Universidade Federal de Alagoas
Instituto de Computação

Dissertação de Mestrado

Mário André de Freitas Farias

UM MODELO CLASSIFICADOR DE BASES DE DADOS NÃO ESTRUTURADAS QUE COMBINA DICIONÁRIO NEUROLINGUÍSTICOS COM ONTOLOGIA: UM ESTUDO DE CASO SOBRE A LISTA DE E-MAIL DO PROJETO *APACHE*

Maceió
2011

MÁRIO ANDRÉ DE FREITAS FARIAS

UM MODELO CLASSIFICADOR DE BASES DE DADOS NÃO ESTRUTURADAS QUE COMBINA DICIONÁRIO NEUROLINGUÍSTICOS COM ONTOLOGIA: UM ESTUDO DE CASO SOBRE A LISTA DE E-MAIL DO PROJETO *APACHE*

Dissertação apresentada como requisito parcial para obtenção do grau de mestre em Modelagem Computacional de Conhecimento na Universidade Federal de Alagoas.

Orientador:

Prof. Dr. Evandro de Barros Costa

Co-orientador:

Prof. Dr. Methanias Colaço Júnior

Maceió
2011

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária: Helena Cristina Pimentel do Vale

F224u Farias, Mário André de Freitas.
Um modelo classificador de bases de dados não estruturadas que combina dicionário neurolinguístico com ontologia : um estudo de caso sobre a lista de e-mail do Projeto Apache / Mário André de Freitas Farias. ï 2011.
90 f. : il.

Orientador: Evandro de Barros Costa.
Co- Orientador: Methanias Colaço Júnior.
Dissertação (mestrado em Modelagem Computacional de Conhecimento) ï Universidade Federal de Alagoas. Programa de Pós-Graduação em Modelagem Computacional de Conhecimento. Maceió, 2011.

Bibliografia: f. 82-87.
Apêndices: 88-90.

1. Mineração de dados (Computação). 2. Textos. 3. Neurolinguística. 4. Ontologia. Software ï Compreensão. I. Título.

CDU: 004.5

UNIVERSIDADE FEDERAL DE ALAGOAS/UFAL
Programa de Pós-Graduação em Modelagem Computacional de Conhecimento
Avenida Lourival Melo Mota, Km 14, Bloco 09, Cidade Universitária
CEP 57.072-900 – Maceió – AL – Brasil
Telefone: (082) 3214-1364

MODELAGEM COMPUTACIONAL DE CONHECIMENTO

Membros da Comissão Julgadora da Dissertação de Mestrado de Mário André de Freitas Farias, intitulada: "Um modelo classificador da lista de e-mail do Projeto Apache que combina dicionário neurolinguístico com ontologia", apresentada ao Programa de Pós-Graduação em Modelagem Computacional de Conhecimento da Universidade Federal de Alagoas em 23 de dezembro de 2011, às 10h00min, na sala de aula do Mestrado em Modelagem Computacional de Conhecimento.

COMISSÃO JULGADORA

Prof. Dr. Evandro de Barros Costa
UFAL – Instituto de Computação
Orientador

Prof. Dr. Patrick Henrique da Silva Brito
UFAL – Instituto de Computação
Examinador

Prof. Dr. Methanias Colaço Rodrigues Júnior
UFSE – Departamento de Sistemas de Informação
Examinador

Maceió, dezembro de 2011.

DEDICATÓRIA

A Deus, pelo dom da vida e pelas oportunidades e graças recebidas.
A minha família, pela cumplicidade e apoio.
Ao meu pai, pela confiança na minha formação acadêmica.
A minha mãe (em memória), pelo amor incondicional e pela presença constante.

AGRADECIMENTOS

A Deus, pela graça da vida e por me proporcionar força e coragem para continuar buscando alcançar meus objetivos acadêmicos.

A minha mãe (em memória), pelos ensinamentos. Mesmo passando um curto espaço de tempo ao seu lado, pude aprender o valor e a importância da formação acadêmica, moral e ética. Através dela, herdei a paixão pela docência, combustível primordial para a continuidade da busca por aprimoramentos na minha formação acadêmica. Mulher devotada, guerreira, espelho de perseverança e garra.

Ao meu pai, por acreditar em um jovem com sede pela vida acadêmica, por acreditar em conceitos e objetivos totalmente desconhecidos por ele. Um homem sem formação acadêmica, sem muita instrução culta, mas com grandes habilidades de gestão, coordenação e liderança. Herdei dele a curiosidade, a inquietação a necessidade pelo novo, pelo desconhecido.

Aos meus irmãos, por me guiarem diante dos desafios da vida. Cresci vendo e ouvindo histórias de como enfrentaram árduas batalhas diárias para ter acesso ao mínimo de educação. Eles são os meus heróis. Particularmente, a Rita de Cássia, por dividir comigo os sonhos e desafios vivenciados por um professor.

A Janete Cahet – uma grande mulher, grande jornalista – por me acompanhar nesta caminhada acadêmica, por estar ao meu lado nos momentos de dúvidas e de dificuldades, por ter contribuído de forma direta neste trabalho, por ter sido uma mulher maravilhosa e companheira durante todos os anos que passamos juntos.

Aos meus orientadores, Prof. Dr. Evandro de Barros Costa e Prof. Dr. Methanias Colaço Júnior, pelos ensinamentos, dedicação e pela oportunidade de desafiar o desconhecido. Educadores, que além do conhecimento técnico compartilhado durante este trabalho, contribuíram para despertar o desejo pela pesquisa científica.

