UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
Programa de Pós-Graduação em Ciência da Computação

RENAN BERGAMIN STUCHI

MAPEAMENTO DE ONTOLOGIAS EMPRESARIAIS PARA MODELOS DE PROCESSOS DE NEGÓCIO EM BPMN, COM APLICAÇÃO EM PROCESSOS DE SOFTWARE

UNESP
2015

RENAN BERGAMIN STUCHI

MAPEAMENTO DE ONTOLOGIAS EMPRESARIAIS PARA MODELOS DE PROCESSOS DE NEGÓCIO EM BPMN, COM APLICAÇÃO EM PROCESSOS DE SOFTWARE

Dissertação elaborada junto ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", com área de concentração em Computação Aplicada, como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação.

Orientadora: Profa. Dra. Hilda Carvalho de Oliveira

UNESP

2015

Stuchi, Renan Bergamin.
Mapeamento de ontologias empresariais para modelos de processos de negócio em BPMN, com aplicação em processos de software / Renan Bergamin Stuchi. -- São José do Rio Preto, 2015
184 f. : il., tabs.

Orientador: Hilda Carvalho de Oliveira
Dissertação (mestrado) – Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas

1. Computação. 2. Engenharia de software. 3. Ontologias (Recuperação da informação) 4. Modelagem de processos. 5. Negócios - Processamento de dados. I. Oliveira, Hilda Carvalho de. II. Universidade Estadual Paulista "Júlio de Mesquita Filho". Instituto de Biociências, Letras e Ciências Exatas. III. Título.

CDU – 518.721

Ficha catalográfica elaborada pela Biblioteca do IBILCE
UNESP - Câmpus de São José do Rio Preto

RENAN BERGAMIN STUCHI

MAPEAMENTO DE ONTOLOGIAS EMPRESARIAIS PARA MODELOS DE PROCESSOS DE NEGÓCIO EM BPMN, COM APLICAÇÃO EM PROCESSOS DE SOFTWARE

Dissertação elaborada junto ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", com área de concentração em Computação Aplicada, como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação.

Banca Examinadora

Profa. Dra. Hilda Carvalho de Oliveira
UNESP – Rio Claro
Orientador

Prof. Dr. Kechi Hirama
USP – São Paulo

Profa. Dra. Kelly Rosa Braghetto
USP – São Paulo

UNESP
2015

AGRADECIMENTOS

Primeiramente, agradeço a Deus pela saúde e força, pois sem isso esta jornada não seria cumprida.

À minha orientadora, Profa. Hilda Carvalho Oliveira, por ter confiado em mim para a realização deste trabalho e pelo imensurável suporte dedicado ao longo desse tempo.

Ao colega de mestrado Alessandro Viola Pizzoleto, pelo grande apoio durante as atividades de pesquisa, inclusive pela contribuição do seu trabalho de Mestrado, que foi utilizado para o estudo de caso nesta dissertação.

Aos meus pais, Luis e Roseli, por nunca terem medido esforços na minha educação. Dedico cada uma de minhas conquistas aos dois.

À minha namorada, Camila, por toda paciência e apoio nos momentos mais difíceis.

“*Sometimes life hits you in the head with a brick. Don´t lose Faith*.”

(Steve Jobs)

RESUMO

O principal objetivo deste trabalho é apresentar uma metodologia para mapear ontologias empresariais em modelos de processos de negócio, com notação gráfica BPMN (*Business Process Model and Notation*). A intenção é obter uma estrutura gráfica que facilite o entendimento da ontologia, incluindo as relações entre os processos e a rede de colaboração entre os atores envolvidos. O processo de mapeamento proposto é parcialmente automatizado e parte do princípio que a ontologia possa ser convertida para a estrutura RDF 1.1 (*Resource Description Framework*). Foi, então, definido um sistema de marcação para a ontologia em RDF, que permite o mapeamento automático para a linguagem XPDL 2.2 (*XML Process Definition Language*), através de uma ferramenta de software (*parser*), denominada *RDFtoXPDL*. Para o desenvolvimento dessa ferramenta, foi utilizada a linguagem SPARQL, para consultas no arquivo RDF devidamente marcado. O arquivo resultante em XPDL pode ser importado por sistemas de modelagem de processos, de modo a permitir a visualização e uso dos diagramas em BPMN. O trabalho apresenta uma aplicação da metodologia proposta para uma ontologia empresarial do Modelo de Referência MPS para Software (MR-MPS-SW), considerando os níveis G e F do modelo. A representação em BPMN do MR-MPS-SW pode apoiar a implantação desse modelo em organizações de desenvolvimento de software, incentivando a adoção futura da abordagem BPM (*Business Process Management*).

Palavras-chave: Ontologia. Ontologia empresarial. Modelagem de processos. BPMN. SPARQL. MPS-SW.

ABSTRACT

The main goal of this work is to present a methodology to map enterprise ontologies in graphical models of business processes, with the graphical notation Business Process Model and Notation (BPMN). The intention is to obtain a graphical structure that contributes to the ontology understanding, including the relation between processes and the collaboration network among the actors involved. The mapping process proposed is partly automated and is based on the principle that the ontology is able to be converted into a Resource Description Framework - RDF - 1.1 structure. So, it was defined a tagging process to the ontology in RDF, which allows the automatic mapping to XPDL 2.2 language (XML Process Definition Language) through a software tool (parser), denominated RDFtoXPDL. For the development of this tool, it was used the SPARQL language to query data in the tagged RDF file. The resulted file in XPDL can be imported by modeling process systems, in a way that allows the viewing and the use of the diagrams in BPMN. This work also presents an application of the proposed methodology to an enterprise ontology of the MPS Reference Model for Software (RM-MPS-SW), considering the levels G and F. The representation in BPMN of the RM-MPS-SW can support the implementation of this model in organizations of software development, stimulating the adoption of the future Business Process Management (BPM) approaching.

***Keywords**: Ontology. Enterprise ontology. Process modeling. BPMN. SPARQL. MPS-SW.*

LISTA DE FIGURAS

Página

LISTA DE TABELAS

LISTA DE SIGLAS

ADS	Ambiente de Desenvolvimento de Software
AP	Atributos de Processo
API	*Application Programming Interface*
BAM	*Business Activity Monitoring*
BID	Banco Interamericano de Desenvolvimento
BPA	*Business Process Analysis*
BPAL	*Business Process Abstraction Language*
BPD	*Business Process Diagram*
BPEL	*Business Process Execution Language*
BPM	*Business Process Management*
BPMI	*The Business Process Management Initiative*
BPMN	*Business Process Model and Notation*
BPMS	*Business Process Management System*
BSC	*Balanced Scorecard*
CMMI	*Capability Maturity Model Integration*
CMMI-DEV	*Capability Maturity Model Integration for Development*
CMMI-SVC	*Capability Maturity Model Integration for Services*
DAML	*DARPA Agent Markup Language*
EPC	*Event-Driven Process Chain*
FINEP	Agência Financiadora de Estudos e Projetos
IA	Instituições Avaliadoras
IDEF	*Integration DEFinition*
II	Instituições Implementadoras
MA-MPS	Método de Avaliação – Melhoria de Processo de Software
MCTI	Ministério da Ciência e Tecnologia e Inovação
MN-MPS	Modelo de Negócio – Melhoria de Processo de Software
mPME	Micro, Pequenas e Médias Empresas
MPS.BR	Melhoria de Processo do Software Brasileiro
MPS-SV	Melhoria de Processo de Serviço
MPS-SW	Melhoria de Processo de Software
MR-MPS-SV	Modelo de Referência para Serviço

MR-MPS-SW	Modelo de Referência para Software
N/n	Não necessário
OIL	*Ontology Interchange Language*
OMG	*Object Management Group*
OPAL	*Object, Process, Actor Modeling Language*
OWL	*Ontology Web Language*
OWL-S	*Ontology Web Language Service*
PMBOK	*Project Management Body of Knowledge*
PNML	*Petri Net Markup Language*
RAP	Resultados de Atributos de Processo
RDF	*Resource Description Framework*
RDFS	*Resource Description Framework Schema*
SBPM	*Semantic Business Process Management*
SEBRAE	Serviço Brasileiro de Apoio às Micro e Pequenas Empresas
SEI	*Software Engineering Institute*
SHOE	*Simple HTML Ontology Extensions*
SOFTEX	Associação para Promoção da Excelência do Software Brasileiro
SQL	*Structured Query Language*
SPARQL	SPARQL Protocol and RDF Query Language
TI	Tecnologias da Informação
UML	*Unified Modeling Language*
URI	*Uniform Resource Identifiers*
W3C	*World Wide Web Consortium*
WFM	*WorkFlow Management*
WfMC	*WorkFlow Management Coalition*
WfMS	*WorkFlow Management System*
WPDL	*Workflow Process Definition Language*
WSML	*Web Service Modeling Language*
XMI	*XML Metadata Interchange*
XML	*eXtensible Markup Language*
XOL	*XML-based Ontology Exchange Language*
XSD	*XML Schema Definition*
XP	*eXtreme Programming*
XPDL	*XML Process Definition Language*

SUMÁRIO

Página

1 INTRODUÇÃO

Considerando a dinâmica e competitividade do mercado, as organizações de desenvolvimento de software vêm buscando mecanismos que tragam melhorias ao negócio e auxiliem nas tomadas de decisão. Segundo Davenport (1993), a chave para melhorias no desempenho dos negócios está na estrutura dos processos. Os negócios não devem ser vistos em função de componentes separados (papéis, produtos, divisões, entre outros), mas sim da cadeia de processos, que corta transversalmente vários departamentos. Essa cadeia de processos deve focar o cliente final e os clientes internos dos processos. Essa ideia vai de encontro ao conceito de processo de negócio, que Verdanat (1996) define como sendo uma sequência lógica de atividades a serem executadas dentro de organizações para entrega de resultados aos clientes. Contudo, focar em processos de negócio exige esforços adicionais para as empresas que, culturalmente, vêm trabalhando com uma visão funcional, setorizada por departamentos.

Contudo, o interesse em estruturar melhor os processos vem crescendo no universo das empresas de desenvolvimento de software. Isso tem levado as organizações a pensarem mais em investimentos estratégicos na gestão de seus processos (McCormack et al., 2009). De fato, a estruturação e autoconhecimento dos processos e atividades ajudam a organização na otimização dos processos e na obtenção de resultados com mais qualidade. É importante que as organizações também busquem atender normas de qualidade, como a ISO 9000, assim como certificações em modelos de qualidade de processos, como CMMI (*Capability Maturity Model Integration*) e Modelo de Referência MPS para software (MR-MPS-SW). Isso agrega valor ao processo e ao produto, mas o processo de implantação desses modelos na cultura das organizações é complexo e requerem muitos investimentos financeiros e de pessoal. Essa complexidade atinge fortemente as micros, pequenas e médias empresas (mPME). Contudo, essas empresas também devem buscar a certificação de qualidade de seus processos, frente aos ganhos internos e no negócio.

Um passo importante antes de se pensar em implantar um modelo de qualidade de processos, é conhecer como os processos têm sido realizados até o momento. Nessa direção, Uschold et al. (1997) introduziu a ideia de ontologias empresariais, de modo a organizar, manipular e representar o conhecimento das organizações. De fato, as ontologias propõem uma forma organizada e não ambígua de representação do domínio empresarial. Elas organizam as informações associadas aos processos, contribuindo para a representação dos relacionamentos entre diferentes elementos envolvidos. As ontologias empresariais usam termos padronizados (vocabulário comum) para a representação dos conceitos e informações da organização, evidenciando as interações entre os conceitos. Na literatura, é possível encontrar alguns trabalhos que destacam o uso de ontologias empresariais, como, por exemplo, Ternai e Torok (2011) e Cerqueira (2007).

Nessa direção, visando auxiliar as empresas de desenvolvimento de software, principalmente as mPME, Pizzoleto (2013) propôs uma ontologia empresarial baseada nos guias textuais do modelo MPS-SW, do Programa de Melhoria de Processo de Software Brasileiro (MPS.BR). Essa ontologia agregou conhecimentos de especialistas no modelo, bem como indicadores de desempenho baseados na técnica *Balanced Scorecard* (BSC). A intenção foi auxiliar o entendimento dos guias do modelo, contribuindo para a modelagem dos processos e informações associadas. Consequentemente, essa forma de representação dos guias MPS-SW contribui para apoiar a redução de custos do processo de implantação. A ontologia de Pizzoleto (2013), contudo, abrange somente os dois primeiros níveis de certificação (maturidade) do MR-MPS-SW: níveis G e F.

Segundo Uschold et al. (1997), todos os conceitos presentes em ontologias empresariais devem ser analisados em conjunto com seus relacionamentos. Isso significa que os conceitos não devem ser analisados de maneira isolada. De modo geral, para Uschold et al. (1997), um dos principais objetivos das ontologias empresariais é estruturar e organizar toda a gama de conhecimento da organização.

Considerando a área de processos de negócio, vale observar que Gruninger e Pinto (1995) já propunham a representação de processos e atividades da organização por meio de ontologias antes de Uschold et al. (1997). A proposta dos autores já era organizar todos os processos, visando a execução das atividades necessárias da maneira mais ágil possível.

Devido à grande riqueza semântica que uma ontologia pode conter, autores como Haller, Gaaloul e Marmolowski (2008) sugerem a criação de ontologias a partir

de modelos de processos, formalizados por meio da linguagem XPDL (*XML Process Definition Language*). Aslam et al. (2006), por outro lado, propõem um mapeamento de modelos de processos, representados em BPEL (*Business Process Execution Language*), para ontologias em OWL-S (*Ontology Web Language Service*). A intenção é inserir um número maior de informações no modelo de processos gerado.

Apesar de fornecerem um meio alternativo para a organização de informações e resolverem problemas de entendimento e compartilhamento de conhecimento, as ontologias possuem um formato que dificulta a visualização do domínio como um todo. Quando se passa a considerar as interligações entre os diferentes conceitos na ontologia, a visualização pode se tornar ainda mais complexa. Assim, Mutton e Golbeck (2003) apresentam uma maneira alternativa de visualização dos conceitos presentes na ontologia, utilizando conceitos de grafos. Stradiotto et al. (2006) também mencionam a relevância de uso de ferramentas para a visualização gráfica das informações contidas nas ontologias. Esses autores também propõem uma visualização por meio de grafos. Eles defendem que a análise e verificação das informações contidas nas ontologias acabam se tornando um trabalho árduo, devido ao formato visual apresentado.

Frente ao exposto, este trabalho foi direcionado a encontrar uma forma visual alternativa para representar as informações contidas nas ontologias empresariais. Pesquisas mostraram que as notações gráficas usadas em modelagem de processos de negócio facilitam o entendimento dos processos, evidenciam determinados conceitos, atividades e relacionamentos, além de propiciarem condições para gestão colaborativa. Alguns exemplos de linguagens gráficas para modelagem de processos de negócio são: UML (*Unified Modeling Language)*, EPC (*Event-Driven Process Chain)*, fluxograma, BPMN (*Business Process Model and Notation*), entre outras. É importante ressaltar que, para este trabalho, a forma visual gráfica do modelo de processos de negócio é considerada como uma forma de apoio à visualização das ontologias.

Além da parte visual, a modelagem de processos é um importante recurso utilizado na abordagem de Gerenciamento de Processos de Negócio, conhecida como BPM (*Business Process Management*). Essa abordagem tem sido muito utilizada já algumas décadas em empresas de grande porte e que não são da área de Engenharia de Software. Contudo, a tecnologia de BPM poderia agregar muito valor à área de desenvolvimento de software. Van der Aalst, ter Hofstede e Weske (2003) definem BPM como um conjunto de técnicas, métodos e sistemas para projetar, controlar,

analisar e manter processos operacionais, que envolvam diferentes atores, como, por exemplo, pessoas, sistemas ou até mesmo organizações.

A modelagem de processos de negócio pode ser considerada como porta de entrada para a gestão de processos com as tecnologias de BPM, voltadas à execução, monitoramento, análise e otimização dos processos. Segundo Indulska et al. (2009), a modelagem de processos de negócio auxilia a organização a analisar seus processos, podendo eliminar ou acrescentar novos processos. Além disso, auxilia a comunicação entre os diferentes usuários envolvidos.

Smith e Fingar (2003) são exemplos de autores que explicitam que uma das primeiras etapas na implantação de BPM é a modelagem dos processos de negócio, com representação visual gráfica. O modelo de processos resultante deve representar graficamente as atividades, os eventos e o fluxo lógico do processo (RECKER et al., 2009). A modelagem de processos de negócio tem como base o conceito de modelagem de *workflow*. Segundo Georgakopoulos, Hornick e Sheth (1995), um *workflow* consiste em um conjunto de tarefas organizadas em função de um fluxo lógico, definindo a execução de processos de negócio. Nos *workflows* se leva em consideração a ordem do fluxo do processo e as condições sob as quais cada tarefa deve ser executada, de forma possibilitar a compreensão, avaliação e, se necessário, redesenho do processo.

Considerando o contexto de BPM, a linguagem de modelagem de processos BPMN, que é um padrão do consórcio OMG (*Object Management Group*), tem sido muito utilizada em abordagens empresariais e nas literaturas, devido à facilidade de entendimento de sua notação. O principal objetivo dessa notação é prover um meio compreensível e uniforme para todos os participantes do processo, usando elementos gráficos com formatos distintos e de baixa complexidade. Assim, essa linguagem foi selecionada, neste trabalho, para prover a forma gráfica alternativa para ontologias empresariais.

Nesse contexto, a *seção 1.1* apresenta os objetivos propostos neste trabalho, enquanto a *seção 1.2* apresenta a metodologia utilizada para o seu desenvolvimento. A *seção 1.3*, por sua vez, traz a organização dos demais capítulos deste trabalho.

1.1 OBJETIVOS DO TRABALHO

O principal objetivo deste trabalho é apresentar uma metodologia para mapear o conteúdo de uma ontologia empresarial para o formato de modelo de processos de negócio, de modo que sua representação gráfica auxilie a compreensão e análise dos processos e relacionamentos representados na ontologia. Mais especificamente, a metodologia constitui uma forma de representar ontologias de processos definidas na linguagem OWL através da notação BPMN 2.0.

Convém observar que as ontologias vêm ganhando destaque na representação de processos de negócio, organizando conceitos envolvidos no domínio, como mostram Aslam et al. (2006) e Haller, Gaaloul e Marmolowski (2008). Por outro lado, a necessidade de se encontrar formas gráficas de visualização dos conceitos representados em ontologias também vem atraindo a atenção de alguns autores, como Stradiotto et al. (2006) e Mutton e Golbeck (2009). Apesar da riqueza semântica, a visualização dos conceitos representados por meio de ontologias dificulta o entendimento do processo de negócio como um todo. Assim, este trabalho apresenta uma forma de mapear ontologias empresariais para uma forma gráfica de fácil visualização, através de uma cadeia de processos em BPMN.

A metodologia foi desenvolvida para ser aplicável a qualquer ontologia empresarial, representando processos e suas inter-relações. Para isso, foi desenvolvido um sistema de marcação para a ontologia, já convertida para o formato RDF (*Resource Description Framework*). Foi também desenvolvida uma ferramenta, denominada RDFtoXPDL, para automatizar a conversão do código RDF, com as devidas marcações, para a linguagem XPDL (*XML Process Definition Language*). Para o desenvolvimento dessa ferramenta, foi utilizada a linguagem SPARQL, usada para consultas de dados no código RDF. O arquivo resultante, em XPDL, pode ser importado por sistemas de modelagem de processos, de modo a permitir a visualização e uso dos diagramas em BPMN.. A **Figura 1** apresenta uma síntese da abordagem deste trabalho.

Para exemplificar a aplicação da metodologia proposta, foi considerada uma ontologia empresarial voltada a processos de desenvolvimento de software. Trata-se da ontologia criada por Pizzoleto (2013), que representa os conceitos e informações referentes aos seguintes processos dos níveis G e F do Modelo de Referência MPS-SW: Gerência de Projetos, Gerência de Requisitos, Garantia da Qualidade, Gerência de Configuração, Gerência de Portfólio de Projetos, Aquisição e Medição. O modelo

de processos em BPMN resultante dessa aplicação deve permitir que, a partir dele, possa ser criado um modelo personalizado, com base nas condições reais da empresa de desenvolvimento de software.

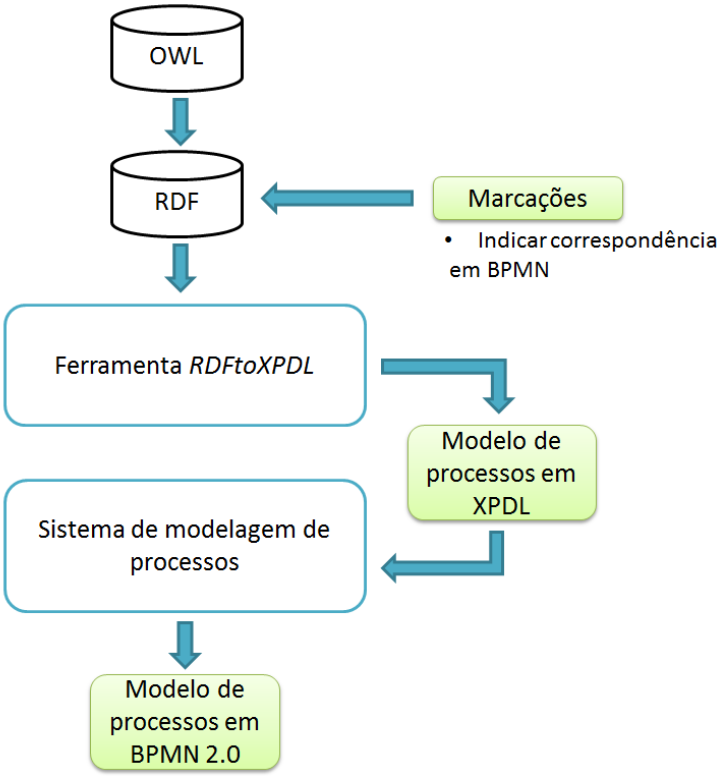


Figura 1 - Esquema representando a metodologia proposta.

1.2 METODOLOGIA DO TRABALHO

Para a realização deste trabalho, foi necessário entender o contexto geral de ontologias, com ênfase em ontologias empresariais. A maior parte dos esforços foi concentrado nos estudos sobre estruturas de ontologias. Para isso, foi realizada uma intensa investigação nas duas seguintes linguagens de representação de ontologias: OWL e RDF. Foram elaboradas algumas ontologias para testes, mas também foram utilizadas ontologias disponíveis em alguns repositórios públicos, principalmente a ontologia empresarial de Pizzoleto (2013).

Após o levantamento sobre ontologias, o próximo passo contemplou estudos sobre modelos de processos de negócio. Foi investigado como fazer modelagem de processos de negócio com BPMN 2.0, bem como avaliado vários sistemas de

interpretação de BPMN. Foram realizadas pesquisas para entender também o contexto da modelagem de processos dentro da abordagem BPM.

Através dos estudos das literaturas, foi possível identificar que certas estruturas de ontologias empresariais poderiam conter elementos que não seriam adequados ou suportados no mapeamento para um modelo de processos. Isso contribuiu para o início do desenvolvimento da metodologia proposta. Foi necessário montar um quadro de correspondência entre elementos da ontologia e elementos do modelo de processos, para o posterior mapeamento. A partir disso, foi definido um processo para marcações no código RDF/XML. Esse processo tem que ser feito manualmente, por enquanto. Essas marcações foram feitas em algumas ontologias disponíveis gratuitamente na Web, para avalição e ajustes no processo.

Foi concebida, então, uma ferramenta, denominada *RDFtoXPDL*, para automatizar a conversão do código RDF, com as devidas marcações propostas, para um respectivo modelo de processos de negócio. Para o desenvolvimento da ferramenta, foi necessário investigar técnicas que pudessem ler dados armazenados em arquivos RDF, bem como técnicas que pudessem converter os dados para um modelo de processos em BPMN 2.0. A partir desses estudos, a linguagem SPARQL foi identificada como um padrão mundialmente utilizado para a leitura de dados em arquivos RDF. De forma similar, a linguagem XPDL foi identificada como um padrão de serialização de modelos de processos. Tecnologias, ferramentas e linguagens que dessem suporte a esses padrões foram então pesquisadas, para serem utilizadas no desenvolvimento da ferramenta proposta.

Para verificar a metodologia, incluindo a técnica de marcação e a ferramenta concebida, foi realizada uma aplicação mais detalhada, considerando a ontologia empresarial criada por Pizzoleto (2013), para os níveis G e F do MR-MPS-SW. Para isso, foi necessário compreender o contexto e a organização do modelo MPS-SW, com ênfase nos níveis G e F. Foram, então, analisados os guias de documentação do modelo. Deve-se ressaltar que o apoio da ontologia ajudou a compreender melhor o conteúdo desses guias. A aplicação da metodologia mostrou, também, que o modelo de processos gerado pode ser utilizado para apoiar a implementação do modelo MPS-SW nas organizações.

1.3 ORGANIZAÇÃO DOS CAPÍTULOS

O do texto deste trabalho foi organizado em cinco capítulos, que visam apresentar os principais resultados obtidos durante a realização deste trabalho.

O **capítulo 2** apresenta uma visão geral sobre ontologias, com ênfase a ontologias voltadas ao domínio empresarial: ontologias empresariais. Adicionalmente, o capítulo traz um levantamento sobre estruturas e informações relevantes para a construção de ontologias, *framework* RDF e linguagem OWL. O capítulo aborda também a linguagem SPARQL, utilizada como um padrão para consultas a dados de arquivos RDF. Por fim, é apresentada uma visão geral da ontologia utilizada para a aplicação da metodologia deste trabalho, contextualizando também o modelo MR-MPS-SW. O capítulo traz também alguns trabalhos relacionados aos tópicos abordados.

O **capítulo 3** apresenta um levantamento sobre modelos de processos de negócio e a relevância desses modelos para o meio empresarial. O capítulo também aborda o tópico de modelagem de processos de negócio com BPMN, bem com o tópico de compartilhamento de processos modelados em BPMN, através de XPDL. Observa-se que XPDL é utilizada para serializar um modelo de processos criado em BPMN. Adicionalmente, o capítulo traz a avaliação de algumas ferramentas de modelagem de processos utilizadas neste trabalho. Uma visão geral sobre a tecnologia BPM também é abordada. Por fim, o capítulo traz alguns trabalhos relacionados aos tópicos abordados.

O **capítulo 4** apresenta a proposta de uma metodologia de mapeamento de ontologias empresariais para modelos de processos de negócio, proposta neste trabalho. O capítulo descreve todas as etapas da metodologia, bem como o desenvolvimento da ferramenta *RDFtoXPDL,* utilizada para automatizar o mapeamento para o modelo de processos. Adicionalmente, o capítulo traz informações sobre testes realizados sobre a ferramenta desenvolvida, a fim de garantir a aderência da mesma à metodologia proposta. Esses tipos de testes são apresentados no APÊNDICE E.

O **capitulo 5**, por sua vez, apresenta uma aplicação da metodologia proposta sobre uma ontologia de processos de desenvolvimento de software, que representa os conceitos e informações dos processos dos níveis G e F do modelo MPS-SW. Todos os modelos de processos gerados durante a aplicação são apresentados no APÊNDICE F.

Frente aos conteúdos abordados nos capítulos anteriores, o **capítulo 6** apresenta as considerações finais sobre o trabalho, evidenciando as contribuições deste trabalho e apresentando possíveis linhas de trabalhos futuros.

2 VISÃO GERAL SOBRE ONTOLOGIAS

Na área de Computação, ontologias começaram a ser utilizadas em Inteligência Artificial por volta da década de 90, com objetivo de organizar grandes bases de conhecimento. Desde então, as ontologias vêm sendo amplamente utilizadas como técnica de representação e reutilização do conhecimento (VAN HEIJST; VAN DER SPEK; KRUIZINGA, 1996).

Segundo Gruber (1993), uma ontologia especifica de maneira explícita uma conceitualização, descrevendo todos os conceitos e as relações existentes entre eles. Ainda, segundo o autor, o termo “conceitualização” refere-se a uma visão simplificada e abstrata do domínio que se quer representar. Segundo Noy e McGuinness (2001), uma ontologia possui uma estrutura com três elementos básicos, apresentados na **Tabela 1**: classes, relações e propriedades.

Tabela 1 – Elementos básicos que compõem uma ontologia.

Elemento	Representação
Classes	Representam um domínio, organizados em uma taxonomia.
Relações	Representam interação entre os conceitos e características de um domínio (classes e propriedades). Também podem representar restrições.
Propriedades	Descrevem características ou atributos de conceitos.

De modo geral, uma ontologia é utilizada para representar o conhecimento de um domínio, de forma que qualquer membro de uma comunidade que utiliza esse domínio possa compreender. Nesse sentido, Mizoguchi (1999) enfatiza algumas importantes características de ontologias, elencadas a seguir:

- uma ontologia expressa o consenso do conhecimento de uma comunidade de pessoas em um domínio;
- ontologias servem como referência de termos definidos de forma precisa;

- ontologias fornecem uma linguagem suficientemente expressiva para que as pessoas possam expressar aquilo que elas querem dizer;
- ontologias são estáveis;
- uma ontologia pode ser utilizada para resolver uma variedade de problemas de um domínio;
- uma ontologia pode ser utilizada como ponto de partida para a construção de múltiplas aplicações.

As estruturas das ontologias podem variar de acordo com os domínios que elas representam. Considerando o domínio empresarial, as ontologias podem representar o conhecimento relevante da empresa, baseando-se em conceitos específicos da empresa e suas relações. Segundo Öhgren e Sandkuhl (2005), ontologias empresariais são descritas como sendo blocos de construção para formação de uma rede de compartilhamento da informação nas empresas.

Atualmente, pode-se contar com algumas linguagens para a construção de ontologias. A linguagem XOL (*XML-Based Ontology Exchange Language*), por exemplo, foi criada para facilitar o compartilhamento de ontologias, com base em XML (*eXtensible Markup Language*). A linguagem DAML+OIL, por sua vez, combina as propriedades das linguagens DAML (*DARPA Agent Markup Language*) e OIL (*Ontology Interchange Language*). Essa linguagem é baseada em um tipo específico de marcação RDF e também baseada em XML. A **Figura 2** ilustra a base formal de representação de algumas linguagens utilizadas para a criação de ontologias: XML. A linguagem SHOE (*Simple HTML Ontology Extensions*), mostrada na **Figura 2**, pode ser vista como uma extensão para HTML, permitindo que autores de páginas Web anotem seus documentos na Web com o conhecimento de leitura óptica. Já a linguagem OML (*Outline Markup Language*), pode ser vista como uma linguagem de apoio à criação de textos em formato de tópicos.

De modo geral, as ontologias podem ser representadas por diferentes linguagens, sendo que uma das mais utilizadas atualmente é OWL (*Ontology Web Language*). OWL é uma linguagem utilizada na área de Web Semântica, baseada em XML, que tem facilidade de ser interpretada por máquinas. OWL é um padrão desenvolvido a partir das linguagens OIL e DAML+OIL, tendo o framework RDF (*Resource Description Framework*) como base. RDF, por sua vez, possui XML como base, sendo um dos primeiros modelos utilizados para compartilhamento de dados na

Web e que possibilita a representação da semântica formal dos conceitos.

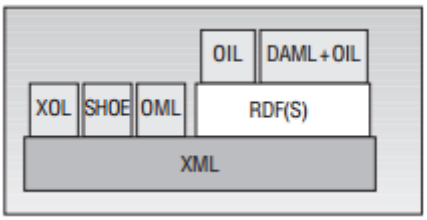


Figura 2 – XML como base formal de algumas linguagens para construção de ontologias.
Fonte: extraído de Gómez-Pérez e Corchos (2002).

Frente ao exposto, e considerando os objetivos deste trabalho, a *seção 2.1* traz uma abordagem específica sobre ontologias empresariais. A *seção 2.2*, por sua vez, apresenta o *framework* RDF, enquanto a *seção 2.3* apresenta aspectos da linguagem OWL, incluindo formatos de representação e algumas das principais estruturas da linguagem OWL. Informações complementares à *seção 2.3* serão encontradas no APÊNDICE A. Adicionalmente, a *seção 2.4* apresenta a linguagem SPARQL, um padrão utilizado para consultas de dados na Web Semântica.

A *seção 2.5* apresenta uma ontologia empresarial, que será utilizada para a aplicação da metodologia proposta neste trabalho, conforme apresentado no *capítulo 5*. Essa ontologia, definida por Pizzoleto (2013), representa o domínio dos níveis G e F do modelo de referência MPS para Software.

Alguns trabalhos encontrados nas literaturas relacionados ao contexto abordado neste capítulo serão apresentados na *seção 2.6*. Por fim, a *seção 2.7* apresenta as considerações finais sobre este capítulo.

2.1 ONTOLOGIAS EMPRESARIAIS

Primeiramente, as ontologias empresariais são específicas para representação do domínio empresarial. Segundo Uschold et al. (1997), as ontologias empresariais devem apoiar a captura, a representação e a manipulação do conhecimento da organização empresarial por meio de um conjunto consistente de conceitos básicos. As ontologias empresariais usam termos específicos para descrever a empresa como um todo, levando também em consideração os processos

aplicados na empresa (USCHOLD; KING, 1995). Assim, o modelo organizacional de uma empresa está estreitamente associado com sua ontologia empresarial.

As ontologias empresariais têm o objetivo de apoiar a interação entre humanos, sejam eles usuários ou desenvolvedores do processo, que podem, inclusive, pertencer a diferentes organizações incluídas no processo. Esse tipo de ontologia também apoia a interação entre sistemas empresariais e humanos, colaborando em diversos fatores, como, por exemplo, a integração de sistemas ou especificações de software.

Segundo Villela et al. (2005), uma ontologia empresarial propicia vários benefícios ao domínio empresarial, listados a seguir:

- fornece uma estrutura para organizar o conhecimento em uma ou mais organizações empresariais;
- permite o desenvolvimento de ferramentas genéricas baseadas em sua própria estrutura, reduzindo o esforço necessário para a construção de ambientes de desenvolvimento de software para diferentes organizações;
- promove a integração entre as ferramentas que manipulam o conhecimento relacionado à ontologia, através do compartilhamento de bancos de dados criados com base na estrutura da ontologia;
- facilita o desenvolvimento de sistemas que manipulam o conhecimento da organização, como, por exemplo, um sistema que providencia suporte a um processo organizacional;
- fornece um vocabulário comum para ser usado por todos os envolvidos em um projeto, permitindo a reutilização do conhecimento da organização para a elaboração dos requisitos;
- auxilia na identificação de profissionais com as competências adequadas para: discussão de ideias sobre um assunto, orientação sobre a execução de uma tarefa ou mesmo para montar equipes para atender às necessidades de um determinado projeto.

O processo para se definir uma ontologia empresarial, segundo Villela et al. (2005), envolve quatro etapas:

- identificação do propósito da ontologia;

- especificação de requisitos, onde as questões gerais de competência são refinadas e agrupadas em subontologias, de acordo com a similaridade de conteúdo;
- projeto da ontologia, onde conceitos, relações e restrições são exemplificadas e descritas em linguagem natural;
- formalização da ontologia, definindo constantes, predicados e axiomas.

Após a definição da ontologia, esta deve ser validada de acordo com os seus propósitos. Maiores informações sobre a criação e validação de uma ontologia empresarial podem ser encontradas em Villela el. al. (2005).

Para Öhgren e Sandkuhl (2005) o desenvolvimento de uma ontologia empresarial também consiste de quatro etapas:

1. identificação do "por que" da construção da ontologia e dos propósitos de seu uso;
2. construção da ontologia através de três passos: captura, codificação e integração. Captura é a fase onde deve ser produzido um texto contendo as definições dos principais conceitos e suas relações. Essas definições são usadas na fase de codificação, representando-as em uma linguagem formal. Na fase de integração, procura-se integrar a ontologia com outras ontologias já existentes (se existirem);
3. avaliação da ontologia para sua validação, ou seja, para ver se a ontologia atende todos os requisitos e se não possui elementos desnecessários;
4. documentação da ontologia.

Blomqvist, Öhgren e Sandkuhl (2006) mostram a fase de avaliação de uma ontologia empresarial criada pela metodologia desenvolvida por Öhgren e Sandkuhl (2005). Segundo eles, essa metodologia gera um resultado com pouca complexidade e poucos axiomas. Os autores mencionam que a abordagem completa do domínio da aplicação depende exclusivamente dos especialistas que foram consultados ou que desenvolveram a ontologia, pois eles que determinam os conceitos específicos que a empresa utiliza.

2.2 RDF: UMA ESTRUTURA PARA DESCRIÇÃO DE RECURSOS

O modelo RDF (*Resource Description Framework)* foi criado originalmente em 1999, pelo Consórcio W3C (*World Wide Web Consortium)*, como um padrão XML (*eXtensible Markup Language)* para codificação de metadados. De modo geral, RDF é um modelo para intercâmbio de dados na Web, pois suas características facilitam a fusão (*merge*) de dados, desejável na Web Semântica. Através desse modelo, pode-se, por exemplo, identificar, na Web, quem é o autor de uma página ou quando uma página foi publicada. A especificação de RDF passou por algumas atualizações desde sua criação, sendo a última atualização (versão 1.1) definida em 2014 (W3C, 2014a).

A Web permite acessos irrestritos às informações distribuídas globalmente através dela. O uso de metadados ajuda na descoberta e no acesso a tais informações. No entanto, o uso efetivo de metadados nos aplicativos requer a adoção de convenções comuns a respeito da semântica, da sintaxe e da estrutura adotada (MILLER, 1998). Essas convenções são abordadas no modelo RDF. Contudo, observa-se que RDF não estipula a semântica para se descrever um recurso, possibilitando que cada domínio defina seus próprios elementos de representação do metadado. Por outro lado, o modelo RDF permite o compartilhamento de metadados, entre sistemas ou qualquer outro mecanismo que utilize metadados, por meio de uma semântica comum.

Hitzler, Krotzsch e Rudolph (2009), bem como Furgeri (2006), definem RDF como um modelo de dados formal para a descrição de recursos. Essa descrição faz uso de termos que são definidos por meio de identificadores de recursos universais, conhecidos como URIs (*Uniform Resource Identifiers)*. De modo geral, URIs são formados por cadeias de caracteres utilizadas para identificar um recurso na Web. Segundo Ramalho (2006), os URIs possibilitam uma maneira global e única de se nomear itens. A intenção é que qualquer aplicação computacional possa trocar metadados com outra aplicação, preservando o significado original da informação.

Observa-se que todo e qualquer nome utilizado para representar um recurso em RDF deve ser único e global, de modo a facilitar o compartilhamento do conhecimento pela Web, Por isso esses nomes são representados por meio de URIs. Os URIs podem possuir a mesma sintaxe de endereço de um *website*, como, por exemplo, "*http://www.w3.org/1999/02/22-rdf-syntax-ns#type*". Considerando que um URI pode atingir um tamanho relativamente grande, é possível "abreviar" essa forma de endereçamento na notação RDF com o uso de *namespaces* de XML. Entende-se

por *namespaces* um agrupamento de conceitos que se referem sempre ao mesmo domínio. *Namespaces* foram criados para evitar que um mesmo elemento possa representar conceitos diferentes.

Segundo Allemang e Hendler (2011), RDF é o pilar da Web Semântica, pois com sua utilização é possível controlar a distribuição de dados pela Web. Um complemento importante para a estrutura RDF é chamada RDFS (*Resource Description Framework Schema*), ou simplesmente RDF Schema. RDFS é uma extensão do vocabulário básico RDF, pois fornece um vocabulário de modelagem de dados para os dados de RDF (W3C, 2014a).

Um modelo RDF deve satisfazer algumas regras básicas, listadas a seguir (TAUBERER, 2006):

- um fato é expresso por meio de uma tripla com a forma “Sujeito-Predicado-Objeto”, denominada “sentença”;
- entidades do tipo sujeito, predicado e objeto são representadas por nomes. Essas entidades podem representar qualquer informação do mundo real, como pessoas, *websites* ou relações entre objetos;
- “nomes” são URIs, que possuem o mesmo significado em todo o escopo, sempre se referindo a mesma entidade no modelo RDF;
- objetos também podem ser representados por valores literais, que podem ou não serem “tipados”, usando os tipos especificados no arquivo *XML Schema* correspondente. Um *XML Schema* é um arquivo que contém a definição da estrutura de um documento XML, as definições de tipo, tamanho, ocorrência e regras de preenchimento dos elementos que compõem o documento.

De certa forma, RDF pode ser entendido como sendo um meio genérico e flexível para “quebrar” qualquer recurso ou informação em partes menores, chamadas de “triplas”. Cada tripla representa um fato sobre o recurso descrito. As triplas são compostas por três partes: sujeito, predicado e objeto. Cada uma das partes da tripla deve ser rotulada com um identificador único (URI); apenas a parte “objeto” pode possuir um identificador URI ou um valor literal (HITZLER; KRÖTZSCH; RUDOLPH, 2009). A **Tabela 2** apresenta alguns exemplos de triplas. A primeira linha da **Tabela 2** mostra três partes de uma tripla identificadas com URIs. Já na segunda linha da tabela, a parte “objeto” é identificada através de um valor literal.

Tabela 2 - Exemplo de RDF representado em formato de triplas.

Sujeito	Predicado	Objeto
<http://example.com/my#Renan>	<http://other.org/possui>	<http://example.com/my#Carro>
<http://example.com/my#Renan>	<http://other.org/aniversario>	“08-04-1988”

Uma forma alternativa de visualização de uma estrutura RDF é um grafo orientado e rotulado. O sujeito e o objeto da tripla podem ser entendidos como sendo dois nós de um grafo, enquanto o predicado é a aresta que une esses dois nós. Tanto os nós quanto as arestas são rotulados com identificadores que os distinguem uns dos outros. A **Figura 3** mostra um exemplo simples dessa forma de visualização. O nó identificado como “*my:Renan*” possui dois objetos vinculado a ele, “08-04-1988” e “*my:Carro*”, que são interligados por meio dos predicados “*e:aniversario*” e “*e:possui*”, respectivamente.

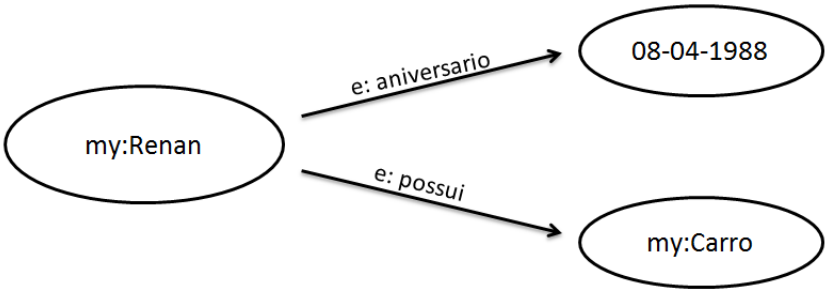


Figura 3 - Exemplo de uma estrutura RDF representada em formato de grafo.

Uma estrutura RDF representada em forma de grafo consegue expressar a mesma informação que uma estrutura RDF representada em formato de triplas. Observa-se que toda estrutura RDF pode ser completamente descrita por meio de nós e arestas, e, posteriormente, transformada em triplas, com sujeitos, predicados e objetos. De modo similar, toda estrutura representada em forma de triplas pode ser transformada em uma estrutura de grafos.

Para a representação dos conceitos da estrutura RDF, existem dois formatos mais comumente utilizados: um que faz uso da linguagem XML e outro formato conhecido como N3 ou *Turtle.*

A representação que utiliza a estrutura XML é definida pelo consórcio W3C. Para isso são definidos dois tipos de nós: os nós fonte e os nós propriedade. Os nós fonte podem representar sujeitos ou objetos, e normalmente possuem um atributo do

tipo “*rdf:about*” especificando a URI do recurso que eles representam. Além disso, podem possuir nós do tipo propriedades dentro dele referenciando um segundo nó fonte, isto é, sujeito referenciando um objeto. Na **Figura 4** pode ser identificado um nó fonte “*rdf:Description*” que se refere ao sujeito “http://example.org/bob#me” e possui quatro predicados (“rdf:type”; “schema:birthdate”;“foaf:knows”; “foaf:topic”) com seus respectivos objetos. Por outro lado, os nós do tipo propriedade podem conter valores literais, uma referência para um objeto através do atributo “*rdf:resource*” ou, ainda, conter um nó completo como objeto.

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
         xmlns:dcterms="http://purl.org/dc/terms/"
         xmlns:foaf="http://xmlns.com/foaf/0.1/"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:schema="http://schema.org/">
   <rdf:Description rdf:about="http://example.org/bob#me">
      <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
      <schema:birthDate rdf:datatype="http://www.w3.org/2001/XMLSchema#date">1990-07-04</schema:birthDate>
      <foaf:knows rdf:resource="http://example.org/alice#me"/>
      <foaf:topic_interest rdf:resource="http://www.wikidata.org/entity/Q12418"/>
   </rdf:Description>
```

Figura 4 - Exemplo de uma estrutura de RDF no formato XML.

Fonte: extraído de W3C (2014b).

A segunda forma de representação dos conceitos da estrutura RDF, conhecida como N3 (*Notation* 3), ou *Turtle* na sua versão mais recente, considera o mesmo modelo de dados e descrição apresentado na forma de XML. A diferença está no modo de ler as triplas formalizadas na estrutura RDF. A **Figura 5** mostra o mesmo exemplo apresentado na **Figura 4**, só que reescrito segundo a forma de representação *Turtle*.

Como mostrado pela **Figura 4**, as declarações são escritas na forma de uma sentença. Os sujeitos e os predicados são representados por URIs, que podem estar, entre os símbolos matemáticos sentenciais “<” e “>” ou estarem “abreviados” por meio de *namespaces*. Já os objetos são representados por URIs, mas também podem representar um valor literal.

```
BASE   <http://example.org/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX schema: <http://schema.org/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX wd: <http://www.wikidata.org/entity/>

<bob#me>
    a foaf:Person ;
    foaf:knows <alice#me> ;
    schema:birthDate "1990-07-04"^^xsd:date ;
    foaf:topic_interest wd:Q12418 .
```

Figura 5 - Exemplo de uma estrutura de RDF no formato *Turtle*.

Fonte: extraído de W3C (2014b).

Para abreviar a representação de múltiplas declarações com o mesmo sujeito, pode-se agrupar todos os predicados e objetos em um único sujeito, usando apenas um ponto e vírgula no lugar do sujeito a partir da segunda vez de utilização, como mostrado na **Figura 5**. Observa-se, na figura, que o sujeito "*<bob#me>*" é utilizado para compor todas as sentenças que fazem parte da sua estrutura, porém não é necessário repetir sua declaração em cada uma das sentenças. De modo geral, cabe ressaltar que o modelo RDF possui uma estrutura flexível, podendo aceitar definições de qualquer tipo de conhecimento (recurso).

2.3 OWL: LINGUAGEM PARA ONTOLOGIAS NA WEB

A linguagem OWL (*Web Ontology Language*) surgiu a partir das linguagens OIL e DAML+OIL e, por inferência, também possui o *framework* RDF como base, que, por sua vez, é baseado na linguagem XML. Pode-se considerar que OWL é uma extensão do vocabulário de RDF e que toda ontologia em OWL é também uma ontologia em RDF (GÓMEZ-PÉREZ; CORCHOS, 2002).

OWL é uma linguagem para definição e instanciação de ontologias Web. Os primeiros passos para sua construção começaram em 2001, mas só a partir de fevereiro de 2004 é que o Consórcio W3C (*World Wide Web Consortium*) começou a adotar a linguagem OWL como um padrão para a construção de ontologias (W3C, 2004).

Uma ontologia construída em OWL pode formalizar um domínio, através de definições de classes e suas propriedades, ou ainda formalizar indivíduos e afirmações sobre eles. Através da linguagem OWL é possível especificar fatos que

não estão explicitamente descritos na ontologia, mas são vinculados semanticamente à ontologia (W3C, 2004).

A linguagem OWL pode ser dividida em três classes de linguagens: OWL DL, OWL Lite e OWL Full. Todas as três linguagens possuem um subconjunto de elementos comum. OWL Lite possui apenas esse subconjunto de elementos, sem operações com conjuntos, como união, complementos e disjunção. Já as linguagens OWL DL e OWL Full possuem alguns elementos de extensão (W3C, 2004).

A linguagem OWL DL inclui todas as construções possíveis da linguagem OWL, bem como as restrições para o uso de cada construção. Como exemplo de restrição, pode-se mencionar que a linguagem não permite o uso de construções OWL e RDF Schema mistas. Além disso, requer a disjunção das classes, indivíduos e propriedades, isto é, uma classe não pode assumir o papel de classe e propriedade ao mesmo tempo. A existência das restrições é para assegurar a existência de um procedimento computacional lógico, que possa computar todas as conclusões definidas, em um tempo finito (W3C, 2004).

A linguagem OWL Full, por sua vez, não é considerada uma sublinguagem de OWL, mas sim a linguagem OWL completa. Possui o mesmo suporte às construções que a OWL DL, porém não tem restrições para o uso de construções OWL e *RDF Schema* mistas, bem como não exige que classes, propriedades e indivíduos sejam disjuntos.

Assim, toda ontologia OWL Lite é também uma ontologia OWL DL, que por sua vez é uma ontologia válida em OWL Full (W3C, 2004). É interessante observar que o caminho contrário não é válido: uma ontologia em OWL Full não é necessariamente uma ontologia válida em OWL DL e nem toda ontologia OWL DL é uma ontologia válida em OWL Lite. Essa ideia vale também para a base conceitual do OWL, ou seja, todas ontologias em OWL Full são também uma ontologia em RDF, porém nem todas ontologias RDF podem ser OWL Lite ou OWL DL.

Assim como RDF, OWL também possui alguns diferentes formatos para representação de seus conceitos, como, por exemplo: o formato RDF/XML, o formato Funcional e o formato Manchester. Esses três tipos de formatos serão abordados na *subseção 2.3.1.* De modo complementar, a *subseção 2.3.2* apresentará algumas das principais estruturas da linguagem OWL.

2.3.1 Formatos de representação dos conceitos de OWL

A representação dos conceitos em OWL pode ser feita de quatro modos: no formato RDF/XML, no formato *Turtle*, no formato Manchester e no formato Funcional. O formato RDF/XML segue exatamente o mesmo padrão da linguagem RDF, sendo todo estruturado em função de "nós". Já o formato Manchester segue um padrão diferente, tentando minimizar a quantidade de declarações e também facilitar o entendimento, adicionando todas as declarações de um indivíduo de maneira agrupada, ou seja, todas juntas. A **Figura 6** mostra um exemplo de um código em formato Manchester, onde o conceito "Person" é definido, juntamente com seus predicados.

