com menos modificações em código fonte. Ademais, o sistema multiagente se integrou com sensores externos para capturar informações de contexto de forma dinâmica e que obtivesse auxílio de recurso humano em situações extremas, o que diminuiu os riscos de falhas de execução do processo de negócio intensivo em conhecimento de atracar uma embarcação permitindo que condições de falhas fossem identificadas previamente e que ações de contorno garantissem a segurança da execução do processo e do objeto físico ao qual esse estava associado.

Para garantir que esses objetivos fossem atendidos pela proposta de arquitetura, vários testes foram realizados no processo de atracação de embarcação com condições que exercitaram os limites possíveis das intempéries climáticas e demais condições de atracação, conforme apresentado na Seção 4.2. Informações de contexto, inferências utilizando ontologia, integração com agentes externos e mudanças de contexto em tempo de execução foram praticados nos testes com o sistema multiagente.

No estudo realizado no contexto de um processo de atracação de embarcação o `AAdapt` apresentou resultados qualitativamente satisfatórios para o problema apresentado e atendeu as expectativas de resolução das exceções identificadas. Mostrou que o problema pode ser resolvido em tempo de execução e que as premissas teóricas e tecnológicas foram confirmadas.

Algumas limitações da proposta foram percebidas, o `AAdapt` pode ser utilizado em outros contextos, mas para isso deve ser modelada uma nova ontologia para o domínio do processo desejado e realizar ajustes no código dos agentes para inferências, axiomas e informações de contexto. Há também a necessidade do processo de negócio ser modelado e configurado no PAIS para permitir a integração com o Java e tomada de decisões em *gateways* do processo.

Essas limitações, que podem ser tratadas em trabalhos futuros, não inviabilizaram

os objetivos dessa pesquisa pois a arquitetura proposta foi capaz de adaptar processo de negócio intensivo em conhecimento em tempo de execução de forma a diminuir os riscos de falhas na execução da instância do processo.

5 TRABALHOS RELACIONADOS

Esse capítulo apresenta um resumo dos trabalhos relacionados e um descritivo das diferenças da proposta com os trabalhos analisados. Os trabalhos são apresentados em ordem cronológica da data de publicação.

Xia e Wei (2008) apresentaram uma solução com decisões baseadas em informações de contexto. A proposição desenhou caminhos alternativos na fase de *design* do processo com as tomadas de decisões nos *gateways*. Na proposta quando uma tarefa falha, uma tarefa alternativa deve ser executada para continuar a execução. O `AAdapt` também utiliza a estratégia de caminhos alternativos no *design* e tomada de decisões nos *gateway* mas amplia os mecanismos de tomada de decisões ao incorporar agentes inteligentes com crenças e desejos, inferência com ontologia de domínio do processo. Ademais, o `AAdapt` não aguarda a ocorrência de um erro para a execução de uma ação, soluções de contorno são acionadas antes da ocorrência de um erro.

Bernal et al. (2010) utilizam uma forma de representar o processo de negócio dinâmico em termos de regras com base na identificação de padrões, ECA (*Event Condition Action*). Para representar as dependências entre as atividades Bernal et al. (2010) criaram uma tabela de verificação de fluxo de controle para mapear o conjunto de regras do ECA que definem as transições entre as atividades. A proposta do *AAdapt* não utiliza tabelas para controlar as dependência mas possui uma função similar nos agentes de BDI. Isso porque os agentes do `AAdapt` possuem *triggers* que disparam quando uma *belief* ou *desire* é violada. Além disso o `AAdapt` utiliza ontologia de domínio do processo para inferências e informações de contexto para tratar um processo intensivo em conhecimento.

Lu et al. (2010) propõem um *middleware* para sistemas RESTfulBP que implementa algoritmos de decisão autônomos voltados para o negócio que decidem quais fragmentos de processo executar em um sistema RESTfulBP. No trabalho de Lu et al. (2010) não é identificada utilização de informações de contexto e nem uso de inteligência artificial e eles apresentam a execução de um subprocesso. A arquitetura proposta no *AAdapt* utiliza uma estratégia diferente implementando as decisões baseado numa estrutura imperativa de processos, utilizando multiagentes e tomando as decisões em *gateways* com a utilização de informações de contexto. Outro ponto é que o *AAdapt* é uma implementação que acompanha um processo de negócio do inicio ao fim.

Tretola e Zimeo (2010) apresentaram uma arquitetura baseada em regras ECA ondes as ações autonômicas são realizadas acionando regras de ECA armazenadas em uma base de conhecimento permitindo mudanças dinâmicas do processo em execução. No `AAdapt` a inteligência está nos agentes com BDI para a tomada de decisões em tempo de execução. O `AAdapt` também trabalha com processo intensivo em conhecimento e ontologia de domínio do processo.

Hu, Wu e Chen (2013) apresentam uma abordagem orientada a regras para gerar processos de negócios dinamicamente de acordo com o contexto atual no tempo de execução. Os autores conseguem a adaptação dinâmica estabelecendo regras em resposta a eventos para gerar a solução com um mecanismo de mapeamento. Dois algoritmos foram projetados para gerar a sequência de atividades de adaptação do processo em tempo de execução. As decisões do `AAdapt` utilizam *belief* e *desire* do BDI do sistema multiagente e mudam os caminhos da instância do processos nos pontos de decisão (*gateways*) estabelecidos no mapeamento dos processos. Além disso, o `AAdapt` trata processo intensivo em conhecimento e faz inferência utilizando ontologia de domínio do processo.