Aos amigos Rodrigo Peixoto, Cledson Gomes, Weslley, Thiago, Leonardo, Douglas, Helio, Lívia Omena, pelo companheirismo e apoio durante os anos vividos na cidade de Maceió.

Aos Docentes do Mestrado em Modelagem Computacional de Conhecimento, que contribuíram com a minha formação acadêmica nesse programa. Particularmente aos professores Henrique Pacca Luna, Patrick Henrique, Eliana Almeida, Arturo Hernández, Fábio Paraguaçu e João Soletti.

Aos colegas docentes do Instituto Federal de Sergipe, por compartilhar os sonhos e os desejos acerca do aprimoramento da formação acadêmica e da melhoria da educação tecnológica. Particularmente ao amigo Prof. Dr. José Osman pelo apoio direto na análise estatística deste trabalho.

RESUMO

Listas de e-mails e grupos de discussão são normalmente usados por programadores para discutir e aperfeiçoar tarefas a serem executadas durante as fases de desenvolvimento de projetos de software. Projetos de softwares *Open Source* utilizam essas listas como uma ferramenta primária para a colaboração e cooperação. Em projetos dessa natureza, normalmente, os desenvolvedores estão em todas as partes do mundo. Desta forma, meios de interação e comunicação são necessários para garantir a colaboração entre os mesmos, bem como a eficácia no processo de construção e manutenção de projetos desse porte. Listas de e-mails podem ser uma importante fonte de dados para a descoberta de informações úteis acerca de padrões de comportamento de desenvolvedores para gerentes de projetos. O Neurominer é uma ferramenta de mineração de texto que determina o Sistema de Representação Preferencial de desenvolvedores de software em um contexto específico. A ferramenta apresenta como inovação a utilização da teoria da Programação Neurolinguística - PNL combinada com técnicas de mineração e estatística. Nesse contexto, é proposta a extensão dessa ferramenta através da aplicação de técnicas de ontologia ao seu dicionário, permitindo a combinação de predicados sensoriais a termos da engenharia de software, proporcionando um poder maior de contextualização ao seu dicionário. Sob esse prisma, a mineração de texto combinada com técnicas de PNL e ontologia surge como candidata natural para compor uma solução que objetiva melhorar a garimpagem de informações textuais, através de listas de discussões, com o propósito de apoiar gerentes de projetos de softwares na tomada de decisão. Essa combinação conduziu a resultados bastante significativos, propondo uma solução eficiente e eficaz.

Palavras-chave: Mineração de texto. Neurolinguística. Ontologia. Compreensão de software.

ABSTRACT

Electronic mailing lists and discussion groups are normally used by programmers to discuss and improve tasks to be performed during software projects development. Open Source Software (OSS) projects use this lists as the primary tool for collaboration and cooperation. In project like that, normally, the developers are around the world. Thus, means of interaction and communication are needed to ensure collaboration between them, as well as efficiency in the construction and maintenance of projects this size. Mailing lists can be an important data source to discovery information useful about patterns of behavior of developer aimed at project manager. The Neurominer is a text mining tool that determines the Preferred Representational System (PRS) of software developers in a specific context. The tool has a new approach which is a combination between the Neuro-Linguistic Programming – NLP theory, text mining and statistic technique. In this context, we propose the extension of this tool by applying of techniques of ontology to dictionary, allowing the combination of sensory predicates with software engineering terms, providing a greater power in the context of the dictionary. This way, the text mining matched with NLP theory and ontology appears as natural candidate that consists a solution that aiming to improve the mining of textual information through mailing lists, in order to support software project managers in making decision. This matching showed significant outcomes, proposing a efficient and effective solution.

Keywords: Text mining. Neuro-Linguistic. Ontology. Software Comprehension.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE SIGLAS

ANLP	*Association for NLP*
df	*Document frequency*
DW	*Data Warehouse*
ETL	*Extraction, Transformation e Load*
IA	Inteligência Artificial
idf	*Inverse document frequency*
LIWC	*Linguistic Inquiry and Word Count*
OSS	*Open Source Software*
PNL	Programação Neurolinguística
RI	Recuperação da Informação
SRP	Sistema de Representação Preferencial
SWEBOK	*Software Engineering Body of Knowledge*
tf	*term frequency*

GLOSSÁRIO

Predicado sensorial:	Palavras que indicam a representação interna de nossa mente, refletindo o sistema representacional.
Sistema de Representação Preferencial:	Representação interna (Visual, Auditiva ou Cinestésica) mais utilizada em uma situação específica.
Matching:	Integração na Comunicação.
Efeito camaleão:	Estímulos imitativos não-verbais e verbais para o aumento da empatia em uma comunicação.
Mineração de texto:	Processo de descoberta de informações escondidas em bases de dados textuais.
Psicometria:	É um ramo da estatística que estuda fenômenos psicológicos e a aplicação de técnicas de mensuração aos fenômenos psíquicos
NEUROMINER:	Ferramenta de análise de texto computadorizada baseada na PNL que possibilita a identificação de SRP de desenvolvedores de *softwares*.
Listas de e-mail *Open Source:*	Lista de discussão de Projetos Open Source.
Open Source Software:	Sistema de código fonte aberto.
Axiomas:	Sentença que não é provada ou demonstrada e é considerada como óbvia ou como um consenso inicial necessário para a construção ou aceitação de uma teoria.
Protégé:	Ferramenta utilizada para criar ontologias.
OWLViz.	Gerador de gráficos com instâncias, herança e outros tipos de relacionamentos. É adicionado à ferramenta Protégé.
Sesame:	Repositório de RDF/RDFS
API GKMT:	Api em Java para manipulação de ontologias.
Function words:	Refletem como as pessoas se comunicam. São compostas por: pronomes, preposições, artigos, conjunções e verbos auxiliares e interligam o conteúdo do texto
Content words	Refletem o que as pessoas querem dizer. São compostas por: verbos, adjetivos, substantivos e advérbios.
CfBT	Entidade de consultoria em educação
Data Warehouse	Banco de dados com dados históricos usados para análise e apoio à decisão.