```
Class: Person
 Annotations: ...
 SubClassOf: owl:Thing that hasFirstName exactly 1
                                and hasFirstName only string[minLength 1]  ,...
 SubClassOf: hasAge exactly 1 and hasAge only not NegInt,...
 SubClassOf: hasGender exactly 1 and hasGender only {female , male} ,...
 SubClassOf: hasSSN max 1, hasSSN min 1
 SubClassOf: not hates Self, ...
 EquivalentTo: g:People ,...
 DisjointWith: g:Rock , g:Mineral ,...
 DisjointUnionOf: Annotations: ... Child, Adult
 HasKey: Annotations: ... hasSSN
```

Figura 6 - Exemplo de código OWL em formato Manchester.
Fonte: extraído de W3C (2012c).

O formato Funcional, que é também o formato utilizado na especificação da linguagem OWL pelo consórcio W3C, propõe um formato voltado mais para a especificação de ontologias através de prefixos. A **Figura 7** apresenta um exemplo do formato Funcional, onde uma ontologia ("*ontology1*") é definida. Observa-se ainda na **Figura 7** que a ontologia sendo definida faz uso de uma segunda ontologia, "*ontology2*".

É interessante observar que todos os formatos apresentados nesta seção são reconhecidos pelo consórcio W3C, porém o formato RDF/XML é o formato considerado "padrão" pelo consórcio. Por isso, esse formato deve aparecer obrigatoriamente como opção de conversão em todas as ferramentas que

implementam a linguagem OWL, para possibilitar a troca entre um formato e outro (W3C, 2012b).

```
Prefix(:=<http://www.example.com/ontology1#>)
Ontology( <http://www.example.com/ontology1>
    Import( <http://www.example.com/ontology2> )
    Annotation( rdfs:label "An example" )

    SubClassOf( :Child owl:Thing )
)
```

Figura 7 - Exemplo de código OWL em formato Funcional.
Fonte: extraído de W3C (2012a).

2.3.2 Estrutura de uma ontologia OWL

OWL permite descrever classes e seus relacionamentos em qualquer aplicação ou documento Web. O primeiro passo para se construir a estrutura de uma ontologia em OWL é indicar qual o vocabulário que será utilizado. Para isso, é utilizada uma declaração que identifica todos os *namespaces* associados com a respectiva ontologia em construção (W3C, 2004). A **Figura 8** mostra como pode ser usada uma declaração dessa natureza.

```
1 <rdf:RDF
 2   xmlns     ="http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#"
 3   xmlns:vin ="http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#"
 4   xml:base  ="http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#"
 5   xmlns:food="http://www.w3.org/TR/2004/REC-owl-guide-20040210/food#"
 6   xmlns:owl ="http://www.w3.org/2002/07/owl#"
 7   xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 8   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 9   xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">
```

Figura 8 – Exemplo de declaração de *namespaces* em uma ontologia OWL.
Fonte: extraído de site W3C (2004).

É importante ressaltar alguns pontos em relação ao uso de *namespaces* em OWL, que podem ser ilustrados pela **Figura 8**. Primeiramente, é necessário indicar um *namespace* padrão para a ontologia, conforme indicado na linha 2, e um URI base para a ontologia (ver linha 4). Conforme indicado na linha 6, é necessário indicar um *namespace* para o vocabulário OWL. Como a linguagem OWL é baseada em RDF,

RDF *Schema* e XML *Schema*, também é necessário indicar um *namespace* para cada um desses vocabulários.

Observa-se que os sufixos ":rdf" e ":rdfs", que aparecem no exemplo da **Figura 8**, indicam que os termos já pertencem à estrutura do modelo RDF. Já o sufixo ":owl" indica que os termos foram introduzidos pela linguagem OWL. Ainda segundo W3C (2004), é interessante a utilização de entidades XML para uma melhor estruturação do código. Assim, torna-se possível, em todo o documento da ontologia, a substituição de declarações expandidas de classes, propriedades ou indivíduos, por uma forma mais compacta. A **Figura 9** apresenta a forma de declarar entidades em XML. Na **Tabela 3** é possível visualizar um exemplo de como a declaração de entidades podem ser utilizadas na forma compactada.

```
<!DOCTYPE rdf:RDF [
    <!ENTITY vin  "http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#" >
    <!ENTITY food "http://www.w3.org/TR/2004/REC-owl-guide-20040210/food#" > ]>
```

Figura 9 - Declaração de entidades XML em uma ontologia.

Fonte: extraído de W3C (2004).

Tabela 3- Exemplo de declarações de entidades de forma compacta.

Tipo	Forma expandida	Forma compacta
Namespace	http://www.w3.org/owl#	&owl
Classe	http://www.w3.org/owl#myClass	&owl:myClass

Após a declaração dos itens de cabeçalho da ontologia, já é possível começar a descrever a ontologia de fato. Os elementos básicos para construção de uma ontologia OWL são as classes, os indivíduos (instâncias das classes) e as propriedades (relacionamentos entre instâncias). As classes possibilitam agrupar diversos recursos com características similares, ou seja, uma classe define um grupo de indivíduos que possuem as mesmas propriedades. O conjunto de indivíduos que estão associados a uma classe é conhecido como extensão da classe. Tais indivíduos são conhecidos como instâncias da classe (BECHHOFER et al., 2004).

Cada indivíduo em uma ontologia em OWL é membro da classe "owl:Thing". Desse modo, a classe "owl:Thing" é superclasse de toda e qualquer classe definida em OWL. Além disso, existe a classe "owl:Nothing" (não possui instâncias), que é uma

subclasse de todas as classes OWL (W3C, 2004).

Uma classe deve ser representada como sendo uma instância do elemento “owl:Class”, que, por sua vez, é uma subclasse de “rdfs:Class”. É obrigatório dar um nome para a instância de uma classe, sendo que para nomear uma classe é utilizado o atributo “rdf:ID’. Segundo os autores Smith, Welty e McGuinness (2004), uma classe pode ser referenciada dentro de uma ontologia na qual ela está declarada utilizando-se apenas a expressão “#nome da classe”. A **Figura 10** apresenta exemplos de declarações de classes.

```
<owl:Class rdf:ID="Winery"/>
<owl:Class rdf:ID="Region"/>
<owl:Class rdf:ID="ConsumableThing"/>
```

Figura 10 - Declaração de estruturas de classes em uma ontologia.

Fonte: extraído de W3C (2004).

A linguagem OWL permite, ainda, descrever hierarquias entre classes. Para tanto, podemos destacar o uso da construção “rdfs:subClassOf”. Na **Figura 11** é possível identificar que a classe “PotableLiquid” é uma subclasse (um subconjunto de indivíduos da classe) “ConsumableThing”.

```
<owl:Class rdf:ID="PotableLiquid">
  <rdfs:subClassOf rdf:resource="#ConsumableThing" />
  ...
</owl:Class>
```

Figura 11 - Declaração de hierarquia de classes em uma ontologia.

Fonte: extraído de W3C (2004).

Convém observar que os elementos e recursos apresentados nesta seção são considerados básicos para o início do desenvolvimento de uma ontologia. Outras estruturas da especificação OWL são apresentadas no APÊNDICE A, observando-se que também são importantes para os propósitos deste trabalho. Exemplos dessas estruturas são: propriedades que permitem relacionar dois elementos; restrições impostas no relacionamento entre dois ou mais elementos; estruturas que possibilitam combinar duas ou mais classes através de operações boolenas; entre outras.

2.4 LINGUAGEM DE CONSULTA DE DADOS SPARQL

Como apresentado na *seção 2.2*, RDF é um modelo de representação de dados na Web Semântica. Contudo, desde seu lançamento e recomendação pelo consórcio W3C há questionamentos sobre qual seria a forma para consultar dados em um documento RDF.

Surgiram, então, diversas propostas de linguagens de consulta a um documento RDF. Em 2004, um órgão pertencente ao W3C, conhecido como Grupo DAWG (*RDF Data Access Working Group*), apresentou uma linguagem chamada SPARQL (*SPARQL Protocol and RDF Query Language)*, que foi rapidamente adotada como padrão para consultas em dados na Web Semântica. A partir de 2008, a linguagem SPARQL se tornou uma recomendação do consórcio W3C. Sua versão mais recente, versão 1.1, encontra-se disponível em W3C (2013).

Segundo Berners-Lee et al. (2006), SPARQL é uma especificação para submissão de consultas (*querys*) em bancos de dados RDF através do conceito de dados interligados (*linked data*). Os autores acrescentam que o conceito de dados interligados consiste em utilizar a Web para criar ligações (*links*) entre dados de diferentes fontes na Web, formando uma rede de dados. As fontes dos dados podem ser de qualquer tipo e de qualquer lugar do mundo, sendo que, historicamente, não precisam possuir interoperabilidade entre si. Resumindo, o conceito técnico de dados interligados refere-se aos dados publicados na Web que podem ser lidos por máquina, tenham significado explicitamente definido e sejam ligados a outros conjuntos de dados externos, que, por sua vez, podem ser ligados a outros conjuntos de dados externos, e assim por diante (BERNERS-LEE et al., 2006).

Arenas, Gutierrez e Pérez (2010) definem SPARQL como sendo uma linguagem projetada para consulta de dados em formato de triplas, sendo que o conceito da linguagem gira em torno da facilidade de retornar resultados de acordo com correspondências padrões dos dados pesquisados.

A sintaxe das consultas em SPARQL é baseada em triplas (sujeito, predicado e objeto), assim como a especificação de RDF é baseada em relacionamentos que se utilizam de triplas. As triplas da linguagem SPARQL são semelhantes às de RDF, com a diferença que em SPARQL cada parte da tripla (sujeito, predicado ou objeto) pode ser substituída por variáveis para retornar um conteúdo.

Qualquer aplicação que utilize a linguagem SPARQL pode extrair informações

complexas que poderão ser retornadas como resultado, ou, se necessário, serem utilizadas como parâmetros para uma nova consulta de forma recursiva. Segundo Berners-Lee et al. (2006), seria inviável utilizar a Web Semântica sem a linguagem SPARQL.

A estrutura sintática da linguagem SPARQL é semelhante à linguagem de consulta de banco de dados SQL (*Structured Query Language*). SPARQL possui três partes essenciais, na seguinte ordem: (1) uma parte que indica os dados a serem retornados; (2) uma parte que identifica em qual repositório os dados serão consultados; (3) uma parte que indica quais os filtros aplicados para o retorno dos dados. A **Figura 12** mostra uma representação da organização das partes sintáticas da linguagem SPARQL.

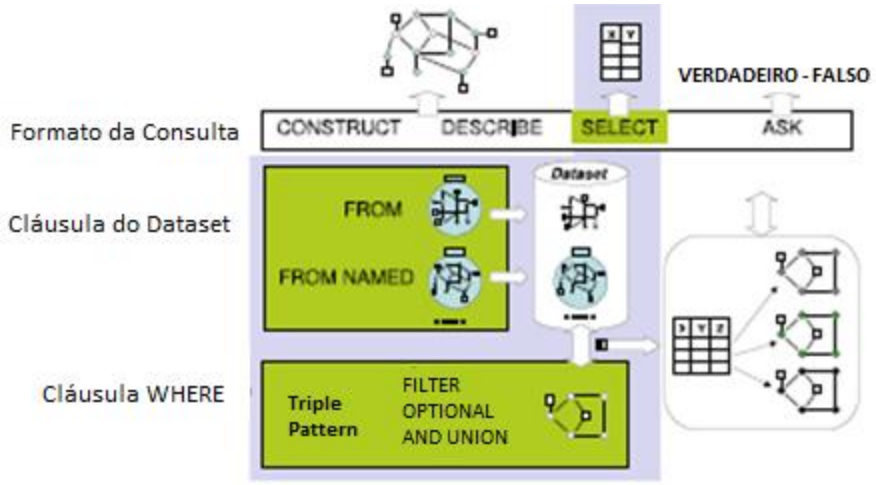


Figura 12 - Representação da organização da sintaxe do SPARQL.
Fonte: extraído e traduzido de Arenas, Gutierrez e Pérez (2010).

Assim como em SQL, a estrutura de uma consulta SPARQL deve seguir algumas regras. Em primeiro lugar, é necessário declarar prefixos para possibilitar abreviação dos URIs, assim como é feito em RDF ou OWL. Depois, é necessário identificar qual será o retorno da consulta. Isso pode ser feito de três modos: (1) através de uma tabela de dados, utilizando a cláusula “SELECT”; (2) através da construção de um novo modelo RDF com os dados retornados, utilizando uma cláusula “CONSTRUCT”; (3) através de uma verificação (“verdadeiro” ou ”falso”) se a pesquisa possui resultados, utilizando a cláusula “ASK”.

A cláusula “FROM” em uma consulta SPARKL especifica a fonte de dados a ser consultada, porém não é uma cláusula obrigatória; senão for informada, a consulta

será feita em todo documento RDF. Já a cláusula “WHERE” serve para identificar um padrão que será utilizado como filtro dos valores a serem retornados. É possível também indicar a cláusula “ORDER BY” para indicar a ordenação com que os dados deverão ser retornados. Porém, também não é uma cláusula obrigatória.

Um ponto interessante a se observar sobre o SPARQL é o tratamento dado à identificação dos termos identificados como “OPTIONAL”, que são utilizados como uma cláusula de filtro, permitindo que possam ser retornadas respostas que não se enquadrem completamente nos filtros pesquisados. Essa opção é de suma importância nas aplicações de Web Semântica, mais especificamente em RDF, onde a maioria das aplicações possui um conhecimento limitado sobre os dados a serem consultados (ARENAS; GUTIERREZ; PÉREZ, 2010). A **Figura 13** mostra um exemplo de uma consulta simples em SPARQL, com a cláusula “OPTIONAL”.

A linguagem ainda permite usar alguns outros recursos de filtragem (modificadores de resultados) com os operadores aqui mencionados. Esses elementos podem ser encontrados na especificação da linguagem em W3C (2013).

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE  { ?x foaf:name  ?name .
         OPTIONAL { ?x  foaf:mbox  ?mbox }
       }
```

Figura 13 - Exemplo de consulta em SPARQL.

Fonte: extraído de W3C (2013).

A definição de uma semântica formal para SPARQL foi fundamental na padronização dessa linguagem de consulta (ARENAS; GUTIERREZ; PÉREZ, 2010) Apesar da baixa complexidade dos recursos e sintaxe da linguagem de maneira isolada, a combinação desses elementos entre si pode tornar a semântica muito complexa. Perez, Arenas e Gutierrez (2006) apresentam uma das primeiras formalizações feitas para a semântica da linguagem. Uma formalização mais completa, contudo, pode ser encontrada em W3C (2013) para a versão atual: 1.1.

2.5 ONTOLOGIA EMPRESARIAL DOS NÍVEIS G E F DO MR-MPS-SW

Esta seção apresenta uma ontologia, definida em OWL, que será utilizada na aplicação da metodologia proposta, conforme apresentado no *capítulo 5*. Essa ontologia é voltada ao contexto de qualidade de software em ambientes de desenvolvimento de software, organizando e representando os processos dos níveis G e F do modelo de referência MPS para software (MR-MPS-SW). Uma de suas finalidades consiste em apoiar a implementação e avaliação do MR-MPS-SW em empresas de desenvolvimento de software, principalmente micro, pequenas e médias empresas (mPME). Essa ontologia foi proposta por Pizzoleto (2013) e apresentada também em Pizzoleto e Oliveira (2013), e Pizzoleto, Oliveira e Oliveira (2012). A ontologia está disponível para *download* no repositório *Protege Ontology Library*, encontrado no endereço indicado por Stanford (2015).

O MR-MPS-SW faz parte do programa de Melhoria de Processo do Software Brasileiro (MPS.BR), mantido pela Associação para Promoção da Excelência do Software brasileiro (SOFTEX), que conta com apoio das seguintes instituições: Ministério da Ciência, Tecnologia e Inovação (MCTI), Agência Financiadora de Estudos e Projetos (FINEP), Banco Interamericano de Desenvolvimento (BID) e Serviço Brasileiro de Apoio às Micro Empresas (SEBRAE). Um dos principais objetivos desse Programa é definir um modelo para melhoria de processos de software, focando preferencialmente as mPME. Esse programa possui dois modelos de referência, sendo um o MPS-SW e o outro modelo de referência para Serviços (MPS-SV), para apoiar organizações prestadoras de serviços. O APÊNDICE B traz uma breve apresentação sobre o programa MPS.BR.

O modelo MPS-SW é composto por treze guias, que descrevem os processos do modelo e as boas práticas para melhoria gradual dos processos de uma organização de produção de software. O modelo é definido através de sete níveis de maturidade, identificados pelas letras de “A” a “G”, sendo “G” o nível inicial (mais baixo) e “A” o nível mais alto. Cada nível possui processos com objetivos a serem alcançados através de uma série de resultados esperados, ou mais comumente conhecidos como RAP (Resultados de Atributos de Processo). Para conseguir a certificação de um nível, é obrigatório que todos os processos dos níveis anteriores já estejam implementados.

Pizzoleto (2013) enfatiza a importância da ontologia empresarial dos níveis G e F porque, segundo ele, os processos iniciais de adoção de modelos de maturidade organizacional são muito complexos, devido às grandes mudanças causadas na organização empresarial. Segundo o autor, o impacto é ainda maior nas mPME, devido às restrições técnicas e financeiras de tais organizações. A **Tabela 4** apresenta a identificação dos processos pertencentes aos níveis G e F do MR-MPS-SW, bem como o que se espera do processo de desenvolvimento de software quando a organização atinge o nível G ou F. Os processos dos demais níveis do modelo MPS-SW, bem como comentários adicionais são encontrados no APÊNDICE B.

Tabela 4 - Processos e complexidade de implementação dos níveis G e F.

Nível	Processos	Complexidade de implementação
F	Medição	Este nível passa a gerenciar os produtos de trabalho produzidos durante a execução do processo.
	Garantia da Qualidade	
	Gerência de Portfólio de Projetos	
	Gerência de Configuração	
	Aquisição	
G	Gerência de Requisitos	Neste nível, a execução do processo deve ser gerenciada, isto é, os planejamentos, métodos e medidas definidas para execução do processo devem ser cumpridos. Além disso, os produtos de trabalho devem ser gerenciados apropriadamente.
	Gerência de Projetos	

A apresentação dos processos nos guias do MPS-SW segue uma estrutura básica, de modo a evidenciar apenas o objetivo de cada processo e os resultados de sua execução. Vale ressaltar que as atividades e tarefas que devem ser executadas para alcançar os resultados do processo não são definidas, ficando por conta das empresas implementadoras a definição de “como” implementá-las. Como se trata de uma apresentação textual (linear), a interpretação dos processos, de suas restrições e inter-relações (interdependências) não é simples. Assim, a representação do modelo MPS-SW em forma de ontologia apoia a organização implementadora a visualizar e entender melhor os guias – o que ajuda nos custos da implantação do modelo. Para contribuir com a compreensão do modelo, Pizzoleto (2013) complementou a ontologia com conhecimentos de especialistas no modelo e conceitos encontrados no Guia de Conhecimentos em Gerência de Projetos, conhecido como PMBOK (*Project Management Body of Knowledge*), além dos indicadores inspirados na técnica *Balanced Scorecard (BSC).* Segundo o autor, a intenção foi proporcionar mais elementos comuns aos desenvolvedores de software, além de contribuir para

uniformizar o entendimento do conteúdo dos guias. Isso reforça os propósitos de apoio às empresas implementadoras, principalmente as mPME, que possuem grandes restrições técnicas e financeiras.

Inicialmente, Pizzoleto (2013) modelou a estrutura padrão de apresentação dos processos nos guias, como forma de apoio à criação da estrutura da ontologia, conforme apresentado na **Figura 14**.

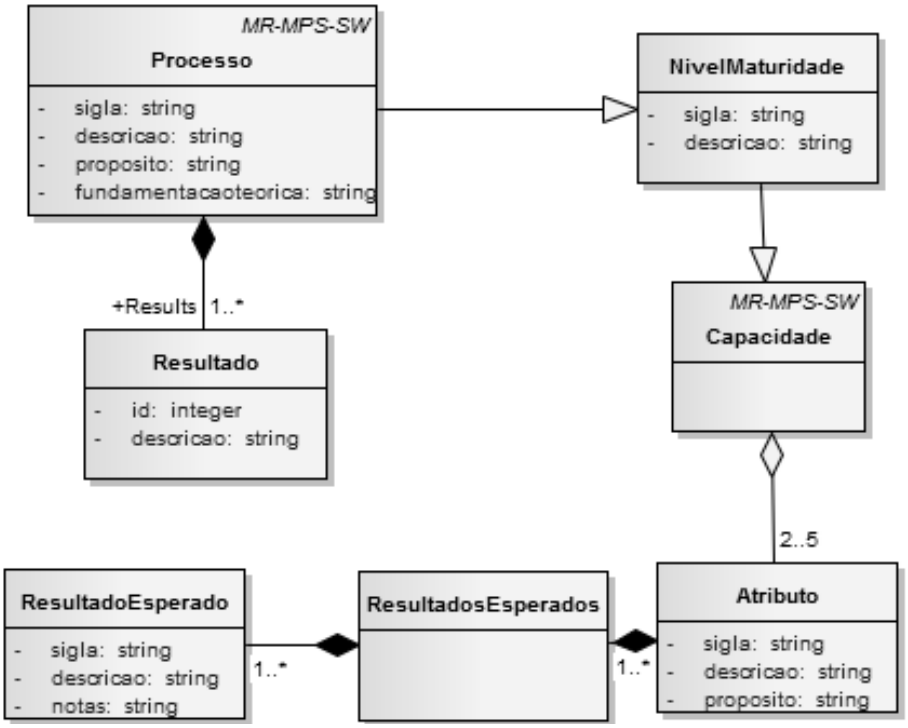


Figura 14 - Organização estrutural dos guias do MR-MPS-SW.

Fonte: extraído de Pizzoleto (2013).

Para a construção da ontologia, foram utilizadas as estruturas mais comumente utilizadas em OWL, para a representação de classes e subclasses, como por exemplo: “equivalentClass”, “subClassOf”, “intersectionOf” e propriedades de relacionamento entre essas classes (“Object Property”). As classes referem-se aos processos e conteúdos associados dos níveis G e F do MR-MPS-SW.. As subclasses expressam a hierarquia entre os conceitos do modelo. Como exemplo, considere o resultado (“Plano de Projeto”) do RAP GPR10, associado ao processo “Gerência de projetos”, do nível G. O processo “Gerência de projetos“ é implementado como uma classe, enquanto seu resultado é indicado como sua subclasse. Para relacionar uma classe com suas subclasses, são utilizadas as “propriedades de objeto” (“Object Property”). É importante ressaltar que cada um dos relacionamentos descritos na

ontologia entre as classes e subclasses representa uma interdependência entre os processos e conceitos do modelo MPS-SW.

É interessante observar que a ontologia foi construída sem definição das atividades realizadas para se chegar aos resultados dos processos. A **Figura 15** mostra como os processos, RAPs e resultados dos processos estão organizados hierarquicamente dentro da ontologia.

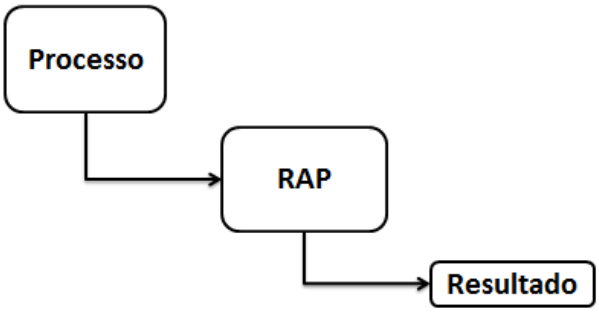


Figura 15 - Organização hierárquica dos processos, RAPs e resultados dos processos.
Fonte: extraído de Pizzoleto (2013).

Um exemplo do código e da estruturação da ontologia pode ser visto na **Figura 16**. Pode-se observar que a estrutura obedece um padrão hierárquico de apenas um nível. Isso significa que todos os processos que são representados na ontologia estão diretamente relacionados com seus resultados esperados. Observa-se que o autor não descreve etapas intermediárias ou tarefas para se chegar aos resultados do RAP, descrevendo diretamente o resultado. Um exemplo disso pode ser observado na **Figura 16**, onde o RAP “GPR11” já possui diretamente relacionado a ele o resultado “AnaliseDeViabiliadeDoProjeto”.

A **Figura 17** identifica as principais classes do modelo MPS-SW presentes na ontologia (dezoito classes). Essa figura foi extraída da interface do sistema Protégé v.4.3. É interessante observar que a classe “Sugestão” foi adicionada pelo autor da ontologia, para facilitar a implementação do modelo, mas não faz parte da sua especificação nos guias do modelo MPS-SW. Já a **Tabela 5** apresenta as principais propriedades de relacionamento entre dois elementos (“Object Property”), utilizadas por Pizzoleto (2013). Na tabela ainda é possível identificar os elementos referenciados (objetos) pelas propriedades.

```
<Class rdf:about="&Ontology1339617941597;GPR11">
    <rdfs:label xml:lang="pt">GPR 11</rdfs:label>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasresult"/>
            <someValuesFrom
             rdf:resource="&Ontology1339617941597;AnaliseDeViabilidadeDoProjeto"/>
        </Restriction>
    </equivalentClass>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;GerenciaDeProjeto"/>
</Class>
```

Figura 16 - Parte de código OWL da ontologia no modelo MPS-SW.

Fonte: extraído de Pizzoleto (2013).

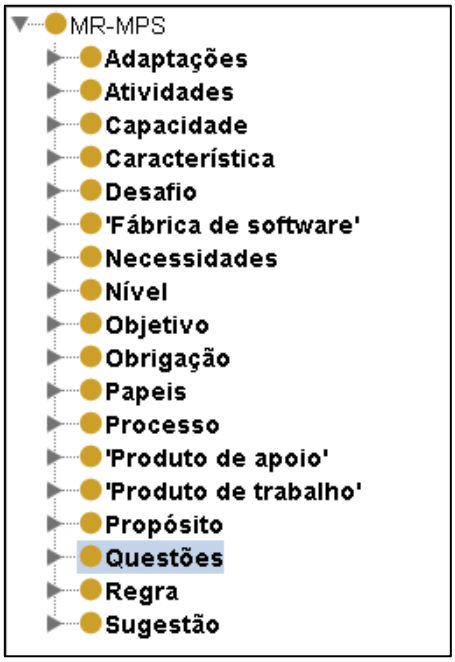


Figura 17 – Principais classes da ontologia.

Convém observar que Pizzoleto (2013) fez uso de outras propriedades na ontologia, mas que não foram mencionadas nesta seção porque não precisam ser utilizadas neste trabalho. É importante ressaltar, que os únicos relacionamentos definidos na ontologia que não utilizam “Object Property” são: relacionamentos entre os processos e seus resultados esperados.

Tabela 5- Principais propriedades de relacionamento da ontologia de Pizzoleto.

Propriedades	Elementos referenciados (objetos)
hasFS	Diferenças em fábrica de software
hasobligation	Obrigação
haspurpose	Objetivo
hasproposal	Sugestão
hasresult	Resultado do processo

2.6 TRABALHOS RELACIONADOS À ABORDAGEM DO CAPÍTULO

Durante o levantamento bibliográfico sobre ontologias, foram encontrados alguns trabalhos relacionados à área de Engenharia de Software, como o de Villela, Travassos e Rocha (2004), Cerqueira (2007), Jenz (2003) e Ternai e Torok (2011). Esses trabalhos ajudaram no entendimento de como as ontologias estão sendo utilizadas no contexto das organizações empresariais

Em Villela, Travassos e Rocha (2004) foi proposta uma solução para introduzir a Gerência de Conhecimento em fábricas de software. A ideia é que o conhecimento gerado durante anos em vários projetos ficasse à disposição de todas as equipes de projetos para projetos futuros. Para tal objetivo ser alcançado, os autores construíram um ambiente de apoio ao desenvolvimento de software, mais comumente conhecidos como ADS (Ambiente de Desenvolvimento de Software). O principal objetivo do trabalho é apoiar o desenvolvimento e a manutenção de software. Considerando apoiar a gestão do conhecimento, os autores propõem uma ontologia voltada para os desenvolvedores de software, para fornecer um vocabulário comum que possa representar o conhecimento das organizações envolvidas em um projeto de software. De forma complementar, uma linguagem de modelagem de processos foi definida, pois segundo os autores, as linguagens existentes não possuem uma forma de representar os conhecimentos produzidos durante os processos de desenvolvimento de software.

Em Cerqueira (2007), é proposto um método para auxiliar a elicitação de requisitos, com objetivo de beneficiar o desenvolvimento de sistemas de informação. O método proposto requer a construção de um modelo ontológico que represente o conhecimento envolvido no negócio, bem como um modelo de processo claro e objetivo, que represente o processo do negócio da empresa. A proposta consiste na integração de ambos os modelos, assim como na simplificação da elicitação dos

requisitos. Para isso, o autor apresenta uma série de passos que visam cruzar as informações da ontologia com o modelo de processos, gerando tabelas de informações cruzadas. Um desses passos é identificar as atividades e os artefatos de entrada e saída no modelo de processos. Outro passo é identificar as atividades e relacionamentos na ontologia, pois através dos relacionamentos destacados na ontologia, pode-se identificar novas atividades que não estão explícitas na modelagem de processos.

Convém ressaltar que foram encontrados trabalhos que utilizaram ontologias empresariais como uma “ponte” (etapa intermediária) entre as etapas de especificação e de modelagem de processos. Esses trabalhos estão voltados à abordagem de Gerenciamento de Processos de Negócio ou BPM (*Business Process Management*), que será abordado no *capítulo 3*. A **Figura 18** apresenta a ontologia como sendo uma forma alternativa para geração de modelos de processos e modelos em UML. Contudo, Jenz (2003) enfatiza que, nesses casos, deve-se ter a preocupação de evitar perdas de informações gerais ou requisitos não funcionais do projeto para o modelo de processos de negócio. A ontologia deve abranger todos os tipos de conceitos relacionados ao processo de negócio, tais como: definição das entidades do processo (atividades, documentos, objetos, eventos, regras, etc.), dos papéis, recursos e fluxo de controle. A ontologia pode, ainda, definir as relações entre as entidades, assim como os sinônimos utilizados no processo, de modo que qualquer pessoa o entenda, e não apenas os especialistas em Tecnologias da Informação (TI). Dessa forma, Jenz (2003) observa que é possível descrever uma atividade de processo em um contexto muito mais completo e geral com uma ontologia. Isso enriquece semanticamente a descrição das atividades do processo, proporcionando aumento da produtividade.

A partir de uma ontologia, diversos artefatos de apoio ao entendimento do domínio podem ser gerados, como, por exemplo, um diagrama de classes UML ou um modelo BPM (ver **Figura 18**). A ontologia pode ser consultada a qualquer momento para eventuais atualizações ou melhorias do processo ou do próprio projeto, se tornando uma base natural e compartilhada de informações (base de conhecimento). Para Jenz (2003), há benefícios quando se tem uma ontologia como base para um modelo de processo de negócio. Para o autor, o maior benefício é a possibilidade de se fazer uma prototipagem rápida, contribuindo com a diminuição do custo do projeto e com a mitigação de riscos.

Outro ponto interessante a se observar é a colocação de Jenz (2003) sobre a possibilidade de o usuário especificar requisitos funcionais e não funcionais em um mesmo lugar (ontologia). Ainda segundo o autor, mesmo que o custo de

desenvolvimento de ferramentas que façam a conversão da ontologia para outros artefatos seja alto, os benefícios de se ter tais ferramentas prevalecem sobre os custos.

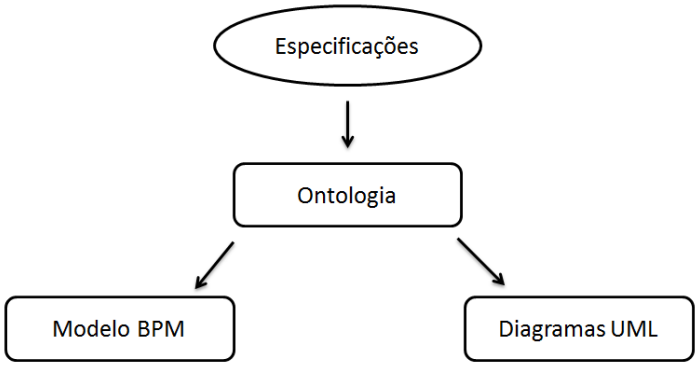


Figura 18 - Ontologia como recurso para a elaboração de modelos em BPM e UML

Por outro lado, autores como Ternai e Török (2011) analisam exatamente o caminho oposto, onde uma ontologia pode ser desenvolvida a partir de um modelo de processos. A intenção é representar os processos, seus dados, informações, colaborações e outros elementos relacionados a partir de ontologias, que podem ser utilizadas para incrementar os modelos executáveis do processo. Segundo Ternai e Török (2011), os modelos de processos já capturam a semântica da aplicação através de uma notação formal, como, por exemplo, BPMN (*Business Process Model and Notation*). Porém, as descrições dessas notações são feitas para serem utilizadas por humanos e não por máquinas (a semântica está implícita nesses modelos), prejudicando a execução do processo.

A conversão do modelo de processos de negócio para uma ontologia divide-se basicamente em duas etapas. A primeira etapa consiste na modelagem do processo através de uma ferramenta de modelagem. A seguir, o modelo é exportado e transformado em uma ontologia em OWL. O fluxo do processo é criado na ontologia a partir de classes, propriedades e atributos. Na segunda etapa deve ser feita a anotação semântica, onde a semântica das tarefas e as decisões no fluxo do processo são especificadas explicitamente na ontologia. Segundo os autores, a ligação entre os elementos da ontologia e os elementos do modelo de processos se dá por meio de propriedades, que especificam a semântica do elemento. A ontologia gerada servirá de apoio à utilização e a execução dos modelos de processos de negócio.

Por meio dos trabalhos apresentados nesta seção, observa-se que ontologias vêm sendo utilizadas para apoiar, diferentes áreas, dentro de organizações

empresariais. Sendo assim, a partir delas podem ser gerados modelos de processos de negócio, na direção dos objetivos deste trabalho. Apesar de Ternai e Torok (2011) ir no sentido de mapeamento oposto deste trabalho, os autores mostraram certas deficiências semânticas dos modelos de processos, em exibir um maior número de informações dos elementos do fluxo do processo. Dessa forma, recomenda-se que ontologias e modelos de processos sejam utilizados conjuntamente para orientação do entendimento e uso dos processos.

2.7 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo apresentou uma visão geral sobre ontologias empresariais, incluindo algumas características fundamentais para sua construção. As ontologias empresariais vêm sendo utilizadas no meio empresarial, principalmente para representação de processos de negócio, como mecanismo de apoio ao entendimento e organização de processos. Assim, elas são ótimos meios de se extrair informações para a construção de modelos de processos.

Dentre as linguagens para representação de ontologias destacam-se a OWL e RDF. A especificação da linguagem OWL inclui propriedades que permitem definir o relacionamento entre dois elementos (“Object Property”). A importância dessas propriedades para o mapeamento de elementos e suas relações da ontologia, fazem da linguagem OWL uma premissa, neste trabalho, para a construção de ontologias. Outros pontos importantes dessa linguagem também influenciaram na sua escolha para este trabalho: como a riqueza de estruturas, que possibilita a representação de conceitos de inúmeras formas, e também, a grande aceitação na área de Web Semântica. Dentre os diversos padrões de representação dessas linguagens, o formato RDF/XML se destaca por ser encontrado em todas as ferramentas que dão suporte à linguagem OWL.

A linguagem SPARQL foi mostrada na seção 2.4, pois é um padrão de consulta a dados em arquivos RDF, e permite que aplicações possam fazer uso de dados contidos em ontologias. Assim, essa linguagem foi utilizada para a construção da ferramenta *RDFtoXPDL*, criada para automatizar parcialmente a metodologia proposta neste trabalho.

Para a aplicação apresentada no *capítulo 5*, a ontologia apresentada na *seção 2.6* foi analisada criteriosamente, com o apoio dos guias e de acordo com os

conhecimentos adquiridos nos estudos do modelo RDF e linguagem OWL, apresentados nas *seções 2.3* e *2.4*, respectivamente.

3 MODELOS DE PROCESSOS DE NEGÓCIO

Segundo Davenport (1994), um processo de negócio é definido como uma série de atividades, ordenadas especificamente para representarem uma ação. Ainda segundo o autor, os processos devem possuir começo e fim, assim como, entradas e saídas bem definidas.

Para Johansson et al. (1995), processo de negócio representa um conjunto de atividades relacionadas que tem como objetivo tomar uma entrada (*input*) e transformar em um resultado (*output*), de forma a adicionar valor à saída.

Já Villela (2000), define processo de negócio como um conjunto de entradas, saídas (resultados), tempo, espaço, ordenação (fluxo do processo), objetivos e valores, que logicamente relacionados, irão compor uma estrutura de entrega de valores aos clientes, isto é, para fornecer produtos ou serviços.

Uma definição mais completa é apresentada por Santoro, Iendrike e Araújo (2011):

> *Um processo é um caminho para uma empresa organizar o trabalho e os recursos (pessoas, equipamentos e informações) para atingir seus objetivos. É composto por um conjunto de atividades bem caracterizadas do trabalho, realizadas em certo momento por responsáveis que assumem papéis específicos. As atividades são realizadas de acordo com um conjunto de regras que estabelecem a ordem e as condições de execução. Na execução de cada atividade são manipulados os produtos de trabalho (dados, documentos ou formulários).*

Ainda de acordo com os autores Santoro, Iendrike e Araújo, os processos podem ser decompostos em: objetivo, evento, atividade, ator, entrada, saída e regras. O objetivo define a razão pela qual o processo está sendo executado. O evento indica um acontecimento, que irá determinar quando uma ação será executada. Regras irão definir as dependências entre as atividades, podendo inclusive interferir no fluxo do processo. As atividades indicam cada passo ou ação a ser executado. O ator é responsável por executar uma ou mais atividades. Entradas são dados ou documentos

utilizados na execução das atividades. E, por fim, as saídas são os resultados gerados pelas ações do processo.

Os processos podem possuir diferentes complexidades, desde um único processo sendo executado uma única vez, por um único ator e com atividades sequenciais, até mesmo um processo com diferentes atores, que executam diversas atividades, podendo estas ser em paralelo. Uma única tarefa possa se quebrar em diversas subtarefas, dando origem a um subprocesso. Já em processos cooperativos, atores podem se comunicar com outros atores, dando origem a eventos de comunicação.

Segundo Smith e Fingar (2003), um processo de negócio é um conjunto completo de atividades transacionais colaborativas, dinamicamente coordenadas, que entregam valor para os clientes. Os autores destacam que um processo de negócio consiste em uma sequência de tarefas, com um objetivo em comum, onde as tarefas podem ocorrer em série ou paralelo. Um processo de negócio possui uma determinada lógica, de modo que ela define a sequência de execução das tarefas.

Para poder compreender os processos de negócio de maneira adequada, bem como gerenciá-los adequadamente, é importante conhecer o ciclo de vida desses processos, conforme ilustrado na **Figura 19** (GEORGAKOPOULOS; TSALGATIDOU, 1998). A fase de “captura” corresponde ao levantamento e modelagem dos processos de negócio. Após a modelagem dos processos, aparece a fase denominada “reengenharia”, onde os processos são analisados de forma que se possa identificar inconsistências, ou qualquer outro tipo de atividade que possa estar criando um gargalo no processo. O objetivo dessa fase é encontrar pontos que possam ser otimizados. Na fase seguinte, de “implementação”, é realizada a automação das atividades do processo de negócio; equivale à realização dos processos de negócio com a aplicação de TI. Na última fase, chamada de “melhoria contínua”, o desempenho dos processos é avaliado, definindo ações para melhoria do processo, caso seja necessário. Para isso podem ser utilizados indicadores, de acordo com a estratégia da empresa.

Face ao exposto, um processo de negócio pode ser entendido como um “caminho” para uma empresa organizar o trabalho e os recursos para atingir seus objetivos, composto por atividades realizadas por responsáveis com papéis específicos. As atividades são realizadas de acordo com regras, seguindo uma ordem de execução. Os produtos de trabalho (dados, documentos, formulários) são utilizados durante a execução de cada atividade do processo.

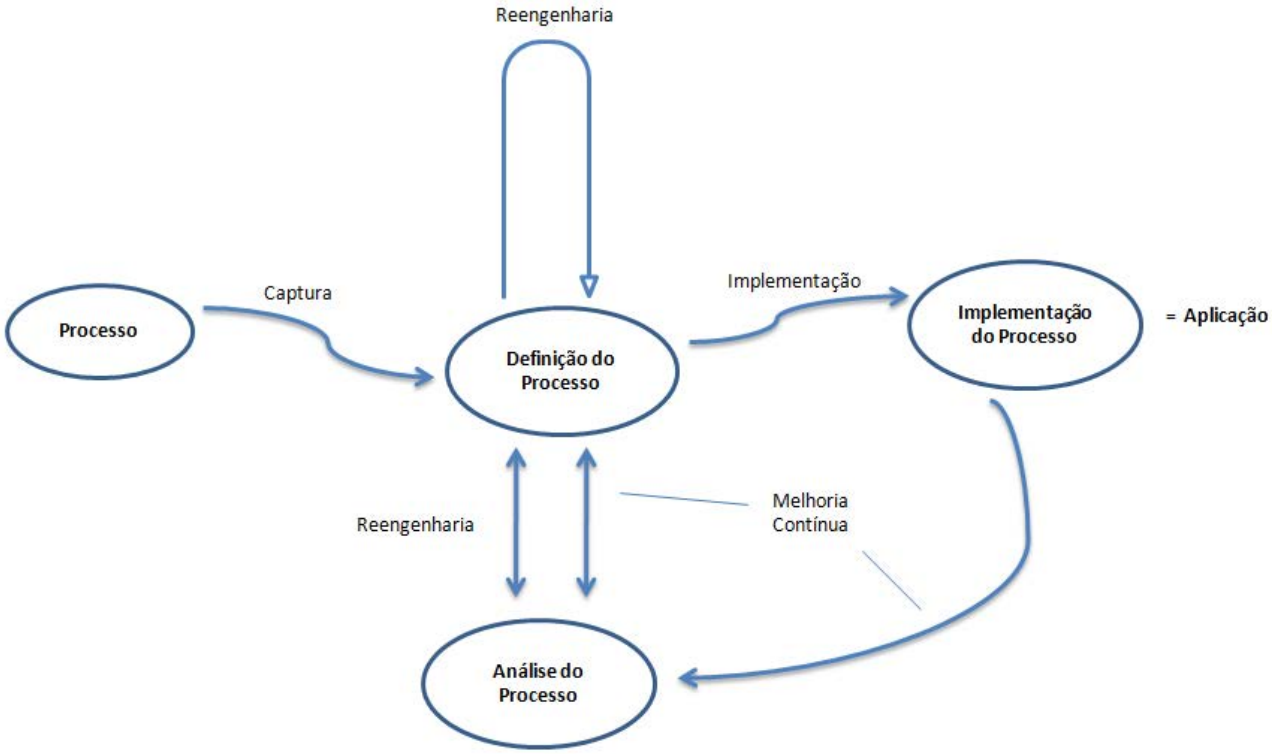


Figura 19 - Ciclo de vida do processo de negócio.

Fonte: extraído de Georgakopoulos e Tsalgatidou (1998).

De acordo com Santo, Iendrike e Araújo (2011), muitas vezes os processos não estão claros, visíveis ou documentados, e para isso, a modelagem de processos de negócio aparece como um conjunto de métodos e técnicas que apoiam a formalização dos processos de negócio. Os autores ainda explicam que a modelagem auxilia no entendimento de questões críticas do negócio. Segundo os mesmos autores, entende-se como formalização do processo a representação textual ou gráfica do negócio como um todo, ou de apenas parte dele.

Um modelo de processo deve representar todas as características possíveis existentes no processo de negócio (como por exemplo: atividades, pessoas, papéis, dados, sistemas que apoiam a execução das atividades, infraestrutura tecnológica, produtos, regras do negócio, entre outros) (SILVA, 2001), podendo assim criar uma visão única dos processos organizacionais, facilitando o entendimento por todos os membros da organização. Enfim, para que uma modelagem de processos funcione, esta deve ser capaz de entender uma quantidade grande de situações que podem fazer parte de um processo de negócio.

Um dos principais fatores que levam os processos de negócio a apresentar problemas durante sua execução é a falta de informações sobre o processo

(SANTORO; IENDRIKE; ARAUJO, 2011). Normalmente os papéis participantes do processo desconhecem o processo como um todo ou parte dele, principalmente quando os processos inter-relacionam várias áreas da organização, ocasionando retrabalho, atrasos na realização das tarefas, atividades mal executadas, e consequentemente um custo mais elevado. Por isso, disponibilizar recursos, como os modelos de processos, para que os participantes envolvidos possam conhecer o processo são de suma importância (SANTORO; IENDRIKE; ARAUJO, 2011).

Dessa forma, a gestão dos processos de negócio auxilia a gestão do conhecimento nas organizações, proporcionando métodos para apoiar a formalização do conhecimento das pessoas, e posteriormente disponibilizar esse conhecimento para o alcance de resultados melhores (SANTORO; IENDRIKE; ARAUJO, 2011).

Existem diversas abordagens para a formalização de modelos de processos, dentre elas: fluxogramas, diagramas UML, EPC (*Event-Driven Process Chain*), Rede Petri e BPMN (*Business Process Model and Notation*). BPMN é uma notação gráfica, e representa o padrão de modelagem utilizado pela tecnologia BPM (*Business Process Management*). É uma das técnicas mais utilizadas atualmente e também foi adotada por este trabalho. A *seção 3.2* apresenta um detalhamento maior sobre os objetivos e os principais elementos que compõe essa notação.

A modelagem de processos de negócio através de uma notação gráfica e padronizada é considerada o primeiro passo para a implementação da abordagem BPM. Segundo Momotko e Nowicki (2003), a modelagem se torna realmente útil com a execução do processo, como parte do ciclo de vida do processo de negócio. Nessa direção, autores como Shapiro (2002) acreditam que as linguagens de execução de modelos de processos, como BPEL (*Business Process Execution Language*) não são capazes de reconstruir o diagrama do processo na íntegra, perdendo informações valiosas. Dessa forma, a linguagem XPDL (*XML Process Definition Language*) aparece como uma forma de representar o modelo de processos, através de XML, mantendo as especificidades do processo, e podendo servir de apoio à execução do mesmo. A *seção 3.3* traz algumas considerações sobre essa linguagem, seus objetivos e sua estrutura.

Devido à enorme quantidade de informações que precisam ser trabalhadas e gerenciadas em um modelo de processos, a utilização de uma ferramenta computacional para apoiar todo o ciclo do processo também é essencial (ARAUJO et al., 2004).O BPMS (*Business Process Management System*) surgiu como um conjunto de ferramentas que apoia a automatização do ciclo de vida dos processos de negócio,

integrando pessoas e sistemas (SANTORO; IENDRIKE; ARAUJO, 2011). Essa ferramenta é a base para a modelagem de processos de negócio e também para a prática de BPM. Porém, ressalta-se que, considerando a etapa de modelagem de processos, existem diversas ferramentas que são específicas para esse objetivo. Essas ferramentas podem possuir diversas notações, mas considerando o interesse deste trabalho em BPMN e XPDL, a *seção 3.4* apresenta a avaliação de três ferramentas utilizadas neste trabalho: *Bizagi Modeler*, *Cameo Business Modeler* e BPMN *Visio Modeler*.

Nessa direção, a *seção 3.1* apresenta informações gerais sobre modelagem de processos. As *seções 3.2* e *3.3* apresentam mais informações sobre a notação BPMN, e a linguagem XPDL, respectivamente. Já a *seção 3.4* apresenta uma avaliação sobre três ferramentas de modelagem de processos, utilizadas neste trabalho. A *seção 3.5*, por sua vez, apresenta conceitos relacionados à tecnologia BPM e *workflows*, e seu ciclo de vida. Embora essa abordagem não faça parte dos propósitos deste trabalho nesse momento, mas podendo fazer em trabalhos futuros. A *seção 3.6* apresenta alguns trabalhos relacionados com o contexto de modelagem de processos. Por fim, a *seção 3.7* apresenta as considerações finais sobre este capítulo.

3.1 MODELAGEM DE PROCESSOS DE NEGÓCIO

A modelagem dos processos de negócio tem como principal objetivo facilitar e melhorar o entendimento da forma como uma determinada organização trabalha, possibilitando o monitoramento e controle dos processos da organização (VERDANAT, 1996). Na modelagem podem ser identificadas informações sobre a execução dos processos, como recursos, custo e duração.

Segundo Davenport (1994), a modelagem de processos permite a compreensão dos processos da organização, facilitando a comunicação entre os profissionais da empresa. Como a modelagem ajuda identificar, previamente, problemas e gargalos nos processos, pode-se buscar soluções de acordo com a política e recursos da empresa. A modelagem dos processos existentes é indispensável para se buscar melhorias nos processos, otimizando-os. A comparação das cadeias com e sem otimização permite avaliar a efetividade da melhoria implementada.

No geral, a modelagem de negócios permite que a organização empresarial possa ter uma representação uniforme (padronizada) do próprio negócio, podendo controlar e monitorar os processos da empresa. Assim, podem ser identificados pontos fortes e fracos, de modo a se atuar em cada um deles, minimizando ou maximizando seus efeitos. A modelagem de processos de negócio ajuda a organização a responder questões críticas sobre o seu negócio, como, por exemplo: o que está sendo feito, por que está sendo feito, onde, por quem, quando e de que forma está sendo feito (PROFORMA, 2000). A **Figura 20** apresenta a posição do modelo de processos frente à explicitação de todos os conceitos envolvidos no negócio, tendo como principal resultado o Modelo de Negócio da organização.

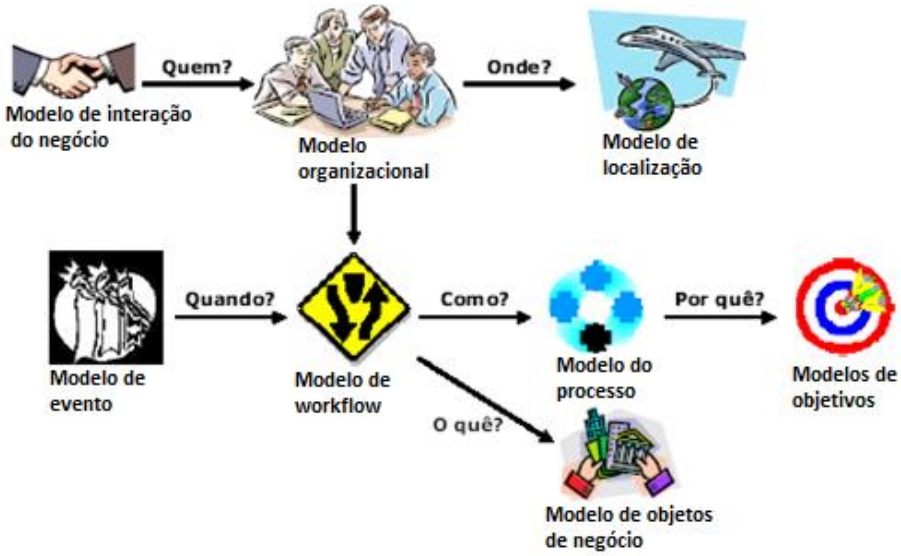


Figura 20 – Posição do modelo de negócio frente aos conceitos do negócio.

Fonte: extraído de PROFORMA (2000).