Beest et al. (2014) propõem uma abordagem para automatizar a geração de processos de intervenção em tempo de execução, usando técnicas de planejamento de IA independentes de domínio. Esse trabalho foi aplicado para acompanhar situações cobertas pela Lei Holandesa de Apoio Social, conhecida como a lei da WMO (*Dutch Law for Societal Support*). A lei da WMO visa permitir que pessoas com doenças crônicas ou com deficiência participem da sociedade e vivam em suas próprias casas pelo maior tempo possível. A semântica do trabalho permite que cada operação de serviço seja representada como uma ação de planejamento, refletindo seu comportamento funcional em termos de pré-condições e efeitos, necessários para possibilitar a geração automática de processos de intervenção. O `AAdapt` utiliza outra estratégia, os *belief* e os *desire* substituem a funcionalidades do planejador. Ademais, o `AAdapt` utiliza um sistema multiagente e ontologia do domínio do processo para tratar a a adaptação em tempo de execução do processo intensivo em conhecimento.

Kahl et al. (2015) propuseram o ADIGE que permite adaptação semiautomática de modelos de processos de negócios para mudanças de disponibilidade, desempenho e relevância de serviços para modelos de processos de negócios em tempo de design e de execução. O trabalho de Kahl et al. (2015) utiliza agentes, informações de contexto, planejamento das tarefas com PDDL e ontologia na implementação. Dois pontos que diferenciam o `AAdapt` do ADIGE é que o `AAdapt` utiliza agentes inteligentes com BDI e lida com processo intensivo em conhecimento(KiP).

Hajmoosaei et al. (2015) apresentam uma análise de impacto da mudança de processo em tempo de execução. Utilizam o *log* do sistema para extrair informações sobre as instâncias de processo, gerando uma *Process Dependency Graph*(PDG). Quando uma mudança acontece, a PDG fornece a base dentro e entre as instâncias e deriva as entidades afetadas. Não há uma referência direta a informações de contexto nesse trabalho, mas pode inferi-las pela análise do *log* para obtenção das informações da instância do processo. O `AAdapt` também utiliza um *log* do sistema que são as informações de contexto e as dependências são mapeadas e implementadas na modelagem do processo. No `AAdapt` não há uma análise de impacto das mudanças, ele utiliza agentes com BDI e ontologia para as tomadas de decisões na execução do processo intensivo em conhecimento.

Awadid e Gnannouchi (2015) apresentam uma solução onde um novo serviço da web é implementado para cada alteração necessária, implementando como web *services* as formas como as atividades são executadas e que podem sofrer alterações. A adaptação dessa proposta só ocorre na fase de implementação de forma a adaptar para outras empresas e outros cenários. Não há uma adaptação em tempo de execução da instância do processo. É uma proposta interessante do ponto de vista de utilizar web *services* mas bem diferente do *AAdapt* que utiliza mecanismos diferentes como sistema multiagente inteligente com BDI com informações de contexto para analisar e sugerir adaptações na instância do processo durante a sua execução.

Oberhauser (2016) propõe um estilo imperativo de processo, baseado em agentes e com uma abordagem de modelagem de processo que especifica a ordem e formas alternativas de execução para um fluxo de trabalho de maneira explícita por meio do uso de construções gráficas. A proposta utiliza contexto e integra com sistemas PAIS. Essa abordagem em estilo imperativo, estratégia de agentes e integração com PAIS também é utilizada no `AAdapt` que adiciona a essa estratégia os mecanismos de carregamento de informações de contexto e inferência por ontologia, BDI na tomada de decisões e interação com ser humano por se tratar de um mecanismo que permite processos intensivos em conhecimento.

Bucchiarone et al. (2017) propuseram uma estrutura de composição baseada em planejamento de IA que apresenta requisitos de composição abstratos e consciência de contexto. Na proposição de Bucchiarone et al. (2017) várias tarefas dependentes de humanos são mapeadas em tempo de *design* e algumas intervenções humanas podem ser necessárias em tempo de execução. Nessa proposta há uma similaridade nos exemplos de aplicação com o `AAdapt`, de uma embarcação. O `AAdapt` se difere desse trabalho por utilizar multiagentes com crenças e desejos e realizar inferência com a ontologia do domínio do processo.

Marrella, Mecella e Sardina (2017) propõem uso de técnicas e *frameworks* bem estabelecidos de Inteligência Artificial, como cálculo de situação, IndiGolog e planejamento clássico. A proposição dá suporte para adaptação em tempo de execução de processos, KiP, planejamento clássico de IA e IndiGolog na especificação da estrutura e fluxo de controle de KiPs. Um trabalho bem interessante e com especificação bem próxima do `AAdapt`, diferenciando de forma geral apenas por não utilizar BDI e inferências com ontologia de domínio do processo.

Nunes et al. (2018a) apresentaram uma adaptação de processos por contexto, *GCAdapt*, utilizando quatro módulos para lidar com elementos de contexto e propiciar caminhos alternativos no processo. O trabalho se utilizou de *Planning Domain Definition Language* (PDDL) para replanejar os processos e foi a base para a proposta do AAdapt.

Seiger et al. (2019) apresentaram uma estrutura para fluxos de trabalho autoadaptativos em *cyber-physical system* (CPS) com base no *loop* de *feedback* MAPE-K

(*Monitor-Analyze-Plan-Execute over a shared Knowledge*). Na proposta o monitoramento e a análise do sensor adicional e os dados de contexto são usados para verificar se há erros imprevistos no mundo físico. O `AAdapt` se difere desse trabalho por utilizar multiagentes com crenças e desejos.

Fakhfakh (2019) apresenta um modelo que visa garantir a exatidão dos processos com consciência do tempo ao considerar as operações de mudança usando o método *Event*-B que suporta um desenvolvimento incremental baseado na técnica de refinamento. O *Event*-B consiste em dois elementos: contexto e máquina. O contexto visa descrever a parte estática do sistema, a máquina *Event*-B representa a parte reativa do sistema e é composta pelas cláusulas VARIABLES que contém a declaração de variáveis, INVARIANTS para definir as propriedades do sistema e EVENT que define o comportamento dinâmico. A proposta utiliza decisões baseadas em informações de contexto e foi aplicada a um processo de embarque de pedido. Embora ambas as propostas utilizem as informações de contexto, a proposta de Fakhfakh (2019) utiliza para descrever a parte estática do sistema, já o `AAdapt` utiliza contexto para as definições estáticas e dinâmicas obtidas durante a sua execução para auxiliar na tomada de de decisões. Além do mais, o `AAdapt` se diferencia por usar sistema multiagente inteligente com crenças e desejos, ontologia de domínio do processo de negócio, integração com PAIS e tratar um processo intensivo em conhecimento.