SUMÁRIO

1 INTRODUÇÃO

Este trabalho trata da exploração de dados textuais visando gerar informação que possa se prestar ao uso por gerentes de projeto de software para apoiar sua atividade de decisão. Particularmente, o foco desta pesquisa é propor um modelo classificador capaz de classificar perfis cognitivos de desenvolvedores a partir de listas de discussão de projeto de softwares.

Os processos decorrentes de atividades gerenciais e/ou desenvolvimento de software, identificados em troca de e-mails, chats, documentos textuais, relatórios de erros e códigos fontes, produzem um grande volume de dados não estruturados nas empresas. Porém, faze-se necessário dispor de ferramentas e técnicas de mineração de dados, buscando informação valiosa.

Informação é uma palavra em destaque que propicia um novo ângulo de observação de eventos e objetos, tornando visíveis pontos de vista antes invisíveis. (NONAKA & TAKEUCHI, 1997 apud MAGALHÃES, 2008). Dispor da informação correta no momento oportuno vem sendo caracterizado como um imprescindível diferencial para gestores de projetos de software. Particularmente, observa-se que informações sobre características cognitivas dos desenvolvedores influenciam de forma direta na interação e colaboração. A produtividade de uma equipe é proporcional ao grau de interação entre seus membros. Neste contexto, detectar as preferências representacionais dos desenvolvedores de software, em projetos de softwares, pode ajudar na melhoria da empatia nas comunicações e na resolução de problemas. Como observa Colaço (2011):

> Detectar [...] que um analista de sistemas pouco usa o sistema de representação visual pode ajudar a solucionar o problema ou a realocá-lo para outra atividade. Muitas vezes, perde-se um grande programador e se ganha um péssimo analista, ou ainda, perde-se um profissional por uma alocação mal feita a um projeto[...]. (COLAÇO, 2011, p 37)

Processos mentais internos, tais como a resolução de problemas, uso da memória e linguagem são formados por representações Visuais (imagens e diagramas), Auditivas (sons) e Cinestésicas (experiências físicas e práticas). Esta representação interna ou sistema de representação é altamente contextualizado, ou seja, varia de acordo com a situação (EINSPRUCH & FORMAN, 1985). Desta forma, em contextos específicos, algumas pessoas têm preferências por formas de representação da informação na hora de compreender algo, definindo o seu Sistema de Representação Preferencial (SRP) para aquele momento. Em

outras palavras, o SRP nada mais é do que a representação interna (Visual, Auditiva ou Cinestésica) mais utilizada em uma situação específica.

Discussões sobre problemas, soluções de desenvolvimento, manutenção e testes de softwares podem ser utilizadas para análises psicológicas, proporcionando a descoberta de perfis de desenvolvedores. Diversas pesquisas têm sido publicadas na área de mineração de texto com o intuito da descoberta de conhecimento e compreender melhor as dificuldades do desenvolvimento de sistemas (COLAÇO et al., 2010; WITTE et al., 2008; RIGBY and HASSAN, 2007; KO et al., 2006; GAIZAUSKAS et al., 2005).

Muito se tem debatido sobre processos e técnicas para construir ferramentas de recuperação da informação (*Information Retrieval*) e mineração de texto. (GOULART et al., 2011; SANTOS et al., 2009;MAGALHÃES, 2008; MOREIRA et al., 2009; KONCHADY, 2006). Esses processos e técnicas requerem atenção especial quando a informação está armazenada em forma textual, ou seja, escrito de forma livre e não estruturada, tornando o processo de mineração do conhecimento uma tarefa difícil e custosa.

Bases de dados históricas, oriundas da construção de sistemas, apresentam-se como mais uma alternativa para a descoberta de conhecimento sobre o processo de desenvolvimento de software. Entre as fontes de dados passíveis de análise estão as listas de e-mails temáticas, as quais armazenam um grande volume de informações não estruturada.

Umas das bases para análise textual de listas de e-mails é a psicometria. A psicometria é um ramo da estatística que estuda fenômenos psicológicos e a aplicação de técnicas de mensuração aos fenômenos psíquicos. Foi criada por estatísticos de formação e faz interface entre a psicologia e a estatística (BRAGA & CRUZ, 2006). A forma mais comum de aplicação da psicometria é a contagem de palavras de um texto, considerando as dimensões significativas (estado emocional ou perfil de aprendizagem) às quais elas pertencem, ou seja, algumas palavras podem indicar o estado emocional do indivíduo, bem como o seu perfil geral. A aplicação desta técnica tem sido automatizada por ferramentas que receberam o nome de LIWC.