Existem diversas abordagens para modelagem de processos de negócio, com variações de linguagens e notações. Contudo, as seguintes ações parecem estar presentes em várias metodologias de alguma forma (SHARP; MCDERMOTT, 2001; ROCHA et al., 2007):

1. Definir escopo de atuação do processo;
2. Identificar e compreender o planejamento estratégico da organização: missão, estratégia, metas e objetivos;
3. Compreender e documentar o estado atual dos processos da organização (estado “AS IS”), considerando macroprocessos e subprocessos, com identificação das atividades vinculadas a cada um;

4. Analisar e avaliar o estado atual dos processos através do modelo obtido na ação 3;
5. Apresentar o modelo aos envolvidos nos processos, de modo a proporcionar oportunidades de discussão e propostas de melhorias para o estado atual, como por exemplo: decidir se abandona algum processo, se algum especialista deve ser contratado, se o processo deve ser mantido como está, se os processos têm de ser redesenhados, etc.;
6. Definir as melhorias desejadas a serem introduzidas no estado atual;
7. Projetar e documentar o estado desejado para os processos da organização (estado “TO BE”).

Conforme apresentado na ação 5, o modelo de processos de negócio pode ser utilizado como instrumento para entendimento, divulgação, discussão, gestão e melhoria contínua da organização, levando em consideração seu funcionamento, metas, produtos, insumos, objetivos e estrutura (ARAUJO et al., 2004).

Santoro, Iendrike e Araújo (2011) apresentam um esquema com os elementos necessários para se realizar uma modelagem de processos, conforme ilustrado na **Figura 21**. É muito importante que, inicialmente, seja definida a sequência de passos (método) para o levantamento das informações a serem modeladas. Para não se perder o foco no levantamento de informações, deve ser definido previamente um metamodelo, que indique o conjunto de informações de interesse para serem utilizadas durante o mapeamento dos processos reais para o modelo formal de processos. Convém observar que para o levantamento das informações, as pessoas envolvidas nos processos devem ser entrevistadas, de modo a se identificar como fazem as atividades do processo, bem como se colher outras informações importantes, como os problemas e conflitos que enfrentam. Isso porque, muitas vezes, a documentação do processo não condiz com a prática. Além disso, é interessante que se tenha contato com as interfaces dos sistemas e outros recursos utilizados.

Assim, a partir do metamodelo, pode-se proceder com o mapeamento e modelagem dos processos, seguindo uma notação gráfica selecionada, que possa ser compreendida por todos os envolvidos. A escolha de uma ferramenta computacional para a modelagem é essencial para o controle de informações, e também, de alterações que normalmente ocorrem durante essa fase. Além disso, poderá apoiar a integração do modelo com outros sistemas de software, bem como a implementação das demais fases relacionadas a gestão de processos de negócio.

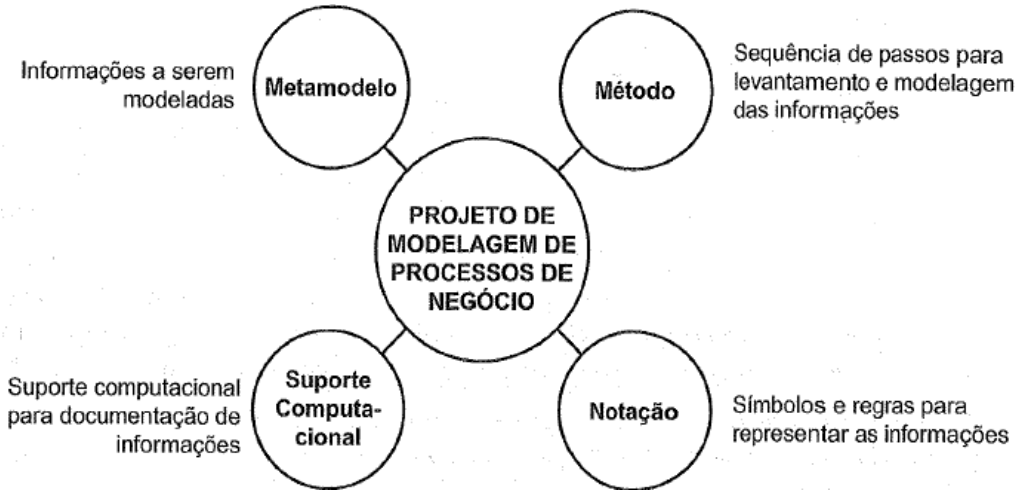


Figura 21 - Elementos necessários para um projeto de modelagem de processos.

Fonte: extraído de Santoro, Iendrike e Araújo (2011).

Convém observar que, para este trabalho, foi selecionada a notação BPMN (*Business Process Model and Notation*), padronizada pelo consórcio OMG (*Object Management Group*), conforme será apresentado na *seção 3.2*. Essa notação foi escolhida por ser intuitiva e aparecer como uma das notações mais utilizadas para modelagem de processos em ambientes empresariais. Considerando os padrões adotados pelos sistemas de modelagem de processos, BPMN também aparece como um dos mais adotados mundialmente. Contudo, outras formas de representação podem ser utilizadas, como a notação de fluxograma, bem como da linguagem de modelagem UML. A linguagem de modelagem do conjunto de métodos IDEF (*Integration DEFinition*) também pode ser utilizada. Observa-se que esse padrão foi originalmente definido pela Força Aérea dos Estados Unidos para a área de Manufatura e tem sido, atualmente, utilizada para modelagem na área de processos de negócio. Outros padrões de representação gráfica, como EPC *(Event-Driven Process Chain*) também têm sido muito utilizados nas empresas.

Uma notação gráfica pode ainda facilitar o entendimento dos fluxos de colaborações e transações de negócio entre organizações parceiras. Segundo Kalpic e Bernus (2002), a principal vantagem de se utilizar modelos estruturais para a descrição de processos está na qualidade da captura e representação do conhecimento formalizado por este.

De modo geral, a modelagem de um processo de negócio pode ser instrutiva por si só, revelando inconsistências e também oportunidades de melhorias. Segundo Koubariakis e Plexousakis (1999), um modelo de processo de negócio pode ser

utilizado para diferentes fins, como por exemplo, avaliar mudanças no lançamento de um novo produto pela organização. Browning (2002) vai mais além, dizendo que a modelagem de processos de negócio pode ser potencialmente utilizada para diversos propósitos:

a) programar o planejamento;

b) servir de base (*baseline*) para melhoria contínua;

c) auxiliar a retenção de conhecimento e aprendizagem por parte da equipe;

d) propiciar visualização do processo como um todo;

e) treinamentos dentro da organização;

f) proporcionar ambiente para a extração de métricas;

g) subsidiar a verificação de conformidades e realização de auditorias;

h) programar a execução dos processos para de acordo com eventos temporais ou ações realizadas.

O modelo de processos resultante pode ser considerado uma ferramenta para suporte ao raciocínio, fazendo com que a modelagem não só contribua para o registro dos processos de negócio, mas também para o raciocínio da melhor maneira de reorganizar o processo (PIDD, 1999).

3.2 NOTAÇÃO BPMN PARA MODELAGEM DE PROCESSOS

BPMN (*Business Process Model and Notation*) é uma notação gráfica para representação de processos de negócio como fluxos de trabalho. Segundo White (2004), o principal objetivo de BPMN é fornecer uma notação de fácil entendimento por todos os usuários do negócio. O conjunto de usuários envolve desde os analistas de negócios, que criam os esboços iniciais dos processos, até os desenvolvedores técnicos, responsáveis pela implementação da tecnologia que irá executar os processos, bem como as pessoas que irão monitorar e controlar os processos (WHITE, 2004).

A especificação inicial da notação BPMN foi publicada em 2004 como resultado de mais de dois anos de trabalho do instituto BPMI (*The Business Process Management Initiative).* A partir de 2005 o BPMI passou a ser integrado com o

consórcio OMG, passando o gerenciamento da notação BPMN ao novo parceiro e, portanto, tornando a BPMN o padrão do consórcio OMG para modelagem de processos de negócio, desde então (OMG, 2005). A versão mais recente de BPMN, versão 2.0, lançada em janeiro de 2011, tem sua notação completa em OMG (2011).

Através de BPMN podem ser criados diagramas de processos de negócio ou BPDs (*Business Process Diagrams*). Um diagrama de processos de negócio deve representar graficamente todo o fluxo de atividades do processo. A linguagem BPMN é constituída por um conjunto de elementos gráficos, representados por meio de formas básicas, como losangos, retângulos, setas, entre outros. Através desses elementos é possível representar vários níveis de refinamento do fluxo de processos, desde os de alta abstração (com macroprocessos) a níveis bem detalhados com subprocessos. Ao mesmo tempo em que BPMN objetiva proporcionar um mecanismo simples para a modelagem de processos, ela permite que a complexidade inerente dos processos de negócio seja expressa.

Para cada elemento do diagrama, podem ser adicionadas variações e informações para ajudar no suporte de requisitos complexos, sem modificar drasticamente a visualização do diagrama (WHITE, 2004).

Conforme apresentado na **Tabela 6**, os elementos gráficos da notação BPMN estão divididos em cinco categorias básicas tornando fácil o reconhecimento dos elementos e o entendimento dos diagramas:

1. Objetos de fluxo: definem o comportamento de um processo de negócio;
2. Objetos de conectividade: utilizados para conectar os objetos de fluxo aos demais elementos do fluxo do processo;
3. Partições (*swinlanes*): utilizadas para duas finalidades: - a primeira para identificação dos participantes e atores do processo; - a segunda para organizar as atividades de acordo com seus participantes dentro do diagrama;
4. Artefatos: transmitem ou mostram informações adicionais pertencentes ao processo sendo modelado;
5. Objetos de dados: utilizados para mostrar itens criados ou consumidos durante a execução do processo de negócio, podendo estes serem físicos ou apenas informativos.

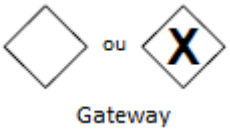
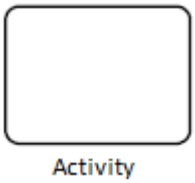
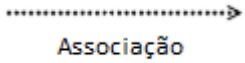
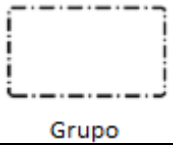
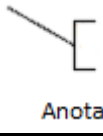
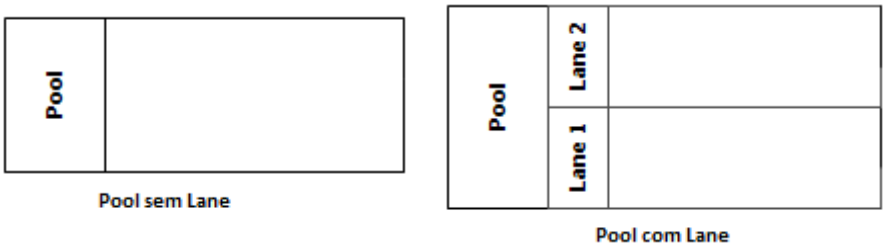
A notação gráfica de BPMN é muito parecida com a de fluxogramas (WHITE, 2004). A especificação de BPMN apresenta também a composição de um conjunto reduzido de elementos, também conhecidos como “core elements”. A finalidade deles é permitir o reconhecimento e compreensão da essência do diagrama de maneira fácil e rápida.

Tabela 6 – Elementos gráficos de BPMN por categoria.

Objetos de fluxo	
Atividade *(Activity)*	Define o trabalho realizado dentro do processo, podendo ser de dois tipos básicos: atividade atômica ou composta. Atividade atômica é uma ação que não pode ser decomposta. Já a composta, pode ser decomposta em várias ações (subprocesso).
Evento *(Event)*	Marcador para indicar um evento que se inicia ou termina no processo, podendo ser de dois tipos básicos: *Start, e End.* Esses eventos indicam o começo e o fim de cada processo ou subprocesso.
Desvio *(Gateways)*	Diverge (divide) ou converge (junta) um fluxo de sequência do diagrama. Gateways podem fazer com que a execução das atividades passe a ser em paralelo, ou até mesmo mudar o caminho de execução.
Objetos de conexão	
Fluxo de sequência *(Sequence Flow)*	Representa a ordem em que as atividades são realizadas.
Fluxo de mensagem *(Message Flow)*	Mostra o fluxo de mensagens entre participantes de diferentes processos (*pools*). Este elemento não pode ser utilizado dentro de um mesmo *pool*.
Associação *(Association)*	Utilizado para associar uma informação (artefato) a atividades ou subprocessos.
Associação de Dados *(Data Association)*	Utilizado para associar um objeto de dados as atividades ou subprocessos.
Raias (Swimlanes)	
Piscinas *(Pools)*	Representa um participante do processo. Pode representar outro processo ou atividade externa.
Divisões *(Lanes)*	Divisão interna de um *pool*. Usadas para representar papeis internos e departamentos de uma organização
Artefatos	
Grupo *(Group)*	Graficamente mostra elementos dentro de uma categoria de grupo. Utilizado apenas como elemento visual, não tendo interferência na execução do processo
Anotações *(Annotation)*	Fornecem informações adicionais para o leitor do diagrama. Não possui nenhuma interferência na execução do processo.
Data	
Objeto de Dados *(Data Objects)*	Mostram dados e informações que são usados, atualizados ou criados durante a execução do processo. Esse elemento pode ser de dois tipos: dados de saída (Data Outputs), ou dados de entrada (Data Inputs).
Bases de dados *(Data Stores)*	São bases que armazenam dados que serão mantidos durante toda a execução do processo.

No total são onze elementos básicos deste conjunto nuclear (*core*) de BPMN, os quais foram apresentados na **Tabela 6**, junto com alguns outros elementos. São eles: (1) Atividade; (2) Eventos; (3) *Gateway*; (4) Fluxo de sequência; (5) Fluxo de mensagens; (6) Objeto de Dados; (7) Anotação; (8) Grupo; (9) Associação; (10) *Pool*; (11) *Lane*. A **Tabela 7** apresenta as formas gráficas associadas a esses elementos nucleares de BPMN.

Tabela 7 - Representação gráfica em BPMN.

Representação gráfica dos elementos essenciais de BPMN	
Eventos	
Desvio	
Atividade	
Fluxo	
Associação	
Grupo	
Anotação	
Piscinas	
Objeto de Dados	

A **Figura 22** apresenta um exemplo genérico de fluxo de processo construído em BPMN. Seguindo o fluxo do “Processo 1”, a “Atividade 1” será executada após o início do processo. Em seguida é verificada se a regra “Regra satisfeita” foi satisfeita, em caso afirmativo, a “Atividade 2” será executada; em caso negativo, a “Atividade 1” deverá ser novamente executada. A execução da “Atividade 2” irá gerar um “Documento” como objeto de dado. Finalizada a “Atividade 2”, o fluxo do processo será encerrado. A **Figura 22** mostra uma anotação explicando que a “Atividade 2” só será executada quando a regra de decisão for satisfeita.

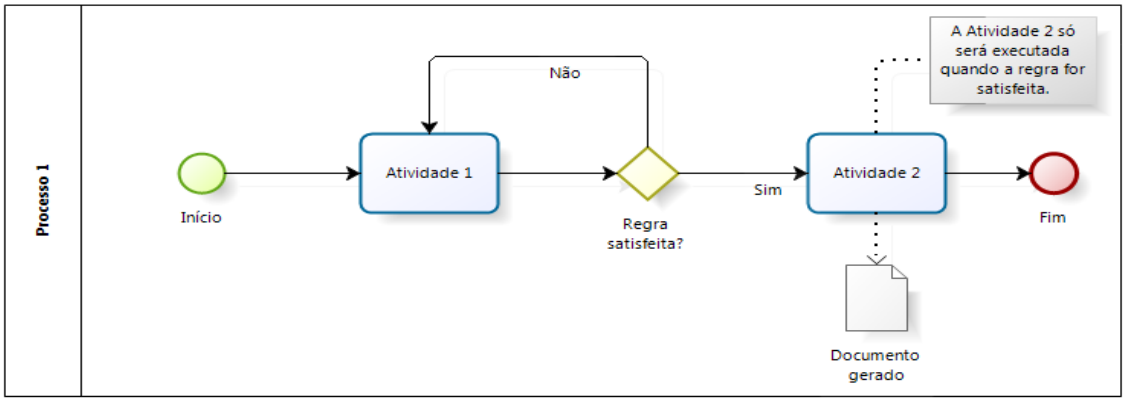


Figura 22 - Exemplo de um diagrama em BPMN.

Atualmente, várias ferramentas de software podem ser utilizadas para facilitar a elaboração da modelagem de processos com BPMN, versão 2.0. Entre elas, algumas ferramentas são de uso livre, como *Bizagi Process Modeler*, *Bonitasoft* e ARIS *Express*. Outras por usa vez são pagas, de uso proprietário, como: *Cameo Business Modeler*, *BPMN Visio Modeler*, *iGrafx* BPM e *Intalio*.

3.3 LINGUAGEM DE SERIALIZAÇÃO XPDL

Desde 1995, quando o Consórcio WfMC (*Workflow Management Coalition*) publicou o primeiro modelo de definição de *workflows* [1] (*Workflow Reference Model*), passou-se a buscar uma forma de permitir a portabilidade dos *workflows* entre diferentes sistemas de gerenciamentos de *workflows* (WfMC, 1995). Em 1998

[1] Segundo o Consórcio WfMC, *workflow* consiste “na automatização de um processo de negócio, no todo ou em parte, durante a qual os documentos, informações ou tarefas são passadas de um participante para outro para ação, de acordo com um conjunto de regras procedimentais” (WfMC, 1999).

apresentou, então, a linguagem WPDL (*Workflow Process Definition Language*) para descrever e compartilhar modelos de processos de negócio. Com a crescente tendência do padrão XML para compartilhamento de dados, o consórcio WfMC começou a repensar a especificação de WPDL. O resultado foi a linguagem XPDL (*XML Process Definition Language)*, que teve a sua primeira versão liberada em 2002. Contudo, não se pode inferir que XPDL é apenas uma versão XML da versão anterior, já que diferentes conceitos foram adicionados e muitos conceitos existentes foram alterados (VAN DER AALST, 2003).

De modo geral, XPDL é uma linguagem para definição de processos de negócio, que possibilita a portabilidade dos modelos de processos, sem risco de perda de informações, apesar de poder ser utilizada por diferentes sistemas editores. (WfMC, 2012). É importante ressaltar que XPDL não pode ser executada, pois sua finalidade é representar e compartilhar processos. A definição de XPDL permite que ferramentas de modelagem de processos possam definir atributos de elementos para serem utilizados como extensão das entidades padrões do XPDL nos modelos de processos (SHAPIRO, 2006). Essas extensões são referenciadas na linguagem pelo atributo “ExtendedAttribute”. Segundo WfMc (2012), devido às necessidades específicas de cada ferramenta, foi necessário criar uma estrutura padronizada de suporte às extensões de cada ferramenta.

Com a criação da notação BPMN em 2004 (ver *seção 3.2*), o WfMC passou a trabalhar em uma nova versão de XPDL que desse suporte a todos os aspectos do padrão BPMN. O resultado foi a versão 2.0 de XPDL, liberada em 2005, que passou a ser recomendada pelo WfMC para o compartilhamento de modelos de processos de negócio criados em BPMN. Atualmente XPDL encontra-se na versão 2.2, com suporte à versão 2.0 de BPMN (versão mais recente).

A **Figura 23** apresenta a arquitetura de XPDL, evidenciando que suas entidades estão completamente relacionadas às da notação BPMN. Pode-se observar que a entidade principal da especificação XPDL é “*Package*”. Essa entidade é um verdadeiro “contêiner” para todos os demais elementos presentes na definição do processo de negócio.

Na **Figura 23**, a entidade “Process” descreve qual será o fluxo do processo de negócio, sendo que um “Package” pode conter um ou mais “Process” em sua definição. A entidade “Process”, por sua vez, serve como “contêiner” para todas as informações relacionadas ao processo de negócio: atividades a serem executadas (“Activity”), transições do fluxo de controle do processo (“Transition”), artefatos

gerados durante a execução do processo (“Artifacts"), entre outros. É interessante observar que cada uma das entidades demonstradas na **Figura 23** possui atributos que permitem identificar a interligação entre os elementos do processo, assim como definir maiores informações do processo de negócio.

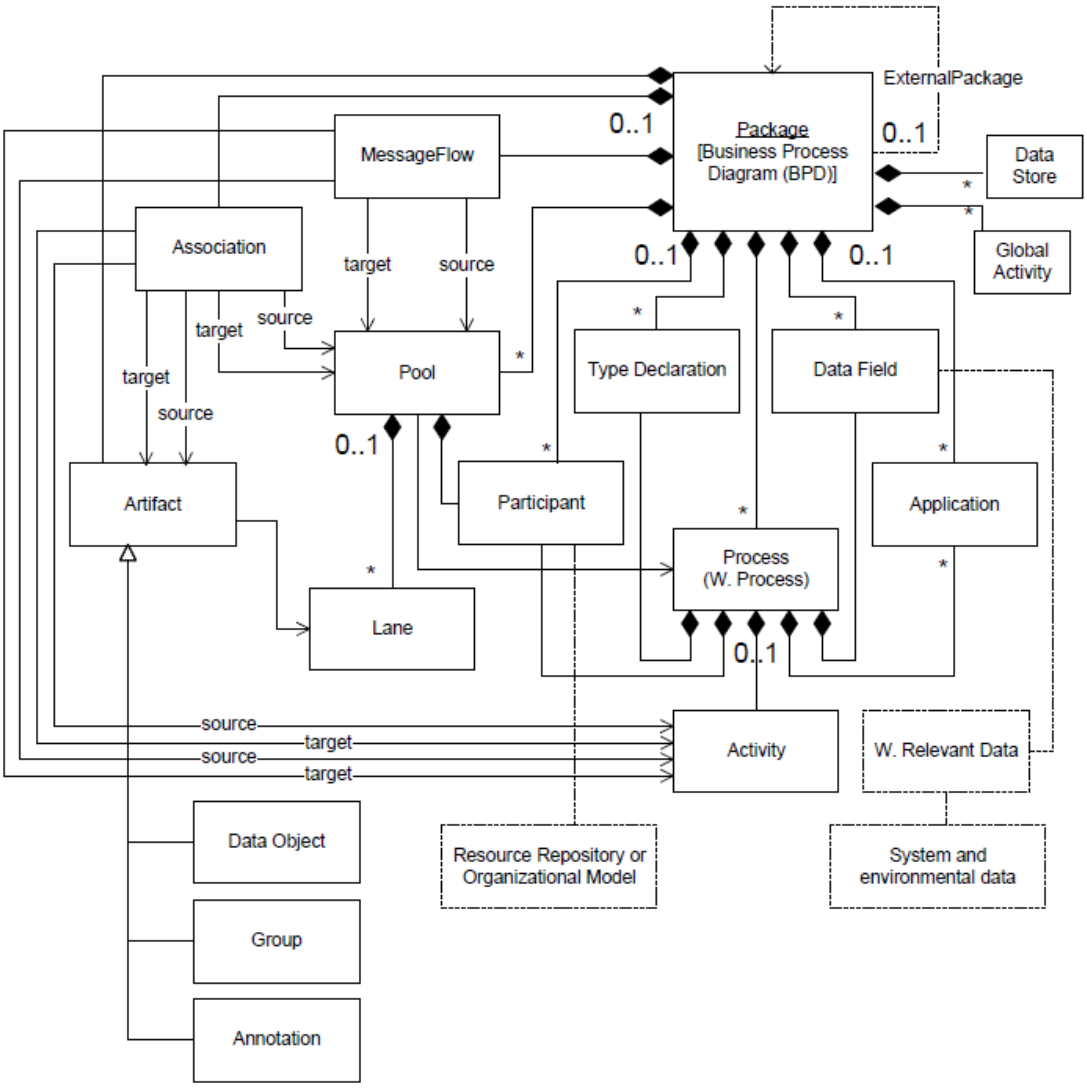


Figura 23 - Entidades da arquitetura da linguagem XPDL.

Fonte: extraído de WfMC (2012).

Segundo o consórcio WfMC (2012), como XPDL trata com elementos gráficos, é necessário indicar posições e coordenadas onde os elementos devem ser representados, para fins de portabilidade entre diferentes ferramentas. Um exemplo dessas coordenadas pode ser visto na **Figura 24**.

```
<NodeGraphicsInfos>
  <NodeGraphicsInfo >
    <Coordinates XCoordinate="730" YCoordinate="120" />
  </NodeGraphicsInfo>
</NodeGraphicsInfos>
```

Figura 24 - Exemplo do uso de coordenadas em XPDL.

Como já mencionado, todos os elementos presentes na especificação de BPMN podem ser representados e compartilhados em XPDL. Adicionalmente, XPDL possibilita que seja feito o uso de elementos que não pertencem à notação BPMN. Esses elementos são descritos na linguagem usando a *tag* “ExtendedAttribute”. Como esse elemento pode ser usado livremente pelas ferramentas de modelagem de processos, elas podem apresentar diferentes abordagens de tratamento. Algumas ferramentas permitem o uso desse elemento pelo usuário, com o objetivo de adicionar maiores informações ao processo sendo modelado. Por outro lado, há ferramentas que acabam utilizando esse elemento para fins próprios, não permitindo a sua utilização pelos usuários.

A **Tabela 8** apresenta a representação XPDL para todos os onze elementos nucleares da notação BPMN. Considerando a importância para este trabalho, a tabela inclui também a representação para o elemento “Subprocess” (atividade composta). O APÊNDICE C traz um exemplo de processo criado em notação BPMN, e sua respectiva representação na linguagem XPDL. Maiores informações sobre todas as definições necessárias para a codificação em XPDL podem ser encontradas em WfMC (2012).

3.4 SISTEMAS DE SUPORTE A MODELAGEM DE PROCESSOS

Como já mencionado na introdução deste capítulo, há várias ferramentas para modelagem de processos de negócios, que podem usar diferentes notações/linguagens, como fluxogramas, BPMN, EPC, IDEF, UML, entre outras. Contudo, para os propósitos deste trabalho, a notação BPMN utilizada depende da linguagem XPDL. Assim, as ferramentas de modelagem candidatas a uso neste trabalho devem providenciar suporte à BPMN e XPDL, e, ainda, às suas versões mais recentes utilizadas neste trabalho: versão 2.0 e 2.2, respectivamente.

Tabela 8 - Mapeamento entre elementos BPMN e XPDL.

BPMN	XPDL

Process	`<WorkflowProcesses>`
	`  <WorkflowProcess />`
	`</WorkflowProcesses>`

Activity	`<Activities>`
	`  <Activity />`
	`</Activities>`

SubProcess	`<ActivitySets>`
	`  <ActivitySet />`
	`</ActivitySets>`

Gateway	`<Activities>`
	`  <Activity >`
	`  <Route />`
	`  </Activity>`
	`</Activities>`

Event	`<Activities>`
	`  <Activity>`
	`  <Event />`
	`  </Activity>`
	`</Activities>`

SequenceFlow	`<Transitions>`
	`  <Transition />`
	`</Transitions>`

MessageFlow	`<MessageFlows>`
	`  <MessageFlow />`
	`</MessageFlows>`

Association	`<Associations>`
	`  <Association />`
	`</Associations >`

Pool (com Lanes)	`<Pools>`
	`  <Pool >`
	`  <Lanes />`
	`  </Pool>`
	`</Pools>`

DataObject	`<DataObjects>`
	`  <DataObject / >`
	`</DataObjects>`

Group	`<Artifacts>`
	`  <Artifact ArtifactType="Group" >`
	`  </Artifact>`
	`<Artifacts />`

Annotation	`<Artifacts>`
	`  <Artifact ArtifactType="Annotation" >`
	`  </Artifact>`
	`<Artifacts />`

Apesar de sistemas de BPM, BPMSs (*Business Process Management Systems*), sempre possuírem ferramentas integradas para a geração de modelos de processos, alguns fabricantes disponibilizam, além de seus BPMSs, ferramentas exclusivas para modelagem de processos, como é o caso da empresa Bizagi Ltda, que possui uma ferramenta BPMS (*Bizagi Engine*, de uso proprietário), e também uma ferramenta exclusiva para modelagem (*Bizagi Modeler*, de uso livre). Geralmente, nesses sistemas, as ferramentas de modelagem têm a preocupação de evitar ambiguidades, de modo a trazer facilidades no entendimento do processo e interpretação dos códigos gerados.

Assim, para este trabalho, alguns sistemas de modelagem de processos foram analisados, de forma a verificar se estes estavam de acordo com a notação BPMN 2.0, e também com suporte a importação de arquivos em linguagem XPDL, versão 2.2. São eles: *Bonitasoft (*código livre*), ADONIS Community Edition, ARIS Express, Intalio, Bizagi Modeler, BPMN Visio Modeler* e *Cameo Business Modeler*. Apesar de todas as ferramentas possuírem suporte a notação BPMN 2.0, apenas três delas possuíam suporte a importação de arquivos XPDL 2.2. São elas: (1) *Bizagi Modeler*, da Bizagi Ltda; (2) *Cameo Business Modeler*¸ da No Magic; (3) *BPMN Visio Modeler*, da Trisotech. Os demais sistemas analisados não puderam ser considerados para este trabalho, ora por não darem suporte a versão 2.2 de XPDL, e ora por não possuírem a funcionalidade de importar arquivos nessa linguagem.

A ferramenta de modelagem *Bizagi Modeler* é de uso livre (*freeware*), podendo ser baixada gratuitamente em *http://www.bizagi.com/en/bpm-suite/bpm-products/modeler.* Apresenta uma interface de fácil compreensão e alto grau de usabilidade. Possui a funcionalidade de exportar um modelo de processos e a de importar um modelo criado em outra ferramenta de modelagem, seja em BPMN ou XPDL. Atualmente a ferramenta trabalha com as versões BPMN 2.0 e XPDL 2.2. É importante ressaltar que a ferramenta *Bizagi Modeler* segue fielmente as especificações da notação BPMN e da linguagem XPDL. A **Figura 25** apresenta um instantâneo da modelagem de processos com a *Bizagi Modeler*. Cabe observar que o BPMS Bizagi permite executar todo o ciclo de BPM, desde a modelagem do processo até o monitoramento e automação do mesmo. Entretanto, com exceção da ferramenta *Bizagi Modeler*, todo o restante desse sistema proprietário é pago. Contudo, pelo contexto acadêmico deste trabalho, foi permitido acesso gratuito a todo o BPMS Bizagi.

A ferramenta de modelagem de processos *Cameo Business Modeler*, diferente do *Bizagi Modeler*, não está integrada a nenhum BPMS específico. Trata-se

de uma ferramenta paga, com uma versão (*trial*) disponível para uso por um período de tempo limitado (255 dias). Permite a modelagem de diagramas em BPMN 2.0, assim como a importação e exportação de modelos em BPMN 2.0 e XPDL 2.2 e 2.1. Essa ferramenta se destaca pela funcionalidade de avaliação de conformidade da modelagem com a especificação da notação BPMN. Além disso, possibilita que a modelagem seja feita de forma colaborativa, com a participação de usuários de um mesmo domínio. A **Figura 26** apresenta um instantâneo da interface da ferramenta *Cameo Business Modeler*.

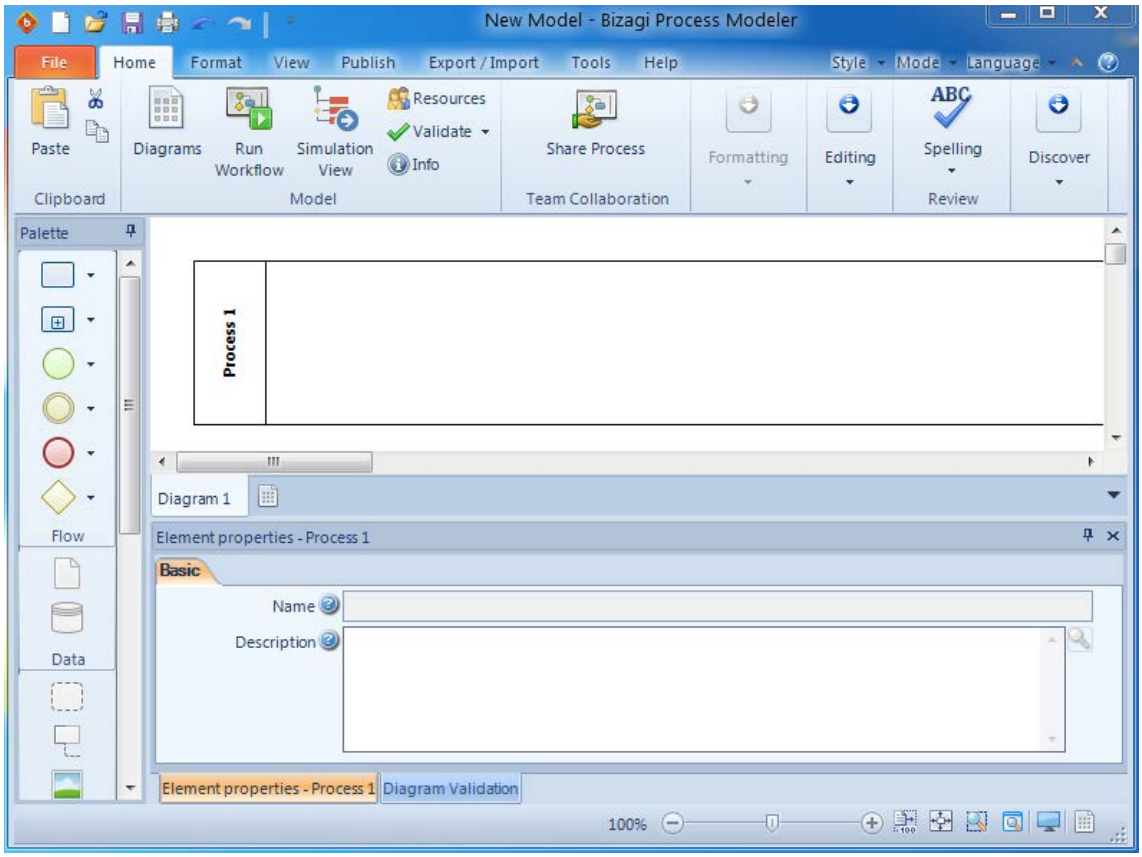


Figura 25 – Instantâneo da interface da ferramenta *Bizagi Modeler*.

A ferramenta BPMN *Visio Modeler* é utilizada como um *plug-in* para a ferramenta Microsoft Visio, que consiste em conjunto de bibliotecas para criação de diagramas de diversos tipos, como fluxogramas, diagramas de rede, diagramas UML, entre outros. Apesar de ser uma ferramenta paga, uma licença do tipo *trial* do sistema Visio é disponibilizada para avaliação em *http://www.microsoft.com/en-us/evalcenter/evaluate-visio-professional-2013*. Atualmente, BPMN *Visio Modeler* tem suporte à especificação de BPMN 2.0. Uma das características que faz a ferramenta se destacar entre as demais é a possibilidade de conexão a um repositório on-line,

mantido pelos próprios criadores da ferramenta. Esse repositório contempla diversas ferramentas de apoio à modelagem, como, por exemplo: validação do diagrama construído em relação à especificação de BPMN e conversão dos arquivos gerados pela ferramenta para outras linguagens, como XML e XPDL. Por outro lado, um ponto negativo da ferramenta está na representação dos modelos em XPDL: a ferramenta não está em conformidade com o padrão de XPDL. Um exemplo disso é que não é permitindo inserir elementos subprocesso do tipo "*embedded*". A **Figura 27** representa um instantâneo da interface da ferramenta Miscrosoft Visio com o *plug-in* BPMN Modeler.

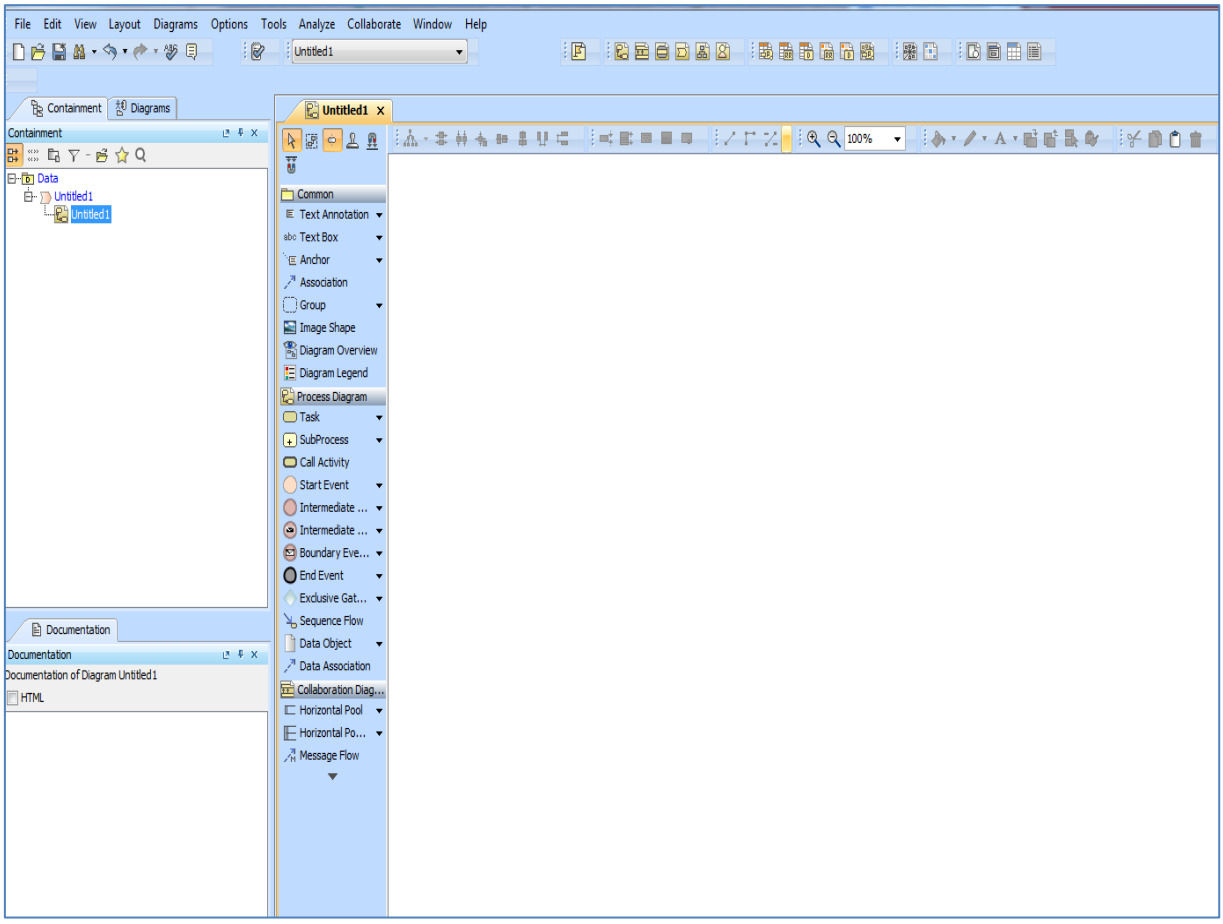


Figura 26 – Instantâneo da interface da ferramenta *Cameo Business Modeler*.

Com a avaliação das três ferramentas descritas nesta seção, pode-se observar que apenas a ferramenta Bizagi Modeler segue fielmente as especificações de BPMN e XPDL. As demais ferramentas, Cameo Business Modeler e BPMN Visio Modeler, apesar de possuírem recursos extras para validação do modelo gerado,

acabam não respeitando regras da especificação, tanto de BPMN, quando de XPDL, sobre a utilização de “subprocessos” do tipo “embedded”.

Um ponto positivo diz respeito à interface amigável e intuitiva de todas as ferramentas, que apresenta um painel contendo todos os elementos nucleares de BPMN, de maneira limpa e visível para o usuário, facilitando assim a busca pelos elementos da notação.

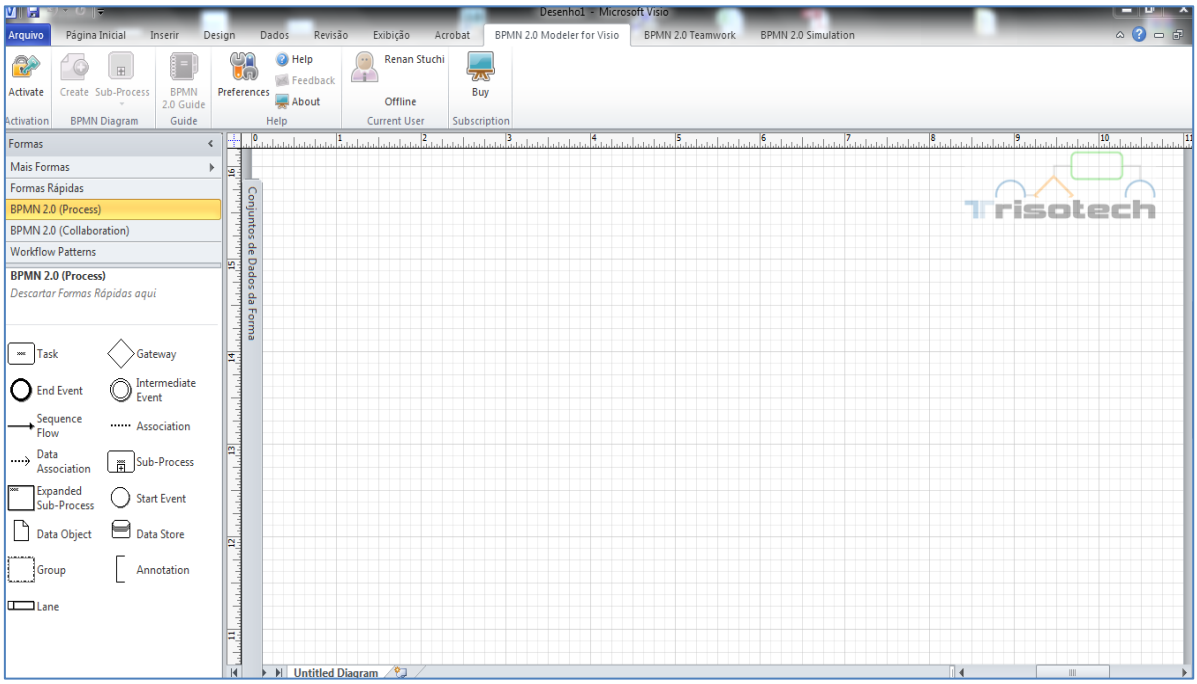


Figura 27 – Instantâneo da interface da ferramenta Microsoft Visio com o *plug-in* BPMN Modeler.

3.5 BPM: GERENCIAMENTO DE PROCESSOS DE NEGÓCIO

A tecnologia de Gerenciamento de Processos de Negócio, conhecida internacionalmente como BPM (*Business Process Management*), consiste em um conjunto de métodos e técnicas que auxiliam uma organização na gestão de seu negócio, através do conhecimento e entendimento dos seus processos. As organizações formalizam, então, seus processos, representando-os através de uma linguagem comum e de entendimento uniforme (JACOBSON; ERICSSON; JACOBSON, 1994). A abordagem BPM contribui com a transparência, melhorias contínuas e padronização dos processos de negócio da organização.

É difícil abordar BPM sem entender o conceito de *workflow*. Assim, a *subseção 3.5.1* apresenta formas de conceituar *workflow*, relacionando-o com processos de negócio.

A *subseção 3.5.2*, por sua vez, apresenta as fases do ciclo de vida de BPM para sua implementação em uma organização.

3.5.1 Conceitos de *Workflow* e processos de negócio

De acordo com Casati et al. (1995), *workflow* é uma coleção de atividades que cooperam entre si para alcançar um objetivo comum. Leymann e Roller (1997), por sua vez, definem *workflow* como um conjunto de atividades que podem ou não ser executadas simultaneamente, contemplando algumas especificações de controle e fluxo de dados entre as atividades. Segundo Bortoli e Price (2000), um *workflow* é formado por um conjunto de atividades relacionadas, executadas por um ou mais atores, com controle de fluxo e condições de execução das atividades. Os autores ainda complementam dizendo que as atividades podem ser processadas ao mesmo tempo (ou não).

De modo geral, há várias definições para *workflow* encontradas na literatura. Segundo Van Der Aalst e Hee (2000), o termo *workflow* é muito utilizado com o significado de “processos de negócio”. De fato, esses conceitos estão relacionados, mas não são sinônimos.

Segundo o Consórcio WfMC, um *workflow* consiste “na automatização de um processo de negócio, no todo ou em parte, durante a qual os documentos, informações ou tarefas são passadas de um participante para outro para ação, de acordo com um conjunto de regras procedimentais” (WfMC, 1999). A tecnologia de *workflow* permite a separação entre a lógica dos procedimentos de negócio e o suporte operacional de TI. O gerenciamento de *workflows* envolve tudo que possa apoiar os processos de negócios, como, por exemplo: ideias, métodos, técnicas e software.

Segundo o consórcio WfMC (1995), um processo de negócio é descrito como um conjunto de atividades relacionadas que visam atingir um objetivo de negócio, no contexto de uma estrutura organizacional. Segundo a especificação da linguagem BPMN 2.0, um processo de negócio consiste de “um conjunto de atividades

empresariais que representam as etapas requeridas para alcançar um objetivo de negócio. Ele inclui o fluxo e uso de informações e recursos" (OMG, 2011).

Neste contexto, Van Der Aalst, Hofstede e Weske (2003) aproveita o ciclo de vida de BPM para fazer uma comparação com o ciclo de vida do Gerenciamento de *Workflow*, ou WFM (*Workflow Management*). Os autores apresentam o ciclo de vida de BPM em quatro fases, como mostra a **Figura 28**. A fase de projeto (1) consiste no desenho dos processos. A fase de configuração (2) consiste na implementação dos processos em um sistema de informação que irá apoiar a execução. Na fase de execução (3) os processos são executados de acordo com as configurações da etapa anterior. E por fim, a fase de diagnóstico (4) consiste na análise dos processos em busca de inconsistências.

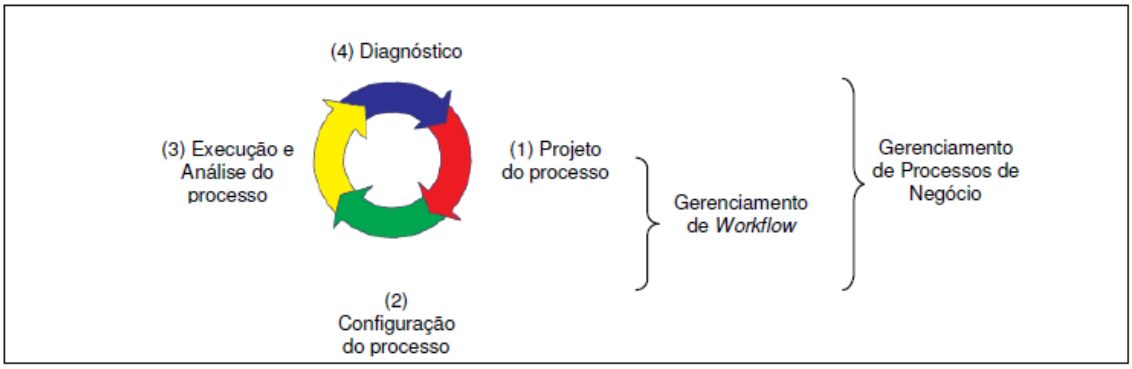


Figura 28 - Ciclo de vida de BPM comparado ao ciclo de vida de WFM.
Fonte: extraído e traduzido de Van der Aalst, Hofstede e Weske (2003).

Ainda de acordo com a **Figura 28**, observa-se que o foco do Gerenciamento de Workflow está na parte inferior do ciclo de vida de BPM, isto é, na fase de execução. Considerando ainda as definições de workflow apresentadas nesta subseção, nota-se que as definições dão enfoque ao termo "execução" e a componentes relacionados ao processo de negócio. Portanto, apesar de sinônimas, as técnicas de workflow e BPM são complementares, onde o workflow apoia a parte de execução e automação dos processos de negócio, e BPM, por sua vez, possui um foco mais gerencial a análise, monitoramento e otimização dos processos.

Por fim, cabe ressaltar que alguns autores, como Netto (2008), expressam que, modelos de workflow precisam de melhorias, como por exemplo, restrições temporais, execução dinâmica, recuperação do histórico do fluxo do processo, entre outras.

A **Figura 29** apresenta um mapa conceitual que relaciona termos utilizados na área de *workflow* (WfMC, 1999). Segundo WfMC (1995), esses termos podem ser conceituados conforme apresentado a seguir::

- **Processo de negócio**: conjunto formado por um ou mais procedimentos ligados entre si ou atividades, que coletivamente compreendem um objetivo de negócio, definindo regras e relacionamentos para seu funcionamento;
- **Negócio**: representação dos processos;
- **Sistema de Gerenciamento de *Workflow***: sistema que define, gerencia e executa o *workflow* (processos de negócio) por meio da execução de software, cuja ordem de execução é direcionada por uma representação computacional da lógica do *workflow*;
- **Instância de Processo**: é a execução do processo;
- **Atividades**: são descrições formais de uma parte do trabalho, que formam um passo lógico dentro de um processo. Uma atividade requer recursos humanos (Atividades Manuais) e/ou máquinas (Atividades Automatizadas) para apoiar a execução de processos;
- **Instâncias de atividade**: representação da execução da atividade, sendo relacionada a uma única instância de processo;
- **Itens de trabalho**: representação do trabalho que deve ser processado por um participante do *workflow*, no contexto de uma atividade dentro de uma instância de processo.

3.5.1 Ciclo de vida de BPM

BPM pode ser visto como uma evolução da abordagem de *workflow,* englobando e expandindo seus conceitos através de outras tecnologias. BPM pode ser entendido como um conjunto de boas práticas para a modelagem, automatização, gerenciamento e otimização dos processos de negócios. Essa tecnologia pode ser utilizada em diferentes tipos de processos, com diferentes abordagens dentro de uma organização.

Segundo Khan (2003), a adoção de BPM normalmente contribui para grande economia financeira, redução de tempo do projeto e chega a triplicar a taxa de

satisfação do cliente. O uso de um modelo de BPM contribui para se prever possíveis erros de processos ou falhas graves na troca de informações.