Patiniotakis et al. (2019) propõem uma abordagem orientada a objetivos que acopla uma estrutura orientada a eventos para detectar e raciocinar em situações que apresentam a necessidade de adaptações de processo para selecionar adaptações. A proposta permite adaptação de processos em tempo de execução com uma extensão orientada a aspectos de um motor BPMN 2.0 utilizando fragmentos de processos limitados à seleção de adaptações de um conjunto pré-definido de alternativas e em adaptações corretivas em vez de otimizadas. Utiliza a técnica de análise de múltiplos critérios de decisão (*Multi-criteria decision making*). A proposta apresentada por Patiniotakis et al. (2019) se assemelha ao `AAdapt` por ser direcionada a objetivos, similar com o que ocorre com sistemas multiagentes com BDI. No entanto o `AAdapt` utiliza sistema multiagente inteligente, trata processo intensivo em conhecimento e utiliza ontologia de domínio do processo para realizar inferências e obter informações de contexto.

Zhang e Liu (2020) apresentaram um aprendizado de máquina (*off-line*) que permite que dado o rastreamento de um caso em execução e sua sequência de dados de contexto correspondente, a estrutura retornará uma previsão do caso em execução, ou seja, se o caso em execução pode alcançar o objetivo de negócios no futuro. Trata-se de um mecanismo para prever se uma instância do processo vai alcançar o êxito na execução. No experimento de avaliação da estrutura preditiva de Zhang e Liu (2020) a predição focou duas de três perguntas de pesquisa em dados de contexto. O `AAdapt` não utiliza um mecanismo de predição de processo, ele trabalha com as informações de contexto durante

o processo de execução da instância utilizando agentes inteligentes e as informações de contexto para a tomada de decisões. Em uma análise mais aprofundada e com a realização de testes, esse mecanismo de predição apresentado por Zhang e Liu (2020) poderia auxiliar os mecanismos de tomada de decisões do `AAdapt`.

Andrews, Steinau e Reichert (2021) apresentaram um conjunto de conceitos, algoritmos e detalhes de implementação que permitem alterações *ad-hoc* para processos com reconhecimento de objeto em tempo de execução, apresentando um mecanismo de processo baseado em microsserviço. O `AAdapt` difere um pouco porque implementa um conjunto de conceitos como multiagente, agentes inteligentes BDI, decisões por informações de contexto e uso de ontologias.

Kir e Erdogan (2021) propõem um sistema intensivo em conhecimento e tecnologias de agentes inteligentes gerando uma estrutura de gerenciamento de processos de negócios inteligentes. Eles propõem a busca por processo compatível num repositório do processo ou, caso não encontrado, a utilização de um planejador externo para compor o processo. A solução apresentada propõe que as perspectivas do ambiente e do processo devem ser capturadas de forma holística, que deve manter o encapsulamento e a componentização e que as execuções de processos não devem ser gerenciadas em termos de fluxos de tarefas que alteram o espaço de informações da empresa, mas sim em termos de entidades que interagem, cada uma com seus próprios objetivos, conhecimento, decisões e ciclo de vida. O trabalho apresentado por Kir e Erdogan (2021) possui várias semelhanças com o `AAdapt`, ficando pequenas diferenças no modo de implementação e integração com o PAIS.

A maioria dos trabalhos utilizou como mecanismo auxiliar informações de contexto do ambiente em que o processo estava executando. De forma geral foi utilizada uma combinação de técnicas cujo principal mecanismo utilizado para identificar e realizar adaptação é apresentado na tabela 11. O `AAdapt` utilizou informações de contexto que foi uma técnica utilizada por quase todos os trabalhos analisados, ontologia que também foi utilizada por (KIR; ERDOGAN, 2021) e (KAHL et al., 2015), sistema multiagentes ((KIR; ERDOGAN, 2021), (KAHL et al., 2015) e (OBERHAUSER, 2016)) com crenças, desejos e intenções (BDI) como mecanismo para detecção e adaptação de processos intensivos em conhecimento (também presente em (KIR; ERDOGAN, 2021)). O `AAdapt` permitiu a integração e funcionamento com um sistema PAIS comercial, atividade igualmente identificada nos trabalhos de (BEEST et al., 2014), (PATINIOTAKIS et al., 2019) e (MARRELLA; MECELLA; SARDINA, 2017).

Uma última observação que pode ser detectada nos dados apresentados na tabela 11 é que as técnicas de utilização de fragmentos de processos e ECA são técnicas que aparentemente não estão sendo mais utilizadas no seu modo padrão, visto que não há trabalhos recentes utilizando elas puramente, porém, os conceitos dessas técnicas são utilizados indiretamente por outras técnicas, visto que eventos e condições são um mecanismo para detectar situações que requerem uma ação que muitas vezes contemplam

Tabela 11 - Técnica principal utilizada para adaptação dos processos.