Ferramentas LIWC - *Linguistic Inquiry and Word Count* – baseiam-se na análise da frequência de palavras através do uso de um dicionário e técnicas de psicometria. Em (Pennebaker et al, 2001), é apresentada uma LIWC com o propósito de ajudar pacientes a tentar descobrir informações sobre o comportamento humano através de análise de amostras de textos. A abordagem LIWC ainda é pouco explorada para avaliações de indivíduos baseadas na teoria da Programação Neurolinguística (PNL), que apresenta possibilidades de

desenvolvimento de um SRP para cognição em seres humanos (COLAÇO, 2011; COLAÇO et al., 2010; TOSEY & MATHISON, 2003, 2009).

Para a PNL, a mente humana é poderosa no que se refere ao uso de sistemas cognitivos. Pessoas diferentes, em contextos diferentes, podem ter preferências de representações diferentes (PETERS et al., 2008 apud COLAÇO et al., 2010). Prova disso é que algumas pessoas podem aprender um esporte empregando preferencialmente um sistema específico e desempenhar suas atividades profissionais utilizando outro sistema com mais intensidade. De fato, a área de psicologia aceita bem a existência de diferentes formas de representação para a cognição (TOSEY & MATHISON, 2009).

Nosso comportamento é programado através da combinação sequencial de processos mentais internos (sistema de representação neural) quando, por exemplo, nossas ações envolvem tomar uma decisão, praticar um esporte, sorrir para o sexo oposto ou visualizar a pronuncia de uma palavra. Um estímulo interno – sinais, sons, sentimentos, cheiros e gostos – é acionado através de uma sequência de representações internas, gerando um resultado comportamental específico (DILTS et al 1980 apud TOSEY and MATHISON, 2009).

Ainda para a PNL, as palavras que usamos diariamente podem indicar a representação interna que estamos utilizando em nossa mente (predicado sensorial), refletindo o sistema representacional usado no momento e, em uma análise mais histórica e profunda, quem somos e em quais relacionamentos sociais estamos inseridos. Segundo Tosey e Mathison (2009), quer seja em uma conversa, escrevendo sobre um tema específico ou lendo um livro, representações internas sensoriais são constantemente formadas e ativadas.

Essa abordagem não é recente no campo da psicologia. A busca de traços psicológicos, através da linguagem, é uma maneira comum e confiável de traduzir pensamentos e sentimentos internos presentes nas pessoas (TAUSCZIK & PENNEBAKER, 2010).

Neste contexto, em (Colaço et al., 2010), foi desenvolvida e validada uma ferramenta de análise de texto computadorizada baseada na PNL, o NEUROMINER[1]. A ferramenta é um tipo de LIWC inovador para identificar o SRP de desenvolvedores de software, combinando técnicas de mineração de texto e psicometria baseada na PNL.

1 Website oficial do NEUROMINER: www.neurominer.com

Ferramentas de análise de textos como o NEUROMINER contam palavras com a finalidade de executar alguma classificação, contudo, uma simples contagem de palavras não é suficiente para medir o significado de um texto. O poder de uma ferramenta para essa finalidade está nas dimensões de significados de seu dicionário (RIGBY & HASSAN, 2007).

Partindo deste princípio, o NEUROMINER utiliza um dicionário com palavras que representam os Sistemas Representacionais (predicados sensoriais) e conceitos da Engenharia de Software (dimensões de significado), bem como uma fórmula específica para cálculo da freqüência e peso destas palavras (COLAÇO et al., 2010). De fato, o que o NEUROMINER não apresenta é uma representação com maior poder semântico da estrutura do seu dicionário, considerando as classes gramaticais de cada palavra, hierarquia dos conceitos e a possibilidade de combinação das palavras que representam os Sistemas Representacionais com outras áreas de conhecimento, visto que seu dicionário foi construído sem observar padrões gramaticais, além de ter sido combinado manualmente sem obedecer regras de combinação definidas.

Neste trabalho, propomos a extensão do NEUROMINER através da aplicação de técnicas de ontologia ao seu dicionário, possibilitando a combinação de predicados sensoriais a termos da engenharia de software, bem como a qualquer área do conhecimento.

Acredita-se que uma ontologia na área da ciência da computação representa um esforço para formular um rigoroso esquema de conceitos, formado por uma estrutura de dados hierárquica, contendo regras e relacionamentos dentro de um domínio (WONGTHONGTHAM et al., 2009). Convém ressaltar que em estruturas textuais, tratamos com relações cognitivas através de linguagens naturais, destarte procuramos formalizá-las mediante a utilização de língua não-natural. As teorias às margens da semântica lexical, aliadas à modelagem computacional, podem subsidiar ontologias a alcançarem esta finalidade.

O uso de ontologias incorporado a sistemas de extração e classificação de informação textual é capaz de resultar ferramentas hábeis em fornecer uma visão mais apurada e aprofundada do domínio estudado. Conforme sintetiza Palmeira e Freitas (2007):

> [...] Essa união de ontologias e sistemas baseados em conhecimento favorece a manipulação de informação – entendida como a realização de mais de uma tarefa, como por exemplo, classificação, recuperação e extração, proporcionando flexibilidade às ferramentas para atuar em diferentes domínios. Observa-se ainda a vantagem de que para trocar de domínio basta trocar a ontologia e parte da base de regras. (PALMEIRA e FREITAS, 2007, p. 2).

De forma análoga, para que o NEUROMINER se apresente como uma solução que possa identificar SPRs em profissionais de outras áreas do conhecimento, basta substituir os conceitos de engenharia de software por conceitos de um domínio específico.