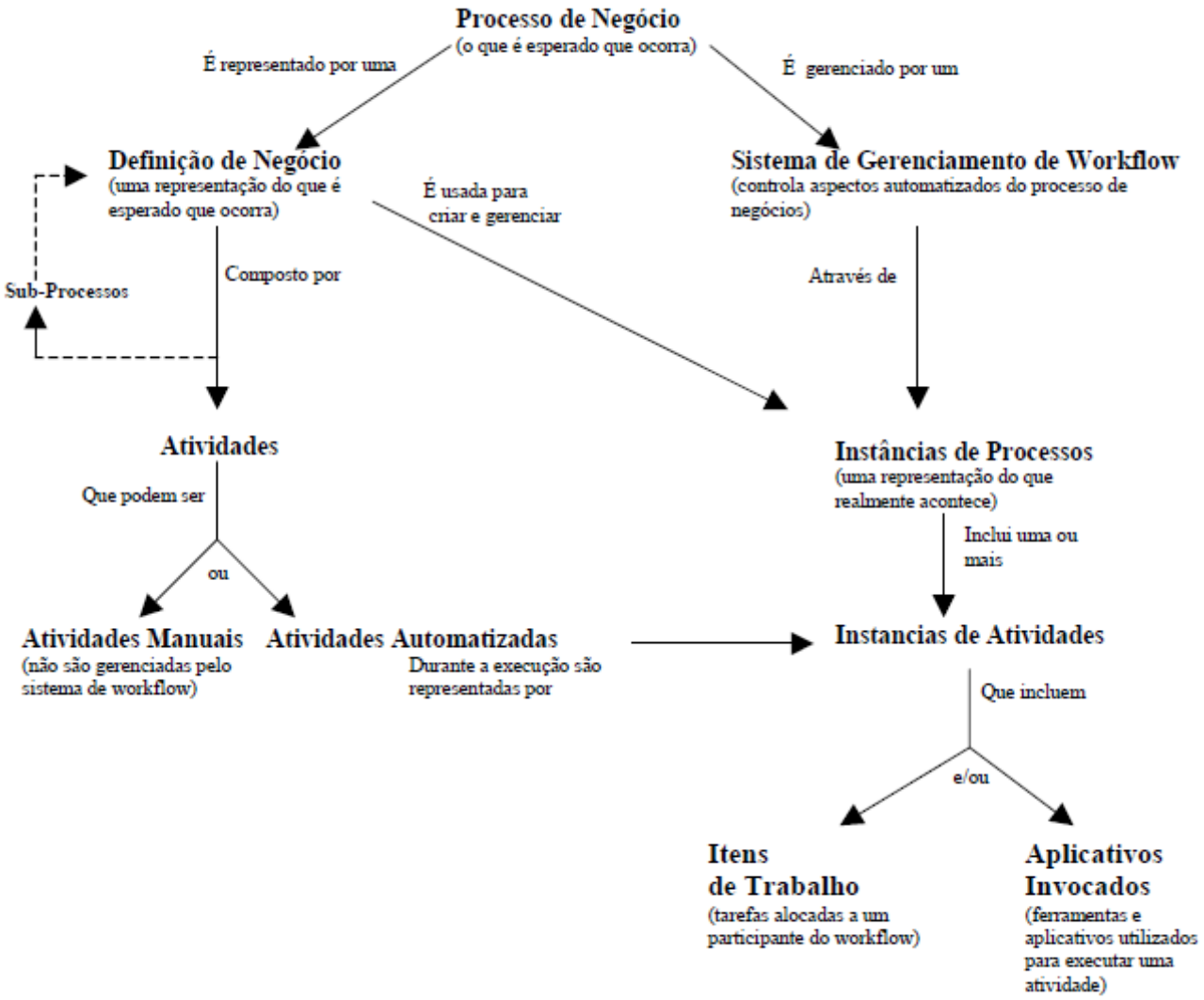


Figura 29 - Terminologia de *Workflow*.

Fonte: extraído e traduzido de WfMC (1999).

A **Figura 30** apresenta um modelo de Gestão de Processos. Nesse modelo, além das entradas próprias do processo de negócio, podem ser vistas também, as principais tarefas do gerente do processo. São elas: estabelecer os objetivos, planejar tarefas, fornecer recursos a execução das tarefas, monitorar a execução e os resultados gerados pelo processo e caso necessário, tomar ações corretivas.

Ainda de acordo com a **Figura 30,** observa-se que as atividades são divididas em duas etapas. Segundo Harmon (2004), na primeira etapa o gerente deve conduzir as atividades de iniciação do processo, isto é, definir objetivos, equipe, orçamento,

equipamentos, e por fim, planejar a sua implementação. A segunda etapa começa após o processo ter sido estabelecido. O gerente deve analisar os resultados obtidos e comparar com o planejamento inicial. Caso os resultados não estejam de acordo com o planejamento, ações corretivas devem ser tomadas.

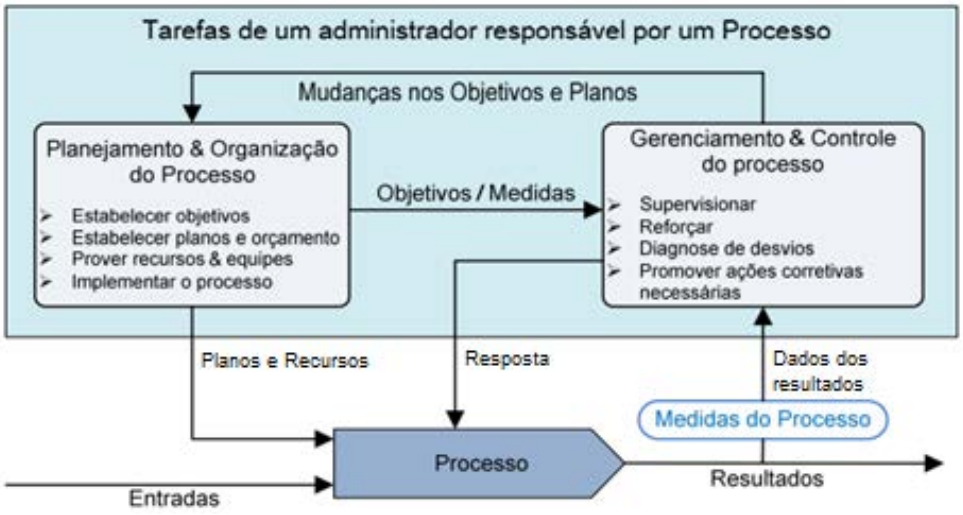


Figura 30 - Modelo básico de Gestão de Processos.

Fonte: traduzido de Harmon (2004).

Na abordagem BPM são utilizadas boas práticas de gestão, visando melhorias contínuas dos processos para se atingir os resultados esperados. Essas boas práticas incluem: mapeamento e modelagem dos processos, definição do nível de maturidade (“AS-IS”), documentação, plano de comunicação, automação do processo, monitoramento através de indicadores de desempenho e ciclo de melhorias contínuas (KHAN, 2003).

Segundo McDaniel (2001), para se compreender a implementação de BPM, é preciso entender que todos os elementos que integram uma organização (clientes, fornecedores, funcionários, sistemas, etc.) são objetos de uso em um processo de negócio e podem fazer parte do mesmo. Cada um desses elementos interagem entre si durante a execução de um processo, e executam uma ou mais funções que contribuem para o objetivo final do processo.

Na abordagem BPM, quatro atributos caracterizam e fundamentam a sua implementação em uma organização (MCDANIEL, 2001):

1) **Modelagem**: permite definir graficamente processos e tarefas que deverão ser executadas num determinado processo de negócio. Este

atributo propicia que o componente humano compreenda e preveja os resultados do processo de negócio;

2) **Integração**: através da integração dos recursos humanos e sistêmicos consegue-se obter uma maior comunicação entre estes. Isso contribui para diminuir erros que possam vir a ocorrer, na execução do processo, por falta de troca de informações;

3) **Monitoração**: a monitoração em tempo real é muito importante e deve propiciar facilidades para que um recurso humano consulte o estado dos processos, subprocessos e procedimentos, bem como as respectivas métricas;

4) **Otimização**: permite encontrar ineficiências em tempo real, estudá-las e corrigi-las, para tornar o processo de negócio cada vez mais eficaz.

Smith e Fingar (2003), por sua vez, apresentam oito fases para se implementar BPM em uma organização, descritas a seguir e ilustradas na **Figura 31**:

1) **Descoberta**: esta fase se refere a levantamentos de “como” as atividades são realizadas, podendo ser feitas manualmente, automaticamente ou combinando as duas formas. Nesta fase são “descobertos” como os processos de negócio, internos e externos, realmente funcionam;

2) **Projeto**: esta fase compreende a modelagem, manipulação e redesenho dos processos, conforme o aprendizado da organização na primeira fase (Descoberta). Todos os elementos que fazem parte do processo participam desta fase: atividades, regras, participantes, interações e relacionamentos. Esta fase inclui também a fixação de métricas para acompanhamento dos processos;

3) **Implantação**: nesta fase, o modelo de processos é distribuído a todos os participantes, visando verificação de ajustes e alterações necessárias. Novos processos podem ser adicionados ao modelo, personalizados ou até redistribuídos;

4) **Execução**: esta fase preza por garantir que todos os participantes irão desempenhar seu papel no processo: pessoas, sistemas, processos e outras organizações envolvidas;

5) **Interação**: esta fase compreende as formas de apoio à interação plena das pessoas com os processos de negócio, através de sistemas ou portais de processos,

6) **Monitoramento e Controle**: Compreende atividades para manter a execução dos processos sob controle, do ponto de vista técnico;

7) **Otimização**: o próprio Sistema de Gerenciamento de Processos utilizado pode ajudar a identificar "gargalos" ou inconsistências no processo de negócio e sugerir ajustes. Essa otimização se apoia completamente na próxima fase (Análise) e pode ser realizada com ou sem intervenção humana;

8) **Análise**: esta fase executa a medição de desempenho do processo para a fixação de métricas. Contempla a inteligência do negócio, necessária para melhorias nas estratégias organizacionais e para descobrir oportunidades de inovação.

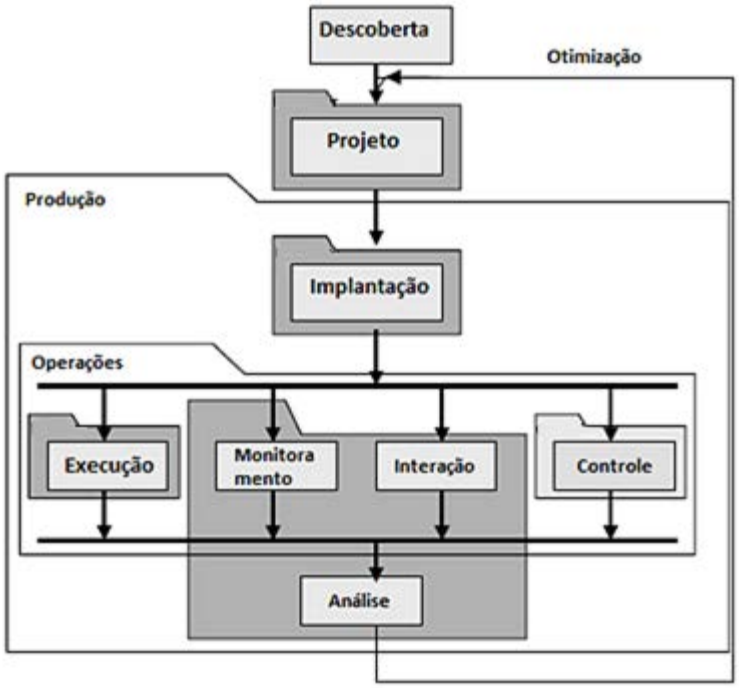


Figura 31 - Ciclo de vida do BPM, segundo Smith e Fingar (2003).

Fonte: extraído e adaptado de Smith e Fingar (2003).

Em relação à última fase, de "Análise", é interessante observar que a área de Análise de Processos de Negócios, conhecida como BPA (*Business Process*

Analysis), vem sendo intensamente estudada nos últimos anos em relação a aspectos de simulação e análise.

Para os propósitos deste trabalho, ressalta-se que o foco está na fase que inclui modelagem de processos, conforme apresentado na *seção 3.1*.

3.6 TRABALHOS RELACIONADOS À ABORDAGEM DO CAPÍTULO

Esta seção apresenta dois trabalhos envolvendo modelagem de processos de negócio, que foram selecionados por apresentarem modos de mapeamento de modelos de processos para outras representações: Francisco et al. (2009) e Ouyang, Dumas e Van Der Aalst (2009).

Ouyang, Dumas e Van Der Aalst (2009) propõem uma nova maneira de converter modelos de processos gerados em BPMN para modelos de execução na linguagem BPEL (*Business Process Execution Language*). BPEL é uma linguagem, baseada em XML, criada para representar e executar processos através de serviços Web (*Web Services*). Segundo os autores, as técnicas convencionais de conversão entre essas duas linguagens consideram um número restrito de estruturas da notação BPMN. A proposta dos autores consiste em um algoritmo que “quebre” o diagrama de processos em pequenos pedaços para serem mapeados por partes, diminuindo a complexidade do diagrama e permitindo que todos as classes possam ser mapeadas.

Já Francisco et al. (2009) apresentam uma solução para mapeamento de modelos de processos em XPDL para modelos formalizados com Rede de Petri. Segundo os mesmos autores, Rede de Petri consiste em uma técnica de modelagem gráfica de modelos de processos através de uma terminologia formal, podendo ser utilizada para validação do gerenciamento do fluxo do trabalho. Os autores objetivam viabilizar análises e simulações em cima do processo de negócio modelado. A solução conta com um *plug-in* que faz a conversão da estrutura XPDL para estruturas na linguagem PNML (*Petri Net Markup Language*), que podem ser importadas por qualquer ferramenta com suporte a Redes de Petri.

Outros trabalhos, voltados à área de Web Semântica, foram selecionados devido ao especial interesse a este trabalho: Missikoff, Proietti e Smith (2010), Haller, Gaaloul e Marmolowski (2008) e Cabral, Norton e Domingue (2009).

Missikoff, Proietti e Smith (2010) consideram que a integração entre ontologias e modelos de processos de negócio pode captar tanto as informações de

fluxo de execução quanto o conhecimento do domínio do processo de negócio. Para isso, os autores propõem um *framework* que integre as linguagens OPAL e BPAL. A linguagem OPAL (*Object, Process, Actor modelling Language*) é usada para representação de ontologias voltadas ao meio empresarial, utilizando alguns elementos essenciais para a modelagem de processos de negócio: Processos, Atores, Objetos e Ações. BPAL (*Business Process Abstraction Language*), por sua vez, representa uma linguagem capaz de capturar e descrever a lógica procedural dos modelos de processos.

Assim, um modelo de processos é gerado em BPAL. A seguir descrições formais da lógica de execução do processo são adicionadas ao modelo e processos. Uma ontologia definida em OPAL é utilizada para capturar o conhecimento do domínio, referente ao processo (atores, objetos e atividades). Por fim, os autores utilizam o conceito de "Anotação Semântica" para ligar os conceitos descritos na ontologia, aos elementos representados no modelo de processos. Segundo os mesmos autores, "Anotação Semântica" refere-se a um conceito a área de Web Semântica, que consiste em descrever recursos através do vocabulário pré-definido em uma ontologia.

Cabral, Norton e Domingue (2009) também apresentam um *framework* para apoiar a modelagem de processos de negócio. O *framework* é baseado em uma ferramenta de modelagem de processo criada pelos autores que cria, automaticamente, com base em conceitos de ontologias e "Anotações Semânticas", propriedades relacionadas ao modelo de processo. Essas propriedades adicionam um nível maior de informações ao modelo de processos. Essas informações adicionais podem ser consultadas através de mecanismos presentes no próprio *framework*. Segundo os autores, a abordagem providencia suporte a outras atividades de BPM, como a execução e análise dos modelos de processos de negócio.

Considerando que a linguagem XPDL é uma forma de representação de BPMN com XML, Haller, Gaaloul e Marmolowski (2008) fazem algumas considerações sobre o código em XPDL. Para os autores, a linguagem XPDL possui suporte limitado à representação dos aspectos e informações reais do fluxo do processo, contidos no diagrama de processos. Assim, propõem que, a partir de um modelo de processos em XPDL, seja construída uma ontologia em formato WSML (*Web Service Modeling Language*). WSML é uma linguagem criada em 2005 pelo consórcio W3C, para descrever recursos através de serviços Web (*Web Services*). O objetivo é que a semântica do modelo XPDL fique explícita no formato de ontologia e possa ser consultada para responder questões críticas sobre o processo. Os autores ainda

complementam dizendo que a ideia consiste em apenas formalizar os conceitos contidos no modelo em XPDL, sem alterar a semântica contida no modelo de processo.

Por meio dos trabalhos apresentados nesta seção, pôde-se notar que a linha de pesquisa de modelagem de processos com ontologias vem sendo amplamente adotada pelo meio acadêmico. Missikoff, Proietti e Smith (2010) e Cabral, Norton e Domingue (2009) apresentam propostas, que integram modelagem de processos e ontologias, para criar ambientes de apoio aos processos de negócio. Já o trabalho de Ouyang, Dumas e Van Der Aalst (2009) foi importante para destacar a real necessidade de se utilizar a linguagem XPDL para a conversão dos elementos da ontologia. Isso porque, as técnicas atuais de conversão de BPMN para linguagens de execução são restritas a poucos elementos do fluxo do processo.

3.7 CONSIDERAÇÕES FINAIS DO CAPÍTULO

A necessidade de compreender o negócio vem exigindo que as organizações empresariais invistam em técnicas que auxiliem no entendimento do negócio como um todo. A modelagem de processos de negócio se destaca nesse sentido, por trazer uma visão clara e organizada de todos os elementos envolvidos no domínio de processos de negócio. Dentre as suas diversas formas de representação, a notação BPMN aparece como uma tendência crescente nos ambientes empresariais. BPMN é uma notação intuitiva que possui elementos gráficos de fácil entendimento para qualquer participante envolvido com o negócio. Um segundo fator positivo, é que BPMN é o padrão adotado pela maioria das ferramentas de suporte a gestão de processos na atualidade e, portanto, com a sua utilização é possível também dar continuidade as demais fases da gestão de processos dentro da organização.

O consórcio OMG apresentou juntamente com a especificação da notação BPMN 2.0, duas novas formas de compartilhamento de modelos de processos em BPMN 2.0: XMI (*XML Metadata Interchange*) e XSD (*XML Schema Definition*). Porém, de acordo com as pesquisas realizadas sobre sistemas de modelagem de processos, a linguagem XPDL, versão 2.2, aparece como uma das mais utilizadas até o presente momento. Essa linguagem prove suporte completo à modelagem de processos em BPMN 2.0.

Os sistema de modelagem de processos apresentados neste capítulo foram escolhidos de acordo com as necessidades deste trabalho: suporte a notação BPMN 2.0; suporte a importação de arquivos em XPDL 2.2; e ferramentas de fácil uso por qualquer pessoa envolvida com o negócio.

4 METODOLOGIA DE MAPEAMENTO DE ONTOLOGIAS EMPRESARIAIS PARA MODELOS DE PROCESSOS DE NEGÓCIO

Como apresentado no *capítulo* 3, a modelagem de processos de negócio vem se destacando mundialmente por possibilitar uma visão geral da cadeia de processos, de modo transversal aos departamentos da organização, com foco nos clientes (internos e externos). Essa modelagem contribui para a análise dos processos da organização, possibilitando identificar partes que podem ser otimizadas e automatizadas. Thiry et al. (2006), entre outros autores, enfatizam a necessidade de modelagem de processos para se entender melhor o domínio abordado. Dentro do contexto empresarial, em decorrência da modelagem de processos, pode-se obter benefícios na produtividade e qualidade dos produtos, bem como na competividade no mercado. Observa-se que as linguagens utilizadas para a modelagem de processos normalmente são simples e intuitivas. Considerando que a notação BPMN, versão 2.0, pode representar a abordagem completa de modelagem de processos, e é aceita pela grande maioria das organizações empresariais, para este trabalho, ela foi escolhida para a representação dos modelos de processos gerados a partir de ontologias.

Por outro lado, como apresentado no *capítulo 2*, as ontologias empresariais apresentam uma visão organizada e uniforme das informações de uma organização, com o objetivo de facilitar o entendimento e visualização do processo de negócio como um todo. Certamente, as ontologias organizam conceitos e processos que estão presentes em documentos e no conhecimento das pessoas envolvidas. São utilizadas na Gestão do Conhecimento de uma organização, devido ao seu grande poder semântico. Contudo, há certas partes da ontologia e inter-relações que podem não se tonar tão claras aos usuários.

De modo a contribuir com os usuários e com a gestão dos processos de uma organização, é interessante que se disponibilize uma forma de visualização alternativa, para complementar a visão propiciada pelas ontologias empresariais. Essa forma alternativa proposta consiste em um modelo de processos de negócio, modelado com BPMN 2.0. Além de contribuir com a visualização do conhecimento da ontologia, o

modelo de negócios possibilita a inserção da cultura de Gestão de Processos nas empresas, que permite maior controle dos processos com foco no cliente.

Nessa direção, este capítulo apresenta uma metodologia para mapear a estrutura e conteúdo de uma ontologia empresarial em OWL, para a estrutura de uma cadeia de processos de negócio em BPMN. Uma aplicação dessa metodologia é apresentada no *capítulo 5*.

A *seção 4.1* apresenta a metodologia proposta, bem como descreve algumas de suas etapas. Contudo, as três principais etapas, para as quais foram criados padrões e métodos, serão apresentadas separadamente em outras seções. Trata-se da etapa onde é definido um método para a identificação dos elementos na ontologia, a etapa onde são definidas marcações (*tags*) para serem adicionadas no código da ontologia e da etapa de interpretação dessas marcações para a conversão para modelos de negócio. Essas três etapas estão descritas nas *seções 4.2, 4.3 e 4.6*, respectivamente.

Ressalta-se que, para a execução da etapa de identificação dos elementos na ontologia, devem ser consideradas regras de uso das propriedades “Object Property”, apresentadas na *seção 4.4*. *A seção 4.5*, por sua vez, apresenta uma etapa de verificação, que deve ser executada ao término das etapas de identificação e marcação, *seções 4.2* e *4.3*, respectivamente. As verificações têm como finalidade identificar se todos os elementos foram abordados pela metodologia. Por fim, a *seção 4.7* apresenta as considerações finais do capítulo.

4.1 METODOLOGIA PROPOSTA

Esta seção apresenta a metodologia desenvolvida para mapear uma ontologia empresarial em um modelo de processos de negócio. Esse modelo poderá ser estendido e ajustado pelas organizações, de modo que possa ser utilizado em variados propósitos, inclusive para dar continuidade à abordagem de Gerenciamento de Processos de Negócio ou BPM (*Business Process Management*).

Para uso da metodologia proposta, pressupõe-se que o usuário da metodologia tenha pleno conhecimento sobre a ontologia que será mapeada, assim como de sua estrutura. Da mesma forma, para que os modelos de processos gerados tenham sentido semântico, é importante que as ontologias representem um domínio com processos – o que normalmente é feito nas ontologias empresariais. Os

elementos relacionados aos processos serão, então, mapeados para o modelo de processos de negócio. Ainda, é importante que as relações entre dois elementos contidos na ontologia sejam obrigatoriamente indicadas através de uma propriedade "Object Property" (ver seção 2.3), sem exceções. Comentários sobre essa propriedade estão apresentados na *seção 4.4*.

Considere, então, uma ontologia empresarial X definida na linguagem OWL. A metodologia consiste em quatro etapas, que devem ser aplicadas sequencialmente:

1. Transformar a ontologia X em formato RDF/XML, gerando um arquivo identificado como "X-RDF". O arquivo "X-RDF" deve conter as relações entre os elementos da ontologia representadas por estruturas "Object Property", segundo as regras apresentadas na *seção 4.4*;
2. Transformar o arquivo "X-RDF" em um arquivo identificado como "X-Tags", com marcações padronizadas. Para isso, são definidas três subetapas, sendo que as duas primeiras devem ser feitas conjuntamente:
    2.1. Identificar os elementos da ontologia X a serem mapeados no arquivo "X-RDF", considerando o método e restrições definidas nas *seções 4.2 e 4.4*;
    2.2. Em cada elemento identificado em 2.1, devem ser adicionadas, manualmente, marcações *(tags*) em XML, de acordo com o padrão definido na *seção 4.3*;
    2.3. Verificar se os elementos pertencentes ao fluxo do processo em "X-RDF" foram marcados na etapa 2.2;
3. Converter o arquivo com as marcações "X-Tags" em um arquivo na linguagem XPDL, identificado como "X-XPDL". Para isso, utilizar a ferramenta (*parser*) *RDFtoXPDL*, desenvolvida neste trabalho, para mapear as estruturas marcadas no arquivo "X-Tags" para estruturas na linguagem XPDL 2.2;
4. Gerar um modelo de processos de negócio na notação BPMN. Para isso, um sistema que interprete a linguagem XPDL 2.2 e tenha a funcionalidade de importação de arquivos nesta linguagem deve ser selecionado. O arquivo "X-XPDL" deve, então, ser importado e interpretado, para visualização do modelo de processos de negócio.

Para a execução da **etapa 1**, deve-se utilizar uma ferramenta com suporte à linguagem OWL, para fazer a conversão do código da ontologia para o formato RDF/XML. Segundo o consórcio W3C (2012b), toda ferramenta com suporte à

linguagem OWL deve obrigatoriamente oferecer a conversão para o formato RDF/XML. O sistema Protégé, que é de uso livre e possui código fonte aberto, é um sistema bastante utilizado para essa finalidade. Sua versão mais recente é a 5.0 e poder ser encontrada no seguinte endereço: *http://protege.stanford.edu/*. Como já mencionado no capítulo 2, há diversos formatos de representação para a linguagem OWL, mas neste caso deve ser feita para o formato RDF/XML.

Para a **etapa 2**, deve-se lembrar que uma ontologia é composta por estruturas que representam as hierarquias de classes (ver *seção 2.4*). Normalmente, essas estruturas aparecem no formato RDF entre as *tags* “<Class></Class>”. No contexto deste trabalho as subetapas definidas na etapa 2 serão referenciadas pelo termo “etapas”. Na **etapa 2.1**, essas estruturas deverão ser identificadas, incluindo todos os elementos representantes das subclasses dessas estruturas. Neste trabalho, serão utilizadas as subclasses que possuam relação por meio de “Object Property”, isto é, identificadas por meio de restrições (“<Restriction></Restriction>”). Isso porque as propriedades apresentadas entre essas *tags* serão relevantes para o mapeamento das estruturas em XPDL, na etapa 3. Essas restrições serão melhor detalhadas na *seção 4.4*. Para a efetivação das marcações na **etapa 2.2**, foi definido um algoritmo, que será apresentado na *seção 4.2*. Para que o completo mapeamento das estruturas da ontologia em OWL seja feito para o formato RDF, recomenda-se verificar, após o término das etapas 2.1 e 2.2, se todos os elementos pertencentes ao fluxo do processo foram de fato contemplados pelo método de marcação (**etapa 2.3**).

As **etapas 2.1** e **2.2** são fundamentais para a metodologia e requereram o desenvolvimento de padrões e métodos. Assim, as *seções 4.2 e 4.4* apresentam todos os elementos necessários para a realização da etapa 2.1, que se refere à identificação dos elementos no arquivo RDF/XML gerado. Já a *seção 4.3* apresenta os elementos necessários para se fazer a marcação no arquivo RDF/XML considerado (etapa 2.2).

Com o arquivo em RDF já com as devidas marcações, pode-se proceder o mapeamento para as estruturas em XPDL. Para isso, foi desenvolvido um sistema de software que automatiza esse mapeamento, com a funcionalidade de um *parser*, denominado *RDFtoXPDL*. Essa ferramenta utiliza a linguagem SPARQL (*SPARQL Protocol and RDF Query Language*) para retornar os dados de interesse que estão no arquivo RDF. Como apresentado na *seção 2.5*, SPARQL é um padrão definido pelo consórcio W3C, que possibilita a realização de consultas de dados em arquivos RDF. Ao se utilizar a ferramenta *RDFtoXPDL*, um arquivo no formato XPDL será gerado, contendo todas as informações mapeadas da ontologia, de acordo com as marcações e restrições de estruturas definidas na etapa 2. A **etapa 3** consiste no uso da

ferramenta *RDFtoXPDL*, conforme será apresentado na *seção 4.6*. Entretanto, a *seção 4.7* traz as tecnologias e abordagens utilizadas para o desenvolvimento desta ferramenta.

Uma vez que a etapa 3 gerou um arquivo em XPDL, a etapa final (**etapa 4**) consiste na visualização do modelo de processos em notação BPMN. Para tal, deve ser utilizada uma ferramenta que possua a funcionalidade de importar arquivos em linguagem XPDL (versão 2.2), como, por exemplo, o sistema mencionado na *seção 3.4*: *Bizagi Process Modeler.* Quando o arquivo em XPDL for importado, o sistema utilizado já transformará o arquivo para a notação gráfica BPMN 2.0. Considerando o que está apresentado na *subseção 4.3.1*, é importante destacar o que se deve fazer caso o usuário tenha optado por exportar os "ExtendedAttribute" na etapa anterior, isto é, exportar os elementos marcados com "Extend1", "Extend2" ou "Extend3". Nesse caso, é necessário escolher uma ferramenta de importação do arquivo XPDL que tenha a opção de uso do "ExtendedAttribute" pelo usuário. Isso porque, como mencionando na *seção 3.3*, nem todas as ferramentas permitem tal uso. A ferramenta *Bizagi Process Modeler* é uma das ferramentas que permitem o uso deste atributo pelo usuário, enquanto *Cameo Business Modeler* e *BPMN Visio Modeler* reservam este atributo para uso próprio.

4.2 IDENTIFICAÇÃO DOS ELEMENTOS RDF (ETAPA 2.1)

Esta seção descreve a execução da **etapa 2.1** da etapa 2 da metodologia proposta neste trabalho. Como apresentado no *capítulo 2*, a ontologia pode ser vista como um modelo de dados usado para descrever recursos. Para isso faz uso de termos em formato de URIs (*Uniform Resource Identifiers*). Os URIs, por sua vez, têm o objetivo de preservar o verdadeiro significado do conceito representado. Porém, ao analisar os URIs de forma isolada, ou seja, apenas como identificadores únicos, não é possível identificar qual o significado de um determinado conceito dentro de seu respectivo domínio.

No exemplo da **Figura 32** é possível identificar o conceito "GPR1", porém não é possível identificar o que este conceito representa. Assim, considerando o domínio de processos de negócio, o conceito "GPR1" poderia corresponder a mais de um elemento: um processo, um resultado, um papel, entre outros. Essa identificação passa a ser mais complexa à medida que os conceitos estão relacionados, como, por exemplo: um resultado de processo sendo utilizado por um segundo processo.

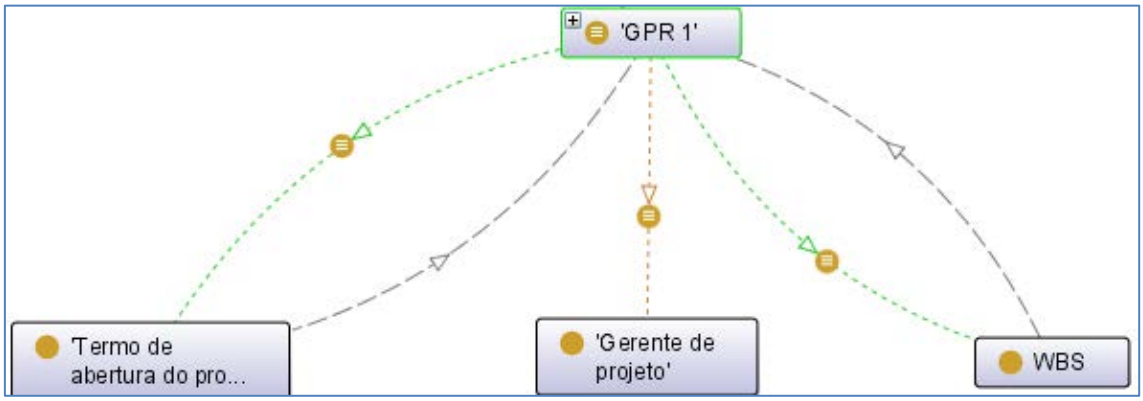


Figura 32 - Exemplo de conceitos de uma ontologia em OWL.
Fonte: extraído de Pizzoleto (2013).

Para resolver esse tipo de problema mencionado, é proposto um método que visa identificar cada um dos elementos da ontologia que deve ser mapeado para o modelo de processos. A esses elementos devem ser adicionadas marcações em formato de *tags* XML, para que tais elementos recebam significado de acordo com conceitos de processos de negócio. O método proposto para identificação dos elementos da ontologia é recursivo e está definido a seguir:

1. Identificar o elemento “X” de maior nível hierárquico a ser mapeado para o modelo de processos
2. Executar procedimento Marcar(X)
3. SE X possui elementos relacionados (Y) e sem marcação
    3.1. SE Y satisfaz Restriçoes(X,Y)
        3.1.1. Identificar elemento “Y”
        3.1.2. Fazer X = Y
        3.1.3. GO TO passo 2
    3.2. SE NÃO, GO TO passo 3
4. SE NÃO, SE X possui elemento imediatamente superior
    4.1. Fazer X = X – 1
    4.2. GO TO passo 3

O **passo 2** do método executa o procedimento “Marcar(a)”, que tem como objetivo inserir marcações no parâmetro “a”. Este procedimento corresponde a etapa 2.2 da metodologia, e é apresentado na *seção 4.3*. Já no **passo 3.1**, a função “Restriçoes(a,b)”, retorna como resultado “verdadeiro”, se o parâmetro “a” for diretamente relacionado ao parâmetro “b” através de “Object Property”. Ou seja, essa função é responsável por verificar se as restrições de relacionamento entre os elementos da ontologia estão de acordo com a metodologia. A *seção 4.4* apresenta os detalhes dessas restrições.

Ressalta-se que caso exista mais de um elemento de maior nível hierárquico na ontologia a ser mapeada (elemento “X” do **passo 1**), o método proposto deve ser repetido por completo para cada um destes elementos.

A fim de facilitar o entendimento de como deve ser a aplicação do método proposto nesta seção, a **Figura 33** apresenta seu fluxo:

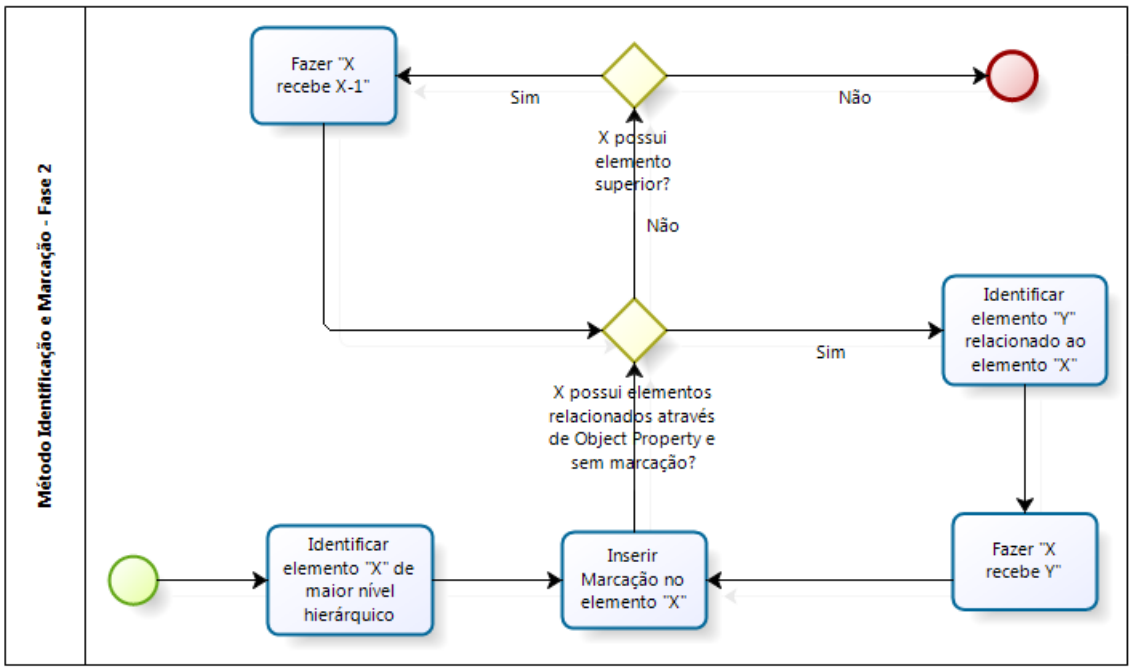


Figura 33 - Fluxo do método de identificação e marcação dos elementos na ontologia.

Observa-se na **Figura 33**, que após a execução das atividades “Identificar elemento X de maior nível hierárquico” e “Inserir Marcação no elemento X”, pela primeira vez, o método realiza uma busca em profundidade dos elementos relacionados ao elemento “X”. Sendo que, após identificar um elemento “Y”

diretamente relacionado com “X”, o elemento “Y” passa a ser o novo “X”. Dessa forma, o novo “X” receberá uma marcação, e será feita a busca de seus elementos “Y”. Ao se alcançar um elemento “X” que não possua nenhuma relação declarada em sua estrutura, ou se todos os elementos “Y” já receberam marcação, o método então retorna para o elemento imediatamente superior de “X” (elemento “X-1”).

Ressalta-se ainda, sobre a **Figura 33**, que a regra “X possui elementos relacionados através de Object property e sem marcação” restringe a identificação de elementos relacionados a “X”, apenas por meio de “Object Property”.

Para os fins deste trabalho, entende-se como “identificar um elemento” o ato de identificar manualmente no código RDF/XML da ontologia a declaração da “classe” do elemento, isto é, a estrutura “<Class></Class>” que define o elemento. Por exemplo, a **Figura 34** mostra a identificação de um “elementoA”, que também possui relação com um “elementoB”. Porém, destaca-se que a estrutura de relação através da propriedade “usa” (quarta linha), seguido do elemento com o qual “elementoA” se relaciona (“elementoB”), não é considerado para fins de identificação. Logo, para identificar o “elementoB”, deve-se encontrar a estrutura de classe que o define, como mostra a figura.

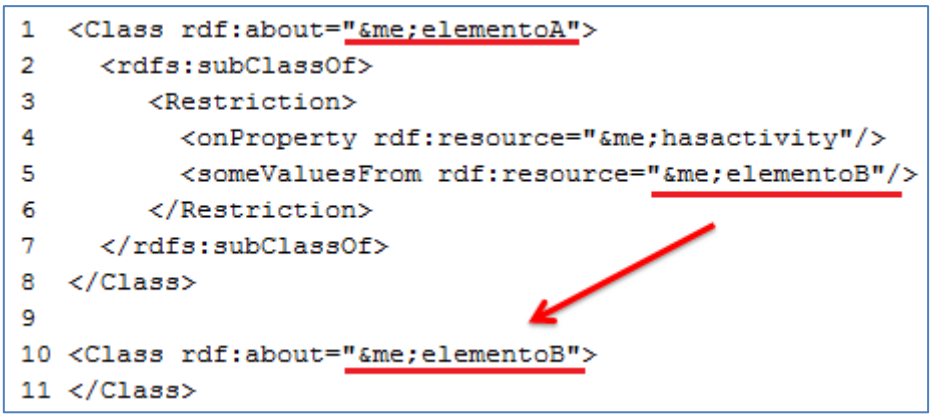


Figura 34 - Exemplo de identificação do elemento A.

Em relação ao **passo 2.2** do método proposto, entende-se como um elemento “Y” diretamente relacionado a um elemento “X”, aquele elemento “Y” que possua uma relação declarada na estrutura da classe do elemento “X”. O relacionamento entre ambos os elementos deve se dar, obrigatoriamente, através de uma “Object Property”. A *seção 4.4* apresenta um maior detalhamento sobre essa restrição. O exemplo na **Figura 34** mostra que o “elementoA” possui uma relação

direta com o “elementoB” em sua estrutura; logo o “elementoB” é diretamente relacionado com o “elementoA” e, nesse exemplo, é o único elemento relacionado com o “elementoA”.

Para o contexto deste trabalho, ”um elemento de maior nível hierárquico”, como apresentado no **passo 1**, será sempre um elemento que represente uma estrutura de “processo”. Para exemplificar, suponha que o elemento de maior nível hierárquico seja um processo denominado “Processo1”. Este deverá ser o elemento de ponto de partida para o início do método (passo 1), para que no próximo passo sejam, identificados os elementos relacionados a ele. A **Figura 35** mostra um exemplo indicando o elemento de maior nível hierárquico (“Processo1”), e todos os elementos relacionados a ele que serão lidos posteriormente, na seguinte ordem: “Tarefa1”, “Subprocesso”, “Tarefa3” e “Tarefa2”.

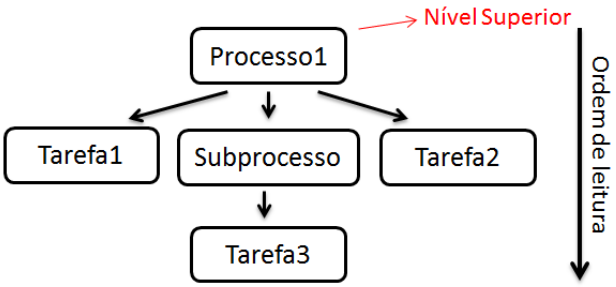


Figura 35 - Exemplo de elemento de maior nível considerado no passo 1 do método.

Como já mencionado, se a ontologia possuir mais de um elemento “de maior nível”, o método deve ser repetido desde o passo 1 para cada um desses elementos. A **Figura 36** mostra um exemplo onde existem dois elementos de maior nível hierárquico, “Processo 1” e “Processo2”. Logo, o método deverá ser repetido para cada um desses elementos e seus respectivos elementos relacionados.

Dessa forma, o primeiro passo do método proposto deve ser identificar o elemento “X” de maior nível hierárquico que deverá ser mapeado para o modelo de processos. A partir desse elemento, deve-se identificar cada um dos elementos que façam parte da sua estrutura hierárquica (passo 3) e que estejam de acordo com as restrições de estruturas impostas pela metodologia (ver *seção 4.4*). Para isso, deve ser utilizada uma abordagem de busca em profundidade. Assim, considerando o exemplo da **Figura 36**, e pressupondo que os elementos à esquerda da figura são escolhidos primeiro, a identificação dos elementos segue a seguinte ordem: 1) “Processo1”; 2) “Tarefa1”; 3) “Subprocesso”; 4) “Tarefa3”; 5) “Tarefa2”. De acordo com

o método apresentado nesta seção, caso exista mais de um elemento “X” de maior nível hierárquico na ontologia, o método deve ser replicado por completo para cada elemento “X”. Portanto, na **Figura 36**, os próximos elementos a serem identificados são: 1) “Processo2”; 2) “Tarefa1”; 3) “Subprocesso”; 4) “Tarefa2”; 5) “Tarefa3”.

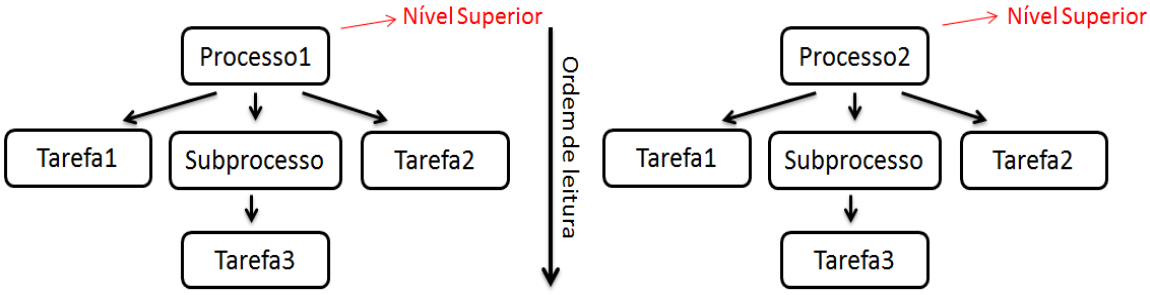


Figura 36 - Exemplo de vários elementos de maior nível.

O método apresentado nesta seção deve ser executado em conjunto com a **etapa 2.2** da metodologia apresentada na *seção 4.1*. Ou seja, para cada elemento “X” identificado pelo método, deve-se, obrigatoriamente, executar também a **etapa 2.2** da metodologia. Essa etapa consiste em adicionar marcações no código do elemento identificado e é melhor descrita na *seção 4.3*. Dessa forma, todos os elementos presentes na ontologia podem ser identificados e marcados.

Destaca-se que apenas os elementos que respeitem as restrições pré-definidas (“Object Property”) pelo método apresentado nesta seção, serão considerados no mapeamento da ontologia para o modelo de processos. Essas restrições são melhor detalhadas na *seção 4.4*.

Recomenda-se uma abordagem de identificação *top-down*, com o apoio de esquemas gráficos como os apresentados nas **Figuras 35** e **36**. Para isso podem ser utilizados diagramas de classes simplificados de UML. Esses esquemas poderiam auxiliar também na **etapa 2.3** da metodologia, apresentada na *seção 4.5.*

4.3 MARCAÇÃO NO ARQUIVO RDF/XML (ETAPA 2.2)

Esta seção descreve a execução da **etapa 2.2**. da metodologia proposta na *seção 4.1*. Essa etapa consiste na adição manual de marcações (*tags*), em formato de atributos de XML no código RDF/XML, mais especificamente nas estruturas

“<Class></Class>” de cada elemento identificado (etapa 2.1 e método da *seção 4.2*). As marcações têm dois objetivos principais: 1) possibilitar a identificação dos significados de cada conceito que está sendo lido no código da ontologia; 2) limitar os conceitos que deverão ser mapeados para o modelo de processos.

Em relação ao primeiro objetivo das marcações, elas possibilitam a identificação do significado correto do elemento, permitindo que este possa ser lido e mapeado para um único elemento correspondente na notação BPMN 2.0. Sem esse tipo de marcação se tornaria impossível identificar, por exemplo, para qual elemento de BPMN (processo, tarefa, evento, entre outros) seria mapeado um elemento “X” da ontologia no modelo de processos de negócio.

Adicionalmente, a linguagem OWL possui um conjunto grande de estruturas que podem ser utilizadas para representar conceitos. Uma mesma estrutura, no entanto, pode ser utilizada para representar todos os conceitos de forma padrão. Assim, considerando o domínio de processos de negócio, a mesma estrutura utilizada para codificar um “processo” seria utilizada também para codificar uma “tarefa do processo”, impossibilitando a distinção dos dois elementos.

Já em relação ao segundo objetivo das marcações, convém lembrar que uma ontologia pode formalizar a representação de qualquer conceito dentro de um domínio (*capítulo 2*). Porém, dentro do contexto de processos de negócio e da notação BPMN, existe um conjunto limitado e pré-definido de elementos possíveis de representação. Assim, nem todos os conceitos possíveis de representação na ontologia podem ser mapeados para o modelo de processos.

Considerando os conceitos de modelagem de processos de negócio apresentados no *capítulo 3*, foi necessário estabelecer um conjunto de elementos das ontologias que podem ser mapeados para um modelo de processos de negócios. O conjunto de elementos foi definido levando em consideração os elementos nucleares (“*core”*) da notação BPMN (*seção 3.2*). É que a partir desses elementos nucleares já é possível criar um fluxo completo do processo de negócio. Dentre esses elementos destacam-se: processos, subprocessos, atividades, papéis participantes do processo, regras para execução de uma tarefa e resultados gerados durante a execução do processo. Assim, as marcações também serão utilizadas como identificadores de correspondência para elementos da notação BPMN.

Dessa forma, foram definidas marcações em formato de atributos XML para serem adicionados manualmente ao código RDF/XML de cada elemento a ser mapeado. De acordo com o consórcio W3C (1998), um elemento em XML pode

possuir um ou mais atributos que tenham a finalidade de apresentar mais informações sobre o respectivo elemento. A **Figura 37** apresenta um exemplo, onde um atributo definido como “Mestrado” é utilizado para um elemento “estudante”, refinando todos os conceitos apresentados para “estudante”, desde que sejam todos do tipo “Mestrado”.

```
<estudante type="Mestrado">
    <elemento> Informações sobre o estudante. </elemento>
</estudante>
```

Figura 37 - Exemplo de uso de atributos em XML.

As marcações definidas são classificadas em três tipos de atributos:

1. “owl:whatType” , que identifica o significado do elemento;
2. “owl:Order”, que indica a ordem que deve ser seguida para o fluxo do processo. Deve ser utilizado em conjunto com a marcação “owl:whatType”;
3. “owl:Event”, que complementa o significado dado pela marcação “owl:whatType” a alguns elementos específicos.

Essas marcações devem ser inseridas no código RDF/XML da ontologia, dentro da estrutura “<Class>” de cada elemento a ser mapeado para o modelo de processos, como mostra a **Figura 38**. Pode-se observar que o “elementoA” possui as marcações “owl:whatType” e “owl:Order” definidas dentro da sua estrutura “<Class>”, sendo que cada uma das marcações possuirá um “Valor”, de acordo com o tipo de elemento identificado.

Para facilitar a compreensão de como utilizar essas três marcações, bem como das restrições associadas a elas, elas serão apresentadas em duas subseções separadamente, embora estejam relacionadas. As informações sobre a marcação “owl:whatType” serão apresentadas na *subseção 4.3.1*, enquanto as informações sobre as marcações “owl:Order” e “owl:Event” serão apresentadas na *subseção 4.3.2*

```
<Class owl:whatType="Valor" owl:Order="Valor" rdf:about="&me;elementoA">
</Class>
```

Figura 38 – Exemplo de inserção de marcação no código RDF/XML.

4.3.1 Marcação para expressar semântica de um elemento ("owl:whatType")

A marcação "owl:whatType" deve ser utilizada para identificar qual a correspondência do elemento "X" da ontologia para o elemento "X" na notação BPMN. Essa marcação é a única obrigatória para que o mapeamento possa ser feito e deve possuir um valor único para cada um dos "tipos" de elementos. As demais marcações são complementares a essa, e, se não forem adicionadas ao código RDF, o mapeamento assumirá um valor padrão (*default*) como será visto posteriormente.

Para a marcação "owl:whatType" há um conjunto de valores válidos que foram definidos. A definição desse conjunto levou em consideração dois fatores: (1) os possíveis elementos que um processo de negócio pode conter; (2) os elementos que podem ser representados através da notação BPMN e que podem ser encontrados e categorizados corretamente em uma ontologia. O conjunto é composto de sete valores, apresentados a seguir, considerando que se referem a determinado elemento lido no código RDF/XML que será identificado como:

- "**Process**": é a principal marcação do fluxo do processo e indica que "X" é um processo. É obrigatório que haja ao menos um processo na ontologia a ser mapeada. Levando em consideração a notação BPMN, ao se mapear esse elemento, ele também deve gerar automaticamente um elemento do tipo *Pool* de BPMN (ver *seção 3.2*);
- "**Subprocess**": indica que "X" é um subprocesso. Considerando a notação BPMN, o elemento será tratado como um subprocesso incorporado (*embedded*), ou seja, um subprocesso que possui início e fim dentro do fluxo do processo principal. Portanto, o subprocesso considerado deve, obrigatoriamente, possuir um elemento superior ao qual ele pertença, podendo ser um processo ou até mesmo outro subprocesso;
- "**Task**": indica que "X" será tratado como uma tarefa a ser executada dentro de um processo ou subprocesso. Obrigatoriamente deve possuir um elemento superior do tipo processo ou subprocesso;
- "**Result**": o elemento "X" com essa marcação será mapeado como um resultado gerado por uma tarefa, um subprocesso ou um processo. Obrigatoriamente deve possuir, como elemento superior, uma tarefa,

subprocesso ou processo. Considerando BPMN, o elemento irá indicar a criação de um “objeto de dado” por uma tarefa, processo ou subprocesso;

- “**Rule**”: esta marcação indica que ”X” será tratado como uma regra, à qual um processo, tarefa ou subprocesso deve obedecer para que sua execução seja considerada válida. Portanto, obrigatoriamente deve possuir um elemento superior do tipo tarefa, processo ou subprocesso. Em BPMN o elemento será tratado como um *gateway* do tipo exclusivo;
- “**Use**”: indica que o elemento “X” utiliza um objeto “Y” durante sua execução. Obrigatoriamente deve possuir, como elemento superior, uma tarefa ou subprocesso. Considerando BPMN, a marcação irá indicar o consumo de um objeto de dados “Y” por uma tarefa ou um subprocesso (elemento “X”);
- “**Role**”: esta marcação identifica que “X” é um ator pertencente ao fluxo do processo. Considerando BPMN, atores podem ser papéis que executam uma determinada tarefa ou subprocesso, e são representados como pistas ou vias (*Lanes*).