Técnica	**Trabalhos**
Adaptação na implementação	Awadid e Gnannouchi (2015)
Análise de impacto da mudança	Hajmoosaei et al. (2015)
Análise de múltiplos critérios de decisão	Patiniotakis et al. (2019)
ECA	Bernal et al. (2010)
	Tretola e Zimeo (2010)
	Hu, Wu e Chen (2013)
Event-B	Fakhfakh (2019)
Fragmentos de processos	Xia e Wei (2008)
	Lu et al. (2010)
Heurísticas de planejamento	Oberhauser (2016)
MAPE-K	Seiger et al. (2019)
Microsserviço - detecta alteração pelo log	Andrews, Steinau e Reichert (2021)
Planejador inteligente	Beest et al. (2014)
	Kahl et al. (2015)
	Marrella, Mecella e Sardina (2017)
	Bucchiarone et al. (2017)
	Nunes et al. (2018a)
	Kir e Erdogan (2021)
Predição de caso de execução	Zhang e Liu (2020)

Legenda: Técnica principal utilizada.

Fonte: 'O autor.'.

tarefas adicionais (fragmentos) ou desvios na instância do processo. Ademais, também se percebe que a técnica com planejadores inteligentes, além de possuir o maior número de trabalhos relacionados, é uma técnica que ainda é explorada em trabalhos acadêmicos.

CONSIDERAÇÕES FINAIS

A arquitetura proposta no `AAdapt` apresentou um sistema multiagente para adaptar processos intensivos em conhecimento em tempo de execução para mitigar riscos de falhas na execução da instância do processo. A arquitetura apresentada foi capaz de resolver o problema do processo de atracação de embarcação integrando com uma ontologia de domínio do processo, permitindo caminhos alternativos, interagindo com conhecimento de recurso humano externo e tomando decisões baseadas em informações de contexto. O conhecimento intensivo no processo, as informações de contexto e o mecanismo de BDI do *AAdapt* permitiu aos seus agentes a integração com a *engine* do PAIS Camunda para analisar em tempo de execução os processos, identificar possíveis riscos ao processo e propor caminhos alternativos para que o PAIS pudesse concluir a execução da instância do processo de forma segura.

A principal contribuição que a proposta trouxe para a área de gerenciamento de processos intensivos em conhecimento foi demonstrar que é possível utilizar sistemas multiagentes com decisões baseadas em crenças, desejos e intenções, integrando e fazendo inferências com ontologia de domínio do processo e utilizar informações de contexto para identificar exceções durante a execução de uma instância do processo. Essa arquitetura permitiu tratar essas exceções ao integrar os agentes do `AAdapt` com o PAIS para adaptar as tarefas da instância do processo em tempo de execução. As crenças, desejos e intenções dos agentes, a integração e inferência da ontologia de domínio do processo se comportaram conforme definido na literatura e foi possível demonstrar que é possível utilizar a arquitetura proposta para adaptar processos intensivos em conhecimento em tempo de execução.

A última contribuição foi a integração de um arcabouço de tecnologias e metodologias que envolveu *Knowledge-Intensive Process*, sistema PAIS Camunda Modeller e Camunda Platform, Java, Jadex, OWL, Protégé, O-Mase, AgentTollIII e a ferramentas auxiliares de teste Python e Insomnia.

A metodologia utilizada no `AAdapt` foi suficiente para realizar a implementação, os testes, validar as premissas e alcançar o objetivo geral propondo uma arquitetura multiagente capaz de adaptar processos de negócio em tempo de execução de forma a diminuir os riscos de falhas na execução da instância do processo. Os objetivos específicos também foram alcançados conseguindo definir e implementar os agentes participantes da arquitetura, especificar o conhecimento sobre o processo por meio de ontologias e definir a forma de interação dos agentes com um sistema PAIS, conforme detalhado nos testes do Capítulo 4.

Algumas limitações foram identificadas durante essa pesquisa, dentre elas a necessidade de ampliar os mecanismos de decisão do BDI para aprender com as decisões

de agentes humanos externos, a necessidade de modificação do código para integração de novos domínios de processo e a adição de interfaces mais amigáveis.

Como trabalho futuro pode ser explorada a utilização de um mecanismo de *machine learning* integrado com a ontologia do domínio do processo para a tomada de decisões em *gateways* do processo e a integração com dispositivos portáteis como *smart watch* e *smart phone* para o recurso humano tomar a decisão.

REFERÊNCIAS

AALST, Wil M. P. van der. *Process-Aware Information Systems: Lessons to Be Learned from Process Mining*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. 1–26 p. ISBN 978-3-642-00899-3. Disponível em: <https://doi.org/10.1007/978-3-642-00899-3_1>.

AALST, W.M.P. van der; ROSA, M. La; SANTORO, F.M. Business process management. *Business and Information Systems Engineering*, v. 58, p. 1–6, 2016.

ANDREWS, Kevin; STEINAU, Sebastian; REICHERT, Manfred. Enabling runtime flexibility in data-centric and data-driven process execution engines. *Information Systems*, Elsevier Ltd, v. 101, 11 2021. ISSN 03064379. Microservice-based process engine.

APPIAN. *Business Process Management (BPM Suite)*. [S.l.], 2021. Disponível em: <https://appian.com/platform/complete-automation/business-process-management-bpm.html>. Acesso em: Outubro, 2021.

AWADID, Afef; GNANNOUCHI, Sonia Ayachi. Approach based on web services for business process adaptation. In: . [S.l.]: Elsevier B.V., 2015. v. 64, p. 832–837. ISSN 18770509.

BEEST, N. R.T.P. Van et al. Automated runtime repair of business processes. *Information Systems*, v. 39, p. 45–79, 2014. ISSN 03064379. The Planning Domain consists of the actions(with their preconditions and effects) that are available in an application domain, along with possible extra domain-specificconstraints that restrict the ways in which they can be applied (e.g. that actiona1should always take place beforea2, or thatthe outcome ofa1should be assigned to the input ofa2, etc).
.

BERNAL, Jose et al. Dynamic context-aware business process: A rule-based approach supported by pattern identification. In: . [S.l.: s.n.], 2010. p. 470–474.

BIZAGI. *Uma plataforma. Todos processos.* [S.l.], 2021. Disponível em: <https://www.bizagi.com/pt>. Acesso em: Outubro, 2021.