Sob esse prisma, a mineração de texto combinada com técnicas de PNL e ontologia surge como candidata natural para compor uma solução que objetiva melhorar a garimpagem de informações textuais, através de listas de discussões, com o propósito de apoiar gerentes de projetos de softwares na tomada de decisão.

1.1 Questões da pesquisa

Diante do exposto, acredita-se que a eficiência e a eficácia de uma ferramenta baseada em mineração de texto e psicometria está no poder contextual de seu dicionário. Assim, este trabalho ataca as seguintes questões de pesquisa:

1. É viável a utilização de um modelo classificador, baseado em psicometria, PNL e ontologia, para a descoberta de conhecimento em listas de discussão?
2. É viável o uso de ontologia para psicometria utilizada no NEUROMINER?
3. A combinação de hierarquia léxica e sintática, através de uma ontologia, entre predicados sensoriais e conceitos de engenharia de software:
    - 3.1 Altera as classificações do NEUROMINER ?
    - 3.2 Facilita a inclusão de novos predicados sensoriais?
    - 3.3 Evidencia as classificações feitas pelo NEUROMINER, aumentando a distância entre os *scores* de cada sistema?
    - 3.4 Aumenta o poder de classificação?
    - 3.5 Aumenta a eficiência do processo (tempo de processamento)?

Logo, a hipótese que queremos negar é:

Hipótese Nula: A aplicação de ontologia ao dicionário do NEUROMINER não altera seu poder de classificação.

Hipótese Alternativa: A aplicação de ontologia ao dicionário do NEUROMINER altera seu poder de classificação.

1.2 Objetivo geral

O objetivo desta dissertação é propor e validar um modelo classificador que permitirá combinar, lexicalmente e sintaticamente, predicados sensoriais neurolinguísticos a conceitos de um domínio específico através de uma ontologia definida.

1.3 Objetivos específicos

Sendo assim, o objetivo geral desta dissertação decompõe-se em um conjunto de tarefas, dentre elas, destacam-se:

- Revisar a literatura buscando identificar conceitos e iniciativas relacionadas ao domínio abordado nesta dissertação;
- Identificar fontes apropriadas para a aquisição dos conceitos de engenharia de software que serão utilizados no povoamento da ontologia;
- Definir técnicas de combinação dos padrões definidos na ontologia com conceitos de engenharia de software;
- Conduzir uma investigação com o propósito de implementar um módulo de gerenciamento da ontologia, apoiando a combinação de predicados sensoriais com conceitos de engenharia de software;
- Implementar e povoar a ontologia;
- Definir requisitos de pré-processamento de textos em listas de discussão de software;
- Implementar módulo de ETL textual;
- Replicar experimento feito anteriormente na listas de discussão do projeto *Open Source Apache*, aplicando dicionário contextualizado;
- Analisar e comparar os resultantes dos experimentos, buscando observar aspectos de qualidade do novo dicionário do NEUROMINER.

1.4 Justificativa

Este trabalho aborda contextos já conhecidos por comunidades especializadas, porém, até então, pouco explorados no que se refere à construção de soluções que extraiam e classifiquem perfis cognitivos de desenvolvedores no domínio da Engenharia de Software. Particularmente, através de comunidades virtuais de projetos *Open Source*, as quais possuem extensa comunicação entre seus participantes através de listas de discussão, possibilitado análises textuais de diversos tipos de indivíduos.

Esse cenário possibilita a descoberta de conhecimento acerca do SRP de membros de equipes e são úteis para a tomada de decisão de gerentes de projetos, contribuindo e influenciando nos processos de construção, teste e manutenção do software. A ferramenta mais importante no processo de desenvolvimento de software são as pessoas. O sucesso de uma equipe depende, principalmente, da interação entre seus membros.

A fundamental contribuição dessa pesquisa será a proposta de um modelo classificador que permitirá combinar, lexicalmente e sintaticamente, predicados sensoriais a conceitos de um domínio específico através do uso de uma estrutura ontológica, objetivando identificar SRP's de desenvolvedores. Atualmente, o uso de ontologias tem sido amplamente empregado em representações do conhecimento de domínios restritos, principalmente em sistemas de busca de informação e processamento de línguas naturais (ALMEIDA & BAX, 2003), nos quais a sua aplicação pode ser mais eficaz por tratar, justamente, de conjuntos léxicos de número finito. (ZAVAGLIA, 2005).

Silva (2007) defende que "[...] o uso de ontologias aliado a sistemas capazes de derivar conclusões e inferir sobre a relevância da informação pode fornecer uma compreensão mais apurada do domínio, resultando numa classificação textual eficiente[...]" (SILVA, 2007, p 2). Assim, ontologias vêm ajudando a padronizar a comunicação entre humanos-humanos e humanos-computadores, mostrando-se qualificada para promover interoperabilidade e facilidade de integração entre unidades e fontes de informação.

Neste contexto, entende-se que esta dissertação perfaz uma importante contribuição para a comunidade especializada em engenharia de software e psicometria, propondo e validando uma estrutura ontológica que possibilitará combinar predicados sensoriais neurolinguísticos a conceitos de um domínio específico. Além disso, com esta proposta, pode-se trocar o domínio a ser minerado, adicionando conceitos pertencentes a outras áreas.

Por fim, o público alvo desta pesquisa pode ser favorecido pela disponibilidade da ontologia proposta e do novo dicionário neurolinguístico, possibilitando o compartilhamento do conhecimento gerado entre a combinação dos predicados sensoriais e os conceitos da Engenharia de Software.