Sobre o valor “**Role**”, é importante que sejam observadas algumas considerações, uma vez que o método de identificação de elementos em RDF/XML (*seção 4.2*) considera apenas subprocessos do tipo *embedded*, que não possuem definição de atores no seu contexto. Assim serão válidos somente atores que possuem como elemento superior um processo ou atividade de um processo ou um subprocesso de um processo. Não serão considerados atores que possuem como elemento superior um subprocesso de subprocesso ou atividade de um subprocesso.

Ainda sobre o valor “**Role**”, lembra-se que os elementos “Subprocess” e “Task” pertencentes ao objeto “Process” possibilitam que seja declarado um ator (“Role”) dentro deles. Contudo, o método (*seção 4.2*) considera que os atores declarados nos dois tipos de elementos mencionados devem passar a ser considerados pelo elemento imediatamente superior, isto é, o elemento “Process”. Essa regra é válida apenas para esses dois elementos (“Subprocess” e “Task”) quando eles possuírem como elemento superior um elemento “Process”. Quando o elemento superior for um elemento “Subprocess”, naturalmente não possuirão suporte ao elemento “Role”, já que todos os subprocessos considerados pelo método descrito são do tipo *embedded*. Isso porque a especificação da notação BPMN não deixa claro se um subprocesso *embedded* pode ou não conter *Pools* e *Lanes* como um de seus

elementos. Na literatura, é possível encontrar diferentes opiniões sobre o assunto. Em relação às ferramentas de modelagem, também ocorre a mesma divergência: algumas permitem o uso desses elementos dentro de um subprocesso *embedded*, e outras não. Considerando esse cenário, neste trabalho não será possível definir *Pools* e *Lanes* dentro de elementos do tipo “subprocesso”.

Assim, a **Figura 39** apresenta um exemplo de declaração do código de um subprocesso “Subprocesso1”, que possui um ator “Gerente” declarado dentro dele. A **Figura 40**, por sua vez, apresenta o modelo de processos gerado pelo código da **Figura 39**. Observa-se que, ao ser mapeado para o modelo de processos, o ator “Gerente” foi considerado como sendo um valor “Role” do elemento “Processo”.

```
<Class owl:whatType="Process" rdf:about="&me;Processo">
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&me;hassubprocess"/>
            <someValuesFrom rdf:resource="&me;Subprocesso1"/>
        </Restriction>
    </rdfs:subClassOf>
</Class>

<Class owl:whatType="SubProcess" rdf:about="&me;Subprocesso1">
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&me;hasrole"/>
            <someValuesFrom rdf:resource="&me;Gerente"/>
        </Restriction>
    </rdfs:subClassOf>
</Class>

<Class owl:whatType="Role" rdf:about="&me;Gerente">
</Class>
```

Figura 39 – Exemplo de código RDF/XML da declaração de um ator dentro de um subprocesso.

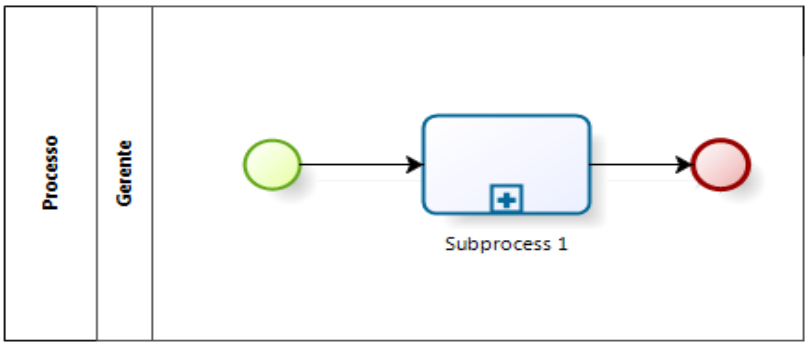


Figura 40 – Modelo de processos gerado a partir das marcações na Figura 49.

Da mesma forma que o elemento “Role”, o método proposto prevê duas tratativas para o elemento “**Result**”, de acordo com o elemento superior associado: (1) caso o elemento superior seja um valor “Task” ou “Subprocess”, o elemento “Result” deve gerar um artefato ligado a respectiva tarefa (“Task”) ou ao respectivo subprocesso (“Subprocess”); (2) caso o elemento seja declarado por um “Process”, o elemento “Result” deve gerar um resultado que pertença ao processo. No caso (2), entende-se que o elemento resultante deve ser gerado somente após a última tarefa ser executada dentro do fluxo do processo. A **Figura 41** apresenta o exemplo da declaração de código de dois resultados, de acordo com as duas possíveis tratativas para a marcação “Result”.

```
<Class owl:whatType="Process" rdf:about="&me;Process1">
   <rdfs:subClassOf>
      <Restriction>
        <onProperty rdf:resource="&me;hasactivity"/>
        <someValuesFrom rdf:resource="&me;Tarefa1"/>
      </Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
      <Restriction>
        <onProperty rdf:resource="&me;hasactivity"/>
        <someValuesFrom rdf:resource="&me;Tarefa2"/>
      </Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
      <Restriction>
        <onProperty rdf:resource="&me;hasresult"/>
        <someValuesFrom rdf:resource="&me;ResultadoB"/>
      </Restriction>
   </rdfs:subClassOf>
</Class>

<Class owl:whatType="Task" rdf:about="&me;Tarefa1">
    <rdfs:subClassOf>
      <Restriction>
        <onProperty rdf:resource="&me;hasresult"/>
        <someValuesFrom rdf:resource="&me;ResultadoA"/>
      </Restriction>
   </rdfs:subClassOf>
</Class>

<Class owl:whatType="Result" rdf:about="&me;ResultadoA">
</Class>

<Class owl:whatType="Result" rdf:about="&me;ResultadoB">
</Class>
```

Figura 41 – Exemplo de código RDF/XML com dois tipos de declaração com valor “Result”.

Observa-se na **Figura 41** que o elemento “ResultadoA” foi declarado por um elemento superior “Tarefa1” (caso (1)), enquanto o elemento “ResultadoB” foi declarado pelo elemento superior “Process1” (caso (2)). Ao se mapear o exemplo da **Figura 41**, o resultado “ResultadoA” irá continuar possuindo, como elemento superior, a “Tarefa1”. Já o “ResultadoB”, que possui, como superior, um elemento marcado como “Process”, deverá passar a ser considerado como um resultado do último passo a ser executado no fluxo do processo: o elemento “Tarefa2”, como mostra a **Figura 42**.

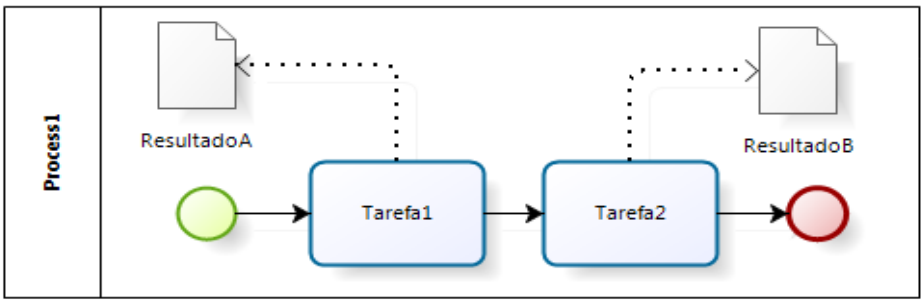


Figura 42 - Modelo de processos gerado à partir das marcações na Figura 41.

O elemento “**Rule**” também é uma exceção que deve possuir duas tratativas, de acordo com o elemento superior que o declara: (1) quando é declarado dentro de um elemento “Task” ou “Subprocess”; (2) quando o elemento “Rule” pertence ao elemento superior “Process”. No caso (1), a regra (“Rule”) declarada deve ser satisfeita por uma tarefa ou por um subprocesso que possui tal regra; se a regra não for satisfeita, o fluxo do processo deve retornar à execução, para que o elemento que possui tal regra seja executado novamente. No caso (2), a regra será considerada apenas para finalizar o fluxo do processo de negócio. Antes de finalizar o fluxo do processo, deve-se verificar se a regra foi satisfeita; em caso negativo, retornar à execução do fluxo para a última tarefa ou subprocesso pertencente ao fluxo do processo. As **Figuras 43** e **44** exemplificam esses dois casos, (1) e (2), respectivamente.

Frente ao exposto, observa-se que com os sete valores apresentados para a marcação “owl:whatType” já é possível construir um fluxo de processo completo. Porém, com o intuito de enriquecer o modelo de processos de negócio, foram definidos mais três tipos de elementos que podem ser mapeados para o modelo de processos, de forma a ajudar na compreensão do processo. Assim, há mais três valores possíveis para a marcação “owl:whatType”:

- “**Extend1**”, “**Extend2**”, “**Extend3**”: estas marcações devem ser utilizadas para indicar elementos que não fazem parte do fluxo do processo (auxiliam o entendimento do processo) e estão relacionados ao elemento “X” considerado no código RDF/XML. Obrigatoriamente, o elemento “X”, isto é, o elemento superior, deve estar marcado como “Task”, “Subprocess”, “Process”, “Rule”.

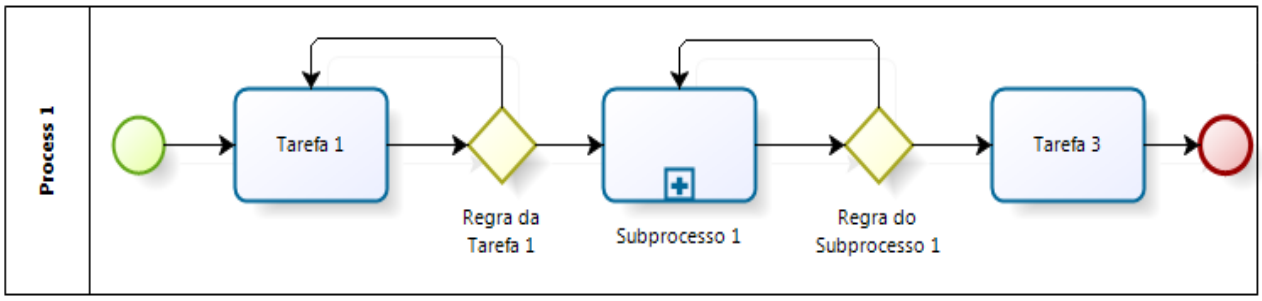


Figura 43 - Exemplo de regra pertencente à uma tarefa ou subprocesso em BPMN.

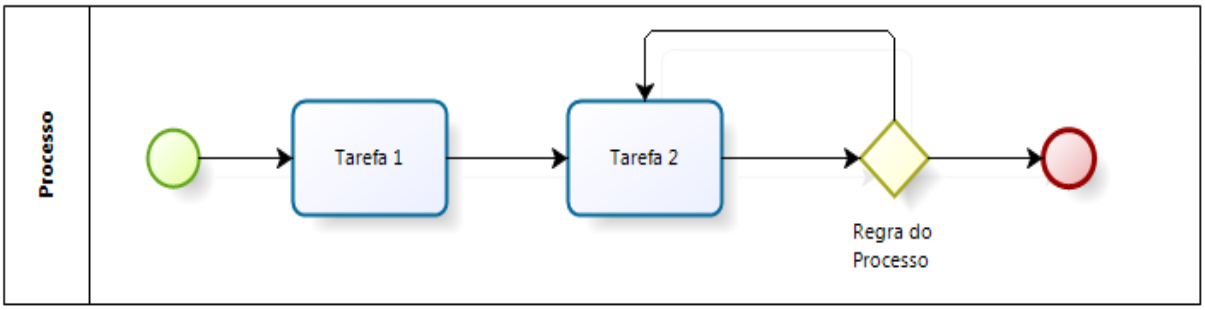


Figura 44 - Exemplo de regra pertencente à um processo.

Os três valores definidos, “**Extend1**”, “**Extend2**” e “**Extend3**”, serão utilizados como forma de descrever, com mais informações, algum outro elemento que faça parte do fluxo do processo. Servirão apenas como apoio no entendimento do modelo de processo. Observa-se que uma maior quantidade de marcações do tipo “Extend” poderiam ser definidas, porém, com o objetivo de não tornar o modelo de processos visualmente poluído, a utilização dessas marcações foi limitada a apenas três tipos. Outro ponto importante diz respeito aos elementos de ordem hierárquica superior que podem ter elementos do tipo “Extend” declarados em sua estrutura. Entende-se que os elementos “Task”, “Subprocess”, “Process” e “Rule”, por representarem atividades e regras a serem executadas durante o processo, podem precisar de informações

adicionais para a execução. Por isso, esses elementos podem receber informações “extras” no modelo de processos, através de elementos do tipo “Extend”.

É importante ressaltar que a notação BPMN não possui nenhum elemento característico para a representação dos valores “Extend1”, “Extend2” e “Extend3”, porém na linguagem XPDL, o elemento “ExtendedAttribute” é de uso livre em algumas ferramentas de modelagem de processos. Isso possibilita a utilização deste elemento para adicionar algum tipo de informação extra na modelagem. Porém, é necessário ressaltar que não são todas as ferramentas de modelagem de processos que permitem o uso deste atributo.

Destaca-se também que quase todos os valores possíveis para a marcação “owl:whatType” possuem restrições quanto ao seu uso. Com exceção do valor “**Process**”, todos os demais precisam obrigatoriamente ter como elemento superior um elemento pré-definido. A **Figura 45** mostra um exemplo, onde o “elementoA” possui relação com o “elementoB”, que por sua vez possui relação com o “elementoC”.

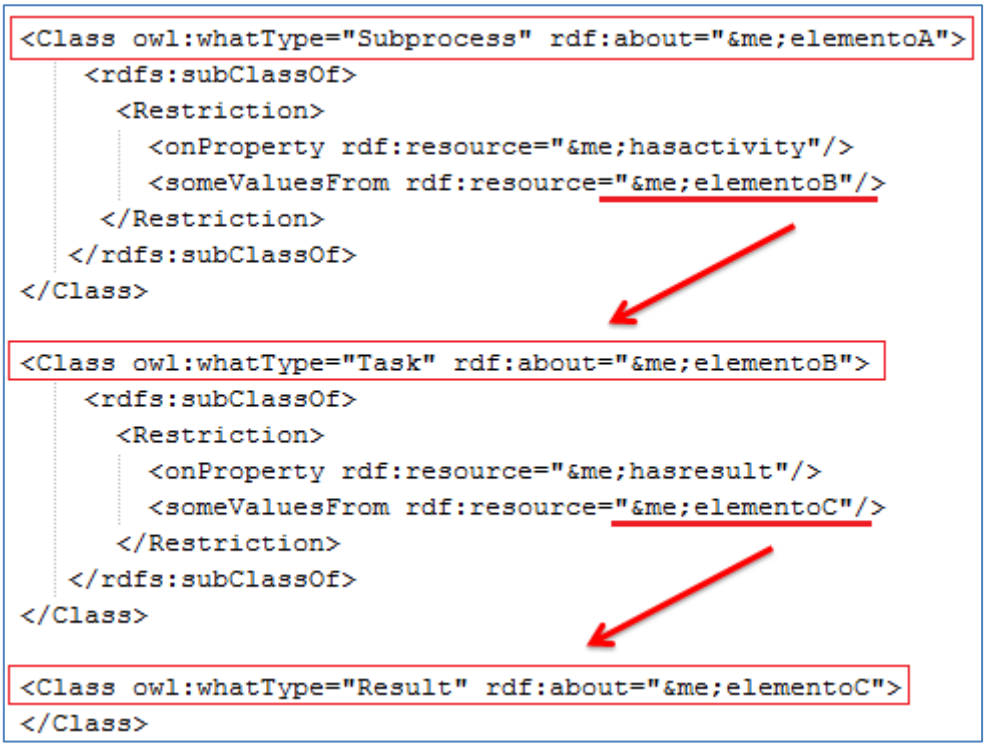


Figura 45 - Exemplo de restrições da marcação owl:whatType.

Na **Figura 45**, o “elementoC” possui valor “Result” na marcação “owl:whatType”, enquanto que seu elemento superior, “elementoB” possui valor “Task” na marcação. O “elementoB” possui o “elementoA”.como superior, marcado com o

valor “Subprocess”. Assim, tanto o “elementoC” quanto o “elementoB” possuem, como superior, um elemento com valor dentro das restrições estabelecidas; logo, ambos podem ser mapeados pela metodologia proposta (seção 4.1). Contudo, caso algum desses elementos (A, B, ou C) não respeitasse as restrições (possuir elemento superior com valor pré-definido), esse elemento não deve ser mapeado, por não possuir uma regra definida para o seu mapeamento correto.

4.3.2 Marcações “owl:Order” e “owl:Event”

A marcação “owl:Order” deve ser utilizada para identificar qual a ordem sequencial que os elementos devem respeitar dentro do fluxo do processo sendo mapeado. Deve ser utilizada em conjunto com a marcação “owl:whatType”, apresentada na subseção anterior. A marcação “owl:Order” não é obrigatória. Se não for utilizada, os elementos serão ordenados de acordo com a ordem de leitura da ontologia, porém é recomendado o seu uso.

Essa marcação pode possuir valores numéricos e superiores a zero. Somente pode ser utilizada quando a marcação “owl:whatType” possuir os valores “Process”, “Subprocess” e “Task”.

Cabe observar que as situações onde a marcação “owl:Order” pode ser utilizada foram definidas levando em consideração a notação BPMN. Assim, caso seja utilizada em conjunto com algum outro valor diferente dos citados, a marcação “owl:Order” deve ser ignorada.

Os valores da marcação “owl:Order” devem sempre seguir uma sequência dentro de um mesmo processo ou subprocesso, não podendo assumir valores repetidos. A **Figura 46** exemplifica um processo que possui duas tarefas e dois subprocessos declarados e ordenados, onde um de seus subprocessos também possui elementos ordenados.

Observa-se através da **Figura 46**, que a ordenação dos elementos pertencentes ao “Processo” segue uma sequência única, e não possui relação com a sequência dos elementos pertencentes ao elemento “Subprocesso 1”, que por sua vez também possui uma sequência única.

Em relação à marcação “owl:Event”, ela deve ser utilizada para identificar qual o tipo de artefato está sendo gerado ou consumido pelo processo de negócio. Deve ser utilizada em conjunto com a marcação “owl:whatType”. Assim como a

marcação de ordenação, “owl:Order”, a marcação “owl:Event” também não é obrigatória. Caso não seja utilizada, o padrão (*default*) a ser considerado deve ser o de valor “DataObject”.

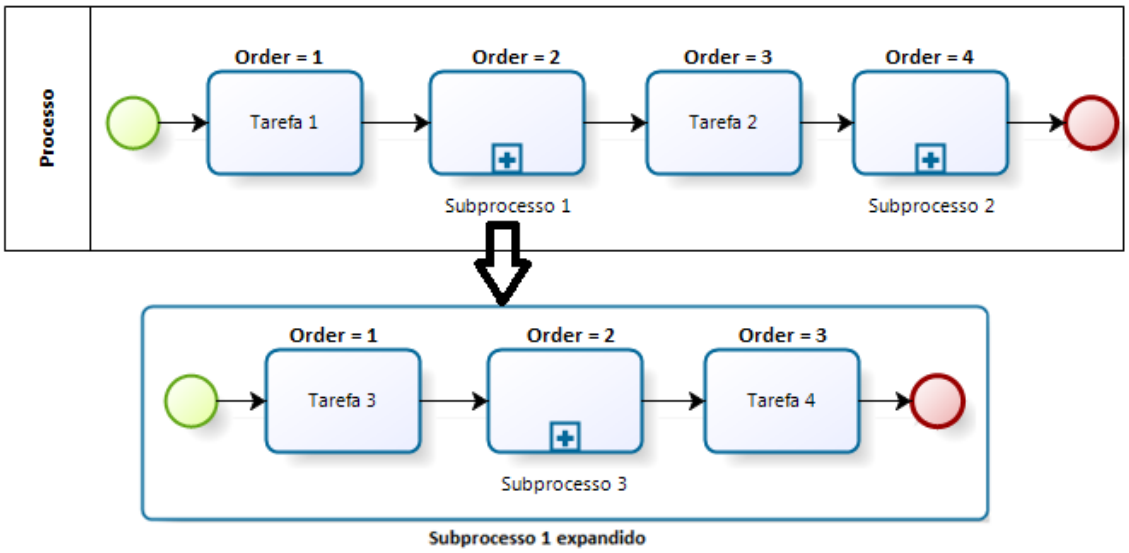


Figura 46 - Exemplo de ordenação dentro de um processo e subprocesso em BPMN.

A marcação “owl:Event” pode ser utilizada somente quando a marcação “owl:whatType” possuir os valores “Result”. Caso a marcação “owl:Event seja utilizada em conjunto com outro elemento que não possua os valores acima, deve ser ignorada. A marcação pode assumir apenas dois valores, considerando a notação BPMN:

- “**Artifact**”: indica que o elemento será considerado como sendo um objeto que não influencia no fluxo do processo, mas apenas contém informações importantes a serem consideradas pelo processo. Em BPMN será mapeado no formato de um artefato do tipo “Annotation”;
- “**DataObject**”: indica que o elemento será considerado como um objeto físico que foi gerado ou consumido dentro do fluxo do processo. Para a notação BPMN, será mapeado como um objeto de dados do tipo “Data Input” ou “Data Output”. O tipo do elemento (“Data Input” ou “Data Output”) será definido considerando a propriedade que o relaciona com o elemento superior “X” (ver *seção 4.4*).

Para facilitar o entendimento de todas as três as marcações (“owl:whatType”. “owl:Order” e “owl:Event”) e das possibilidades de combinação entre elas, com seus

possíveis valores, foi criado um diagrama, apresentado na **Figura 47,** que considera as possibilidades de combinação entre elas. As marcações “owl:Order” e “owl:Event”, por não serem obrigatórias, apenas estão indicadas na **Figura 47** nos respectivos elementos que possam contê-las.

Para exemplificar a compreensão da **Figura 47**, pode-se observar que o elemento “Task”, que possui como elemento superior “Process”, pode ser diretamente relacionado com os elementos “Rule”, “Role”, “Result”, “Use”, “Extend1”, “Extend2”, “Extend3”. Por outro lado, o mesmo elemento “Task”, com elemento superior “Subprocess”, pode ser diretamente relacionado com os elementos “Result”, “Use”, “Rule”, “Extend1”, “Extend2” e “Extend3”, não sendo permitida a relação com o elemento “Role”. Isso porque, de acordo com as restrições, apresentadas na *subseção 4.3.1*, o elemento “Task” só pode ser diretamente relacionado com o elemento “Role” quando possuir o elemento “Process” como superior.

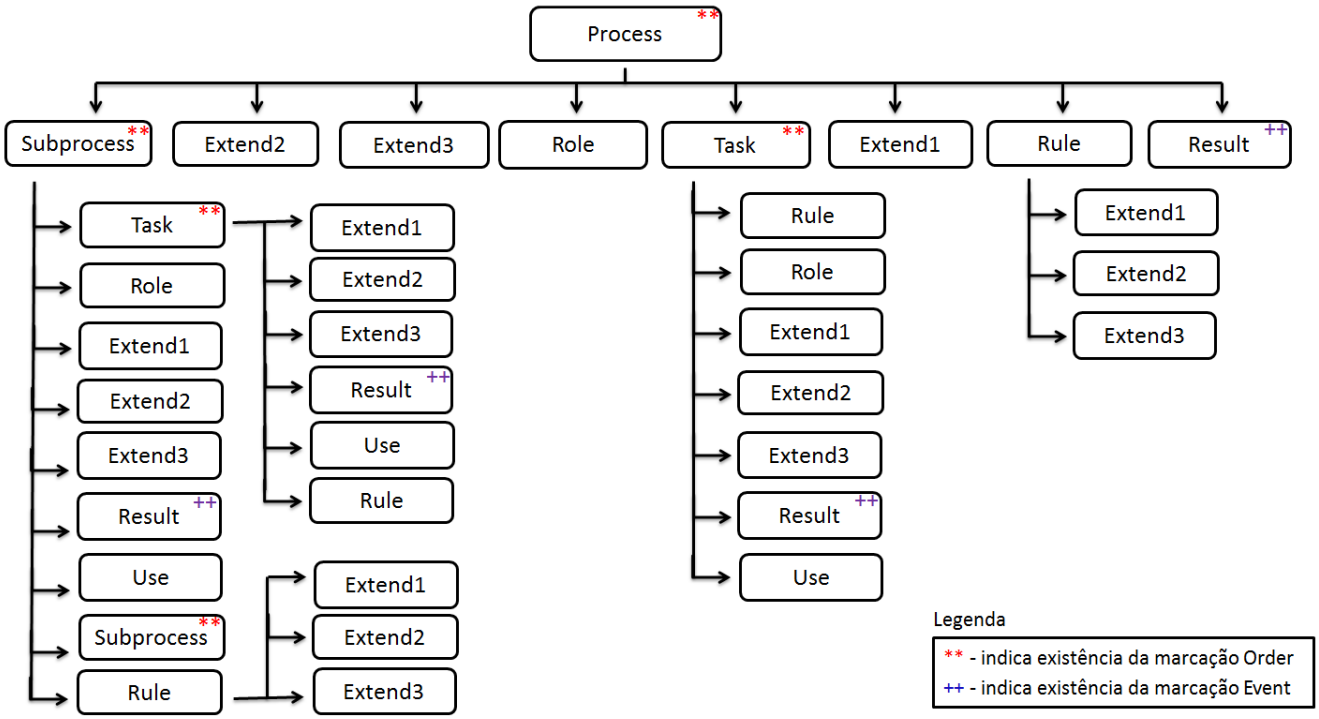


Figura 47 – Tabela com as possibilidades de marcações no código RDF/XML.

4.4 REGRAS PARA USO DE PROPRIEDADES "OBJECT PROPERTY" NO ARQUIVO RDF/XML

As ontologias possuem uma estrutura voltada para as inter-relações de seus elementos, isto é, determinado elemento "A" pode se relacionar de diversas formas diferentes com outros elementos contidos na ontologia, ou até mesmo em outras ontologias. Dessa forma, apenas com as marcações apresentadas na *seção 4.3* é impossível identificar a relação entre dois elementos da ontologia. A **Figura 48** apresenta um exemplo onde um mesmo elemento que recebeu a marcação "owl:whatType=Result" está diretamente relacionado a dois elementos. Suponha que o elemento "ClasseA" considera o elemento "ResultadoA" como um resultado de fato. Suponha também que o elemento "ClasseB" apenas utilize o resultado gerado pela "ClasseA", isto é, "ClasseB" faz uso de "ResultadoA". Ao tentar interpretar as relações de cada um dos elementos de maneira isolada, apenas com as marcações da *seção 4.3* o elemento "ResultadoA" acabaria sendo interpretado como mostra a **Figura 49**. Isso significa que a relação entre os elementos "ResultadoA" e "ClasseB" acabaria sendo interpretada erroneamente, gerando uma relação inexistente entre tais elementos.

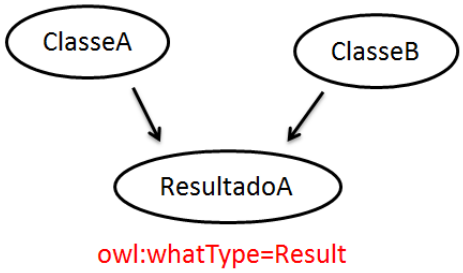


Figura 48 – Análise das relações do elemento "ResultadoA".

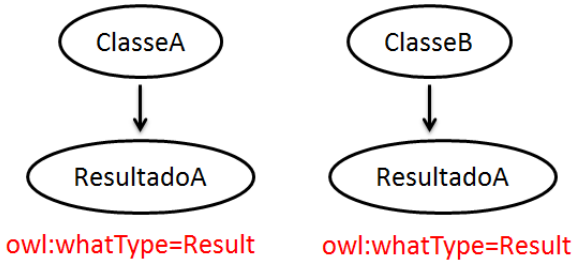


Figura 49 - Análise isolada da relação entre dois elementos sem "Object Property".

Para solucionar situações como essas, foi considerado que a linguagem OWL possibilita o uso de propriedades denominadas “Object Property”, como apresentado no APÊNDICE A. Essas propriedades são utilizadas por algumas estruturas para indicar a relação entre dois elementos. Dessa forma, considerando o uso desses elementos “Object Property” para indicar a relação entre dois elementos, a **Figura 50** mostra como é interpretada a relação entre os elementos apresentados na **Figura 49**.

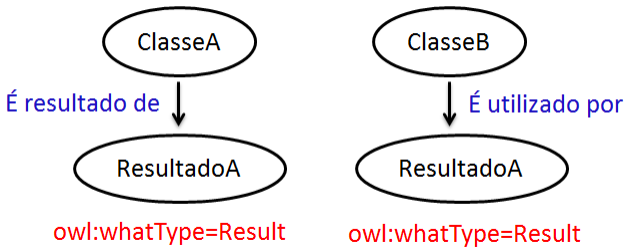


Figura 50 - Análise isolada da relação entre dois elementos com “Object Property”.

Observa-se na **Figura 50** que mesmo o elemento “ResultadoA” estando marcado como um “resultado”, a propriedade “Object Property” que o relaciona com “ClasseB” (“é utilizado por”) indica que, para o elemento “ClasseB”, “ResultadoA” deve receber uma tratativa diferente e não deve ser considerado como um “resultado”.

Dessa forma, para a completa marcação da metodologia, deve-se levar em conta o uso, obrigatório, das propriedades “Object Property” no código do arquivo RDF/XML. Elas representam, como já mencionado, a relação direta entre dois elementos da ontologia, e em conjunto com as marcações propostas na *seção 4.3*, permitirão identificar qual o tipo de elemento que está sendo lido e mapeado. É por isso que todas as relações entre dois elementos contidos na ontologia devem obrigatoriamente ser indicadas através de uma “Object Property”, sem exceção.

A **Figura 51** mostra a estrutura hierárquica que dever ser considerada na metodologia proposta (*seção 4.1*), sendo que qualquer elemento da ontologia que não siga a estrutura mencionada não poderá ser levado em consideração no mapeamento da ontologia. Pode-se observar, na **Figura 51**, que a relação entre os elementos “ClasseA” e “ClasseB” é representada por meio de uma propriedade (“Object Property”) denominada “possuirelação”.

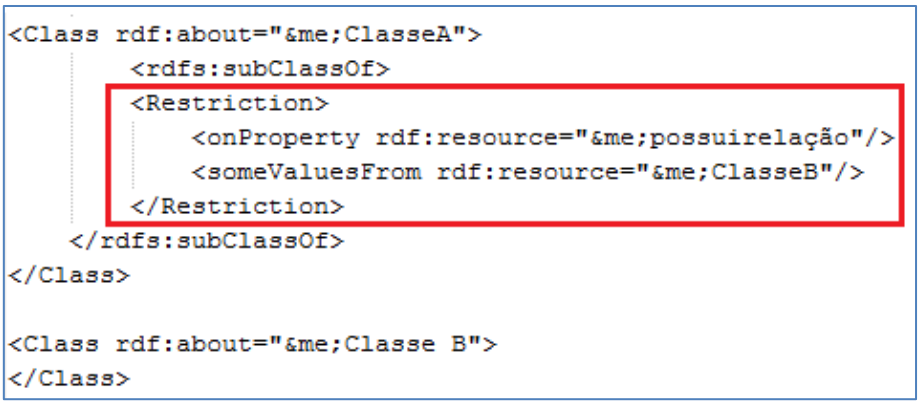


Figura 51 - Estrutura hierárquica contendo “Object Property” considerada pela metodologia.

Considerando o padrão definido pela linguagem OWL, também destacado na **Figura 51,** para a utilização de “Object Property” na metodologia proposta, é necessário que as quatro regras listadas a seguir sejam satisfeitas:

1. A propriedade “Object Property” deve sempre estar contida numa estrutura que inicia e termina com “<Restriction>”, ou seja, entre as *tags* <Restrition></Restriction>;
2. O valor da “Object Property” deve estar na estrutura “<onProperty>”;
3. Imediatamente após a estrutura “<onProperty>”, deve ser declarada uma restrição, e nela deve ser indicado com qual elemento se dá a relação. Nota-se que a declaração de restrição pode ser feita utilizando qualquer estrutura definida na linguagem OWL. Na **Figura 51**, foi utilizada a estrutura “<someValuesFrom>”;
4. Considerando a marcação “owl:whatType”, a relação entre uma determinada “Object Property” e o elemento com o qual se dá a relação deve, obrigatoriamente, ser de “um para um”. Isso significa que uma “Object Property” deve sempre referenciar, em todo o contexto da ontologia, um único valor da marcação “owl:whatType”.

A fim de padronizar a estrutura da ontologia a ser mapeada para BPMN, é proposto um conjunto de valores para serem utilizados ao se modelar uma ontologia e suas respectivas “Object Property”. Considere “X” um elemento que seja diretamente relacionado, através de “Object Property”, ao elemento “Y”. Os dez possíveis valores que a propriedade “Object Property” pode assumir, assim como os respectivos valores

da marcação “owl:whatType” à qual o elemento “Object Property” se refere, estão listados a seguir:

- “**hassubprocess**”: esta propriedade indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Subprocess”;
- “**hasactivity**”: a propriedade indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Task”;
- “**hasrole**”: a propriedade indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Role”;
- “**hasrule**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Rule”;
- “**hasresult**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Result”;
- “**hasuseresult**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Result”. Esta propriedade indica que um resultado está sendo utilizado pelo elemento “X”;
- “**hasuse**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Use”. Esta propriedade indica que um elemento qualquer “Y”, que não é um resultado gerado durante o processo sendo lido, está sendo utilizado pelo elemento “X”;
- “**hasextendone**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Extend1”;
- ‘**hasextendtwo**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Extend2”;
- “**hasextendthree**”: indica que o elemento “X” possui relação com um elemento “Y”, que, por sua vez, possui a marcação “Extend3”.

Como já citado anteriormente, cada um dos valores de “Object Property” deve, obrigatoriamente, referenciar a relação com um mesmo tipo de elemento. A **Figura 52** apresenta um exemplo do uso da “Object Property” “hassubprocess”. Observa-se na **Figura 52**, que a propriedade “hassubprocess” referencia um elemento que possui marcação “Subprocess”. Portanto, durante todo o contexto dessa

ontologia, a propriedade “hassubprocess” irá, obrigatoriamente, referenciar elementos que sejam do tipo “Subprocess”.

É interessante observar que apesar desta seção apresentar dez valores padrões para a propriedade “Object Property”, outros valores quaisquer são de livre uso, desde que sejam satisfeitas as quatro regras apresentadas anteriormente nesta seção.

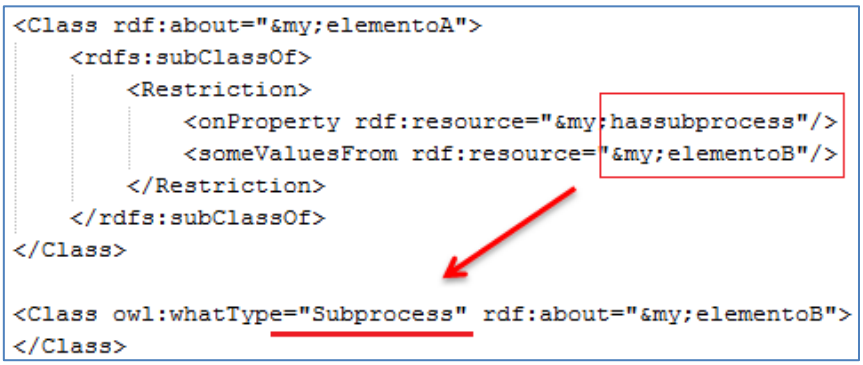


Figura 52 – Exemplo de uso da propriedade "hassubprocess".

4.5 VERIFICAÇÃO DE COBERTURA DOS ELEMENTOS (ETAPA 2.3)

Após os elementos do arquivo RDF/XML terem sido marcados nas etapas integradas 2.1 e 2.2 da metodologia proposta (*seção 4.1*), é importante verificar que todos os elementos existentes no fluxo do processo foram devidamente marcados.

Vale lembrar que, como dito na *seção 4.2*, à medida que os elementos são identificados no código RDF/XML para a devida marcação, é recomendado o uso de esquemas gráficos de apoio, como os apresentados anteriormente nas **Figuras 35** e **36**. Foi também mencionado que podem ser utilizados diagramas de classes simplificados de UML para ir acompanhando a identificação dos elementos. Todos esses recursos, se usados, podem auxiliar na etapa 2.3 da metodologia, quando se deve verificar se todos os elementos elegíveis para o mapeamento foram identificados e marcados de fato.

Essa verificação de cobertura de todos os elementos possíveis no arquivo RDF/XML deve incluir outro processo de verificação. Esse segundo processo deve verificar se a correspondência dos elementos da ontologia, seus respectivos marcadores e suas respectivas “Object Property” estão de acordo com as restrições

apresentadas nas *seções 4.3* e *4.4*. Como base, a **Tabela 9** mostra a correspondência entre cada um dos elementos possíveis de serem mapeados, de acordo com as marcações “owl:whatType” e “owl:Event”, e suas respectivas “Object Property”. A tabela mostra ainda a coluna “Elemento Superior”, em função da marcação “owl:whatType”, que mostra qual o elemento superior relacionado ao elemento sendo identificado. Os valores dessa coluna devem estar de acordo com a hierarquia de elementos proposta na *seção 4.3*. Observa-se ainda, na **Tabela 9**, que o elemento que possui a marcação “Process” não possui uma “Object Property” que o referencia. Isso acontece, pois o elemento “Process” é o primeiro elemento da hierarquia e, portanto, não existe nenhum outro elemento acima dele, que faça parte do fluxo do processo, e que possa referenciá-lo através de “Object Property”. A sigla “**N/n**” indica que para o respectivo elemento, não há necessidade de tratativa pelo processo de marcação.

Tabela 9 - Correspondência entre os elementos das marcações e de BPMN.

BPMN	**Ontologia**	**“Object Property”**	**Elemento Superior**
	“owl:whatType” (“owl:Event”)		
Processo	Process	N/n	-
Atividade	Task	hasactivity	Process
Gateways	Rule	hasrule	Process
Objeto de Dados (Saída)	Result (DataObject)	hasresult	Task
Objeto de Dados (Entrada)	Result (DataObject)	hasuseresult	Task
Objeto de Dados (Entrada)	Use	hasuse	Task
Anotação	Result (Artifact)	hasresult	Task
Fluxo de Sequência	N/n	N/n	-
Fluxo de Mensagens	-	-	-
Associação	N/n	N/n	-
Grupo	-	-	-
Pool	N/n	N/n	-
Lane	Role	hasrole	Process
Eventos (Início/Fim)	N/n	N/n	-
Subprocesso	Subprocess	hassubprocess	Process
-	Extend1	hasextendone	Task
-	Extend2	hasextendtwo	Task
-	Extend3	hasextendthree	Task

Assim, a partir da **Tabela 9**, pode-se montar uma tabela para referência cruzada de todos os elementos pertencentes ao fluxo do processo, com as marcações e “Object Property” de cada elemento. Esse recurso pode colaborar com três tipos de verificação: (1) garantia de que nenhum elemento ficou de fora do processo de marcação; (2) garantia de que todos seguem o padrão de estrutura com suas

respectivas “Object Property”; (3) garantia de que nenhum elemento recebeu marcação indevida. A **Tabela 10** mostra um exemplo de uma tabela de referência.

É importante ressaltar que apenas os elementos da notação BPMN “**Fluxo de mensagens**” e “**Grupo**” não puderam ser tratados na metodologia proposta, devido à impossibilidade de se identificar tais elementos na ontologia, conforme mostrado na **Tabela 9**, com o símbolo “-“.

Tabela 10 - Exemplo do cruzamento de informações inseridas durante o processo de marcação.

Elementos	Marcadores			Object Property	Elemento Superior
	owl:whatType	owl:Event	owl:Order		
elemento1	Process	-	1	-	-
elemento2	Task	-	1	hasactivity	Process
elemento3	Task	-	2	hasactivity	Process
elemento4	Rule	-	-	hasrule	Process
elemento5	Result	DataObject	-	hasresult	Task

Os elementos da **Tabela 9** identificados com “**N/n**” (Não necessário) não precisam fazer parte da ontologia e, portanto, devem ser tratados da seguinte forma:

- Para o elemento “**Processo**”, não é necessário possuir uma “Object Property”, pois este elemento será sempre o primeiro elemento do fluxo, ou seja, deve sempre ser o elemento de maior nível hierárquico e não pode ser referenciado por nenhum outro elemento;
- Para o elemento “**Fluxo de sequência**” não é necessário um correspondente na ontologia, pois este elemento deve ser inserido automaticamente no fluxo do processo, sempre conectando dois elementos, de acordo com a ordenação proposta pela marcação “owl:Order”;
- Para o elemento “**Associação**” não é necessário possuir um correspondente na ontologia, pois é utilizado para conectar “Anotações” ou “Objeto de Dados” a seus respectivos pares e, portanto, deve ser inserido automaticamente sempre que identificado um elemento “Anotação” ou “Objeto de Dados”;
- O elemento “**Pool**” não possui um correspondente na ontologia, pois serve de contêiner para um processo e, portanto, deve ser automaticamente inserido no

fluxo do processo sempre que identificado um elemento com a marcação “owl:whatType” e valor “Process”;

- O elemento “**Eventos (início/fim)**” não possui correspondência na ontologia, pois indica o início ou o fim de um processo ou subprocesso. Ao se identificar um elemento com a marcação “owl:whatType” e valor “Process” ou “Subprocess”, um evento do tipo “início” deve ser gerado, e, ao término, um evento do tipo ”fim” deve ser gerado.

Por fim, é importante ressaltar que, apesar da linguagem OWL permitir que sejam definidos valores relacionados à propriedade inversa das “Object Property”, mais conhecidas como “owl:inverseOf”, esses valores não estão sendo considerados neste trabalho. Essas propriedades “owl:inverseOf” representariam um fluxo inverso ao pretendido pelo processo sendo mapeado.

4.6 CONVERSÃO PARA O MODELO DE PROCESSOS (ETAPA 3)

Esta seção descreve a execução da **etapa 3** da metodologia apresentada na *seção 4.1*, que consiste na conversão de uma ontologia empresarial em formato RDF/XML, com as marcações e restrições definidas nas *seções 4.3* e *4.4*, em um arquivo na linguagem XPDL, versão 2.2.

Para a execução da etapa 3 foi desenvolvida uma ferramenta para automatizar o processo de conversão dos dados da ontologia, em um modelo de processos, em linguagem XPDL. A ferramenta é denominada *RDFtoXPDL*. Nesta seção será apresentado como se utiliza a ferramenta *RDFtoXPDL* para executar a etapa 3 da metodologia proposta. As informações sobre o seu desenvolvimento, assim como seu executável, podem ser encontradas no APÊNDICE D.

A ferramenta *RDFtoXPDL* pode ser executada nos ambientes com sistema operacional Windows Vista ou superior. Para execução da ferramenta, basta clicar sobre o arquivo “Parser.exe”. A tela principal da ferramenta será então exibida, como mostra a **Figura 53**. Clicando sobre o botão “Select File”, uma janela será aberta, possibilitando a escolha do arquivo em formato RDF da ontologia, com as devidas marcações.

Após a escolha do arquivo da ontologia, pode-se escolher mudar os valores padrões definidos para as “Object Property” propostas neste trabalho e apresentados na *seção 4.4*. Para tanto, deve-se acessar o menu “Configuration” e ir para a opção “RDF Properties”. Na janela que se abrirá, pode-se escolher um valor para cada uma das “Object Property” definidas. Como já apresentado na seção 4.4, os valores padrões definidos neste trabalho, para serem utilizados nas “Object Property”, são apenas sugestões.

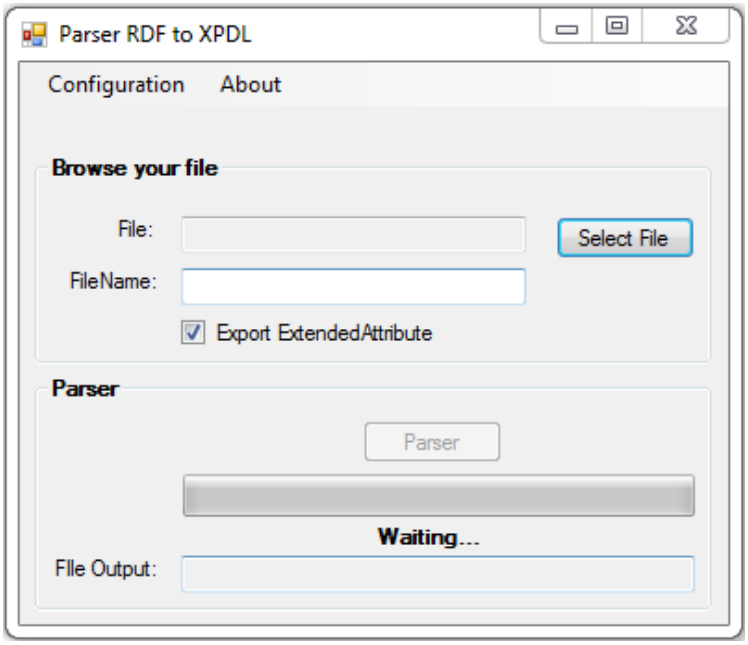


Figura 53 - Tela principal da ferramenta *RDFtoXPDL*.

Para prover maior flexibilidade para o usuário, a ferramenta também possibilita a troca dos valores definidos para as marcações “owl:whatType”, “owl:Order” e “owl:Event” definidos na *seção 4.3*. Para isso, deve ser utilizado o menu “Configuration”, apresentado na **Figura 53**, com a opção “RDF Marks”. Essa opção visa flexibilizar o uso da metodologia apresentada na *seção 4.1*, em ontologias que já apresentem um padrão parecido de marcações, mas que difiram apenas nos valores definidos neste trabalho.

Caso nenhum valor tenha sido alterado com a funcionalidade de configuração “Configuration”, a ferramenta irá assumir, por padrão (*default*), todos os valores indicados nas *seções 4.3* e *4.4*.

O último passo necessário para a realização do mapeamento para XPDL 2.2 é o de verificação do checkbox “Export ExtendedAttribute” indicado na **Figura 53**.

Recomenda-se que o usuário já tenha selecionado o sistema de interpretação de XPDL que será utilizado na etapa 4 da metodologia. Isso porque, como já mencionado, nem todos os sistemas que têm suporte à linguagem XPDL permitem o uso do atributo “ExtendedAttribute” - o que pode ocasionar erros no momento da importação do arquivo pelo sistema. Contudo, se o sistema escolhido não possuir suporte ao atributo “ExtendedAttribute”, é necessário desmarcar o checkbox “Export ExtendedAttribute”.

Observadas as informações mencionadas para uso da ferramenta *RDFtoXPDL,* deve-se clicar sobre o botão “Parser” e aguardar que a ferramenta indique o término do mapeamento. A **Figura 54** mostra que o mapeamento foi completado com sucesso. Caso tivesse tido algum problema, seria apresentado um dos seguintes erros: 1) “Erro na leitura dos dados RDF”, caso exista algum problema com os dados contidos na ontologia; 2) “Erro na conversão dos dados para a estrutura XPDL”, caso ocorra algum problema de conversão dos dados.

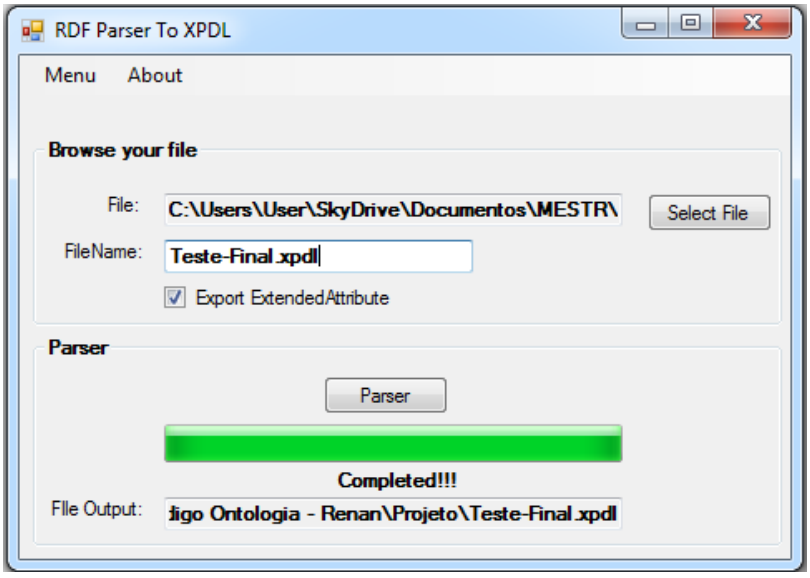


Figura 54 - Mapeamento realizado com sucesso.

Ao término do mapeamento, um arquivo em formato XPDL 2.2 será gerado no mesmo diretório onde se encontra a ontologia escolhida. O arquivo, com extensão “.xpdl” contempla, então, o modelo de processos mapeado de acordo com a ontologia. A **Figura 55** mostra parte de um arquivo gerado pela ferramenta *RDFtoXPDL*, correspondente a uma ontologia utilizada durante os testes da ferramenta. Devido à extensão do código do arquivo, apenas uma parte deste é mostrada na imagem.

Ressalta-se que a ferramenta irá gerar um único arquivo ao final do mapeamento da ontologia, sendo que mesmo que a ontologia sendo mapeada possua mais de um elemento “Process”, o arquivo XPDL irá conter todos os elementos de maior nível hierárquico que foram marcados na ontologia, assim como seus respectivos elementos relacionados. Portanto, todos os processos contidos no arquivo da ontologia serão gerados em um mesmo modelo de processos.

```
<?xml version="1.0" encoding="utf-8"?>
<Package xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
         Id="IdProcess"
         Name="ProcessDiagram"
         xmlns="http://www.wfmc.org/2009/XPDL2.2">
  <PackageHeader>
    <XPDLVersion>2.2</XPDLVersion>
    <Created>2015-01-18T15:57:24.5488281-02:00</Created>
    <ModificationDate>2015-01-18T15:57:24.5644531-02:00</ModificationDate>
    <Description>This XPDL was generated by Parser - RDF To XPDL</Description>
    <Documentation>RDF To XPDL Tool - V1.0</Documentation>
  </PackageHeader>
  <RedefinableHeader>
    <Author>Renan Stuchi</Author>
    <Version>1.1</Version>
    <Countrykey>BR</Countrykey>
  </RedefinableHeader>
  <ExternalPackages />
  <Pools>
    <Pool Id="Id1" Name="Pool1" Process="Process1" BoundaryVisible="true">
      <Lanes>
        <Lane Id="Id2" Name="Ator1" ParentPool="Pool1">
          <NodeGraphicsInfos>
            <NodeGraphicsInfo Height="200" Width="800" BorderColor="0" FillColor="0">
              <Coordinates XCoordinate="30" YCoordinate="30" />
            </NodeGraphicsInfo>
          </NodeGraphicsInfos>
        </Lane>
```

Figura 55 - Exemplo de parte de um arquivo XPDL gerado pela ferramenta *RDFtoXPDL*.