BOISSIER, Fabrice; RYCHKOVA, Irina; GRAND, Bénédicte Le. Challenges in knowledge intensive process management. In: *2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW)*. [S.l.: s.n.], 2019. p. 65–74.

BRANDER, S. et al. Refining process models through the analysis of informal work practice. *9th International Conference on Business Process Management*, v. 6896, p. 116–131, 2011.

BRASIL, MARINHA DO. *Escala Beaufort*. [S.l.], 2021. Disponível em: <https://www.marinha.mil.br/chm/sites/www.marinha.mil.br.chm/files/u2035/escala_beaufort.pdf>. Acesso em: Agosto, 2021.

BRATMAN, Michael. Intention and means-end reasoning. *The Philosophical Review*, [Duke University Press, Philosophical Review], v. 90, n. 2, p. 252–265, 1981. ISSN 00318108, 15581470. Disponível em: <http://www.jstor.org/stable/2184441>.

_____. Two faces of intention. *The Philosophical Review*, [Duke University Press, Philosophical Review], v. 93, n. 3, p. 375–405, 1984. ISSN 00318108, 15581470. Disponível em: <http://www.jstor.org/stable/2184542>.

_____. *Faces of Intention: Selected Essays on Intention and Agency*. [S.l.]: Cambridge University Press, 1999. (Cambridge Studies in Philosophy).

BRAUNNAGEL, Daniel et al. Bpm adoption in small and medium-sized companies in bavaria. *European Conference on Information Systems (ECIS)*, Europe, September 2016.

BUCCHIARONE, Antonio et al. A context-aware framework for dynamic composition of process fragments in the internet of services. *Journal of Internet Services and Applications*, Springer London, v. 8, 12 2017. ISSN 18690238. As composable components we useprocessfragments(or simply fragments). Fragments [9] are a wayto represent reusable process knowledge in service com-positions and encode elementary subprocesses that can beused as constructing blocks for more complex processes. Context-aware dynamic service composition. composition problem is solved through AI planning techniques.

CAMUNDA. *Automate Any Process, Anywhere.* [S.l.], 2021. Disponível em: <https://camunda.com/>. Acesso em: Outubro, 2021.

CHALUPSKY, Hans et al. An overview of kqml: A knowledge query and manipulation language. 07 1993.

CHOI, Injun et al. Ipm-epdl: an xml-based executable process definition language. *Computers in Industry*, v. 56, n. 1, p. 85–104, 2005. ISSN 0166-3615. Disponível em: <https://www.sciencedirect.com/science/article/pii/S016636150400079X>.

CICCIO, C. Di; MARRELLA, A.; RUSSO, A. Knowledge-intensive processes: Characteristics, requirements and analysis of contemporary approaches. *Journal of Data Semantics*, 2014.

COMPANY, Oracle. *Seu Guia Completo para o ERP Moderno.* [S.l.], 2021. Disponível em: <https://www.oracle.com/br/erp/complete-guide-to-modern-erp/>. Acesso em: Outubro, 2021.

de Matos, Everton et al. Context information sharing for the internet of things: A survey. *Computer Networks*, v. 166, p. 106988, 2020. ISSN 1389-1286. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128619310400>.

DUMAS, M. et al. *Fundamentals of Business Process Management.* 2. ed. Berlim, Germany: Springer, 2018. ISBN 9783662565087.

ERTEL, Wolfgang. *Introduction to Artificial Intelligence.* 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2011. ISBN 9780857292988.

FAKHFAKH, Fairouz. A refinement-based approach for verifying dynamic changes on time-aware processes. In: . [S.l.]: Elsevier B.V., 2019. v. 159, p. 1489–1498. ISSN 18770509. We propose a formal model which guarantees the correctness of time-aware processes while considering change operations. We used the Event-B method which supports an incremental developmentbased on the refinement technique.

FININ, Tim et al. Kqml - a language and protocol for knowledge and information exchange. 1994.

FOUNDATION, YAWL. *YAWL BPM Powerful, concise, extensible*. [S.l.], 2021. Disponível em: <https://yawlfoundation.github.io/>. Acesso em: Novembro, 2021.

GARCíA-OJEDA, Juan; DELOACH, Scott. *The O-MaSE Methodology*. [S.l.: s.n.], 2014. 253-285 p. ISBN 978-3-642-39974-9.

GONCALVES, Gedir. *Execução do AADAPT em condições climáticas extremas*. 2022. <https://drive.google.com/file/d/1WzdWuuAiqWep4bLZSgBwMm_Gqsa4aKVf/view?usp=sharing>.

GUARINO, Nicola. *Ontologies and knowledge bases: towards a terminological clarification*. [S.l.: s.n.], 1995. 25-32. p.

GUARINO, Nicola; OBERLE, Daniel; STAAB, Steffen. *What Is an Ontology?* Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. 1–17 p. ISBN 978-3-540-92673-3. Disponível em: <https://doi.org/10.1007/978-3-540-92673-3_0>.

HAGEN, C. R.; RATZ, D.; POVALEJ, R. Towards self-organizing knowledge intensive processes. *Journal of Universal Knowledge Management*, v. 0, p. 148–169, 2005.

HAJMOOSAEI, Mojtaba et al. Impact analysis of process change at run-time. In: . [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2015. p. 156–161. ISBN 9781467376921. Análise para deduzir os elementos de processo afetados e avaliar o impacto da mudança usando métricas quantitativas estendidas do fluxo de trabalho Qualidade de Serviço (QoS).

HINDRIKS, Koen V. The shaping of the agent-oriented mindset. In: DALPIAZ, Fabiano; DIX, Jürgen; RIEMSDIJK, M. Birna van (Ed.). *Engineering Multi-Agent Systems*. Cham: Springer International Publishing, 2014. p. 1–14. ISBN 978-3-319-14484-9.