1.5 Aspectos metodológicos

A pesquisa em pauta utilizou as dimensões qualitativas e quantitativas, visto que "O conjunto de dados quantitativos e qualitativos [...] não se opõem [...], ao contrário, se

complementam, pois a realidade abrangida por eles interage dinamicamente, excluindo qualquer dicotomia" (MINAYO, 1994, p. 22).

A pesquisa bibliográfica e documental constitui-se como um procedimento metodológico para a coleta de informações. Sabe-se que "a pesquisa bibliográfica se utiliza fundamentalmente das contribuições dos diversos autores sobre determinado assunto e [...] a pesquisa documental vale-se de materiais que não receberam ainda tratamento analítico, ou que ainda podem ser reelaborados de acordo com os objetivos da pesquisa" (GIL, 1999, p. 66).

Após o levantamento das fontes bibliográficas - acerca de engenharia de software, mineração de texto, ferramentas de análise de texto computadorizada, programação neurolinguística e ontologia - ocorreu a leitura e organização das idéias empregadas no estudo em questão. A pesquisa documental se deu através da análise dos e-mails de projetos *OSS* e dos conceitos de engenharia de *software*[2] disponíveis através da ontologia proposta por Wongthongtham et al. (2009).

Após as etapas descritas acima, os trabalhos para melhoria da ferramenta de mineração de texto NEUROMINER foram traçados. O primeiro passo foi nomear os predicados sensoriais que compuseram nosso dicionário LIWC. No caso da obtenção desses dados complementares, recorreu-se à pesquisa de campo, momento em que se utilizou da experiência de outros trabalhos nos campos da psicometria e psicologia. (COLAÇO, 2011, TAUSCZIK & PENNEBAKER, 2010, IRELAND et al., 2010 , KAHN et al., 2007, KO et al., 2006, YOUNG et al., 2005).

Em seguida, identificou-se que os arquivos contendo os e-mails de projetos *OSS*, no idioma inglês, possuíam estruturas distintas em torno da organização dos textos. A partir do estudo desses padrões, foi possível desenvolver o módulo ETL[3] textual com o intuito de transformar os dados e classificar os termos.

Após consolidar os dados obtidos com o processo descrito acima, foi desenvolvida uma ontologia contendo restrições e axiomas[4] que incidem sobre as classes e instâncias, as quais refletem regras de combinação semântica lexical entre predicados sensoriais e conceitos de engenharia de software.

2 Disponível através do website: http://natsuda.tomcat.debii.curtin.edu.au/webprotege/#SpecificOnto

3 Sigla em inglês para Extract, Transform and Load

4 Sentença que não é provada ou demonstrada e é considerada como óbvia ou como um consenso inicial necessário para a construção ou aceitação de uma teoria.

Para a construção dessa ontologia, utilizou-se a ferramenta Protégé[5] por ser uma ferramenta visual, gratuita e de fácil manipulação para construção de ontologias. Aproveitamos também o OWLViz[6], um gerador de gráficos com instâncias, herança e outros tipos de relacionamentos. Além disso, foram utilizados o Sesame[7] e a API GKMT[8] no processo de povoamento da ontologia e manipulação do novo dicionário.

Para validar os resultados através de técnicas estatísticas de comparação de conjuntos de medidas, Wainer (2007) preconiza que em casos mais completos deve-se executar um novo programa *Pn* e um ou mais programas competitivos *P1, P2,...Pk* nos mesmos dados de *benchmark*[9], objetivando validar seus resultados através de técnicas estatísticas de comparação de conjuntos de medidas. Assim sendo, um estudo experimental foi realizado, em duas etapas, para validar a proposta dessa dissertação, com o objetivo de avaliar a influência da ontologia no dicionário neurolinguístico e na ferramenta.

No primeiro, utilizou-se o NEUROMINER com a ausência da ontologia e, portanto, com a presença de um dicionário simples para identificar SRP's de desenvolvedores de software do Projeto *Apache*. No segundo experimento, foram utilizados a ontologia proposta e o novo dicionário na identificação dos SRP's dos desenvolvedores. Por fim, utilizamos técnicas estatísticas de comparação para validar diferenças significativas entre os resultados.

1.6 Organização da Dissertação

O capítulo após a introdução descreve a contextualização do domínio da pesquisa, seguido de um capítulo sobre as técnicas e ferramentas usadas neste trabalho. A descrição da ontologia desenvolvida encontra-se no Capítulo 4. Os experimentos realizados antes e depois da ontologia encontram-se no Capítulo 5. Após o Capítulo 6, reservado à conclusão, são apresentadas as referências bibliográficas utilizadas nesta dissertação.

Desta maneira, o restante desta dissertação encontra-se dividida nos capítulos descritos abaixo:

5 Disponível através do website http://protege.stanford.edu

6 Disponível através do website http://www.co-ode.org/

7 Disponível através do website: http://www.openrdf.org.

8 Api em Java para manipulação de ontologias. Desenvolvida pelo Grupo de Otimização da Web (GrOW) do Instituto de Computação da UFAL. Disponível através do website: www.grow.ic.ufal.br

9 Conjunto de dados que devem em princípio representar a possível diversidade dos "dados reais". (WAINER, 2007)

CAPÍTULO 1 – descreve os objetivos, motivação, questões da pesquisa, aspectos metodológicos, justificativa, assim como a organização do presente trabalho.

CAPÍTULO 2 – esse capítulo apresenta a contextualização do domínio da pesquisa, descrevendo os trabalhos relacionados e aspectos de gerenciamento de projetos de software. Alem disso, apresenta conceitos e utilidade da programação neurolinguística e como a descoberta de informação pode apoiar gerentes de softwares.