4.7 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo apresentou uma metodologia para mapeamento de ontologias para processos de negócio em BPMN. Destaca-se que a metodologia foi criada com o

intuito de ser aplicada a qualquer ontologia que represente o domínio de processos. A ferramenta *RDFtoXPDL* também foi criada de modo a suportar as estruturas mais comumente utilizadas em OWL, para relacionar dois elementos, desde que respeitadas as restrições de “Object Property”. Apesar de exigir um esforço manual moderado, as marcações apresentam uma forma de restringir os elementos a serem mapeados para o modelo de processos. Sem essas marcações, seria necessário assumir como premissa, que a ontologia utilizada na aplicação da metodologia iria conter apenas definições de elementos possíveis de mapeamento para o modelo de processos. Dessa forma, a riqueza semântica da ontologia se tornaria restrita a elementos contidos na notação BPMN 2.0.

5 APLICAÇÃO DA METODOLOGIA EM UMA ONTOLOGIA EMPRESARIAL DO MR-MPS-SW, NÍVEIS G E F

Este capítulo apresenta uma aplicação realizada seguindo a metodologia apresentada no *capítulo 4* deste trabalho.

Dentre as formas existentes para representação de processos de negócio, a forma textual é uma abordagem que gera ambiguidade e dificulta o entendimento por parte dos participantes. A representação dos processos com suas informações, ações, fluxo, artefatos, atores, por essa abordagem, acaba se tornando uma difícil tarefa, devido a complexidade nos relacionamentos de todas essas informações. Considerando-se o domínio de modelos de qualidade de processos, como, por exemplo, o MR-MPS-SW, as dificuldades aumentam, visto que as interdependências entre diferentes níveis de maturidade tornam complexo o entendimento das informações. Assim, com o intuito de facilitar e padronizar o conteúdo do modelo MR-MPS-SW, Pizzoleto (2013) criou uma forma de representação alternativa para o modelo, com foco nos níveis G e F.

Como já mencionado no capítulo 2, a ontologia contempla todos os processos dos níveis G e F do modelo MR-MPS-SW, incluindo todos os resultados gerados pelos processos. Além dos processos e seus resultados, a ontologia também apresenta as regras que os resultados dos processos devem seguir, atributos de processos (AP), sugestões inseridas pelo autor, diferenças do modelo para fábricas de software, entre outros.

Adicionalmente, o autor complementa a ontologia com conceitos do PMBOK e BSC, enriquecendo a representação da ontologia. Em alguns pontos também é possível encontrar algumas sugestões inseridas pelo autor, essas sugestões se baseiam em experiências adquiridas durante o processo de implementação e avaliação da ontologia. Essa ontologia provê uma forma de apoio para implementadores e avaliadores do modelo MR-MPS-SW.

Assim, para realizar uma aplicação da metodologia proposta neste trabalho, foi utilizada a ontologia empresarial, criada por Pizzoleto (2013), do modelo de

processos MR-MPS-SW, níveis G e F. Além de contribuir com a comunidade científica com a metodologia proposta, a aplicação da metodologia em ontologias, como a de Pizzoleto (2013), contribui para as comunidades acadêmicas e privadas que tenham interesse na implementação de modelos de qualidade.

Para a aplicação da metodologia, a ontologia do MR-MPS-SW passou por algumas modificações, tendo dois objetivos principais: 1) adequar a ontologia aos pré-requisitos da metodologia; 2) inserir mais informações na ontologia, de modo a gerar um modelo de processos mais rico. Essas alterações são mais bem detalhadas na *seção 5.1*. Assim, pôde-se iniciar com a aplicação da metodologia. Com a execução da **etapa 1** da metodologia, a ontologia que se encontrava em formato OWL foi convertida para o formato RDF/XML. Para essa conversão, foi utilizado o software Protégé (versão 4.3).

Devido às muitas estruturas existentes na ontologia, a demonstração da aplicação da **etapa 2** da metodologia, neste capítulo, será demonstrada através dos processos do nível G: “Gerência de Projeto”, considerando apenas o resultado esperado “GPR2”, e “Gerência de Requisitos”, considerando apenas o resultado esperado “GRE4”. A *seção 5.2* apresenta a aplicação dessa etapa. Ressalta-se que a mesma mecânica aplicada para esses processos foi replicada para os outros cinco processos restantes: “Garantia da qualidade”, “Gerência de portfólio de projetos”, “Gerência de configuração”, “Medição” e “Aquisição” e seus respectivos resultados esperados. Dessa forma, a **etapa 3** pôde ser executada, utilizando-se da ferramenta *RDFtoXPDL*. Para a **etapa 4** foi escolhido a ferramenta *Bizagi Modeler* para a visualização do modelo de processos gerado. A execução dessas duas últimas etapas é descrita na *seção 5.3*.

Nesse contexto, a *seção 5.1* traz as adequações realizadas na ontologia para a aplicação da metodologia. A *seção 5.2* apresenta a execução da etapa 2, enquanto a *seção 5.3* apresenta a execução das etapas 3 e 4. Por fim, a *seção 5.4* traz as considerações finais do capítulo.

5.1 ADEQUAÇÃO DA ONTOLOGIA

De acordo com estudos realizados nos guias do MR-MPS-SW, com foco no processo “Gerência de Requisitos”, foi identificado que um maior número de elementos poderia ser acrescentado à ontologia, de modo a torná-la mais detalhada e

possibilitar a geração de um modelo de processos com maior nível de detalhes. Ressalta-se que os elementos inseridos na ontologia baseiam-se no guia do MR-MPS-SW e, portanto, não possuem caráter de definição de “como” o processo deve ser implementado. Dessa forma, o processo “Gerência de Requisitos” passou pela seguinte reformulação com a decomposição de alguns objetivos em atividades detalhadas:

- Foi adicionado ao processo o papel de “Analista de Requisitos”;
- GRE1 foi decomposto nas atividades: “Entrar em contato com cliente”, “Levantar Requisitos”, “Documentar requisitos”;
- GRE2 foi decomposto nas atividades: “Avaliar requisitos com equipe técnica”, “Avaliar requisitos com cliente”;
- GRE3 foi decomposto nas atividades: “Gerar rastreabilidade Requisitos X Produtos de trabalho”;
- GRE4 foi decomposto nas atividades: “Revisar requisitos e elementos do projeto”, “Registrar inconsistências”, “Corrigir inconsistências”;
- GRE5 foi decomposto nas atividades: “Registrar mudanças nos requisitos”.

Dessa forma, o processo “Gerência de requisitos” foi reformulado, considerando algumas atividades básicas para se chegar aos resultados esperados. Ressalta-se que as estruturas já existentes na ontologia não sofreram nenhuma alteração.

Antes de aplicar a metodologia de mapeamento, foram necessárias duas alterações na estrutura da ontologia: 1) todos os processos e seus resultados esperados não estavam relacionados através de “Object Property”; 2) uma mesma “Object Property” referenciava mais de um tipo de elemento. Ambas as alterações foram necessárias para adequar a ontologia aos pré-requisitos da metodologia (ver capítulo 4). Portanto, a primeira alteração consistiu em relacionar todos os processos e seus resultados esperados através de “Object Property”. Como já mencionado neste capítulo, para a demonstração da aplicação da metodologia foram escolhidos os processos “Gerência de projetos” e “Gerência de requisitos”. Dessa forma, para exemplificar as alterações descritas nesta seção também utilizaremos os mesmos elementos. Assim, foi necessário rever a estrutura de relacionamento entre os elementos “Gerência de projetos” e seu resultado “GPR2”, e “Gerência de requisitos” e seu resultado “GRE4”, para passarem a utilizar “Object Property”.

Na ontologia utilizada nesta aplicação, o elemento “GPR2” representa o resultado “GPR2”, referente ao processo de “Gerência de Projetos”, do nível G do modelo MR-MPS-SW e, portanto, pode possuir uma série de atividades que o definam. Dessa forma, o elemento foi tratado como sendo um “Subprocess” do elemento “Gerência de Projeto”, e para o relacionamento entre ambos foi utilizada a propriedade (“Object Property”) “hassubprocess”. A **Figura 56** mostra a estrutura do elemento “Gerência de Projeto” já alterada para se relacionar com o elemento “GPR2” por meio de “Object Property”.

```
<Class rdf:about="&Ontology1339617941597;GerenciaDeProjeto">
   <rdfs:label xml:lang="pt">Gerência de projeto</rdfs:label>
   <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Processo"/>
   <rdfs:subClassOf>
      <Restriction>
         <onProperty rdf:resource="&Ontology1339617941597;hasFS"/>
         <someValuesFrom rdf:resource="&Ontology1339617941597;DiferencaEscopo"/>
      </Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
      <Restriction>
         <onProperty rdf:resource="&Ontology1339617941597;hassubprocess"/>
         <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
      </Restriction>
   </rdfs:subClassOf>
```

Figura 56 - Estrutura do elemento "Gerência de Projeto" adequada com Object Property.
Fonte: extraído de Pizzoleto (2013).

Ressalta-se que essa alteração precisou ser realizada em todos os processos existentes na ontologia e seus respectivos resultados. Dessa forma, a mesma alteração foi replicada para os processos de: “Gerência de Requisitos”, “Garantia de qualidade”, “Aquisição”, “Medição”, “Gerência de Portfólio” e “Gerência de Configuração”, incluindo todos os seus resultados esperados.

Outro ponto que precisou ser revisado, diz respeito às “Object Property” utilizadas por Pizzoleto (2013). Uma mesma “Object Property” era utilizada para referenciar diferentes tipos de elementos, o que não pode ocorrer segundo a metodologia proposta neste trabalho (ver *seção 4.4*). A **Figura 57** apresenta a estrutura do elemento “GPR2” presente na ontologia. Observa-se que o “GPR2” possui relação com o elemento “DimensaoDaTarefa”, através da propriedade “hasresult”.

```
<Class rdf:about="&Ontology1339617941597;GPR2">
    <rdfs:label xml:lang="pt">GPR 2</rdfs:label>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;haspurpose"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;DecomporEscopo"/>
        </Restriction>
    </equivalentClass>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasresult"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;DimensaoDaTarefa"/>
        </Restriction>
    </equivalentClass>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasFS"/>
            <someValuesFrom
            rdf:resource="&Ontology1339617941597;DimensaoDeTarefaMaisConcreta"/>
        </Restriction>
    </equivalentClass>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;GerenciaDeProjeto"/>
</Class>
```

Figura 57 - Estrutura do elemento “GPR2”.
Fonte: extraído de Pizzoleto (2013).

Na ontologia utilizada nesta aplicação, a propriedade “hasresult” é utilizada para indicar que o elemento “GPR2” possui relação com um resultado de processo gerado. Porém, ao analisar a estrutura do elemento “DimensaoDaTarefa”, na **Figura 58**, observa-se que existe uma relação com o elemento “GPR2” através da mesma propriedade “hasresult” utilizada na **Figura 57**. Ou seja, o elemento “GPR2” aparece como um resultado gerado por “DimensaoDaTarefa” gerando uma interpretação errônea dos elementos, como demonstra as triplas na **Tabela 11**. Para adequar as estruturas às premissas da metodologia, foi necessário realizar alterações na estrutura do elemento “DimensaoDaTarefa”, utilizando-se da propriedade “inverseOf” de OWL, tornando assim a estrutura deste elemento adequada a aplicação da metodologia.

A **Figura 59** mostra como ficou a estrutura alterada do elemento “DimensaoDaTarefa”, onde a propriedade “isresultof” foi utilizada para indicar que o elemento “DimensaoDaTarefa” é um resultado gerado por “GPR2”. É importante ressaltar que essa alteração precisou ser replicada para todos os elementos da ontologia, que possuíam relação através de “Object Property”, e que faziam parte do escopo do mapeamento.

```
<Class rdf:about="&Ontology1339617941597;DimensaoDaTarefa">
    <rdfs:label xml:lang="pt">Dimensionamento da tarefa</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;ProdutoTrabalho"/>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasresult"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
        </Restriction>
    </rdfs:subClassOf>
</Class>
```

Figura 58 - Estrutura do elemento "DimensaoDaTarefa".

Fonte: extraído de Pizzoleto (2013).

Tabela 11 - Estruturas em formato de triplas.

Sujeito	Predicado	Objeto
GPR2	hasresult	DimensaoDaTarefa
DimensaoDaTarefa	hasresult	GPR2

```
<Class rdf:about="&Ontology1339617941597;DimensaoDaTarefa">
    <rdfs:label xml:lang="pt">Dimensionamento da tarefa</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;ProdutoTrabalho"/>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;isresultof"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
        </Restriction>
    </rdfs:subClassOf>
</Class>
```

Figura 59 - Estrutura do elemento “DimensaoDaTarefa” corrigida.

Fonte: extraído de Pizzoleto (2013).

5.2 APLICAÇÃO DA ETAPA 2 DA METODOLOGIA

Para a demonstração da aplicação da etapa 2, neste capítulo, foram selecionados os processos “Gerência de projetos”, com seu resultado “GPR2” e “Gerência de requisitos”, com seu resultado “GRE4”. As **Figuras 60** e **61** apresentam uma visão gráfica das estruturas do “GPR2” e “GRE4”, respectivamente.

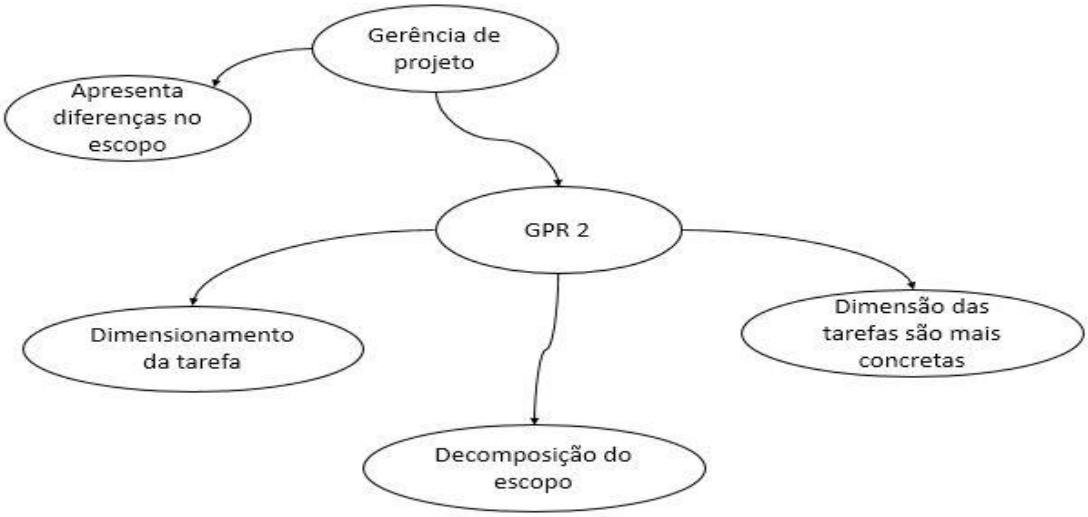


Figura 60 - Visão gráfica da estrutura GPR2.
Fonte: extraído de Pizzoleto (2013).

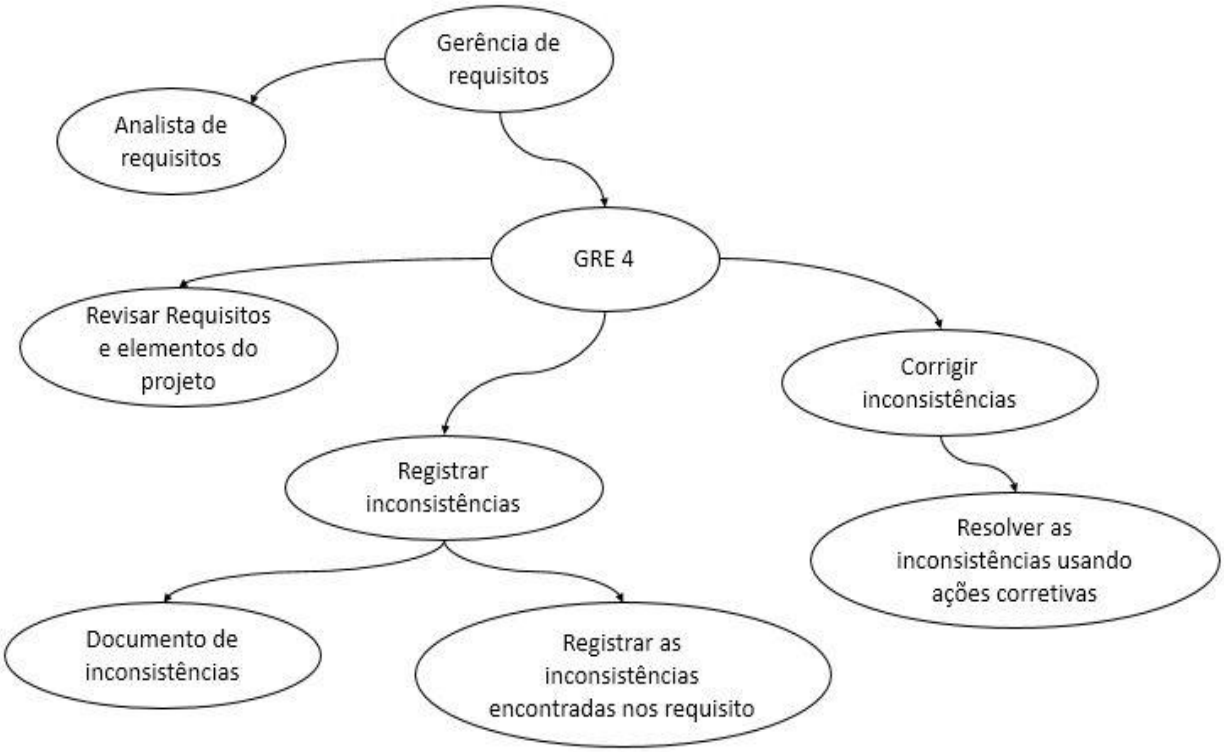


Figura 61 - Visão gráfica da estrutura GRE4.
Fonte: extraído de Pizzoleto (2013).

Dessa forma, pôde-se dar início a execução do método proposto na **etapa 2.1** da metodologia**.** É importante ressaltar que, como apresentando na *seção 4.2*, para

cada elemento identificado na etapa 2.1, deve-se obrigatoriamente executar também a **etapa 2.2** (marcação do elemento).

Portanto, para a execução do **passo 1** do método, foi escolhido iniciar pelo processo “Gerência de Projetos”. Assim, a estrutura do processo foi identificada e recebeu a marcação “owl:whatType=Process”, como mostra a **Figura 62**. Ressalta-se que a figura apresenta apenas uma parte da estrutura do elemento “GerênciaDeprojeto”. Esse elemento, de acordo com o modelo MR-MPS-SW, representa o processo “Gerência de Projetos”, e possui um total de dezenove resultados esperados (RAP) ligados a ele.

```
<Class owl:whatType="Process" rdf:about="&Ontology1339617941597;GerenciaDeProjeto">
    <rdfs:label xml:lang="pt">Gerência de projeto</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Processo"/>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasFS"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;DiferencaEscopo"/>
        </Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hassubprocess"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
        </Restriction>
    </rdfs:subClassOf>
```

Figura 62 - Estrutura do elemento "Gerência de projeto".
Fonte: extraído de Pizzoleto (2013).

O próximo passo (**passo 2.1**) foi a identificação dos elementos diretamente relacionados ao elemento “Gerência de projeto”. Observa-se na **Figura 62** a relação com dois elementos, “DirerencaEscopo” e “GPR2”, onde optou-se por iniciar pelo elemento “DiferencaEscopo”. Na ontologia utilizada nesta aplicação, a propriedade “hasFS”, que liga o elemento “GerenciaDeProjeto” ao “DiferencaEscopo”, indica uma relação de diferença na implementação do processo para organizações do tipo “Fábrica de Software”. Ou seja, á uma relação que não é possível de ser representada no modelo de processos e, portanto, o elemento “DiferencaEscopo” foi identificado e marcado como um “Extend1”, de forma a apenas enriquecer o modelo de processos. A **Figura 63** apresenta a estrutura do elemento já com a marcação.

Observa-se ainda na **Figura 63** que o elemento “DiferencaEscopo” não possui nenhum outro elemento relacionado a ele. A única relação existente na estrutura do elemento, se dá por meio da propriedade “inversa”, que volta para o elemento “GerenciaDeprojeto”, e que segundo a metodologia, não deve ser considerada. Ressalta-se também, mesmo que o elemento “DiferencaEscopo” tivesse outros elementos diretamente relacionados a ele, estes não seriam considerados no mapeamento, pois de acordo com as definições apresentadas na *seção 4.3*, elementos com marcação “Extend1” não podem possuir nenhum outro elemento relacionado a ele.

```
<Class owl:whatType="Extend1" rdf:about="&Ontology1339617941597;DiferencaEscopo">
    <rdfs:label xml:lang="pt">Apresenta diferenças no escopo</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;FabricaSoftware"/>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;isFSof"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GerenciaDeProjeto"/>
        </Restriction>
    </rdfs:subClassOf>
</Class>
```

Figura 63 - Estrutura do elemento "DiferencaEscopo" com marcação.
Fonte: extraído de Pizzoleto (2013).

Portanto, dando continuidade ao método de identificação de elementos (**passo 2.1**), deve-se voltar ao elemento “GerenciaDeprojeto” e analisar o próximo elemento diretamente relacionado a ele, isto é, o elemento “GPR2”. De volta a **Figura 62**, observa-se que o elemento “GPR2” está relacionado através da propriedade “hassubprocess”, que na respectiva ontologia indica que o elemento “GPR2” é um subprocesso de “GerenciaDeprojeto”. Assim, a estrutura do elemento “GPR2” foi identificada e marcada com o valor “Subprocess”, como mostra a **Figura 64**.

Observa-se ainda na **Figura 64** que, o elemento recebeu a marcação de ordenação “owl:Order=2”, isto é, será o segundo elemento a aparecer no fluxo do processo. O passo seguinte foi identificar todos os elementos diretamente relacionados ao “GPR2”. O primeiro elemento escolhido foi “DecomporEscopo” que está relacionado através da propriedade “haspurpose”. Essa propriedade indica que o “GPR2” possui como objetivo o elemento “DecomporEscopo”. Esse tipo de elemento não possui uma representação característica no modelo de processos e, portanto, foi

marcado como “Extend2”. Observa-se que a marcação “Extend1” já foi utilizada anteriormente para referenciar outro tipo de elemento, através da “Object Property” “hasFS”, e por isso, essa mesma marcação não pode ser utilizada para o elemento “DecomporEscopo”. A **Figura 65** mostra a estrutura marcada desse elemento.

```
<Class owl:whatType="Subprocess" owl:Order="2" rdf:about="&Ontology1339617941597;GPR2">
    <rdfs:label xml:lang="pt">GPR 2</rdfs:label>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;haspurpose"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;DecomporEscopo"/>
        </Restriction>
    </equivalentClass>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasresult"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;DimensaoDaTarefa"/>
        </Restriction>
    </equivalentClass>
    <equivalentClass>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;hasFS"/>
            <someValuesFrom
            rdf:resource="&Ontology1339617941597;DimensaoDeTarefaMaisConcreta"/>
        </Restriction>
    </equivalentClass>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;GerenciaDeProjeto"/>
</Class>
```

Figura 64 - Estrutura do elemento "GPR2".

Fonte: extraído de Pizzoleto (2013).

```
<Class owl:whatType="Extend2" rdf:about="&Ontology1339617941597;DecomporEscopo">
    <rdfs:label xml:lang="pt">Decomposição do escopo</rdfs:label>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;ispurposeof"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
        </Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Objetivo"/>
</Class>
```

Figura 65 - Estrutura do elemento "DecomporEscopo".

Fonte: extraído de Pizzoleto (2013).

Assim, de acordo com as definições das marcações (ver *seção 4.3*), elementos do tipo “Extend2” não devem possuir elementos relacionados. Portanto, deve-se retornar ao elemento “GPR2” e identificar o próximo elemento relacionado a ele.

Dessa forma, o próximo elemento identificado foi o “DimensaoDaTarefa”, relacionado ao “GPR2” através da propriedade “hasresult”, que indica a geração de um resultado. Portanto, o elemento “DimensaoDaTarefa” recebeu a marcação “owl:whatType=Result”. De acordo com o modelo MR-MPS-SW, o resultado esperado “GPR2” consiste em uma estimativa de tamanho das funcionalidades e, portanto, entende-se que um artefato contendo estimativas deve ser produzido nesta etapa. O elemento “DimensaoDaTarefa” representa um resultado físico gerado, e por isso, recebeu a marcação de “owl:Event=DataObject”. A **Figura 66** mostra a estrutura marcada do elemento.

```
<Class owl:whatType="Result" owl:Event="DataObject"
       rdf:about="&Ontology1339617941597;DimensaoDaTarefa">
    <rdfs:label xml:lang="pt">Dimensionamento da tarefa</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;ProdutoTrabalho"/>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;isresultof"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
        </Restriction>
    </rdfs:subClassOf>
</Class>
```

Figura 66 - Estrutura do elemento "DimensaoDaTarefa".
Fonte: extraído de Pizzoleto (2013).

De acordo com as definições apresentadas na *seção 4.3*, elementos do tipo “Result” não devem possuir nenhum elemento diretamente relacionado, portanto, deve-se retornar ao elemento “GPR2” e continuar com a identificação dos próximos elementos relacionados a ele.

O elemento “DimensaoDeTarefaMaisConcreta” foi o próximo e último elemento identificado, relacionado ao “GPR2”. Esse elemento está relacionado através da propriedade “hasFS”, que como já explicado anteriormente, representa a relação com um elemento que não pode ser representado no modelo de processos. Ressalta-se também, que em passos anteriores desta aplicação, um elemento que possuía

relação através da mesma propriedade (“hasFS”) recebeu a marcação “Extend1”. Portanto, como o elemento “DimensaoDeTarefamaisConcreta” utiliza mesma “Object Property”, ele pode receber a marcação “Extend1”. A **Figura 67** demonstra a estrutura do elemento “DimensaoDeTarefaMaisConcreta” com a marcação.

```
<Class owl:whatType="Extend1"
    rdf:about="&Ontology1339617941597;DimensaoDeTarefaMaisConcreta">
    <rdfs:label xml:lang="pt">Dimensão das tarefas são mais concretas</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;FabricaSoftware"/>
    <rdfs:subClassOf>
        <Restriction>
            <onProperty rdf:resource="&Ontology1339617941597;isFSof"/>
            <someValuesFrom rdf:resource="&Ontology1339617941597;GPR2"/>
        </Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <Class>
            <intersectionOf rdf:parseType="Collection">
               <Restriction>
                  <onProperty rdf:resource="&Ontology1339617941597;hasFS"/>
                  <someValuesFrom
                    rdf:resource="&Ontology1339617941597;EstimativaDeEsfocoCusto"/>
               </Restriction>
            </intersectionOf>
        </Class>
    </rdfs:subClassOf>
</Class>
```

Figura 67 - Estrutura do elemento "DimensaoDeTarefaMaisConcreta".
Fonte: extraído de Pizzoleto (2013).

Nota-se na **Figura 67** que o elemento possui diretamente relacionado a ele o elemento “EstimativaDeEsfocoCusto”, porém como a metodologia (ver *seção 4.3*) não permite que elementos do tipo “Extend1” possuam elementos relacionados, o elemento não será considerado no mapeamento.

Com isso, todos os elementos de “GPR2” foram identificados e marcados conforme as etapas 2.1 e 2.2 da metodologia. O próximo passo foi voltar ao elemento “GerenciaDeProjeto” e identificar o próximo elemento diretamente relacionado a ele. Ressalta-se que o elemento “GerenciaDeProjeto”, como já descrito anteriormente, possui dezenove RAPs e, portanto, cada um de seus RAPs deve passar pelos mesmos passos detalhados para o elemento “GPR2” neste capítulo.

De acordo com método detalhado na *seção 4.2*, após o término de identificação de todos os elementos relacionados ao elemento de maior nível hierárquico (“GerenciaDeprojeto”), deve-se identificar se existem mais elementos de maior nível e repetir o mesmo procedimento para eles. Portanto, dando continuidade à execução do método (**passo 1**), o próximo elemento de maior nível hierárquico escolhido foi o processo “Gerência de requisitos”. Assim, o elemento foi identificado e recebeu as marcações “owl:whatType=Process” e “owl:Order=2”, pois este será o segundo processo a ser representado no fluxo do processo. A **Figura 68** mostra a estrutura desse elemento já com a marcação.

```
<Class owl:whatType="Process" owl:Order="2"
    rdf:about="&Ontology1339617941597;GerenciaDeRequisito">
    <rdfs:label xml:lang="en">Requirement Management</rdfs:label>
    <rdfs:label xml:lang="pt">Gerência de requisitos</rdfs:label>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Processo"/>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;hassubprocess"/>
      <someValuesFrom rdf:resource="&Ontology1339617941597;GRE4"/>
    </Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;hasrole"/>
      <someValuesFrom rdf:resource="&Ontology1339617941597;AnalistaRequisitos"/>
    </Restriction>
  </rdfs:subClassOf>
</Class>
```

Figura 68 - Estrutura do elemento "Gerência de requisitos".
Fonte: extraído de Pizzoleto (2013).

Da mesma forma que o processo “Gerência de projetos”, ao processo “Gerência de requisitos” possui um total de cinco resultados esperados, porém na **Figura 68** apenas o elemento “GRE4” é exibido. De acordo com a estrutura apresentada na **Figura 68**, o elemento possui diretamente relacionado a ele mais dois elementos: “GRE4” e “AnalistaRequisitos”. Para execução do **passo 2.1** do método, o primeiro elemento escolhido para prosseguir com a identificação foi o “GRE4”. Assim, o elemento “GRE4” foi identificado e recebeu as marcações “owl;whatType=Subprocess” e “owl:Order=4”, indicando que o elemento será mapeado

como um “subprocesso” e tendo a ordenação igual a quatro. A **Figura 69** mostra a estrutura desse elemento já marcada.

```
<Class owl:whatType="Subprocess" owl:Order="4"
   rdf:about="&Ontology1339617941597;GRE4">
  <rdfs:label xml:lang="pt">GRE 4</rdfs:label>
  <equivalentClass>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;hasactivity"/>
      <someValuesFrom
      rdf:resource="&Ontology1339617941597;RevisarRequisitoseProdutos"/>
      </Restriction>
    </equivalentClass>
  <equivalentClass>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;hasactivity"/>
      <someValuesFrom
      rdf:resource="&Ontology1339617941597;RegistrarInconsistencias"/>
    </Restriction>
  </equivalentClass>
  <equivalentClass>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;hasactivity"/>
      <someValuesFrom
      rdf:resource="&Ontology1339617941597;CorrigirInconsistencias"/>
    </Restriction>
  </equivalentClass>
</Class>
```

Figura 69 - Estrutura do elemento "GRE4".
Fonte: extraído de Pizzoleto (2013).

Observa-se ainda na **Figura 69** que o elemento possui diretamente relacionado a ele os elementos: “RevisarRequisitoseProdutos”, “RegistrarInconsistencias” e “CorrigirInconsistencias”. Na ontologia utilizada, esses três elementos representam “atividades” que são realizadas dentro do subprocesso “GRE4”, e devem ser realizadas respeitando o seguinte fluxo: 1) “RevisarRequisitoseProdutos”; 2) “RegistrarInconsistencias”; 3) “CorrigirInconsistencias”. Portanto, para o próximo passo foi escolhido o elemento “RevisarRequisitoseProdutos”, que foi identificado e recebeu a marcação, “owl:whatType=Task”, e a marcação “Owl:Order=1” para indicar a sua ordem dentro do fluxo. A **Figura 70** mostra a estrutura do elemento “RevisarRequisitoseProdutos” já marcada.

```
<Class owl:whatType="Task" owl:Order="1"
  rdf:about="&Ontology1339617941597;RevisarRequisitoseProdutos">
  <rdfs:label xml:lang="pt">Revisar Requisitos e elementos do projeto</rdfs:label>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;isactivityof"/>
      <someValuesFrom rdf:resource="&Ontology1339617941597;GRE4"/>
    </Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Atividades"/>
</Class>
```

Figura 70 - Estrutura do elemento "RevisarRequisitoseProdutos".
Fonte: extraído de Pizzoleto (2013).

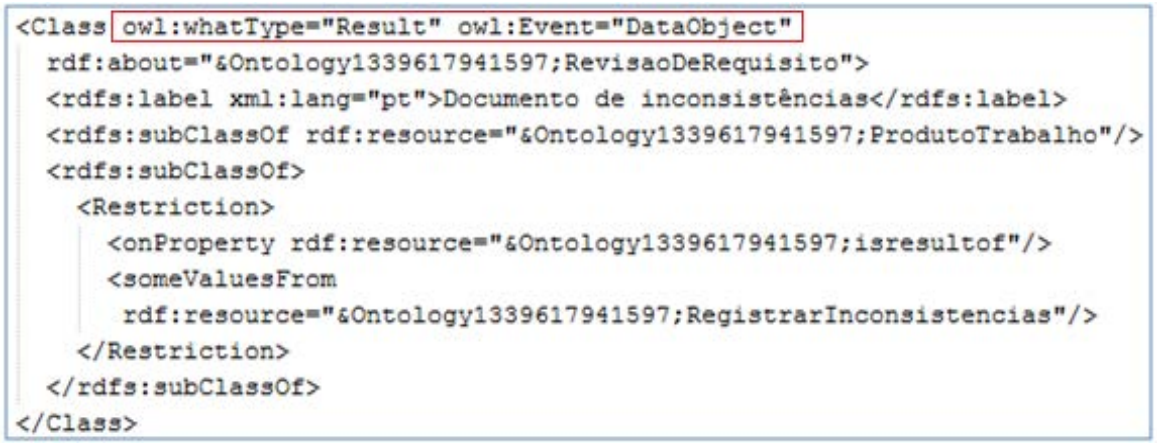
Como o elemento “RevisarRequisitoseProdutos” não possui nenhuma relação direta com outro elemento, o próximo elemento relacionado ao “GRE4” foi escolhido. O elemento “RegistrarInconsistencias” foi identificado e marcado com as marcações “owl:whatType=Task” e “owl:Order=2”, como mostra a **Figura 71**.

```
<Class owl:whatType="Task" owl:Order="2"
    rdf:about="&Ontology1339617941597;RegistrarInconsistencias">
   <rdfs:label xml:lang="pt">Registrar inconsistências</rdfs:label>
    <rdfs:subClassOf>
     <Restriction>
       <onProperty rdf:resource="&Ontology1339617941597;hasobligation"/>
       <someValuesFrom
       rdf:resource="&Ontology1339617941597;RegistrarInconsitenciaRequisito"/>
     </Restriction>
    </rdfs:subClassOf>
    <equivalentClass>
     <Restriction>
       <onProperty rdf:resource="&Ontology1339617941597;hasresult"/>
       <someValuesFrom rdf:resource="&Ontology1339617941597;RevisaoDeRequisito"/>
     </Restriction>
    </equivalentClass>
    <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Atividades"/>
  </Class>
```

Figura 71 - Estrutura do elemento "RegistrarInconcistencias".
Fonte: extraído de Pizzoleto (2013).

Na **Figura 71**, observa-se que o elemento “RegistrarInconsistencias” possui relação direta com outros dois elementos: o primeiro, “RevisaoDeRequisito”, que indica um artefato gerado e o segundo, “RegistrarInconsistenciaRequisito”, que representa um elemento que não pertence ao fluxo do processo, isto é, tem o objetivo apenas de enriquecer o processo. Portanto, o elemento “RevisaoDeRequisito” recebeu as marcações “owl:whatType=Result” e “owl:Event=DataObject”.

Já o elemento “RegistrarInconsistenciaRequisito” recebeu a marcação “owl:whatType=Extend3”, pois ele não é um elemento pertencente ao fluxo do processo, e foi considerado como uma informação extra no modelo de processo. Para esse elemento foi utilizada a marcação “Extend3”, pois a “Object Property” “hasobligation”, que o relaciona com o elemento imediatamente superior, é diferente das propriedades consideradas para as marcações “Extend1” e “Extend2” feitas anteriormente. As **Figuras 72** e **73** mostram as estruturas marcadas dos elementos “RevisaoDeRequisito” e “RegistrarInconsistenciaRequisito”, respectivamente.



```
<Class owl:whatType="Result" owl:Event="DataObject"
  rdf:about="&Ontology1339617941597;RevisaoDeRequisito">
  <rdfs:label xml:lang="pt">Documento de inconsistências</rdfs:label>
  <rdfs:subClassOf rdf:resource="&Ontology1339617941597;ProdutoTrabalho"/>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;isresultof"/>
      <someValuesFrom
       rdf:resource="&Ontology1339617941597;RegistrarInconsistencias"/>
    </Restriction>
  </rdfs:subClassOf>
</Class>
```

Figura 72 - Estrutura do elemento "RevisaoDeRequisito".
Fonte: extraído de Pizzoleto (2013).

Destaca-se que ambos os elementos apresentados nas **Figuras 72** e **73** não podem possuir nenhum outro elemento diretamente relacionado a eles, de acordo com as definições apresentadas na *seção 4.3*. Portanto, como todos os elementos relacionados à atividade “RegistrarInconsistencias” foram abordados, deve-se retornar ao elemento “GRE4” e identificar se ainda existe algum elemento diretamente relacionado a ele, e sem marcação.

```
<Class owl:whatType="Extend3"
  rdf:about="&Ontology1339617941597;RegistrarInconsitenciaRequisito">
  <rdfs:label xml:lang="pt">Registrar as inconsistência encontradas
                           nos requisitos</rdfs:label>
  <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Obrigacao"/>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;isobligationof"/>
      <someValuesFrom
       rdf:resource="&Ontology1339617941597;RegistrarInconsistencias"/>
    </Restriction>
  </rdfs:subClassOf>
</Class>
```

Figura 73 - Estrutura do elemento "RegistrarInconsistenciaRequisito".
Fonte: extraído de Pizzoleto (2013).

O próximo elemento a ser considerado foi o “CorrigirInconsistencias”. Esse elemento foi marcado com as marcações “owl:whatType=Task” e “owl:Order=3”, pois no contexto da ontologia do MR-MPS-SW esse elemento representa uma atividade, como mostra a **Figura 74**.

```
<Class owl:whatType="Task" owl:Order="3"
  rdf:about="&Ontology1339617941597;CorrigirInconsistencias">
  <rdfs:label xml:lang="pt">Corrigir Inconsistências</rdfs:label>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;hasobligation"/>
      <someValuesFrom
       rdf:resource="&Ontology1339617941597;ResolverInconsistencia"/>
    </Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Atividades"/>
</Class>
```

Figura 74 - Estrutura do elemento "CorrigirInconsistencias".
Fonte: extraído de Pizzoleto (2013).

Observa-se na **Figura 74** que o único elemento diretamente relacionado com “CorrigirInconsistencias” é “ResolverInconsistencia”. Na ontologia utilizada nesta aplicação, esse elemento refere-se a uma informação que não pertence ao fluxo do processo. Da mesma forma, a relação desse elemento com o elemento imediatamente superior, “CorrigirInconsistencias”, se dá por meio da “Object Property”

“hasobligation”, que já foi considerada pela marcação “Extend3”. Portanto, o elemento recebeu a marcação “owl:whatType=Extend3”, como mostra a **Figura 75**.

```
<Class owl:whatType="Extend3"
  rdf:about="&Ontology1339617941597;ResolverInconsistencia">
  <rdfs:label xml:lang="pt">Resolver as inconsitências dos requisitos
                           usando ações corretivas</rdfs:label>
  <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Obrigacao"/>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;isobligationof"/>
      <someValuesFrom
      rdf:resource="&Ontology1339617941597;CorrigirInconsistencias"/>
    </Restriction>
  </rdfs:subClassOf>
</Class>
```

Figura 75 - Estrutura do elemento "ResolverInconsistencia".

Fonte: extraído de Pizzoleto (2013).

Dessa forma, todos os elementos relacionados ao resultado “GRE4” foram identificados e devidamente marcados. Dando continuidade ao **passo 2** do método de identificação (ver *seção 4.2*), deve-se retornar ao elemento superior de “GRE4”, “GerenciaDeRequisito”, e procurar por outros elementos ainda sem marcação. Assim, o elemento “AnalistaRequisitos” foi identificado. Esse elemento representa um papel pertencente ao fluxo do processo, e por isso, recebeu a marcação “owl:whatType=Role”, como mostra a **Figura 76**.

```
<Class owl:whatType="Role"
  rdf:about="&Ontology1339617941597;AnalistaRequisitos">
  <rdfs:label xml:lang="pt">Analista de requisitos</rdfs:label>
  <rdfs:subClassOf>
    <Restriction>
      <onProperty rdf:resource="&Ontology1339617941597;isroleof"/>
      <someValuesFrom
       rdf:resource="&Ontology1339617941597;GerenciaDeRequisito"/>
    </Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&Ontology1339617941597;Papeis"/>
</Class>
```

Figura 76 - Estrutura do elemento "AnalistaRequisitos".

Fonte: extraído de Pizzoleto (2013).

Assim, todos os elementos pretendidos foram identificados e marcados. Ressalta-se que o elemento “GerenciaDeRequisitos”, como já descrito anteriormente, possui cinco resultados esperados e, portanto, cada um destes deve passar pelos mesmos passos detalhados para o elemento “GRE4”.

De acordo com a etapa 2.1 da metodologia (ver *seção 4.2*), o método de identificação deve ser repetido por completo para cada elemento de maior nível hierárquico definido na ontologia. Na ontologia utilizada nesta aplicação, além dos dois processos apresentados nesta seção, existem mais cinco elementos de maior nível hierárquico: “Gerência de configuração”, “Gerência de Portfólio de Projetos”, “Garantia de qualidade”, “Medição” e “Aquisição”. Portanto, a mecânica apresentada nesta seção deve ser repetida para cada um desses elementos, até que todos os processos e elementos relacionados tenham sido abordados pela metodologia.

Dessa forma, pôde-se concluir a aplicação das etapas 2.1 e 2.2 da metodologia. Para a execução da **etapa 2.3**, foi construída uma tabela de referência cruzada de todos os elementos e informações consideradas nas etapas anteriores (2.1 e 2.2). Por meio dessa tabela foi possível verificar que todos os elementos pertencentes aos processos descritos pela ontologia (“Gerência de Projeto”, “Gerência de Requisitos”, “Garantia da Qualidade”, “Gerência de Portfólio de Projetos”, Gerência de Configuração”, “Medição” e “Aquisição”), foram abordados pelas etapas anteriores. A tabela criada nesta etapa encontra-se disponível no APÊNDICE F.

Portanto, ao término da aplicação da **etapa 2** da metodologia, em todos os processos do nível G e F, as seguintes marcações foram utilizadas:

- Para os processos “Gerência de projetos”, “Gerência de Requisitos”, “Gerência de Portfólio”, “Garantia de Qualidade”, “Gerência de Configuração”, “Medição” e “Aquisição” a marcação “owl:whatType” assumiu o valor “Process”, contendo também o marcador “owl:Order”;
- Para os RAP dos processos, a marcação “owl:whatType” assumiu o valor “Subprocess” em todos os casos, contendo também o marcador “owl:Order”;
- Para os papéis relacionados aos processos ou subprocessos (RAP), a marcação “owl:whatType” assumiu o valor “Role”;
- Para as atividades, que devem ser executadas para se chegar a um resultado esperado, foram utilizados as marcações “owl:whatType” com valor “Task”, e “owl:Order”;

- Para os resultados gerados pelos processos, a marcação “owl:whatType” assumiu o valor “Result” em todos os casos, porém diferindo na marcação “owl:Event” que em casos de geração de objetos físicos, como documentos, termos, entre outros, assumiu valor “DataObject”, e em casos que se referiam apenas a “marcos” de atividades desenvolvidas assumiu o valor “Artifact”;
- Os elementos “diferenças para fábrica de software”, “objetivos” e “obrigação” foram mapeados através da marcação “owl:whatType” assumindo os valores ”Extend1”, “Extend2” e “Extend3” respectivamente.

5.3 APLICAÇÃO DAS ETAPAS 3 E 4 DA METODOLOGIA

Após a execução da etapa 2 em todos os processos da ontologia, a execução da **etapa 3** consistiu em utilizar a ferramenta *RDFtoXPDL,* desenvolvida neste trabalho, para a conversão dos dados da ontologia em um modelo de processos. De acordo com as “Object Property” utilizadas pela ontologia do MR-MPS-SW, a ferramenta *RFtoXPDL* precisou ser configurada de modo a atender a ontologia. Para isso, através do menu “Configuration”, opção “RDF Properties”, pôde-se configurar as “Object Property” necessárias, como mostra a **Figura 77**.

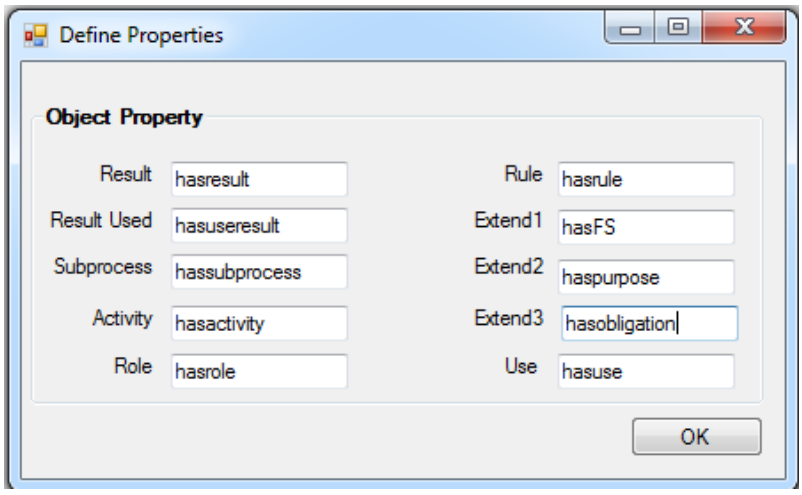


Figura 77 - Configuração dos valores de “Object Property” na ferramenta *RDFtoXPDL*.

Já os valores das marcações utilizadas na aplicação da metodologia, seguiram os valores padrões apresentados na *seção 4.3*. Como a ferramenta

RDFtoXPDL já traz os valores padrões apresentados para esses elementos, não houve a necessidade de maiores alterações, como mostra a **Figura 78**.

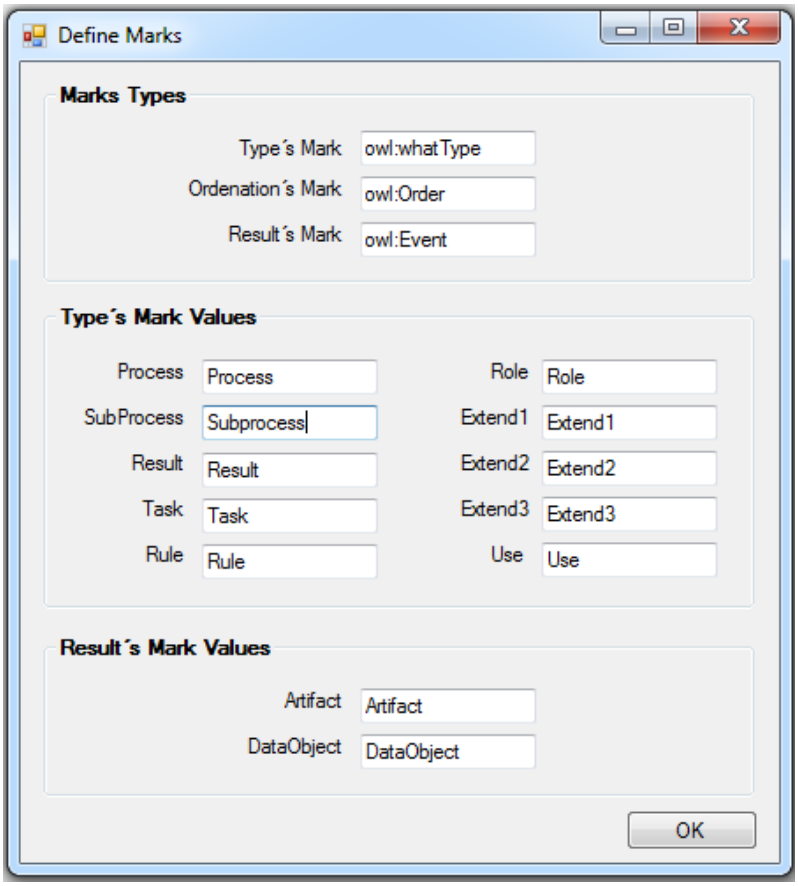


Figura 78 - Configuração dos valores das marcações na ferramenta *RDFtoXPDL*.

Por fim, a execução da **etapa 4** da metodologia, consistiu em utilizar uma ferramenta de modelagem de processos para visualizar o modelo de processos gerado pela ferramenta *RDFtoXPDL*. Para execução dessa etapa, o software *Bizagi Modeler* (versão 2.6.0.4) foi utilizado. A **Figura 79** apresenta o modelo de processos gerado para o processo “Gerência de requisitos” do nível G, do modelo MR-MPS-SW. Devido ao tamanho dos modelos de processos gerados, todos os modelos de processos gerados para os níveis G e F do modelo MR-MPS-SW, resultantes desta aplicação, encontram-se no APÊNDICE F, em mídia digital.

Dessa forma, com o término da execução das **etapas 3 e 4**, o modelo de processos gerado pela ferramenta *RDFtoXPDL*, foi importado pela ferramenta *Bizagi Modeler,* sem nenhuma inconsistência. Assim, com o término da aplicação da metodologia proposta neste trabalho, na ontologia do modelo MR-MPS-SW, níveis G e F, os seguintes resultados foram observados:

- Os processos “Gerência de projetos”, “Gerência de Requisitos”, “Garantia da Qualidade”, “Gerência de Configuração”, “Medição”, “Aquisição” e “Gerência de Portfólio de Projetos” foram mapeados juntamente com seus resultados;
- O processo “Gerência de Requisitos”, que teve elementos adicionados a sua estrutura, foi mapeado com todos os elementos adicionais: papéis, atividades, resultados e obrigações.
- Os elementos que possuíam relacionados a ele elementos do tipo “Extend1”, “Extend2” e “Extend3” foram mapeados com sucesso para os atributos “ExtendedAttribute” da linguagem XPDL.