HORROCKS, Ian et al. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML - W3C Member Submission 21 May 2004*. 2004. Disponível em: <https://www.w3.org/Submission/SWRL/#1>.

HU, Guangchang; WU, Budan; CHEN, Junliang. Lncs 8377 - dynamic adaptation of business process based on context changes: A rule-oriented approach. *LNCS*, v. 8377, p. 492–504, 2013. Algortimo In response to these events and new requirements, Rule Engine generates the corresponding activity sequence according to various rules which describe the relationships between states of resources and activities. Rules are stored in the Rule Base, and activities are stored in the Service Base. Regras associadas a eventos que acionam tarefas.

HULER, Scott. *Defining the Wind: The Beaufort Scale and How a 19th-Century Admiral*. 1. ed. United States: Broadway Books, 2005. ISBN 1400048850.

IBM. *IBM Content Manager*. [S.l.], 2021. Disponível em: <https://www.ibm.com/products/content-manager>. Acesso em: Novembro, 2021.

_____. *Introduction to IBM Records Manager*. [S.l.], 2021. Disponível em: <https://www.ibm.com/docs/en/rmfz/8.5.0?topic=introduction-records-manager>. Acesso em: Novembro, 2021.

ILAHI, Latifa; GHANNOUCHI, Sonia Ayachi. Improving telemedicine processes via bpm. *Procedia Technology*, v. 9, p. 1209–1216, 2013. ISSN 2212-0173. CENTERIS 2013 - Conference on ENTERprise Information Systems / ProjMAN 2013 - International Conference on Project MANagement/ HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2212017313002892>.

JAGGA, Ankit; JUNEJA, Dimple; SINGH, Aarti. Updates in knowledge query manipulation language for complex multiagent systems. *International Journal of Computing Academic Research*, MEACSE Publications, v. 4, n. 1, p. 19–26, 2015. ISSN 2305-9184. Disponível em: <http://www.meacse.org/ijcar>.

JAMES, Katy; RANDALL, Nicola; HADDAWAY, Neal. A methodology for systematic mapping in environmental sciences. *Environmental Evidence*, v. 5, p. 7, 04 2016.

KAHL, Gerrit et al. Adige: Semantic business process management for smart retail environments. In: . [S.l.]: Association for Computing Machinery, Inc, 2015. ISBN 9781450334914. The ADIGE system facilitates the semi-automated adapta-tion of business process models to changes of the availability,performance, and relevance of services for business processmodels at design time and at process service execution time. Utiliza Montologia e PDDL. Processo de varejo. Agente Jadex.

KALIBATIENE, D; VASILECAS, O. A survey on modeling dynamic business processes. *PeerJ Computer Science 7*, e609, 2021.

KARAGIANNIS, Dimitris. Bpms: Business process management systems. *SIGOIS Bull.*, Association for Computing Machinery, New York, NY, USA, v. 16, n. 1, p. 10–13, ago. 1995. ISSN 0894-0819. Disponível em: <https://doi.org/10.1145/209891.209894>.

KIR, Huseyin; ERDOGAN, Nadia. A knowledge-intensive adaptive business process management framework. *Information Systems*, v. 95, p. 101639, 2021. ISSN 0306-4379. Disponível em: <https://www.sciencedirect.com/science/article/pii/S030643792030106X>.

KITAMURA, Yoshinobu; MIZOGUCHI, Riichiro. Ontology-based systematization of functional knowledge. *Journal of Engineering Design*, Taylor & Francis, v. 15, n. 4, p. 327–351, 2004. Disponível em: <https://doi.org/10.1080/09544820410001697163>.

LEE, Haein; PARK, Hyejin; KIM, Jinwoo. Why do people share their context information on social network services? a qualitative study and an experimental study on users' behavior of balancing perceived benefit and risk. *International Journal of Human-Computer Studies*, v. 71, n. 9, p. 862–877, 2013. ISSN 1071-5819. Social Networks and Ubiquitous Interactions. Disponível em: <https://www.sciencedirect.com/science/article/pii/S107158191300044X>.

LIN, C. et al. Logical inference of horn clauses in petri net models. *IEEE Transactions on Knowledge and Data Engineering*, v. 5, n. 3, p. 416–425, 1993.

LIZáN, Francisco José Mora; RIZO-MAESTRE, Carlos. *Intelligent Buildings: Foundation for Intelligent Physical Agents*. 2017.

LU, Qinghua et al. Integration of restfulbp with bdim decision making. In: *Middleware '10 Posters and Demos Track*. New York, NY, USA: Association for Computing Machinery, 2010. (Middleware Posters '10). ISBN 9781450306010. Disponível em: <https://doi.org/10.1145/1930028.1930030>.

LUCK, Michael; PADGHAM, Lin. *Agent-Oriented Software Engineering VIII: 8th International Workshop, AOSE 2007, Honolulu, HI, USA, May 14, 2007, Revised Selected Papers*. [S.l.: s.n.], 2008. v. 4951. ISBN 978-3-540-79487-5.

MALLE, Bertram F.; MOSES, Louis J.; BALDWIN, Dare A. *Intentions and intentionality : foundations of social cognition*. Cambridge, Mass: MIT Press, 2001. ISBN 0262133865.

MARRELLA, Andrea; MECELLA, Massimo; SARDINA, Sebastian. Intelligent process adaptation in the smartpm system. *ACM Transactions on Intelligent Systems and Technology*, Association for Computing Machinery, v. 8, 11 2017. ISSN 21576912. Uso de técnicas e frameworks bem estabelecidos de Inteligência Artificial, como cálculo de situação, IndiGolog e planejamento clássico. SmartPM, um modelo e um protótipo de Sistema de Gerenciamento de Processos que apresenta um conjunto de técnicas que fornecem suporte para a adaptação automatizada de processos intensivos em conhecimento em tempo de execução. Planejamento clássico. customizaram o IndiGolog
Usa Indigolog - IndiGolog is a high-level programming language for robots and intelligent agents that supports on-line planning and plan execution in dynamic and incompletely known environments. Usa o Indigolog para programar e controlar o fluxo de kips

.