CAPÍTULO 3 – discorre sobre técnicas de mineração de texto, detalhando as etapas concernentes ao processo de mineração. Logo após, descreve o Neurominer, uma ferramenta de mineração de texto que tem como propósito a descoberta de informações sobre o sistema cognitivo preferencial de desenvolvedores de software. Ao fim do capítulo, é apresentada conceitos e técnicas de construção de ontologias.

CAPÍTULO 4 – apresenta a ontologia desenvolvida neste trabalho.

CAPÍTULO 5 – descreve os experimentos realizados com o objetivo de avaliar a influência da ontologia no dicionário neurolinguístico e na ferramenta.

CAPÍTULO 6 – são revisadas as contribuições, feitas as considerações finais e relatados os trabalhos futuros.

2 CONTEXTUALIZAÇÃO DO DOMÍNIO DA PESQUISA

Este capítulo tem por objetivo situar melhor o domínio da pesquisa em questão. Para isso, serão discutidos alguns aspectos relacionados à engenharia de software, ressaltando a importância da descoberta de conhecimento para gerentes de software no apoio a decisão. A área da Programação Neurolinguística também é investigada, evidenciando como a mente humana trabalha as habilidades de acuidade sensorial, e como elas podem levar a uma maior compreensão do comportamento humano.

2.1 Trabalhos Relacionados

Em uma abordagem mais próxima desta pesquisa, Riby e Hassan (2007) analisaram os e-mails do Projeto OSS *Apache* em busca de pistas de personalidade e características emocionais dos desenvolvedores. A pesquisa utilizou uma ferramenta LIWC, porém não foi desenvolvida para minerar e-mails e não é portável, ou seja, os e-mails foram tratados manualmente e a mudança do contexto explorado não é trivial. Em Colaço et al. (2010), foi apresentado um relatório parcial e inicial do uso de classificações neurolinguísticas a partir da mineração de listas de discussão de projetos de software. Esse trabalho motivou e norteou a necessidade de aprimoramento da capacidade de seu dicionário através de combinações de predicados sensoriais e termos da engenharia de software.

Em se tratando de trabalhos de mineração de repositórios de softwares relacionados à descoberta de conhecimento, alguns estudos consideraram listas de e-mails, código fonte e *bugs* de erros como grandes fontes de dados não estruturados. Pattison et al. (2008) pesquisou a relação existente entre alterações em entidades de software e suas referências através de e-mails. Nia et al. (2010) analisou comunidades de projetos OSS com o objetivo de estudar a colaboração e descobrir padrões no fluxo das informações entre os desenvolvedores. Nesse mesmo caminho, Farias et al. (2011) realizou um estudo na lista de discussão de desenvolvedores do Projeto OSS *Apache* com o fito de colaborar para a análise visual da rede de colaboração e cooperação dos desenvolvedores.

Ainda no contexto de mineração de repositório de software, Santos et. al. (2009) propôs uma ferramenta capaz de sugerir módulos que deverão ser alterados pelo desenvolvedor no processo de manutenção do software. Essa pesquisa utilizou dados históricos de código fonte armazenados em sistemas de controle de versão como fonte de dados para a mineração e descoberta de conhecimento.

Tratando-se de ferramentas LIWC, que se baseiam na análise da frequência de palavras através do uso de um dicionário e técnicas de psicometria, Tausczik e Pennebaker (2010) faz um apanhado em vários métodos de análise de texto computadorizados e descreve um sucinto histórico acerca de LIWC, incluído como foi criado, validado e como pode ser usada. Além de oferecer um apanhado do uso da psicometria através das palavras, Newman et al. (2008) pesquisou a diferença na maneira em que homens e mulheres usam a linguagem. A pesquisa teve como fonte de dados 14 mil textos de 70 estudos distintos e demonstrou uma pequena, porém sistemática, diferença na maneira como homens e mulheres se expressam. Além de identificar diferenças na aplicação de *function words*[10] em textos femininos e masculinos.

Em se tratando de PNL, a base para seus modelos e técnicas pode ser encontrada em estudos psicológicos que envolvem o chamado 'efeito camaleão', o qual diz respeito a estímulos imitativos não-verbais e verbais (*matching*) para o aumento da empatia em uma comunicação. Ireland et al (2010) investigou o grau de integração ou *matching* em relacionamentos românticos de casais. O estudo identificou que a correspondência no estilo de linguagem prediz significativamente a estabilidade em um relacionamento. Em uma abordagem semelhante, Van Baaren et al. (2003) investigou o 'efeito camaleão' em um experimento executado em um restaurante, no qual um grupo de garçonetes tentou imitar o cliente e/ou usar a mesma linguagem empregada (*matching*) na confirmação dos pedidos. O estudo mostrou que a empatia entre os clientes e garçonetes aumentou no grupo que utilizou o 'efeito camaleão', aumentando significativamente os valores das gorjetas.

Em uma abordagem inédita, Bailenson e Yee (2005) analisaram indivíduos que interagiram com um software baseado em inteligência artificial – um agente para simulação de um indivíduo emitindo uma explicação. O agente que imitou movimentos dos participantes foi mais convincente, recebendo avaliações mais positivas. Foi o primeiro estudo de realidade virtual que demonstrou os efeitos de um imitador não-verbal automático para ganho de empatia. Essas evidências estabelecem uma base empírica para uma investigação mais aprofundada.