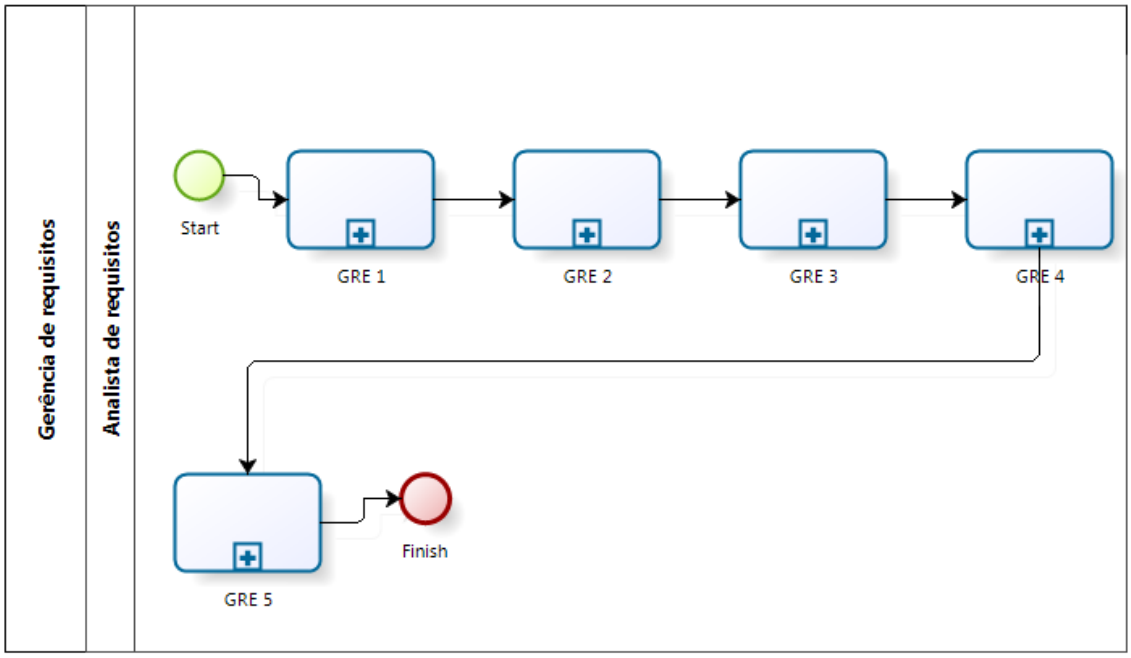


Figura 79 - Modelo de processos de "Gerência de requisitos".

5.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Com a aplicação da metodologia desenvolvida neste trabalho, na ontologia do MR-MPS-SW, foi possível verificar a aderência da metodologia em processos de negócio reais, na área de qualidade de software. Além de verificar a metodologia, a aplicação resultou em um modelo de processos do MR-MPS-SW, níveis G e F, que pode servir de apoio as organizações empresariais que pretendem implementar esse modelo.

Nem todos os elementos presentes na ontologia puderam ser mapeados para o modelo de processos, dentre os quais: os elementos pertencentes as classes “Propósito”, Questões”, “Sugestões” e “Necessidades”, não puderam ser levados para o modelo, pois representam conceitos que não fazem parte do fluxo do processo. Ou seja, esses elementos precisariam ser mapeados por meio das marcações “Extend1”, “Extend2” ou “Extend3”. Porém, a aplicação da metodologia considerou três outros tipos de elementos para essas marcações, de acordo com a ordem de leitura dos elementos na ontologia.

Já os elementos pertencentes às classes “Produtos de apoio” e “Regras” não puderam ser mapeados por não estarem de acordo com a hierarquia de valores possíveis de elementos, que a metodologia propõe, de acordo com a marcação “owl:whatType” (ver *seção 4.3*).

6 CONSIDERAÇÕES FINAIS

Com um mercado de trabalho cada vez mais voltado para o cliente, as organizações passam a buscar exaustivamente por melhorias na qualidade de seus produtos. A procura por métodos e tecnologias que apoiem a organização no entendimento de suas necessidades e no autoconhecimento de seus processos vem atraindo a atenção das organizações desenvolvedoras de software. Para essas organizações, é fundamental que existam processos bem definidos e conhecidos por todos os seus participantes. O ambiente dessas organizações é muito dinâmico e evolui muito rapidamente, dificultando o entendimento de seus processos, e, consequentemente, contribuindo com a má qualidade dos produtos desenvolvidos. Frente ao exposto, a exigência dos consumidores aumenta cada vez mais, e a procura por fornecedores de software que possuam certificados de qualidade e uma gestão bem definida cresce a cada dia.

Para tanto, a gestão dos processos de negócio é essencial, tornando possível a análise completa do negócio, e apoiando sua otimização. Nesse contexto, as ontologias vêm ganhando cada vez mais espaço no contexto das organizações empresariais. Para isso, a representação de processos através de ontologias acaba sendo uma forma alternativa de representar os conceitos envolvidos no domínio de processos. As ontologias organizam as informações, estruturam os elementos dos processos, possibilitam identificar com maior clareza as relações entre cada elemento, e tornam o seu entendimento uniforme. Porém, quando se passa a considerar os diversos processos da organização, assim como suas interdependências, as ontologias ainda possuem um formato visual que dificulta o entendimento do domínio como um todo. Por esses fatores, ontologias não representam uma alternativa efetiva para representação de processos de negócio, porém são ótimos repositórios para se extrair informações para geração de modelos de processos.

Um dos primeiros passos para que a organização possa gerir adequadamente seus processos é através da modelagem de processos. Através da modelagem é possível entender cada elemento envolvido no processo, desde as pessoas, sistemas, tarefas, departamentos, até os resultados gerados por cada um

destes. A modelagem também pode ser utilizada para a disseminação do conhecimento dentro da organização, mantendo, assim, todas as pessoas envolvidas no processo alinhadas com o negócio.

Com a intenção de proporcionar um melhor entendimento de processos de negócio, representados por meio de ontologias empresariais, este trabalho apresentou o desenvolvimento de uma metodologia para geração de um modelo de processos estruturado como *workflow*, utilizando como base uma ontologia empresarial. Nessa direção, a *seção 6.1* apresenta as contribuições deste trabalho. Já a *seção 6.2* apresenta algumas propostas de trabalhos futuros.

6.1 CONTRIBUIÇÕES DO TRABALHO

Nessa direção, o principal objetivo deste trabalho foi propor uma metodologia que tornasse possível o mapeamento de processos representados através de ontologias empresariais para modelos de processos de negócio em BPMN. Através da notação gráfica BPMN, padrão mundialmente utilizado, o modelo de processos gerado permitirá um melhor entendimento do fluxo do processo como um todo, facilitando a comunicação entre os envolvidos, fornecendo visão clara das atividades a serem realizadas e também a identificação de inconsistências no fluxo do processo. O objetivo é que o modelo de processos possa servir como modelo base para a organização. Com o modelo em mãos, a organização pode personalizá-lo com informações da própria organização, transformando o modelo em um artefato particular. O modelo pode, inclusive, servir de base para dar continuidade à abordagem de Gerenciamento de Processos de Negócio.

O trabalho visa contribuir, junto ao meio acadêmico, com a metodologia de mapeamento desenvolvida para o mapeamento de ontologias em modelos de processos de negócio, assim também como uma ferramenta criada para automatizar o processo de conversão dos elementos da ontologia para o modelo de processos.

Através da ontologia empresarial do modelo MPS-SW, níveis G e F, criada por Pizzoleto (2013), foi possível realizar uma aplicação com a metodologia. O modelo de processos gerado pela aplicação mostrou que a metodologia se adequou perfeitamente à ontologia empresarial utilizada. Porém, ressalta-se que nem todos os elementos puderam ser mapeados para o modelo de processos, ora por não se

enquadrar nas restrições impostas pela metodologia, e ora por possuir elementos que não podem ser interpretados pelo modelo de processos.

Como parte da contribuição deste trabalho, destaca-se que o modelo de processos gerado pela aplicação da metodologia com a ontologia empresarial do modelo MPS-SW pode ser utilizado como apoio por organizações que desejam implementar os níveis G e F do modelo MPS-SW. O modelo gerado pode ser utilizado como base para que a organização comece a entender como estruturar seus processos. A organização pode ainda inserir informações, de acordo com sua realidade, no modelo, de modo a torná-lo um modelo de processos próprio da organização.

É importante destacar que a metodologia proposta neste trabalho pode ser aplicada para qualquer ontologia empresarial, desde que sejam respeitadas as definições e restrições impostas por ela. Tendo em vista que a metodologia foi desenvolvida para ser genérica, é muito importante que esta seja aplicada em outros casos, de modo a validar, e até mesmo melhorar sua proposta.

6.2 TRABALHOS FUTUROS

Visando à continuidade deste trabalho, a realização de estudos que criem uma alternativa gráfica para a realização da etapa 2 da metodologia proposta, de modo que não seja mais necessário fazer marcações manuais no código da ontologia, tornaria a metodologia mais ágil e simples de ser aplicada. Através dessa alternativa, o usuário poderia encontrar os elementos na ontologia de forma rápida, de modo a ir marcando os elementos conforme a necessidade.

Considerando que processos também podem possuir caminhos paralelos, como continuidade deste trabalho pode ser estudada maneiras de possibilitar o mapeamento das informações da ontologia em caminhos paralelos, ou até mesmo, que levem para caminhos distintos dentro do fluxo do processo. É interessante observar, que para isso pode ser necessário revisar alguns pontos da metodologia criada, como por exemplo, a marcação “owl:Order”, que não poderá mais possuir caráter sequencial.

Como outra abordagem para continuidade deste trabalho, podem ser feitos estudos de modo a identificar uma forma de se instanciar o modelo de processos criado pelo mapeamento em uma organização, tendo em vista que a respectiva

organização poderia adicionar informações ao modelo, de acordo com sua realidade, inclusive com a adição de regras de negócio para a execução do modelo de processo. Dessa forma, a utilização do modelo de processos poderia se estender a outras partes da tecnologia BPM. Algumas ideias iniciais foram analisadas, como por exemplo: disponibilizar o modelo criado como *WebService*, para que a empresa pudesse ter uma versão própria do modelo e pudesse personalizá-la de acordo com suas necessidades. Uma segunda ideia seria apenas disponibilizar o modelo como sendo uma forma de "*checklist*", no qual a organização poderia se apoiar e controlar o que já foi finalizado, o que ainda está sendo feito ou o que falta ser implantado.

Visando enriquecer o modelo de processos gerado pela metodologia, estudos que criem sistemas de modelagem de processos que tenham formas alternativas para o uso do atributo "ExtendedAttribute" da linguagem XPDL. A ferramenta poderia possuir diferentes mecanismos de interpretação do respectivo atributo, de modo a adicionar elementos textuais, ou até mesmo gráficos no modelo de processos. As informações inseridas por este atributo não poderiam ser levadas em consideração na leitura e execução do modelo de processos, porém serviriam para apoiar no entendimento do fluxo do processo.

Outra linha de trabalho seria a extensão dos elementos de BPMN considerados pela metodologia para serem mapeados, de modo a enriquecer o modelo de processos gerado. Assim, outros elementos como, por exemplo, "eventos intermediários" poderiam tornar o modelo de processos mais completo.

REFERÊNCIAS

ALLEMANG, D; HENDLER, J. **Semantic web for the working ontologist: effective modeling in RDF and OWL 2**. 2º ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 2011. 384 p.

ARAUJO, R.; CAPPELLI, C.; GOMES JUNIOR, A.; PEREIRA, M.; IENDRIKE, H. S.; IELPO, D.; TOVAR, J.A. A definição de Processos de Software sob o ponto de vista da Gestão de Processos de Negócio. In: SIMPÓSIO INTERNACIONAL DE MELHORIA DE PROCESSOS DE SOFTWARE (SIMPROS 2004). 2004. **Anais... ,**2004.

ARENAS, M.; GUTIERREZ, C.; PÉREZ, J. On the Semantics of SPARQL. In: VIRGILIO, R., GIUNCHIGLIA, F., TANCA, L. (Org). **Semantic Web Information management: A model based perspective.** Springer. 2010. p. 281-307.

ASLAM, M.; AUER, S.; SHEN, J.; HERRMMAN, M. Expressing Business Process Models as OWL-S Ontologies. In: INTERNATIONAL CONFERENCE ON BUSINESS PROCESS MANAGEMENT (BPM 2006), 4. 2006.

BECHHOFER, S; HARMELEN, F. V; HENDLER, J; HORROCKS, I; MCGUINNESS, D. L; PATEL-SCHNEIDER, P. F; STEIN, L. A. **OWL Web Ontology Language Reference**. 2004. Disponível em: <http://www.w3.org/TR/2004/REC-owl-ref-20040210/>. Acesso em: 28 maio 2014.

BERNERS-LEE, T.; CHEN, Y.; CHILTON, L.; CONNOLLY, D.; DHANARAJ, R.; HOLLENBACH, J.; LERER, A.; SHEETS, D. Tabulator: exploring and analyzing linked data on the semantic web. In: INTERNATIONAL SEMANTIC WEB USER INTERACTION WORKSHOP, 3. 2006, Athens. **Proceedings…** Athens, 2006. p. 3-10. Disponível em: <http://swui.semanticweb.org/swui06/papers/Berners-Lee/Berners-Lee.pdf>. Acesso em: 12 maio 2014.

BLOMQVIST, E.; ÖHGREN, A.; SANDKUHL, K. Ontology construction in an enterprise context: comparing and evaluating two approaches. In: INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS, 8. 2006, Paphos. **Proceedings...** Paphos, 2006. p. 86-93.

BORTOLI, L. A.; PRICE, A. M. A. O uso de Workflow para Apoiar a Elicitação de Requisitos. In: WORKSHOP DE ENGENHARIA DE REQUISITOS (WER 2000), 2000, Rio de Janeiro. **Anais…** Rio de Janeiro, 2000. pp. 22-37.

BROWNING, T. R. **Process Integration Using the Design Structure Matrix**. Systems Engineering, 2002. v. 5.

CABRAL, L.; NORTON, B.; DOMINGUE, J. The Business process modeling ontology. In: INTERNATIONAL WORKSHOP ON SEMANTIC BUSINESS PROCESS MANAGEMENT (SBPM 2009). 4. 2009.

CASATI, F.; CERI, S.; PERCINI, B.; POZZI, G. Conceptual Modeling of Workflows. In: OBJECT-ORIENTED AND ENTITY-RELATIONSHIP CONFERENCE (OOER 1995). Gold Coast, Austrália, 1995. **Proceedings...** , 1995. pp. 341-354.

CERQUEIRA, A. L. A. **Integração de Ontologia com Modelagem de Processo: Um método para facilitar a elicitação de requisitos**. Dissertação (Mestrado). PUCRJ, 2007.

CRUZ, T. **BPM & BPMS – Business Process Management & Business Process Management Systems**. Rio de Janeiro: Brasport, 2008.

DAVENPORT, T. **Process Innovation. Reengineering Work through Information Technology**. Boston,MA: Harvard Business School Press. 1993.

DAVENPORT, T. **Reengenharia de Processos.** 5 ed. Rio de Janeiro: Campus,1994.

DUBOULOUZ, B. **Business Process Management Systems (BPMS)**. Ensures Consulting. 2004.

FRANCISCO, R.; LOURES, E. F. R.; SANTOS, E. A. P.; PAULA, M. A. B.; DESCHAMPS, F. Traduzindo a definição de processo em XPDL para modelos em Redes de Petri. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO (ENEGEP 2009), 29. 2009. Salvador. **Anais...** Salvador: ENEGEP, 2009.

FURGERI, S. **O papel das linguagens de marcação para a Ciência da Informação**. Campinas: TransInformação, v. 18, n. 3, 2006. Disponível em: <http://periodicos.puc-campinas.edu.br/seer/index.php/transinfo/article/view/670>. Acesso em: 20 Setembro. 2014.

GEORGAKOPOULOS, D.; HORNICK, M.; SHETH, A. **An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure**. Boston, MA: Distributed and Parallel Databases.1995. v.3.

GEORGAKOPOULOS, D.; TSALGATIDOU, A. Technology and Tools for Comprehensive Business Process Lifecycle Management. In: DOGAC, A., KALINICHENKO, L., OZSU, T., SHETH, A. **Workflow management Systems and Interoperability**. v. 164. Springer, 1998. pp. 356-395.

GÓMEZ-PÉREZ, A.; CORCHO, O. Ontology languages for the semantic web. In: **IEEE Intelligent Systems**. v. 17, n. 1, 2002. p. 54-60.

GRUBER, T. R. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. In: **International Journal Human-Computer Studies,** v. 43.1993. p.907-928.

GRUNINGER, M.; PINTO, J. A. A Theory of Complex Actions for Enterprise Modelling. In: BOUTILIER C., GOLDSZMIDT, M. (Org). **Symposium Extending Theories of action: Formal Theory and Practical Applications**, Stanford, California: AAAI Press. 1995. pp. 94-99.

HALLER, A.; GAALOUL, W.; MARMOLOWSKI, M. Towards an XPDL compliant process ontology. In: IEEE CONGRESS ON SERVICES. 2008. **Proceedings…** 2008. pp. 83-86.

HARMON, P. **Evaluating an Organization's Business Process Maturity. Business Process Trends**. , v. 2, n.3. 2004. 11 p. Disponível em: <http://www.bptrends.com/publicationfiles/03-04%20NL%20Eval%20BP%20Maturity%20-%20Harmon.pdf>. Acesso em: 12 setembro 2014.

HITZLER, P.; KRÖTZSCH, M.; RUDOLPH, S**. Foundations of Semantic Web technologies**. Chapman & Hall/CRC, 2009. 455 p.

INDULSKA, M.; GREEN, P.; RECKER, J.; ROSEMANN, M. Business Process Modeling: Perceived Benefits. In: CASTANO, S., DAYAL, U., LAENDER, H. F. (Org). **Conceptual Modeling – ER 2009**. Gramado, Brasil:Springer. 2009. pp. 458-471.

JACOBSON, I.; ERICSSON, M.; JACOBSON, A. **The Object Advantage: Business Process Reengineering with Object Technology**. New York: Addison-Wesley Publishing Co., 1994.

JENZ, D. E. **Business Process Ontologies: Speeding up Business Process Implementation**. 2003. Disponível em: <http://www.bptrends.com/publicationfiles/07-03%20WP%20BP%20Ontologies%20Jenz.pdf>. Acesso em: 10 abril 2014.

JOHANSSON, H. J.; MCHUGH, P.; PEDLEBURY, A. J.; WHELLER, W. A. **Processos de negócio**. São Paulo: Pioneira. 1995.

KALPIC, B.; BERNUS, P. Business process modeling in industry: the powerful tool in enterprise management. In: **International Journal of Computers in Industry**, v. 47, n. 3, 2002. pp. 299-318.

KHAN, R. Evaluating BPM Software. In: **Business Integration Journal**, 2003. pp. 24-30.

KOUBARIAKIS, M.; PLEXOUSAKIS, D. **Business Process Modeling and Design – a formal model and methodology**. 1999. Disponível em: <http://www.ics.forth.gr/isl/publications/paperlink/final-bt.pdf>. Acesso em: 23 Março. 2014.

LEITE, L. O.; REZENDE, D. A. Gestão corporativa por processos na administração pública municipal: estudo de caso da implantação de BPM no Instituto Curitiba de Informática. In: Encontro de Administração da Informação, 2007. **Anais...** Florianópolis: EDANI, 2007.

LEYMANN, F; ROLLER, D. Workflow-based aplications. In: **IBM Systems Journal**, v.36, n.1, 1997. pp. 102-123.

MCCORMACK, K.; WILLEMS, J.; VAN DEN BERGH, J.; DESCHOOLMEESTER, D.; WILLAERT, P.; STEMBERGER, M. I.; SKRINJAR, R.; TRKMAN, P.; LADEIRA, M. B.; OLIVEIRA, M. P. V.; VUKSIC, V. B.; VLAHOVIC, N. A global investigation of key turning points in business process maturity. In: **Business Process Management Journal**. v. 5, n. 5. 2009. pp. 792-815.

MCDANIEL, T. Ten Pillars of Business Process Management. In: **Journal of Enterprise Application Integration**. 2001. pp. 29-32.

MILLER, E. An Introduction to the Resource Description Framework. In: D-Lib Magazine, v. 4, n. 5, 1998. Disponível em: <http://www.dlib.org/dlib/may98/miller/05miller.html>. Acesso em: 16 agosto 2014.

MISSIKOFF, M.; PROIETTI, M.; SMITH, F. **Linking Ontologies to Business Process Schemas**. 2010. Disponível em: <http://www.iasi.cnr.it/reports/R10020/R10020.html>. Acesso em: 14 outubro 2014.

MIZOGUCHI, R. **Tutorial on Ontological Engineering**. 2003. Disponível em: <http://www.unipamplona.edu.co/unipamplona/portalIG/home_23/recursos/general/06032011/onto_parte1.pdf>. Acesso: 15 dezembro 2014.

MOMOTKO, M; NOWICKI, B. Visualisation of (distributed) Process: Execution based on Extended BPMN. In: INTERNATIONAL WORKSHOP ON DATABASE AND EXPERT SYSTEMS APPLICATIONS, 14. 2003. **Proceedings…** 2003. pp. 280-284.

MUTTON, P.; GOLBECK, J. Visualization of Semantic Metadata and Ontologies. In: INTERNATIONAL CONFERENCE ON INFORMATION VISUALIZATION, 7. 2003, London. **Proceedings...** London, 2003.

NETTO, F. S**.** Gerenciamento de Processos de Negócio: um estudo teórico-comparativo sob as óticas da Gestão Empresarial e da Tecnologia da Informação. In: SIMPÓSIO DE EXCELÊNCIA EM GESTÃO E TECNOLOGIA, 5.2008. **Anais…** Resende,2008.

NOY, N. F.; MCGUINNESS, D. L. **Ontology Development 101: A Guide to Creating Your First Ontology**. 2001. Disponível em: <http://protege.stanford.edu/publications/ontology_development/ontology101.pdf>. Acesso em: 28 dezembro 2014.

ÖHGREN, A.; SANDKUHL, K. Towards a methodology for ontology development in small and medium-sized enterprises. In: INTERNATIONAL CONFERENCE ON APPLIED COMPUTING, 2005, Algarve. **Proceedings…** Algarve. 2005. pp. 369-376.
OMG – Object Management Group. **BPMI.org and OMG announce strategic merger of Business Process Management Activities**. 2005. Disponível em: <http://www.omg.org/news/releases/pr2005/06-29-05.htm>. Acesso em: 15 agosto 2014.

_____. **Business Process Model and Notation (BPMN). Version 2.0**, 2011. Disponível em: <http://www.omg.org/spec/BPMN/2.0/> .Acesso em: 12 fevereiro 2014.

OUYANG, C.; DUMAS, M.; VAN DER AALST, W. M. P. From Business Process Models to Process-oriented Software Systems**.** In: **ACM Transactions on Software Engineering Methodology.** v.19. 2009. pp. 1-37.

PEREZ, J.; ARENAS, M.; GUTIERREZ, C**.** Semantics and Complexity of SPARQL. In: INTERNATIONAL SEMANTIC WEB CONFERENCE (ISWC 2006), 5. 2006. **Proceedings…** 2006. pp. 30-43.

PIDD, M. Just Modeling Through. A Rough Guide to Modeling. In: **Interfaces**. v.29. 1999. pp. 118–132.

PIZZOLETO, A. V.; OLIVEIRA, H. C.; OLIVEIRA, C. S. An ontology on the level G of the Software Process Model MPS.BR to assist business processes modeling. In: INTERNATIONAL CONFERENCE WWW/INTERNET, 2012, Madrid. **Proceedings...** 2012. pp. 404-408.

PIZZOLETO, A. V. Ontologia Empresarial no modelo MPS.BR visando modelagem de processos de negócios, com foco nos níveis G e F. Dissertação (Mestrado). Universidade Estadual Júlio de Mesquita Filho, 2013.

PIZZOLETO, A. V.; OLIVEIRA, H. C. Methodology for ontology development in support to the MPS model for software. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING RESEARCH AND PRACTICE (SERP 2013), 11. 2013, Las Vegas, Nevada. **Proceedings…** Las Vegas, 2013. 7 p.

PROFORMA. Enterprise Application Modeling – Vision and strategy for the ongoing development of ProVision Workbench. Proforma Technical White Paper, 2000.

RAMALHO, R. A. S. Web Semântica: aspectos interdisciplinares da gestão de recursos **informacionais no âmbito da Ciência da Informação**. 2006. Dissertação (Mestrado). Faculdade de Filosofia e Ciências – Universidade Estadual Paulista, Marília, 2006. Disponível em: <http://www.marilia.unesp.br/Home/Pos-Graduacao/CienciadaInformacao/Dissertacoes/ramalho_ras_me_mar.pdf>. Acesso em: 02 junho 2014.

RECKER, J.; ROSEMANN, M.; INDULSKA, M.; GREEN, P. Business Process Modeling. A comparative analysis. In: **Journal of the Association for Information Systems**. v.10.n. 4. 2009. pp. 333-363.

ROCHA, A. C. C; CAVALCANTI, E. X.; CUNHA, E. R.; KOBLITZ, L. F.; CRUZ, P. O. S.; BITTENCOURT, R.; SILVA, T. A. A. A modelagem de Processos de negócios em empresa pública – A Experiência da Comissão Nacional de Energia Nuclear – CNEN na modelagem dos processos de negócio do Serviço de Tecnologia da Informação – SETIN. In: CONGRESSO CIENTÍFICO DA UNIVERCIDADE, 2. 2007. Rio de Janeiro. **Anais...** Rio de Janeiro, 2007.

SANTORO, F. M.; IENDRIKE, H.; ARAUJO, R. M. Colaboração em Processos de Negócio, In: PIMENTEL, M., FUKS, H. (Org.). **Sistemas Colaborativos**. Elsevier, 2011. pp. 173-185.

SHARP, A.; MCDERMOTT, P., **Workflow Modeling: Tools for Process Improvement and Application Development**. Boston: Artech House, 2001.

SHAPIRO, R. M. **A comparison of XPDL, BPML and BPML4WS**. 2002. Disponível em: <https://www.google.com.br/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCYQFjAA&url=http%3A%2F%2Fwww.ebpml.org%2FA_Comparison_of_XPDL_and_BPML_BPEL.doc&ei=bOD8VP_BGarPsQT0sYHIAw&usg=AFQjCNFkL-SdYJZ2V-KrZ2AWte6kOIMdig&sig2=MjwFLU8ICwQNOSwcMWrCSg&bvm=bv.87611401,d.cWc&cad=rja>. Acesso em: 13 novembro 2014.

SHAPIRO, R. M. XPDL 2.0: Integrating Process Interchange and BPMN. In: FISCHER, L. **2006 Workflow Handbook including Business Process Management**. 2006.

SILVA, A. V. **Modelagem de processos para implementação de workflow: uma avaliação crítica**. Dissertação (Mestrado). COPPE- Universidade Federal do Rio de Janeiro, 2001.

SMITH, H.; FINGAR, P. **Business Process Management: Third Wave**. Tampa, Florida, USA: Meghan Kiffer Press, 2003.

SMITH, M. K; WELTY, C; MCGUINNESS, D. L. **OWL Web Ontology Language Guide.** 2004**.** Disponível em: <http://www.w3.org/TR/2004/REC-owl-guide-20040210/>. Acesso em: 27 maio 2014.

SOFTEX - Associação para Promoção da Excelência do Software Brasileiro. **MPS.BR Melhoria de processo do software brasileiro**: **guia geral de software**. 2012a. Disponível em: <http://www.softex.br/wp-content/uploads/2013/07/MPS.BR_Guia_Geral_Software_2012.pdf>. Acesso em: 10 abril 2013.

_____. **MPS.BR Melhoria de processo do software brasileiro**: Website do programa MPS, 2012b. Disponível em: <http://www.softex.br/mpsbr>. Acesso em: 20 janeiro 2013.

SORDI, J. O. **Gestão por processos: uma abordagem da moderna administração**. São Paulo: Saraiva. 2005.

STANFORD. **Protégé Ontology Library**. 2015. Disponível em: <http://protegewiki.stanford.edu/wiki/Protege_Ontology_Library>. Acesso em: 03 janeiro 2015.

STRADIOTTO, C.; PACHECO, E.; BORTOLON, A.; HOESCHL, H. A Graphic Tool for Ontology Viewing Based on Graph Theory. In: WORLD COMPUTER CONGRESS (WCC 2006), 19. 2006. **Proceedings…** Santiago, Chile, 2006.

TAUBERER, J. **What is RDF**. 2006. Disponível em: <http://www.xml.com/pub/a/2001/01/24/rdf.html>. Acesso em: 24 setembro 2014.

TERNAI, K.; TÖRÖK, M. A New Approach in the Development of Ontology Based Workflow Architectures. In: INTERNATIONAL CONFERENCE ON CONCURRENT ENTERPRISING (ICE 2011), 17. 2011, Aachen, Germany. **Proceedings…** Aachen, 2011. 10 p.

THIRY, M.; WANGENHEIM, C.G .V.;ZOUCAS, A.; PICKLER, K. **Uma abordagem para a Modelagem Colaborativa de Processos de Software em Micro e Pequenas Empresas**. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE (SBQS 2006), 5. 2006.

USCHOLD, M.; KING, M. **Towards a Methodology for Building Ontologies**. In: WORKSHOP ON BASIC ONTOLOGICAL ISSUES IN KNOWLEDGE SHARING. Edinburgh. 1995.

USCHOLD, M.; KING, M.; MORALEE, S.; ZORGIOS, Y. **The Enterprise Ontology**. 1997 Disponível em: <http://ipm.lviv.ua/library/0/004/004.8/98-ker-ent-ontology.pdf>. Acesso em: 12 dezembro 2014.

VAN DER AALST, W. M. P. **Patterns and XPDL: A Critical evaluation of the XML Process Definition Language**. Queensland University of Technology, 2003. Disponível em: <http://www.workflowpatterns.com/documentation/documents/ce-xpdl.pdf>. Acesso em: 5 setembro 2014.

VAN DER AALST, W. M. P.; HEE, V. K. **Workflow management: models, methods and systems**. 2000. Disponível em: <http://wwwis.win.tue.nl/~wvdaalst/publications/p120.pdf>. Acesso em: 17 janeiro 2014.

VAN DER AALST, W. M.P.; TER HOFSTEDE, A.H.M.; WESKE, M. Business process management: a survey. In: INTERNATIONAL CONFERENCE ON BUSINESS PROCESS MANAGEMENT (BPM 2003). 2003. Berlin. **Proceedings…** Berlin, 2003. pp. 1-12.

VAN HEIJST, G.; VAN DER SPEK, R.; KRUIZINGA, E. **Organizing Corporate Memories**. – In: CIBIT Consultants. 1996. Disponível em: <http://www.dnvusa.com/Binaries/CorporateMemory_tcm153-295650.pdf>. Acesso em: 12 outubro 2014.

VERNADAT, F. B. **Enterprise modeling and integration: principles and applications**. London, UK: Chapman and Hall. 1996. 513 p.

VILLELA, C. S. S**. Mapeamento de processos como ferramenta de reestruturação e aprendizado organizacional**. Dissertação (Mestrado). Universidade Federal de Santa Catarina. Florianópolis. 2000. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/78638/171890.pdf?seque>. Acesso em: 22 junho 2014.

VILLELA, K.; TRAVASSOS, G. H.; ROCHA, A. R. C. **Definição e Construção de Ambientes de Desenvolvimento de Software Orientados a Organização**. 2004. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/sbqs/2004/032.pdf>. Acesso em: 11 abril 2014.

VILLELA, K.; SANTOS, G.; SCHNAIDER, L.; ROCHA, A. R.; TRAVASSOS, G. H. The use of an enterprise ontology to support knowledge management in software development environments. In: **Journal of the Brazillian Computer Society**, v. 11, n.2, 2005. pp. 45-59.

WHITE, S. A. **Introduction to BPMN.** 2004. Disponível em: <http://www.omg.org/bpmn/Documents/Introduction_to_BPMN.pdf>. Acesso em: 13 dezembro 2014.

WfMC - WORKFLOW MANAGEMENT COALITION. **The Workflow Reference Model**. 1995. Disponível em: <http://www.wfmc.org/docs/tc003v11.pdf>. Acesso em: 20 agosto 2014.

_____. **The Workflow Management Coalition – Terminology and Glossary**. 1999. Disponível em: <http://www.wfmc.org/docs/TC-1011_term_glossary_v3.pdf>. Acesso em: 20 agosto 2014.

_____. **Process Definition Interface. XML Process Definition Language**. **Version 2.2**. 2012. Disponível em: <http://www.xpdl.org/standards/xpdl-2.2/XPDL%202.2%20%282012-08-30%29.pdf>. Acesso em: 05 Agosto 2014.

W3C – WORLD WIDE WEB CONSORTIUM. **Extensible Markup Language (XML) 1.0**. 1998. Disponível em: <http://www.w3.org/TR/1998/REC-xml-19980210>. Acesso em: 15 setembro 2014.

_____. **OWL Web Ontology Language Guide**. 2004. Disponível em: <http://www.w3.org/TR/owl-guide/>. Acesso em: 14 agosto 2014.

_____. **OWL2 Web Ontology Language. Structural Specification and Functional-Style Syntax (Second Edition)**. 2012a. Disponível em: <http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/>. Acesso em: 28 maio 2014.

_____. **OWL2 Web Ontology Language. Document Overview (Second Edition)**. 2012b. Disponível em: <http://www.w3.org/TR/owl2-overview/>. Acesso em: 14 setembro 2014.

_____. **OWL2 Web Ontology Language. Manchester Syntax (Second Edition)**. 2012c. Disponível em: <http://www.w3.org/TR/2012/NOTE-owl2-manchester-syntax-20121211/>. Acesso em: 14 setembro 2014.

_____. **SPARQL 1.1 Query Language**. 2013. Disponível em: <http://www.w3.org/TR/sparql11-query/>. Acesso em: 20 agosto 2014.

_____. **RDF Schema 1.1**. 2014a. Disponível em: <http://www.w3.org/TR/rdf-schema/>. Acesso em: 17 agosto. 2014.

_____. **RDF 1.1 Primer**. 2014b. Disponível em: <http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140225/>. Acesso em: 17 agosto. 2014.

APÊNDICE A – ESTRUTURAS DE OWL UTILIZADAS NO TRABALHO

Este Apêndice complementa a apresentação das estruturas básicas da linguagem OWL apresentadas na seção 2.4.2. Trata-se de estruturas relevantes para este trabalho, pois são estruturas essenciais para a construção de ontologias e também para o correto entendimento sobre a metodologia apresentada no *capítulo 4*. Tais estruturas foram utilizadas tanto na definição da metodologia, quanto no desenvolvimento da ferramenta *RDFtoXPDL*, As estruturas apresentadas são: propriedades "Object Property", "Data Property"; restrições "someValuesFrom", "allValuesFrom", "hasValue"; estruturas de equivalência "equivalentClass"; e combinações boolenas "unionOf", "oneOf", "intersectionOf", "complementOf".

As propriedades são utilizadas para descrever relacionamentos entre indivíduos ou entre um indivíduo e um valor literal. As propriedades permitem afirmar fatos sobre os membros das classes, assim como também podem afirmar fatos sobre indivíduos das classes. As propriedades estão divididas em duas categorias (SMITH; WELTY; MCGUINNESS, 2004):

- propriedades de dados tipados: indicam uma relação entre indivíduos e valores literais. Este tipo de propriedade é definido como uma instância da classe "owl:DataTypeProperty";
- propriedades de objetos: indicam uma relação entre indivíduos. Este tipo de propriedade é definido como instância da classe "owl:ObjectProperty".

Nas **Figuras 80** e **81** é possível observar a declaração de uma propriedade do tipo "objeto" e uma do tipo "dado tipado", respectivamente. Todas as referências de valores literais possíveis para propriedades do tipo "dado tipado" e também maiores explicações sobre declarações de valores internos das propriedades podem ser encontradas em (SMITH; WELTY; MCGUINNESS, 2004).

```
<owl:ObjectProperty rdf:ID="madeFromGrape">
  <rdfs:domain rdf:resource="#Wine"/>
  <rdfs:range rdf:resource="#WineGrape"/>
</owl:ObjectProperty>
```

Figura 80 - Declaração de uma propriedade de objeto.

Fonte: extraído de W3C (2004).

```
<owl:DatatypeProperty rdf:ID="yearValue">
  <rdfs:domain rdf:resource="#VintageYear" />
  <rdfs:range  rdf:resource="&xsd;positiveInteger"/>
</owl:DatatypeProperty>
```

Figura 81 - Declaração de uma propriedade de dado tipado.

Fonte: extraído de W3C (2004).

As propriedades do tipo objeto servem como meio de interligação entre dois elementos, ou seja, considerando a definição de triplas, a “Object Property” atua como sendo o predicado da sentença. A **Figura 82** demonstra o uso dessa propriedade, onde a “Classe A“ é o sujeito e a “Classe B” é o objeto, sendo interligado pela propriedade “temLigacao”.

```
<owl:ObjectProperty rdf:ID="temLigacao">
</owl:ObjectProperty>

<Class ID="Classe A">
<subClassOf>
    <Restriction>
        <onProperty rdf:resource="#temLigacao"/>
        <someValuesFrom rdf:resource="Classe B"/>
    </Restriction>
</subClassOf>
</Class>

<Class ID="Classe B">
</Class>
```

Figura 82 - Exemplo de uso da “Object Property” “temLigacao”.

As “Object Property” podem ser utilizadas sempre que houver a necessidade de relacionar dois indivíduos. Destaca-se que a linguagem OWL permite descrever o caminho inverso ao seguido pelas “Object Property” através da propriedade simétrica “owl:inverseOf”. Observa-se, contudo, que o consórcio W3C menciona que não é necessário tal declaração, pois já fica subtendido, através do uso da “Object Property”,

a existência do caminho inverso (W3C, 2004). A **Figura 83** apresenta um exemplo do uso da propriedade inversa: “éLigadoA”, onde o elemento “ClasseA” relaciona com “ClasseB” através da propriedade “temLigacao”, e para indicar o caminho inverso a estrutura do elemento “ClasseB” indica uma relação inversa com “ClasseA” através da propriedade “éLigadoA”.

```
<owl:ObjectProperty rdf:ID="temLigacao">
    <owl:inverseOf rdf:resource="#éLigadoA"/>
</owl:ObjectProperty>

<Class ID="Classe A">
<subClassOf>
    <Restriction>
        <onProperty rdf:resource="#temLigacao"/>
        <someValuesFrom rdf:resource="Classe B"/>
    </Restriction>
</subClassOf>
</Class>

<Class ID="Classe B">
<subClassOf>
    <Restriction>
        <onProperty rdf:resource="#éLigadoA"/>
        <someValuesFrom rdf:resource="Classe A"/>
    </Restriction>
</subClassOf>
</Class>
```

Figura 83 - Exemplo do uso da propriedade inversa.

A linguagem OWL permite ainda que seja possível incluir restrições sobre as propriedades já mencionadas: “Object Property” e “DatatypeProperty”. Uma restrição pode ser descrita como a definição de uma classe anônima de indivíduos que satisfazem as seguintes restrições: “allvaluesFrom”, “someValuesFrom” e “hasValue”.

A restrição “allValuesFrom” indica que quando uma propriedade for utilizada dentro de uma instância de classe, esta deve ter vinculada a ela apenas valores que sigam a restrição determinada. A **Figura 84** mostra que, para cada instância da classe “PotableLiquid” que utiliza a propriedade “hasMaker”, todos os valores associados a essa propriedade devem ser do tipo “Winery”. O uso da propriedade “hasMaker” não é obrigatório, mas se essa propriedade for utilizada, os valores associados devem ser do tipo “Winery”.

É interessante observar o uso da construção “owl:Restriction” na **Figura 84**,

que indica uma classe anônima dentro de outra classe (neste caso, a classe “Wine”). A definição de classes anônimas evita a necessidade de se nomear cada classe dentro de outra. Além disso, fornece um poderoso mecanismo de definição de classes baseado na satisfação de uma propriedade (W3C, 2004).

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid" />
  ...
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMaker" />
      <owl:allValuesFrom rdf:resource="#Winery" />
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Figura 84 – Exemplo de declaração de uma restrição “allValuesFrom”.

Fonte: extraído de W3C (2004).

Já a restrição “someValuesFrom” indica que pelo menos um dos valores indicados na restrição deve ser associado à propriedade. Através da **Figura 85** pode se observar que pelo menos uma instância da classe “Wine” que possua a propriedade “hasMaker”, deve ter obrigatoriamente o valor associado a essa propriedade como sendo do tipo classe “Winery” (W3C, 2004).

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid" />
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMaker" />
      <owl:someValuesFrom rdf:resource="#Winery" />
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Figura 85 – Exemplo de declaração da propriedade “someValuesFrom”.

Fonte: extraído de W3C (2004).

Por fim, a restrição “hasValue” indica que apenas um valor pode ser associado à elementos que possuam tal propriedade. Na **Figura 86** é representada uma situação onde um indivíduo será membro da classe “Burgundy” se e somente se possuir o valor “Dry” na propriedade “hasSugar” (W3C, 2004).

Dentro da definição de uma ontologia pode-se indicar que diferentes classes são equivalentes entre si, ou seja, que possuem as mesmas extensões de classe e são sinônimas entre si. A estrutura “equivalentClass” ainda pode indicar a equivalência de indivíduos de ontologias diferentes. Na **Figura 87** pode-se identificar que a classe “Wine” é também instância da classe “&vin:Wine” de uma segunda ontologia.

```
<owl:Class rdf:ID="Burgundy">
  ...
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasSugar" />
      <owl:hasValue rdf:resource="#Dry" />
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Figura 86 – Exemplo de declaração da propriedade “hasValue”.

Fonte: extraído de W3C (2004).

```
<owl:Class rdf:ID="Wine">
  <owl:equivalentClass rdf:resource="&vin;Wine"/>
</owl:Class>
```

Figura 87 – Exemplo de declaração da propriedade “equivalentClass”.

Fonte: extraído de W3C (2004).

A linguagem OWL permite que se possa definir, já na construção de uma classe, todo e qualquer indivíduo que será instância da respectiva classe através da propriedade “owl:oneOf” (W3C, 2004). Na **Figura 88** é possível visualizar que apenas os indivíduos “White”, “Rose” e “Red” podem ser indivíduos da classe “WineColor”.

```
<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
    <owl:Thing rdf:about="#Rose"/>
    <owl:Thing rdf:about="#Red"/>
  </owl:oneOf>
</owl:Class>
```

Figura 88 – Exemplo de declaração da propriedade “oneOf”.

Fonte: extraído de W3C (2004).

Por fim, a linguagem OWL possibilita realizar algumas combinações booleanas

para manipular extensões de classes. Essas combinações são chamadas de “conjuntos de operadores”. Também podem ser visualizados como sendo uma representação dos operadores de lógica descritiva AND, OR e NOT, segundo Bechhofer et al.(2004). De acordo com os autores, os operadores podem ser de três tipos: “owl:intersectionOf”, “owl:unionOf” *e* “owl:complementOf”:

O operador “owl:intersectionOf” declara uma lista de indivíduos que, através da intersecção destes, irá estender a classe que os declarou. Na **Figura 89** pode-se notar que a classe “WhiteWine” é a intersecção de todas as classes “Wine” com o conjunto de classes que possuírem a propriedade “hasColor” com o valor “White”.

```
<owl:Class rdf:ID="WhiteWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#White" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Figura 89 - Declaração do operador “intersectionOf”.

Fonte: extraído de W3C (2004).

O operador “owl:unionOf” declara uma lista de indivíduos onde a união de todos os indivíduos irá se extender para uma segunda classe. Na **Figura 90** pode-se observar que a classe “Fruit” é formada pela união das classes “SweetFruit” e “NonSweetFruit”.

```
<owl:Class rdf:ID="Fruit">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

Figura 90 - Declaração do operador “unionOf”.

Fonte: extraído de W3C (2004).

O operador “owl:complementOf”, por sua vez, declara uma classe que possui elementos não contidos em uma segunda classe. Na **Figura 91** a classe “NonConsumableThing” contem todos os indivíduos que não pertencem a extensão de classe “ConsumableThing”, indicando portanto uma negação.

Face ao exposto, nesta seção foram abordadas estruturas que foram relevantes para este trabalho. Mais informações sobre a estrutura do OWL podem ser encontradas na especificação da linguagem em W3C (2004). A especificação da versão mais atual do OWL, também chamada de OWL2, respeita a estrutura presente na primeira versão da linguagem, e pode ser encontrada em W3C (2012a).