MICROSOFT. *Microsoft Dynamics 365 Agilidade sem limites*. [S.l.], 2021. Disponível em: <https://dynamics.microsoft.com/>. Acesso em: Outubro, 2021.

MONK, Ellen; WAGNER, Bret. *Concepts in Enterprise Resource Planning*. 4th. ed. Boston, MA, USA: Course Technology Press, 2012. ISBN 1111820392.

NORLING, Emma. Folk psychology for human modelling: extending the bdi paradigm. In: . [S.l.: s.n.], 2004. p. 202– 209. ISBN 1-58113-864-4.

NUNES, Vanessa T. *Dynamic Process Adaptation: Planning In A Context-Aware Approach*. 211 p. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, Engenharia de Sistemas e Computação - COPPE, Rio de Janeiro, 2014.

NUNES, V. T. et al. A context-aware replanning approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems.*, v. 48, p. 99–118, 2018.

NUNES, Vanessa T. et al. Real-time process adaptation: A context-aware replanning approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, v. 48, n. 1, p. 99–118, 2018.

OBERHAUSER, Roy. A hypermedia-driven approach for adapting processes via adaptation processes. In: . [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2016. p. 73–80. ISBN 9781467398374. Os padrões fundamentais de fragmento de

processo de inserção, exclusão e movimentação foram implementados com REST com os parâmetros gerais esperados necessários para um PAIS passado como parâmetros JSON, e o padrão de substituição foi realizado como um subprocesso que usa os padrões de inserção e exclusão.

O'CONNOR, Martin et al. Supporting rule system interoperability on the semantic web with swrl. In: GIL, Yolanda et al. (Ed.). *The Semantic Web – ISWC 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 974–986. ISBN 978-3-540-32082-1.

OLIVEIRA, A. B. F. de; WERNECK, V. M. B. Ontologias. *Cadernos do IME.*, IME - Instituto de Matemática e Estatística / UERJ, v. 15, dezembro 2003.

OPENTEXT. *The Future of Business*. [S.l.], 2021. Disponível em: <https://www.opentext.com.br/>. Acesso em: Novembro, 2021.

ORACLE. *Cloud Platform for Digital Business*. [S.l.], 2021. Disponível em: <https://www.oracle.com/middleware/>. Acesso em: Outubro, 2021.

_____. *Oracle Universal Records Management*. [S.l.], 2021. Disponível em: <https://docs.oracle.com/cd/E14571_01/doc.1111/e10724/c04_urm.htm#UCMOV163>. Acesso em: Novembro, 2021.

PATINIOTAKIS, Ioannis et al. Assessing flexibility in event-driven process adaptation. *Information Systems*, Elsevier Ltd, v. 81, p. 201–219, 3 2019. ISSN 03064379. In execution of process adaptations, we detect situations with Complex Event Patterns which can be implemented with technologies that can handle big-data and streaming data
Nosso trabalho difere, pois considera situações mais genéricas do que Qualidade de Serviço ao usar BPMN2.0 em vez de BPEL.

PEOPLESOFT, Oracle. *PeopleSoft*. [S.l.], 2021. Disponível em: <https://www.oracle.com/applications/peoplesoft/>. Acesso em: Outubro, 2021.

PEREIRA, J. L.; FREITAS, A. P. Towards a characterization of bpm tools simulation support: the case of bpmn process models. *International Journal for Quality Research*, 2019.

PILLAT, Raquel M.; SANTOS, Renata M. S.; OLIVEIRA, Toacy C. Systematic literature review on bpmn-based process adaptation approaches. In: *Proceedings of the XV Brazilian Symposium on Information Systems*. New York, NY, USA: Association for Computing Machinery, 2019. (SBSI'19). ISBN 9781450372374. Disponível em: <https://doi.org/10.1145/3330204.3330242>.

PIPEDRIVE. *Pipedrive gets your sales organized*. [S.l.], 2021. Disponível em: <https://www.pipedrive.com/>. Acesso em: Novembro, 2021.

PIPEFY. *Crie workflows automatizados em minutos*. [S.l.], 2021. Disponível em: <https://www.pipefy.com/>. Acesso em: Novembro, 2021.

POKAHR, Alexander et al. Programming bdi agents with pure java. In: MÜLLER, Jörg P.; WEYRICH, Michael; BAZZAN, Ana L. C. (Ed.). *Multiagent System Technologies*. Cham: Springer International Publishing, 2014. p. 216–233. ISBN 978-3-319-11584-9.

POKAHR, Alexander; BRAUBACH, Lars; LAMERSDORF, Winfried. A goal deliberation strategy for bdi agent systems. In: EYMANN, Torsten et al. (Ed.). *Multiagent System Technologies*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 82–93. ISBN 978-3-540-28741-4.

POSLAD, Stefan. Specifying protocols for multi-agent systems interaction. *ACM Trans. Auton. Adapt. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 2, n. 4, p. 15–es, nov. 2007. ISSN 1556-4665. Disponível em: <https://doi.org/10.1145/1293731.1293735>.

PROCESSMAKER. *Design even the most complex workflows easily with ProcessMaker.* [S.l.], 2021. Disponível em: <https://www.processmaker.com/>. Acesso em: Outubro, 2021.

RAO, Anand S; GEORGE, Michael P. Bdi agents: From theory to practice. In: *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95).* [s.n.], 1995. p. 312–319. Disponível em: <http://www.agent.ai/doc/upload/200302/rao95.pdf>.

RECKER, Jan. *Scientific Research in Information Systems*. Springer Berlin Heidelberg, 2021. Disponível em: <https://doi.org/10.1007/978-3-030-85436-2>.

REICHERT, Manfred; RINDERLE-MA, Stefanie; DADAM, Peter. *Flexibility in Process-Aware Information Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. 115–135 p. ISBN 978-3-642-00899-3. Disponível em: <https://doi.org/10.1007/978-3-642-00899-3_7>.

RUSSELL, Stuart; NORVIG, Peter. *Artificial Intelligence: A Modern Approach*. 4th. ed. USA: Prentice Hall Press, 2020. ISBN 0134610997.

SALHOFER, Peter; STADLHOFER, Bernd; TRETTER, Gerald. Ontology driven e-government. *In Politics, Democracy and E-Government: Participation and Service Delivery*, IGI Global, p. 383–401, 2010. ISSN 1615209336. Disponível em: <http://doi:10.4018/978-1-61520-933-0.ch023>.

SANCHEZ-LOPEZ, Yanet; CEREZO, Eva. Designing emotional bdi agents: good practices and open questions. *The Knowledge Engineering Review*, Cambridge University Press, v. 34, p. e26, 2019.

SANTORO, F.M.; BAIAO, F.A. Knowledge-intensive process: A research framework. *Teniente E., Weidlich M. (eds) Business Process Management Workshops*, Springer, v. 308, jan. 2018.

SAP. *Tenha controle sobre seus negócios com o SAP Business One e concorra com os grandes*. [S.l.], 2021. Disponível em: <https://www.sap.com/brazil/cmp/dg/br-sap-business-one-brasil/index.html?campaigncode=CRM-BR21-PPC-GBUBUSN&gclid=Cj0KCQjwiNSLBhCPARIsAKNS4_fB4n8TXpx2GPqvLVhVxKZA4tRlcrFKPBchwlu3mPQFXpAoyrsTCl4aAmgoEALw_wcB&gclsrc=aw.ds>. Acesso em: Outubro, 2021.

SCHWERZ, André Luis et al. Robust and reliable process-aware information systems. *IEEE Transactions on Services Computing*, v. 14, n. 3, p. 820–833, 2021.

SEIGER, Ronny et al. Toward a framework for self-adaptive workflows in cyber-physical systems. *Software and Systems Modeling*, v. 18, 04 2019.

SHEHORY, Onn; STURM, Arnon (Ed.). *Agent-Oriented Software Engineering: Reflections on Architectures, Methodologies, Languages, and Frameworks*. Berlin: Springer, 2014. ISBN 978-3-642-54431-6.

S.L., Open Document Management System. *Document Management System*. [S.l.], 2021. Disponível em: <https://www.openkm.com/>. Acesso em: Novembro, 2021.

SZELAGOWSKI, M. Evolution of the bpm lifecycle. *Communication Papers of the Federated Conference on Computer Science and Information Systems*, v. 17, p. 205–211, 2018.

TER, Arthur et al. *Modern Business Process Automation: YAWL and Its Support Environment*. [S.l.: s.n.], 2010.

TOTVS. *O sistema de gestão ERP que cresce junto com a sua empresa*. [S.l.], 2021. Disponível em: <https://www.totvs.com/>. Acesso em: Outubro, 2021.

TRETOLA, Giancarlo; ZIMEO, Eugenio. Autonomic internet-scale workflows. In: *Proceedings of the 3rd International Workshop on Monitoring, Adaptation and Beyond*. New York, NY, USA: Association for Computing Machinery, 2010. (MONA '10), p. 48–56. ISBN 9781450304221. Disponível em: <https://doi.org/10.1145/1929566.1929573>.

TURING. *Inteligência artificial*. [S.l.], 1950. Disponível em: <https://pt.wikipedia.org/wiki/Intelig%C3%AAncia_artificial>.

UNGER, M.; LEOPOLD, H.; MENDLING, J. How much flexibility is good for knowledge intensive business processes: A study of the effects of informal work practices. *HICSS 2015*, p. 4990–4999, 2015.

VACULIN, R. et al. Declarative business artifact centric modeling of decision and knowledge intensive business processes. *The Fifteenth IEEE International Enterprise Computing Conference*, p. 151–16, 2011.

VIRIYASITAVAT, W.; XU, L. Da; BI, Z. et al. Blockchain-based business process management (bpm) framework for service composition in industry 4.0. *Journal of Intelligent Manufacturing*, v. 31, p. 1737–1748, 2020.

WATERS, Max; SARDINA, Sebastian. Improving domain-independent intention selection in bdi systems. *Autonomous Agents and Multi-Agent Systems*, Springer, v. 29, p. 683–717, 2015. ISSN 1573-7454. Disponível em: <https://doi.org/10.1007/s10458-015-9293-5>.

WOOLDRIDGE, Michael. *An Introduction to MultiAgent Systems*. 2nd. ed. [S.l.]: Wiley Publishing, 2009. ISBN 0470519460.

WOOLDRIDGE, Michael; JENNINGS, Nicholas R. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, Cambridge University Press, v. 10, n. 2, p. 115–152, 1995.

XIA, Yonglin; WEI, Jun. Context-driven business process adaptation for ad hoc changes. In: . [S.l.: s.n.], 2008. p. 53–60. ISBN 9780769533957.

ZHANG, Junbao; LIU, Guohua. Prediction of incompliance with business goals with business-related data and context data. *IEEE Access*, Institute of Electrical and Electronics Engineers Inc., v. 8, p. 187008–187020, 2020. ISSN 21693536. Framework Off-line training and runtime prediction - baseado em contexto. Machine Learning com Scikit. Implementaram o processo de negócios na plataforma Java JDK 1.8 e Java expression language (JEXL 3.1). Usaram um processo empresarial para ilustrar como os dados de contexto afetam as previsões e quão eficaz é o quadro de monitoramento do processo preditivo proposto.