```
<owl:Class rdf:ID="NonConsumableThing">
  <owl:complementOf rdf:resource="#ConsumableThing" />
</owl:Class>
```

Figura 91 - Declaração do operador “complementOf”.

Fonte: extraído de W3C (2004).

APÊNDICE B – PROGRAMA DE MELHORIA DE PROCESSO DE SOFTWARE BRASILEIRO (MPS.BR)

O programa de Melhoria de Processo de Software Brasileiro (MPS.BR), criado em dezembro de 2003, é mantido pela Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), que conta com o apoio de outras instituições como: Ministério da Ciência e Tecnologia e Inovação (MCTI), Agência Financiadora de Estudos e Projetos (FINEP), Serviço Brasileiro de Apoio às Micros e Pequenas Empresas (SEBRAE) e Banco Interamericano de Desenvolvimento (BID).

O principal objetivo do programa MPS.BR é definir modelos de melhoria e avaliação de processos de software. Como já dito na *seção 2.4* o programa visa atender, preferencialmente, as micros, pequenas e médias empresas (mPME), embora se mostre adequado às grandes empresas produtoras de software. Através de recursos captados pela Softex (Associação para Promoção da Excelência do Software Brasileiro), o programa propicia financiamentos para grupos de mPME implementarem os modelos (SOFTEX, 2012b). A intenção é que o programa seja reconhecido nacional e internacionalmente, provendo modelos aplicáveis à indústria de software (SOFTEX, 2012a).

O programa MPS.BR contempla quatro componentes, descritos através de documentos denominados “Guias”, conforme ilustrado na **Figura 92** (SOFTEX, 2012a):

(1) Modelo de Referência para software (MR-MPS-SW);

(2) Modelo de Referência para Serviço (MR-MPS-SV);

(3) Método de Avaliação (MA-MPS);

(4) Modelo de Negócio (MN-MPS).

Tanto o MR-MPS-SW como o MR-MPS-SV contêm os requisitos que os processos das unidades organizacionais devem atender para estar em conformidade com o programa MPS-BR. Conforme ilustrado na **Figura 92**, a construção dos modelos de referência foi baseada em algumas normas internacionais significativas para o desenvolvimento de software: NBR ISO/IEC 12207, referente ao processo do ciclo de vida do software, NBR ISO/IEC 15504, voltada ao processo de avaliação do

software, e NBR ISO/IEC 20000, referente a gerenciamento de serviços. Também deve ser observado na **Figura 92** a influência do modelo CMMI (*Capability Maturity Model Integration*) sobre o MPS.BR, sendo CMMI-DEV (*Capability Maturity Model Integration for Development*) como apoio ao MR-MPS-SW, e CMMI-SVC (*Capacility Maturity Model Integration for Services*) ao MR-MPS-SV. Observa-se que o modelo MPS-SW contempla dois guias em destaque na **Figura 92:** Guia de Aquisição, contendo boas práticas para aquisição de softwares correlatos, e Guia de Implementação, sugerindo formas de implementação para cada um dos níveis do modelo MR-MPS-SW em diversos tipos de organizações. Como o foco desse trabalho recai sobre a produção de software, e não sobre serviços de software, o MR-MPS-SW, também referenciado como modelo MPS-SW, será melhor descrito no tópico seguinte.

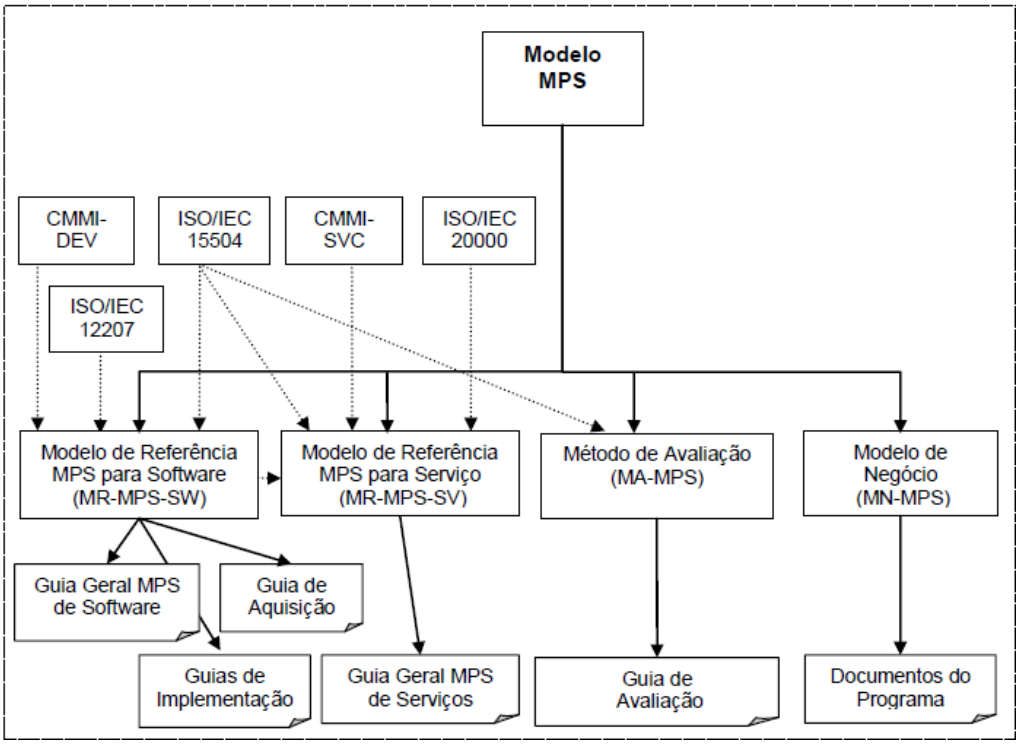


Figura 92 - Componentes do programa MPS.BR.

Fonte: extraído de Softex (2012a).

O Modelo de Negócio (MN-MPS) descreve regras de negócio necessárias para diversas atividades do programa, como, por exemplo: a implementação dos modelos MR-MPS-SW e MR-MPS-SV, assim como a avaliação por Instituições Avaliadoras (IA) seguindo o Modelo de Avaliação (MA-MPS). O modelo MA-MPS, por sua vez, descreve o processo e o método de avaliação que as IA dos modelos devem seguir.

Modelo de referência MPS para software (MR-MPS-SW)

De modo geral, o modelo MPS-SW descreve práticas para a melhoria gradual e contínua de processos. Como ilustrado na **Tabela 12**, o modelo define sete níveis de maturidade, de G a A, sendo G o nível inicial, já passivo de certificação, e A o nível mais alto. Cada nível possui processos com objetivos específicos que devem ser alcançados através de resultados esperados.

Tabela 12 - Níveis de maturidade (MPS.BR).

Fonte: extraído de Softex (2012a).

Níveis	Descrição dos Níveis
A	Em otimização
B	Gerenciado Quantitativamente
C	Definido
D	Largamente Definido
E	Parcialmente Definido
F	Gerenciado
G	Parcialmente Gerenciado

Os níveis de maturidade representam a evolução dos processos, que em cada nível de maturidade precisam ter sua implementação. O nível de maturidade em que se encontra uma organização indica qual o desempenho esperado ao executar um processo (SOFTEX, 2012a). Essa divisão de níveis, maior que o CMMI-DEV, torna o modelo atraente para as mPME, pois é possível avaliar e perceber resultados em um menor prazo de tempo.

A capacidade do processo define atributos para alcançar os resultados esperados, onde cada atributo expressa um grau de refinamento com o qual o processo foi executado. Em cada um dos níveis de maturidade são definidas “áreas” de processos que indicam onde a organização deve focar o esforço de melhoria. O alcance de um determinado nível de maturidade se obtém quando os atributos de processo e todos os resultados esperados de cada processo daquele determinado nível são atendidos.

Os processos no MR-MPS-SW são descritos em termos de propósito, resultados e informações adicionais, descritos na **Tabela 13**.

Tabela 13 - Descrição dos termos dos processos do MR-MPS-SW.

Termos	Descrição
Propósito	Objetivo geral a ser atingido durante a execução daquele processo específico.
Resultados esperados	Resultados a serem obtidos com a implementação do processo.
Informações adicionais	Referências que podem ajudar na definição do processo pela organização.

A **Figura 93** apresenta os níveis de maturidade do MR-MPS-SW, com seus respectivos processos e atributos de processo.

Nível	Processos	Atributos de Processo
A		AP 1.1, AP 2.1, AP 2.2, AP 3.1, AP 3.2, AP 4.1, AP 4.2 , AP 5.1 e AP 5.2
B	Gerência de Projetos – GPR (evolução)	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2, AP 4.1 e AP 4.2
C	Gerência de Riscos – GRI	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Desenvolvimento para Reutilização – DRU	
	Gerência de Decisões – GDE	
D	Verificação – VER	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Validação – VAL	
	Projeto e Construção do Produto – PCP	
	Integração do Produto – ITP	
	Desenvolvimento de Requisitos – DRE	
E	Gerência de Projetos – GPR (evolução)	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência de Reutilização – GRU	
	Gerência de Recursos Humanos – GRH	
	Definição do Processo Organizacional – DFP	
	Avaliação e Melhoria do Processo Organizacional – AMP	
F	Medição – MED	AP 1.1, AP 2.1 e AP 2.2
	Garantia da Qualidade – GQA	
	Gerência de Portfólio de Projetos – GPP	
	Gerência de Configuração – GCO	
	Aquisição – AQU	
G	Gerência de Requisitos – GRE	AP 1.1 e AP 2.1
	Gerência de Projetos – GPR	

Figura 93 – Níveis de maturidade, processos e atributos de processo do MR-MPS-SW.

Fonte: extraído de Softex (2012a).

O atendimento aos atributos do processo (AP), feito através do atendimento aos resultados esperados (RAP), é necessário para todos os processos no nível de maturidade correspondente, embora eles não sejam detalhados dentro de cada processo. Já a capacidade do processo é representada por um conjunto de atributos de processo descrito através de resultados esperados. A capacidade do processo diz respeito ao nível de evolução com que o processo é executado na organização. Conforme a organização evolui nos níveis de maturidade, a capacidade de processo também evolui, exigindo um maior desempenho do que no nível anterior (SOFTEX, 2012a).

Os níveis de capacidade e maturidade são acumulativos, ou seja, se a organização está no nível F, esta possui o nível de capacidade do nível F, que inclui os atributos de processo dos níveis G e F para todos os processos pertencentes ao nível de maturidade F, que, por sua vez, inclui os processos do nível G. Portanto ao passar de um nível para outro, os processos executados anteriormente passam a ser executados com uma maior capacidade no nível atual (SOFTEX, 2012a).

Cada processo possui seus próprios resultados esperados, que precisam obrigatoriamente ser alcançados para que o processo seja considerado atendido, de acordo com os AP. Considerando o propósito deste trabalho, a **Tabela 14** apresenta os processos dos níveis G e F, e as suas quantidade de resultados esperados.

Tabela 14 - Processos dos níveis G e F e seus RAPs.

Processos	Quantidade de RAP	Descrição dos RAP
Gerência de Projetos	19 (até o nível F)	GPR1 ao GPR19
Gerência de Requisitos	5	GRE1 ao GRE5
Aquisição	8	AQU1 ao AQU8
Gerência de Configuração	7	GCO1 ao GCO7
Gerência de Portfólio de Projetos	8	GPP1 ao GPP8
Garantia da Qualidade	4	GQA1 ao GQA4
Medição	7	MED1 ao MED7

Vale ressaltar que alguns processos e até mesmo alguns RAP podem ser excluídos, total ou parcialmente, do escopo de uma avaliação por não serem

pertinentes ao negócio da organização em questão. Maiores informações podem ser encontradas em Softex (2012a).

A capacidade do processo no MR-MPS-SW possui nove AP: AP 1.1, AP 2.1, AP 2.2, AP 3.1, AP 3.2, AP 4.1, AP 4.2, AP 5.1 e AP 5.2. Cada AP está detalhado em termos de resultados esperados do atributo de processo (RAP), que precisam ser completamente atendidos para que o AP seja alcançado, conforme definido na **Tabela 15.**

Tabela 15 - Atributos de processos (AP) do MR-MPS-SW.

Atributo	Descrição (evidencia o quanto ...)
AP 1.1	O processo atinge o seu propósito.
AP 2.1	A execução do processo é gerenciada.
AP 2.2	Os produtos de trabalho produzidos pelo processo são gerenciados apropriadamente.
AP 3.1	Um processo padrão é mantido para apoiar a implementação do processo definido.
AP 3.2	O processo-padrão é efetivamente implementado como um processo definido para atingir seus resultados.
AP 4.1	Os resultados de medição são usados para assegurar que a execução do processo atinge seus objetivos de desempenho e apoia o alcance dos objetivos de negócio
AP 4.2	O processo é controlado estatisticamente para produzir um processo estável, capaz e previsível dentro de limites estabelecidos.
AP 5.1	As mudanças no processo são identificadas a partir da análise de defeitos em geral e da investigação de enfoques inovadores para a definição e implementação do processo.
AP 5.2	As mudanças na definição, gerência e desempenho do processo têm impacto efetivo para o alcance dos objetivos relevantes de melhoria de processo.

APÊNDICE C – EXEMPLO DE PROCESSO REPRESENTADO EM XPDL

Este apêndice apresenta um exemplo da conversão, de um processo modelado em BPMN, para linguagem XPDL. A **Figura 94** apresenta um processo contendo seis elementos (uma piscina, dois fluxos de sequência, uma atividade, um evento de início e um evento de término) representados em notação gráfica (BPMN).

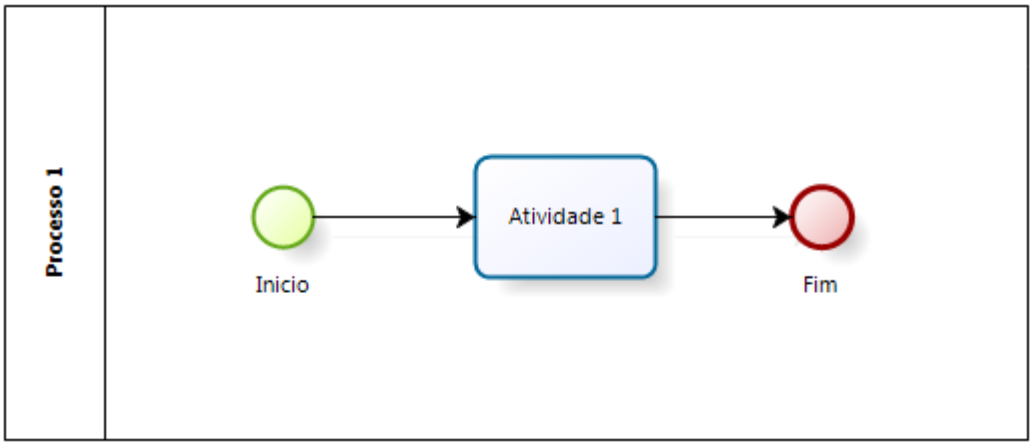


Figura 94 - Exemplo de processo em notação gráfica.

A **Figura 95** apresenta o mesmo processo da **Figura 94**, agora representado em linguagem XPDL. O processo identificado como “Processo 1” possui apenas três elementos: um evento de início, uma atividade (“Atividade 1”) e um evento de término do fluxo do processo. A figura mostra também os elementos de “fluxo de sequência”, que representam a ordem do fluxo, indo do elemento “Inicio” para “Atividade 1”, e de “Atividade 1” para o elemento “Fim”. Observa-se que devido ao tamanho das estruturas, as estruturas que representam as posições (coordenadas X e Y) dos elementos não são exibidas na figura. Porém, é importante ressaltar que tais estruturas são obrigatórias (ver *seção 3.3*), e em um exemplo real apareceriam em cada um dos elementos pertencentes ao diagrama do processo. As estruturas de código apresentadas estão baseadas nas estruturas constantes da **Tabela 8**, na *seção 3.3*, correspondentes aos elementos nucleares da notação BPMN.

```
<?xml version="1.0" encoding="utf-8"?>
<Package >
  <PackageHeader />
  <RedefinableHeader />
  <ExternalPackages />
  <Pools>
    <Pool Name="Processo 1" Process="Processo 1" >
      <Lanes />
    </Pool>
  </Pools>
  <Associations />
  <Artifacts />
  <WorkflowProcesses>
    <WorkflowProcess Id="Processo 1" Name="Processo 1">
      <ActivitySets />
      <DataInputOutputs />
      <Activities>
        <Activity Name="Início">
          <Event>
            <StartEvent Trigger="None" />
          </Event>
        </Activity>
        <Activity Name="Atividade 1">
          <Implementation>
            <Task />
          </Implementation>
          <Loop LoopType="None" />
        </Activity>
        <Activity Name="Fim">
          <Event>
            <EndEvent Result="None" />
          </Event>
        </Activity>
      </Activities>
      <Transitions>
        <Transition From="Início" To="Atividade 1">
          <Condition />
        </Transition>
        <Transition From="Atividade 1" To="Fim">
          <Condition />
        </Transition>
      </Transitions>
      <ExtendedAttributes />
    </WorkflowProcess>
  </WorkflowProcesses>
  <ExtendedAttributes />
</Package>
```

Figura 95 - Exemplo de código em XPDL.

APÊNDICE D – FERRAMENTA RDFTOXPDL

Este apêndice traz informações sobre - o processo de desenvolvimento da ferramenta *RDFtoXPDL* (*parser*), utilizada na etapa 3 da metodologia proposta na *seção 4.1.* O código e o executável da ferramenta RDFtoXPDL encontram-se em formato digital, em CD-R anexo a esta dissertação, em pasta denominada "ApendiceD-FerramentaRDFtoXPDL". A ferramenta foi concebida com a finalidade de automatizar o mapeamento de uma ontologia empresarial em formato RDF/XML, com as devidas marcações propostas neste trabalho, para um modelo de processos com a notação BPMN 2.0.

A ferramenta foi desenvolvida para a plataforma Windows Vista ou superior. Possui aproximadamente três mil linhas de código e foi implementada utilizando a linguagem de programação C#, através do *framework Microsoft .NET 4.5,* com a plataforma de desenvolvimento Visual Studio 2013.

Considerando que o arquivo de entrada deve estar em formato RDF/XML, foi utilizada uma linguagem de consulta de dados em arquivos RDF conhecida como SPARQL (*SPARQL Protocol and RDF Query Language*), versão 1.1, já apresentada no *capítulo 2* (*seção 2.5*). SPARQL pode ser utilizada através de interfaces de programação conhecidas como API (*Application Programming Interface*). Considerando que este trabalho utiliza a tecnologia *Microsoft .NET Framework*, foi necessário encontrar APIs voltadas para essa tecnologia. Para isso, foi selecionada a API "*dotnetrdf*", que se encontra na versão 1.0.6 e está disponível em *http://dotnetrdf.org/*. A API foi selecionada por ser de uso livre, receber atualizações de dois em dois meses e implementar a linguagem SPARQL na íntegra, de maneira fiel ao padrão especificado pelo consórcio W3C.

Já para fazer a conversão dos elementos da ontologia em RDF para um modelo de processos, que, posteriormente, pudesse ser interpretado como notação BPMN, foi utilizada a linguagem de serialização XPDL, versão 2.2. Essa versão foi escolhida porque as versões anteriores não possuem suporte a versão atual do BPMN 2.0.

A lógica de funcionamento do sistema *RDFtoXPDL* pode ser representada no algoritmo apresentado na **Figura 96**.

```
1   Consultar os dados no arquivo RDF através de uma consulta em SPARQL
2   Se existir resultados
3       Criar estrutura RDFSchema para armazenar os dados da ontologia
4       Buscar elementos retornados pela consulta SPARQL, considerando suas hierarquias
5          Para cada um dos elementos retornados, fazer
6               Verificar o tipo de elemento sendo lido, de acordo com suas marcações
7               Inserir os elementos na estrutura RDFSchema, caso ainda não exista
8       Criar estrutura XPDLSchema para armazenar os dados que serão convertidos
9       Buscar elementos da estrutura RDFSchema
10         Para cada um dos elementos retornados, fazer
11              Verificar se já não existe na estrutura XPDLSchema;
12              Identificar o elemento correspondente em XPDL, e converter
13              Criar elementos "automáticos" de acordo com a necessidade
14              Gravar na estrutura XPDLSchema;
15      Adicionar fluxo de sequência para todos os elementos da estrutura XPDLSchema
16      Adicionar posições coordenadas para todos os elementos da estrutura XPDLSchema
17      Serializar estrutura XPDLSchema e gerar arquivo XPDL
18  Se não FIM
```

Figura 96 - Lógica de funcionamento da ferramenta *RDFtoXPDL*.

A primeira linha da **Figura 96** mostra que uma consulta em SPARQL foi criada para execução a fim de retornar os dados a serem mapeados da ontologia. A consulta toma por base somente as marcações e as "Object Property" consideradas na etapa 2 da metodologia proposta. Logo, a consulta só retornará os elementos considerados nessas marcações.

A **Figura 97** apresenta uma parte da estrutura da consulta criada para o retorno dos dados. Observa-se que a quinta linha do código da **Figura 97** ("?superClass owl:whatType ?superType") limita qualquer dado a ser retornado através do marcador "owl:whatType", conforme requerido pela metodologia. A oitava linha, por sua vez, mostra a restrição das propriedades "Object Property", também exigida pela metodologia.

Após a consulta dos dados em RDF, foi utilizado um algoritmo de busca em profundidade para identificar cada um dos elementos retornados e, assim, montar a hierarquia correta dos elementos, de acordo com suas respectivas relações. O algoritmo assume como elemento de maior nível o elemento com a marcação "owl:whatType" e de valor "Process". A partir desse elemento, busca cada um dos seus respectivos elementos relacionados até que sejam encontrados todos os elementos pertencentes ao fluxo do processo.

O algoritmo apresentado na **Figura 96** repete os passos de busca em profundidade (linhas 4, 5, 6 e 7) para cada elemento de maior nível encontrado. Assim, todos os elementos e suas respectivas relações terão sido armazenados em uma estrutura intermediária, chamada de *RDFSchema*. A estrutura segue exatamente a mesma hierarquia de valores propostos na etapa 2.2 da metodologia, de acordo com a marcação "owl:whatType". A **Figura 98** apresenta a classe principal da estrutura

gerada. O diagrama de classes completo da estrutura *RDFSchema* pode ser encontrado em formato digital, em CD-R anexo a esta dissertação, em pasta denominada “ApendiceD-FerramentaRDFtoXPDL”.

```
SELECT distinct ?superClass ?superType ?superLabel ?superOrder ?superEvent
    ?propert ?subClass ?subType ?subLabel ?subOrder ?subEvent
WHERE {
    ?superClass a owl:Class.
    ?superClass owl:whatType ?superType.
    ?superClass rdfs:label ?superLabel.
    ?superClass rdfs:subClassOf ?object.
    ?object owl:onProperty ?propert;
       owl:someValuesFrom|owl:allValuesFrom|owl:hasValue ?subClass.
    ?subClass owl:whatType ?subType.
    ?subClass rdfs:label ?subLabel.
    OPTIONAL { ?superClass owl:Order ?superOrder. }
    OPTIONAL { ?superClass owl:Event ?superEvent. }
    OPTIONAL { ?subClass owl:Order ?subOrder. }
    OPTIONAL { ?subClass owl:Event ?subEvent. }
    }
```

Figura 97 – Parte do código da consulta de dados RDF na ferramenta *RDFtoXPDL*.

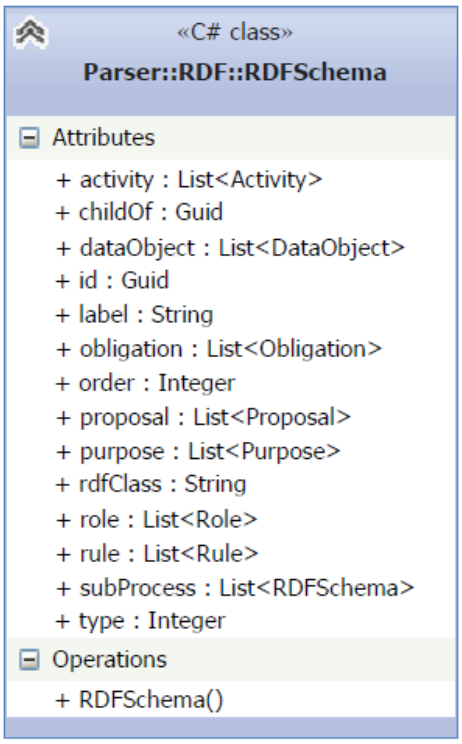


Figura 98 – Classe principal da estrutura *RDFSchema*.

Os elementos lidos da ontologia são inseridos na estrutura *RDFSchema*, considerando três identificadores básicos: (1) um identificador único para cada elemento (“id”); (2) um identificador que define o tipo de cada elemento (“type”); (3) um identificador que aponta para o elemento de ordem superior do elemento (“childOf”), isto é, seu elemento “pai”. Os elementos que possuem ordenação, de acordo com o marcador “owl:Order”, também possuem um identificador que indica a sequência do elemento no fluxo do processo (“order”). Porém, de acordo com a metodologia, essa marcação não é obrigatória e, portanto, caso nenhum valor seja enviado da ontologia, o identificador “order” assumirá como valor (*default*) a ordem de leitura dos elementos vindos da ontologia.

Com a estrutura devidamente preenchida, o próximo passo é converter os elementos contidos na estrutura *RDFSchema* para elementos de um modelo de processos, considerando a linguagem XPDL, utilizada para a conversão. Dessa forma, foi criada uma segunda estrutura, chamada *XPDLSchema*, respeitando a especificação da linguagem XPDL 2.2.

É importante ressaltar que a estrutura *XPDLSchema* não foi criada levando em consideração a especificação completa da linguagem, mas apenas os elementos necessários para a realização deste trabalho. A **Tabela 16** mostra os elementos considerados e seus respectivos tipos, de acordo com a especificação da notação BPMN 2.0. Além dos elementos apresentados na **Tabela 16**, o elemento “ExtendedAttribute” presente na linguagem XPDL também foi considerado na estrutura.

Tabela 16 - Elementos de BPMN considerados na estrutura *XPDLSchema*.

Estrutura BPMN	
Elemento	**Tipo**
Events	Start/End (Default)
Task	-
Gateways	Exclusive
Artifacts	Text Annotation
Data	Data Object
Subprocess	Embedded
Pool	-
Lane	-
Sequence Flow	-
Association	-

Neste trabalho, optou-se por não utilizar o "*Schema*" pronto da linguagem XPDL, disponibilizado pelo consórcio WfMc e também pelo endereço *http://www.xpdl.org/standards/xpdl-2.2/bpmnxpdl_40a.xsd*. Isso porque essa estrutura contempla elementos que já não fazem mais parte das novas versões da linguagem XPDL, o que torna confuso seu uso. Portanto, a estrutura *XPDLSchema* foi criada de modo a ser similar à estrutura contida no XPDL "*Schema"* padrão, porém contendo um conjunto menor de elementos. A **Figura 99** apresenta a classe principal da estrutura *XPDLSchema* gerada. O diagrama de classes completo da estrutura *XPDLSchema* pode ser encontrado em formato digital, em CD-R anexo a esta dissertação, em pasta denominada "ApendiceD-FerramentaRDFtoXPDL".

Com a estrutura criada, um segundo algoritmo, também baseado em busca em profundidade, foi aplicado na estrutura *RDFSchema.* O algoritmo também se baseia no elemento de maior nível hierárquico ("Process"). A partir desse elemento, busca cada um dos elementos pertencentes à sua hierarquia, de modo: - a ler o elemento; - identificar sua correspondência com elementos de XPDL, através do identificador "*type*" da estrutura *RDFSchema*; - converter o elemento para o padrão XPDL; - gravar na estrutura *XPDLSchema.*

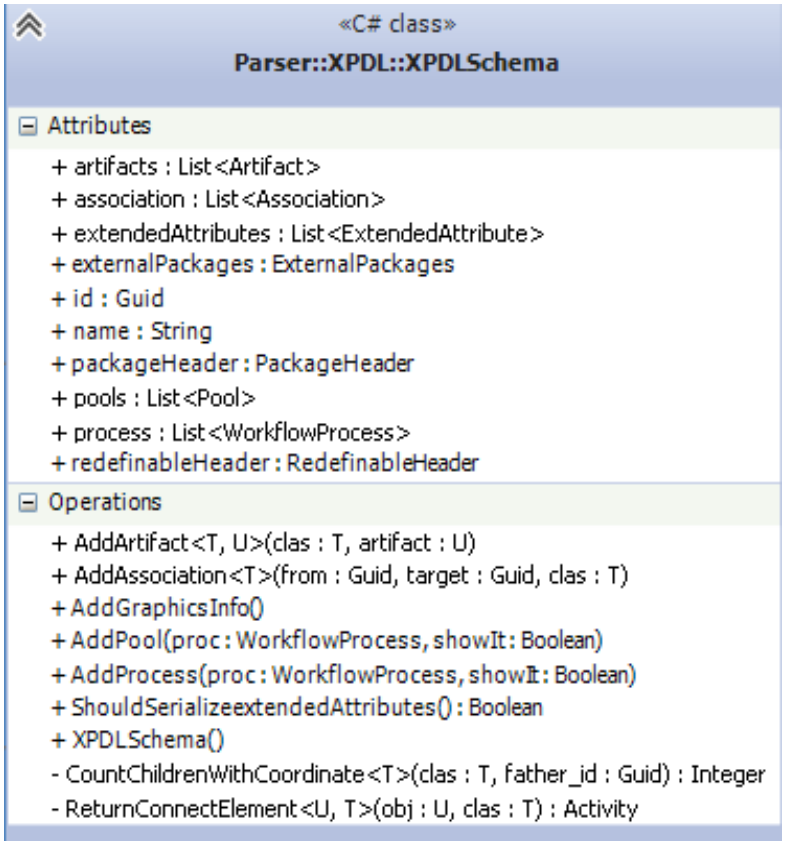


Figura 99 - Classe da estrutura XPDLSchema.

Considerando que um mesmo elemento “X” pode ser utilizado em diferentes lugares do fluxo do processo, verificações são feitas no momento de inserção dos elementos na estrutura XPDLSchema. A intenção é garantir que nenhum elemento seja inserido mais de uma vez, e que a referência em cima do elemento “X” seja a mesma durante todo o fluxo do processo, respeitando, assim, os conceitos de URIs. A **Figura 100** mostra um exemplo onde um objeto “ResultA” gerado a partir da “Tarefa A” é consumido pela “Tarefa B” e pela “Tarefa C”. Nesse caso, as três representações referem-se ao mesmo elemento.

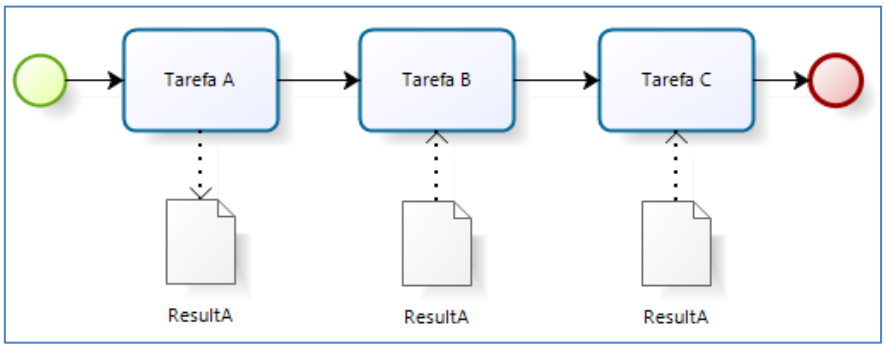


Figura 100 - Exemplo de representações de um mesmo elemento.

Assim, os elementos são convertidos para a estrutura *XPDLSchema*, seguindo as regras consideradas na etapa 2 da metodologia proposta. No momento da conversão do elemento, também já é realizada a tratativa para determinados elementos, como segue:

- Um elemento “Pool” é gerado no momento de criação de um elemento do tipo “Process”;
- Um elemento “Associação” é gerado no momento de criação de um elemento do tipo “Anotação” ou “Objeto de Dados”. Esse elemento sempre aponta para o elemento “pai” contido na estrutura do elemento “Anotação” ou “Objeto de Dados”;
- Um elemento “Evento de início” é gerado no momento de criação de um elemento do tipo “Subprocesso” ou “Processo”;
- Um elemento “Evento de fim” é gerado ao término da leitura de todos os elementos contidos dentro de um “Processo” ou “Subprocesso”.

Com todos os elementos mapeados, o próximo passo necessário é o de interligar cada um dos elementos lidos, isto é, conectar os elementos através de um "fluxo de sequência". Assim, através da ordenação estipulada pela marcação "owl:Order", caso ela tenha sido utilizada, ou através da ordenação feita pela própria ferramenta *RDFtoXPDL*, o fluxo do processo é criado, conectando sempre os elementos aos pares. Isso significa que, se considerados os elementos "X", "Y" e "Z" de um processo, para interliga-los, primeiramente são conectados os elementos "X" e "Y" e, posteriormente, os elementos "Y" a "Z".

Por fim, como mencionado na *seção 3.3*, a linguagem XPDL exige que sejam indicadas posições em formato de coordenadas X e Y, para que os elementos possam ser representados corretamente no modelo de processos em BPMN. Portanto, o último passo da ferramenta *RFDtoXPDL* (linha 16 da **Figura 96**) é fornecer coordenadas para todos os elementos presentes na estrutura *XPDLSchema*.

Para tanto, foi criado um algoritmo que percorre o fluxo do processo do primeiro elemento ao último, inserindo coordenadas para cada um deles. A cada elemento novo, o algoritmo verifica se a posição que deveria ser usada para o novo elemento já é utilizada; Em caso positivo, calcula uma nova posição. O algoritmo permite que sejam inseridos até no máximo cinco elementos por linha, dentro de um elemento "Pool" - acima desse valor, o algoritmo irá passar para a próxima linha. Caso exista mais de um "Process" no mesmo modelo de processo, o algoritmo irá repetir os passos para cada "Process", e, por padrão, assume que os elementos "Process" devem ser posicionados um após o outro verticalmente.

Para o desenvolvimento do algoritmo de posições, foram criadas variáveis com valores padrões para determinadas situações. A seguir são listadas algumas dessas variáveis com seus valores possíveis:

- Uma linha pode conter no máximo cinco elementos sequenciais;
- Artefatos e Objetos de Dados são inseridos sempre acima dos elementos aos quais estão conectados;
- O menor tamanho possível para um elemento "Pool" é o de 550 pixels de altura por 850 pixels de largura;
- Cada elemento possuirá uma distância fixa de 140 pixels para seus elementos adjacentes.

Considerando a interface do usuário da ferramenta *RDFtoXPDL*, a **Figura 53** (*seção 4.6*) apresentou a tela inicial. Nessa tela, é possível selecionar o arquivo em formato RDF que será utilizado para o mapeamento. O campo “FileName” permite ao usuário editar o nome do arquivo que será gerado pela ferramenta.

Como já mencionado na *seção 4.6*, foi considerado que nem todos os sistemas de modelagem de processos permitem o uso do atributo “ExtendedAttribute”, utilizado para mapear elementos marcados como “Extend1”, “Exten2” e “Extend3”. Assim, a ferramenta *RDFtoXPDL* permite ao usuário escolher se esses campos serão incluídos no mapeamento ou não, através do *checkbox* “Export ExtendedAttribute”,

Para iniciar o mapeamento, o usuário deve clicar sobre o botão “Parser”. A ferramenta irá gerar um arquivo em formato XPDL no mesmo diretório que se encontra o arquivo da ontologia selecionado. Por motivos de consulta, o campo “File Output” exibirá qual o diretório de geração do arquivo.

Visando facilitar e prover maior flexibilidade ao usuário, a ferramenta *RDFtoXPDL* permite que sejam configurados os valores das marcações “owl:whatType” e “owl:Event”, assim como das propriedades “Object Property” que serão utilizados no mapeamento. A marcação “owl:Order” não possui tal opção de parametrização, visto que permite apenas valores numéricos.

É interessante observar que a ferramenta *RDFtoXPDL*, por padrão, vem com os valores apresentados neste capítulo. Porém, entende-se que possam existir ontologias que difiram apenas nos valores padrões propostos. Por isso, a ferramenta *RDFtoXPDL* tem a flexibilidade de apoiar o usuário, para que não seja necessário reestruturar toda uma ontologia que já se encontre em formato adequado. Ressalta-se, contudo, que mesmo possuindo tal flexibilidade, a ferramenta *RDFtoXPDL* segue as regras estipuladas pela metodologia, possuindo um valor único para cada uma das marcações e também das “Object Property”.

As telas de configuração das marcações e “Object Property” podem ser acessadas através do menu “Configuration”, opções “RDF Marks” e “RDF Properties” respectivamente. As **Figuras 101** e **102** apresentam as telas para a configuração dos valores de “Object Property” e marcações, respectivamente.

É importante mencionar que foram feitos vários testes sobre a ferramenta *RDFtoXPDL*, utilizando critérios pré-estabelecidos e uma ontologia genérica. Apesar de não conter elementos do domínio de processos, a ontologia utilizada possui um número grande de estruturas da linguagem OWL, ajudando assim, os testes de aderência da ferramenta *RDFtoXPDL* em função da linguagem OWL

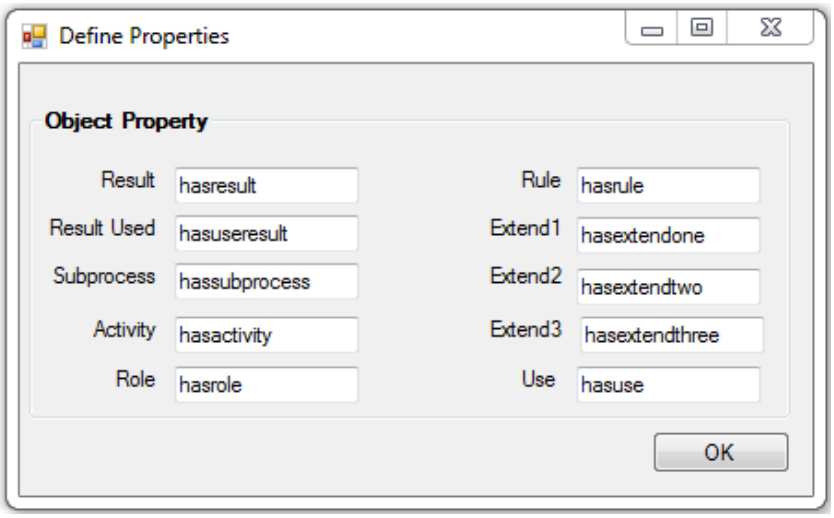


Figura 101 - Tela de configuração dos valores de Object Property.

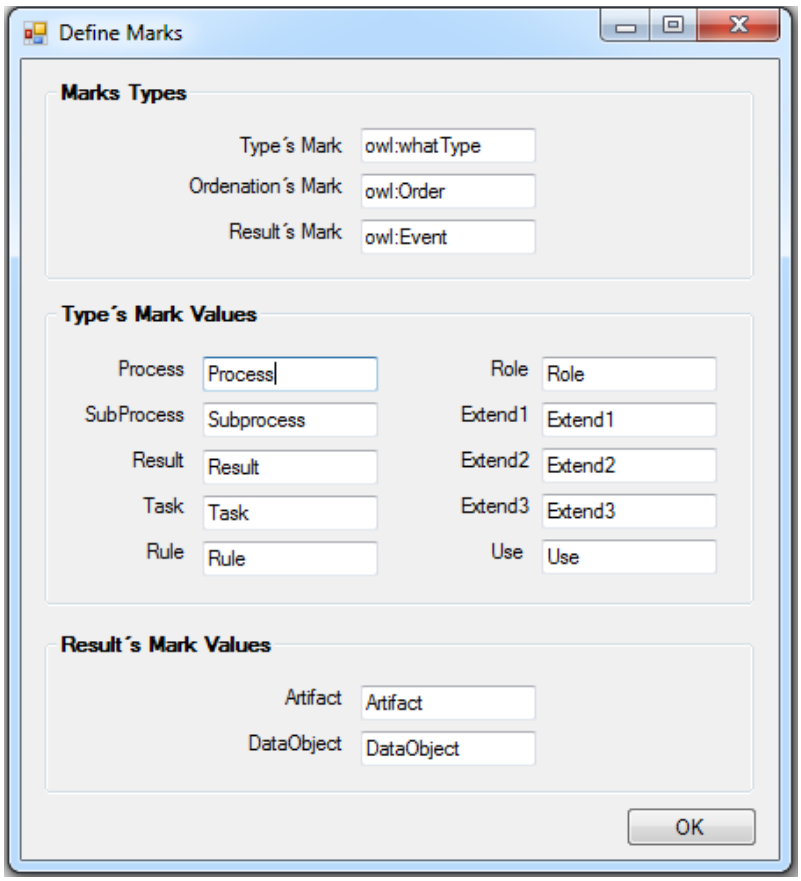


Figura 102 - Tela de configuração dos valores de marcações.

A ontologia utilizada para os testes da ferramenta *RDFtoXPDL* está disponível gratuitamente na internet, e é utilizada para fins de tutoriais da ferramenta Protégé. Vários testes foram realizados para verificar a aderência da ferramenta RDFtoXPDL com os diversos tipos de estruturas presentes na linguagem OWL. A partir dos testes

com a ontologia genérica, também foram testados os modelos de processos gerados. O APÊNDICE E traz o roteiro de testes realizado, assim como seus resultados.

APÊNDICE E – ROTEIRO DE TESTES DA FERRAMENTA *RDFTOXPDL*

Este apêndice apresenta o roteiro de tarefas e perguntas executado a fim de testar se a ferramenta *RDFtoXPDL*, desenvolvida para a conversão dos elementos da ontologia empresarial para um modelo de processos, segue as premissas e regras estipuladas pela metodologia apresentada no *capítulo 4*.

Para testar a ferramenta *RDFtoXPDL* de acordo com a metodologia de mapeamento, foi proposto um roteiro de tarefas a serem executadas com a finalidade de gerar o modelo de processos. As **Tabelas 17 a 20** apresentam os critérios de avaliação utilizados para a execução das tarefas de testes da ferramenta *RDFtoXPDL*. Os números e letras de cada critério indicam a ordem em que os critérios serão aplicados.

Observa-se que, cada um dos critérios de ordem “2” deve ser executado em conjunto com todos os critérios de ordem “3” e “4”.

Assim, uma ontologia genérica disponível gratuitamente na internet foi utilizada para a realização dos testes. A ontologia escolhida é utilizada como ontologia modelo, e também para fins de tutoriais da ferramenta Protégé, e pode ser encontrada em http://www.di.unipi.it/~simi/AI/SI2006/Lucidi/pizza.owl. A ontologia foi criada considerando o domínio de “pizzas”, e nela é possível encontrar os elementos estruturais de OWL, descritos na seção 2.4 e também no APÊNDICE A. Os testes realizados possibilitaram verificar a aderência da ferramenta *RDFtoXPDL* as diversas estruturas existentes na linguagem OWL.

É importante ressaltar que apesar de não conter nenhum tipo de elemento relacionado ao domínio de “processos”, a ontologia foi utilizada apenas para fins de testes comportamentais da ferramenta, não sendo verificado, portanto, a consistência da respectiva ontologia referente ao domínio de processos. Observa-se também que a estrutura da ontologia não foi alterada, apenas elementos que já existiam na ontologia foram utilizados, preservando assim todo o formato de estruturas utilizados por esta validação.

Tabela 17 - Critérios de ordem 1, para testes da ferramenta *RDFtoXPDL*.

	Critérios
1.a	Importar um arquivo RDF
1.b	Importar outro tipo de arquivo

Tabela 18 - Critérios de ordem 2, para testes da ferramenta RDFtoXPDL.

	Critérios
2.a	Inserir todas as três marcações com valores estipulados para cada um, apenas em elementos que possuam relação através da “Object Property”
2.b	Não inserir nenhuma marcação
2.c	Inserir marcação “owl:whatType” e utilizar apenas elementos que possuam relação através da “Object Property”. Inserir marcação “owl:Event” em elementos que não possuam suporte a essa marcação
2.d	Inserir marcação “owl:whatType” e utilizar apenas elementos que possuam relação através da “Object Property”. Inserir marcação “owl:Order” em elementos que possuam suporte a essa marcação
2.e	Inserir marcação “owl:whatType” e utilizar apenas elementos que possuam relação através da “Object Property”. Não inserir marcação “owl:Event” em elementos que necessitam deste
2.f	Inserir marcação “owl:whatType” e utilizar apenas elementos que possuam relação através da “Object Property”. Não inserir marcação “owl:Order” em elementos que necessitam deste
2.g	Inserir valores diferentes dos permitidos na marcação “owl:whatType”
2.h	Inserir marcação “owl:whatType” e utilizar apenas elementos que possuam relação através da “Object Property”. Inserir valores diferentes dos permitidos na marcação “owl:Event
2.i	Inserir marcação “owl:whatType” e utilizar apenas elementos que possuam relação através da “Object Property”. Inserir valores diferentes dos permitidos na marcação “owl:Order
2.j	Inserir marcações com valores estipulados para cada um, apenas em elementos que não sejam relacionados através da “Object Property

Tabela 19 - Critérios de ordem 3, para testes da ferramenta RDFtoXPDL.

	Critérios
3.a	Remover as “Object Property” do *Parser*
3.b	Incluir um mesmo valor para mais de uma “Object Property”
3.c	Utilizar um valor diferente para cada “Object Property”

Tabela 20 - Critérios de ordem 4, para testes da ferramenta RDFtoXPDL.

	Critérios
4.a	Algum elemento que deveria estar mapeado ficou de fora?
4.b	Algum elemento a mais foi gerado?
4.c	Os elementos foram gerados de acordo com a tabela de correspondência apresentada na etapa 2.3 da metodologia?
4.d	Os elementos de geração automática foram gerados corretamente?
4.e	O fluxo está de acordo com o pretendido?

O primeiro passo consistiu em executar na ferramenta RDFtoXPDL os critérios de testes de ordem “1”, conforme mostrado na **Tabela 17**. Sendo que a ferramenta permitiu apenas arquivos que possuam formato RDF.

Para a realização do segundo passo foram executados na ontologia cada um dos critérios de testes de ordem ‘2”, sendo que para cada um desses critérios foram realizadas as seguintes atividades:

- “2.a”: foram utilizadas as marcações: “owl:whatType” com valores “Process”, “Task”, “Subprocess”, “Result”; a marcação “owl:Event” com os valores “DataObject” e “Artifact”; e a marcação “owl:Order”;
- “2.c”: a marcação “owl:Event” foi inserida nos elementos que possuíam marcação “owl:whatType” com valor “Process”, “Subprocess”, “Task”;
- “2.d”: a marcação “owl:Order” foi inserida nos elementos que possuíam marcação “owl:whatType” com valor “Process”, “Subprocess”, “Task”;
- “2.g”: a marcação “owl:whatType” foi utilizada com os valores “Process1”, “Subprocess1”, “Task1”;
- “2.h”: a marcação “owl:Event” foi utilizada com os valores “DataObject1”, “Artifact1”;
- “2.i”: a marcação “owl:Order” foi utilizado com valores alfanuméricos;
- “2.j”: as marcações “owl:whatType” com valor “Subprocess”, “Task” e “Result” foram utilizados em elementos diretamente relacionados através da estrutura “disjointWith”.

A seguir, cada um dos critérios de ordem “2”, e suas respectivas atividades, foram executados separadamente em conjunto com os critérios de ordem “3” e “4”, gerando resultados expressos nas tabelas a seguir. As **Tabelas 21 a 25** apresentam os resultados obtidos com a execução de cada critério de ordem “2”, com todos os critérios de ordem “3” e “4”.

Analisando as ações tomadas pela ferramenta *RDFtoXPDL,* de acordo com os critérios de testes propostos, observa-se que a ferramenta atendeu para todos os critérios as regras impostas pela metodologia (*capítulo 4*), não permitindo em nenhum momento que fossem geradas exceções. É interessante observar também que caso o usuário tente executar alguma ação com valores não permitidos para as marcações e “Object Property”, a ferramenta volta aos valores padrões indicados pela metodologia, como mostram os critérios “3a” e “3b”, nas **Tabelas 21**, **23**, **24** e **25**.

Tabela 21 – Resultados do critério “2a” aplicado sobre a ferramenta *RDFtoXPDL.*

Tarefas/Perguntas	Ações/Respostas para os critério 2a
2.a	Os elementos são considerados e mapeados.
3.a	Os valores padrões propostos pela metodologia são considerados.
3.b	Não é permitido. A ferramenta volta ao valor padrão.
3.c	Em conjunto com o respectivo valor da marcação "owl:whatType" irá considerar como um elemento a ser mapeado.
4.a	Todos os elementos com marcações e "Object Property" foram mapeados.
4.b	N/A
4.c	Todos elementos respeitam seu devido correspondente em XPDL.
4.d	Todos os elementos que necessitam de um "Fluxo de sequência" ou "associação" tiveram estes elementos gerados automaticamente.
4.e	O fluxo do processo mapeado seguiu a ordem estipulada pelo "owl:Order". Na ausência dessa marcação, a ordem se torna aleatória.

Tabela 22 – Resultados dos critérios “2b”,”2g,”2j” aplicados sobre a ferramenta *RDFtoXPDL*.

Tarefas/Perguntas	Ações/Respostas para os critérios 2b, 2g, 2j
2.b	Nenhum elemento foi considerado.
3.a	N/A
3.b	N/A
3.c	N/A
4.a	N/A
4.b	N/A
4.c	N/A
4.d	N/A
4.e	N/A

Tabela 23 – Resultados dos critérios “2c”, “2d” aplicados sobre a ferramenta *RDFtoXPDL*.

Tarefas/Perguntas	Ações/Respostas para os critérios 2c, 2d
2.c	A marcação é ignorada.
3.a	Os valores padrões propostos pela metodologia são considerados.
3.b	Não é permitido. A ferramenta volta ao valor padrão.
3.c	Em conjunto com o respectivo valor da marcação "owl:whatType" irá considerar como um elemento a ser mapeado.
4.a	Todos os elementos com marcações e "Object Property" foram mapeados.
4.b	Apenas elementos com marcações e "Object Property" foram mapeados.
4.c	Todos elementos respeitam seu devido correspondente em XPDL.
4.d	Todos os elementos que necessitam de um "Fluxo de sequência" ou "associação" tiveram estes elementos gerados automaticamente.
4.e	O fluxo do processo mapeado seguiu a ordem estipulada pelo "owl:Order". Na ausência dessa marcação, a ordem se torna aleatória.

Tabela 24 – Resultados dos critérios “2e”, “2h” aplicados sobre a ferramenta *RDFtoXPDL*.

Tarefas/Perguntas	Ações/Respostas para os critérios 2e, 2h
2.e	O elemento é considerado como sendo um "DataObject".
3.a	Os valores padrões propostos pela metodologia são considerados.
3.b	Não é permitido. A ferramenta volta ao valor padrão.
3.c	Em conjunto com o respectivo valor da marcação "owl:whatType" irá considerar como um elemento a ser mapeado.
4.a	Todos os elementos com marcações e "Object Property" foram mapeados.
4.b	Apenas elementos com marcações e "Object Property" foram mapeados.
4.c	Todos elementos respeitam seu devido correspondente em XPDL.
4.d	Todos os elementos que necessitam de um "Fluxo de sequência" ou "associação" tiveram estes elementos gerados automaticamente.
4.e	O fluxo do processo mapeado seguiu a ordem estipulada pelo "owl:Order". Na ausência dessa marcação, a ordem se torna aleatória.

Tabela 25 – Resultados dos critérios “2f”, “2i” aplicados sobre a ferramenta *RDFtoXPDL*.

Tarefas/Perguntas	Ações/Respostas para os critérios 2f, 2i
2.f	Uma ordenação é gerada automaticamente para o elemento.
3.a	Os valores padrões propostos pela metodologia são considerados.
3.b	Não é permitido. A ferramenta volta ao valor padrão.
3.c	Em conjunto com o respectivo valor do marcador "owl:whatType" irá considerar como um elemento a ser mapeado.
4.a	Todos os elementos com marcações e "Object Property" foram mapeados.
4.b	Apenas elementos com marcações e "Object Property" foram mapeados.
4.c	Todos elementos respeitam seu devido correspondente em XPDL.
4.d	Todos os elementos que necessitam de um "Fluxo de sequência" ou "associação" tiveram estes elementos gerados automaticamente.
4.e	O fluxo do processo mapeado seguiu a ordem estipulada pelo "owl:Order". Na ausência desse marcação, a ordem se torna aleatória.

A fim de verificar com mais exatidão a aderência da ferramenta *RDFtoXPDL* com ontologias e estruturas em geral, foi realizada uma segunda etapa de testes, utilizando-se da mesma ontologia genérica. Essa etapa teve como objetivo avaliar a aderência da ferramenta *RDFtoXPDL,* com as diversas estruturas existentes na linguagem OWL que possibilitam a relação entre dois elementos (“Object Property”). Dessa forma a metodologia, proposta no *capítulo 4*, foi aplicada na ontologia, de modo a abranger o maior número possível de estruturas. Ressalta-se que a ontologia utilizada nestes testes não possui nenhum valor semântico referente ao domínio de “processos”. Assim, os elementos escolhidos para serem mapeados foram escolhidos de acordo com as possíveis variações de estruturas da linguagem OWL, não

possuindo qualquer valor semântico. Dessa forma, a **Tabela 26** apresenta cada uma das estruturas consideradas nesta validação.

Tabela 26 - Estruturas de relação entre elementos em OWL avaliadas.

Estruturas da linguagem OWL		
Estrutura	**ObjectProperty**	**Aderência da ferramenta**
subClassOf	Sim	Sim
subClassOf/unionOf	Sim	Sim
subClassOf/OneOf	Sim	Sim
equivalentClass/intersectionOf	Sim	Sim
equivalentClass/intersectionOf/complementOf	Sim	Sim
equivalentClass/unionOf	Não	Não
equivalentClass/oneOf	Não	Não
equivalentClass/intersectionOf/unionOf	Não	Não
equivalentClass/intersectionOf/oneOf	Não	Não
equivalentClass	Sim	Sim
intersectionOf	Sim	Sim
disjointWith	Não	Não

Na **Tabela 26**, é possível observar que algumas estruturas não possuem a restrição “Object Property” como parte de sua estrutura e, portanto, não são consideradas pela ferramenta *RDFtoXPDL*. Por outro lado, todas as demais estruturas presentes na ontologia e que possuem “Object Property” puderam ser interpretadas com sucesso pela ferramenta.

Com a finalidade de testar os modelos de processos gerados pela ferramenta *RDFtoXPDL,* o modelo de processos gerado de acordo com a segunda etapa de testes foi importado em três ferramentas de modelagem de processos. Os testes com as ferramentas de modelagem possuem os seguintes objetivos: o primeiro é poder visualizar se o processo foi gerado com uma sequência lógica correta, de maneira a ser interpretado por ferramentas de modelagem de processos; e o segundo ponto é verificar se o processo está de acordo com a especificação padrão da linguagem XPDL e respeita todas as regras detalhadas por este.

Para isso, foram selecionadas três ferramentas com suporte a linguagem XPDL 2.2, e ao BPMN 2.0 e que possuíssem a funcionalidade de importação de modelos de processos gerados em linguagem XPDL, versão 2.2: *Bizagi Modeler* (versão 2.6.0.4)*, Cameo Business Modeler* (versão 17.0.5) *e BPMN Visio Modeler*

(versão 4.2.0). Todas as três ferramentas já foram apresentadas na *seção 3.4* e, portanto, neste apêndice só listaremos os resultados das validações realizadas.

Com relação a ferramenta *Bizagi Modeler,* o modelo de processos gerado pela ferramenta *RDFtoXPDL* foi importado com sucesso, sem nenhuma inconsistência. Já para as ferramentas "*Cameo Business Modeler*" e "*BPMN Visio Modeler*" foi encontrado o mesmo problema em ambas, onde as ferramentas não permitem a declaração de um subprocesso do tipo "embedded". Para essas ferramentas um subprocesso deve ser declarado de forma separada do seu processo principal. Ou seja, na prática estas ferramentas tratam qualquer tipo de subprocesso declarado como sendo do tipo "reutilizável", não respeitando a especificação padrão do XPDL. Destaca-se que as ferramentas não geram erro ao tentar importar os arquivos gerados pela ferramenta *RDFtoXPDL*, porém por não existir a tratativa para esses subprocessos, esses elementos são posicionados de forma errônea na tela para o usuário.

Diante da possibilidade da ferramenta *RDFtoXPDL* de permitir exportar alguns elementos da ontologia, através do atributo "ExtendedAttribute", foi verificado que a única ferramenta que permite tal uso é o *Bizagi Modeler*, tanto a Cameo Business Modeler quanto a *BPMN Visio Modeler* não permitem tal uso, e acabam gerando inconsistências na importação do modelo de processos.

Portanto, de acordo com os testes realizados nesta última etapa, o modelo de processos gerado pela ferramenta *RDFtoXPDL* pode ser considerado consistente e de acordo com os padrões da linguagem XPDL, versão 2.2

A ontologia utilizada para esta validação, assim como o modelo de processos gerado encontra-se disponível na mídia digital, deste apêndice, em pasta chamada "Apêndice E-Roteiro de Validação da ferramentaRDFtoXPDL".

APÊNDICE F – APLICAÇÃO DA METODOLOGIA PROPOSTA EM UMA ONTOLOGIA EMPRESARIAL (CD-R ANEXO)

Este apêndice apresenta os modelos de processos e a tabela de cruzamento dos dados (etapa 2.3 da metodologia) gerados durante a aplicação da metodologia apresentada no capítulo 5 deste trabalho. Devido à dimensão do conteúdo deste Apêndice, encontra-se em formato digital, em CD-R anexo a esta dissertação, em pasta denominada “Apêndice F – AplicaçãoDaMetodologia”.