10 Refletem como as pessoas se comunicam. São compostas por: pronomes, preposições, artigos, conjunções e verbos auxiliares.

Tratando-se de trabalhos que utilizam ontologias para apoio a descoberta de informação, Silva (2006) propôs uma extensão da ferramenta MASTER Web para a classificação de artigos baseados em ontologia. Esse trabalho propôs uma ontologia do domínio de Inteligência Artificial combinada à ferramenta de mineração textual e sistemas de regras para a extração de informações e classificação de artigos científicos no domínio de IA. Considerando a importância da semântica em documentos escritos em linguagem natural no processo de construção de um software, Witte et al. (2008) propôs uma ferramenta de mineração de texto capaz de minerar semanticamente documentos de engenharia de software escritos em linguagem natural e povoar uma ontologia com o propósito de estabelecer uma relação entre artefatos descritos nos documentos e o código fonte do software.

O trabalho de Wongthongtham et al. (2009) tem como objetivo apresentar uma ontologia de engenharia de software para representar seu conhecimento. A ontologia é baseada no livro 'Software Engineering' escrito por Sommerville e no artigo científico 'Software Engineering Body of Knowledge (SWEBOK)'. A ontologia permite a padronização da comunicação e do compartilhamento de informações semânticas em projetos de softwares e usuários, possibilitando a comunicação humanos-humanos e humanos-computadores (processos de software).

2.2 Programação Neurolinguística (*Neuro-Linguistic Programming*)

A Programação Neurolinguística (PNL), criada na década de 1970 (BANDLER & GRINDER, 1979), alcançou considerável popularidade como uma abordagem para comunicação e desenvolvimento pessoal. O termo PNL é exaustivamente usado nas áreas de educação, gestão e treinamento. Contudo, apesar de terem sido publicadas evidências da PNL como modelo para compreensão e aprendizado (TOSEY & MATHISON, 2003), existem poucos trabalhos acadêmicos sobre o assunto.

A PNL foi desenvolvida por Richard Bandler e John Grinder. Bandler, cuja experiência era em matemática e gestáltica, foi estudar na Universidade de Santa Cruz na década de 70, onde desenvolveu uma efetiva colaboração com John Grinder, um professor de lingüística. Para Grinder e Bandler, o nome derivou dos estudos dos padrões ou programações criadas pela interação entre o cérebro (neuro – derivado do Grego neuron), a linguagem (lingüística – derivada do Latin lingua) e o corpo, que resultam em comportamentos tanto eficientes quanto ineficientes. Em outras palavras, a mente de um indivíduo é formada de

conexões padronizadas entre processos neurológicos, linguagem e estratégias comportamentais de aprendizado (programação) (DILTS et al., 1980).

As definições para a PNL são as mais variadas, por exemplo, na literatura promocional, encontrada em centenas de sites na internet, a mesma é conceituada como a "arte da excelência na comunicação", e por algumas vezes anunciada como um método de sedução e manipulação associada ao controle da mente. Por outro lado, seus criadores a conceituam como "o estudo da experiência subjetiva e o que pode ser calculado disso" (DILTS et al., 1980). O objetivo é analisar o impacto da comunicação verbal e não-verbal sobre a realidade intrapessoal e a qualidade da comunicação. Considerada como ciência aplicada pelos seus defensores, a PNL supostamente oferece procedimentos específicos e eficazes nos campos da educação, treinamento e administração. Em se tratando de relacionamentos, buscam-se os melhores tipos de interação e cooperação baseados em como as pessoas são e não como se pensa que devem ser.

A PNL trabalha as habilidades de acuidade sensorial, ajudando a desenvolver características de observação visual, cinestésica e auditiva. Um estudo mais aprofundado da PNL oferece diversas técnicas para alcançar esse objetivo, incluindo a utilização de padrões de linguagem e a maneira que as pessoas codificam e expressam suas experiências subjetivas. Segundo Tosey e Mathison (2009), PNL representa uma distintiva síntese inovadora acerca da interdisciplinaridade do conhecimento sobre a comunicação das pessoas, especialmente na sutileza de seus entendimentos das relações entre linguagem, traços psicológicos e comportamento. Por outro lado, oferece um desafio a qualquer forma de desenvolvimento dos indivíduos através de suas percepções das maneiras pelas quais a linguagem se comunica com o subconsciente influenciando, deste modo, as pessoas.

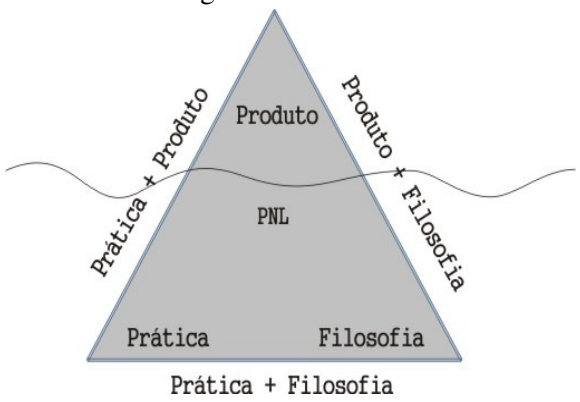
Entre as principais características da PNL está a orientação para o aprendizado como chave para mudança e desenvolvimento pessoal, pressupondo que as pessoas são intrinsecamente criativas e capazes, agindo de acordo com a forma que entendem e representam o mundo e não de acordo como o mundo é. A literatura constantemente cita a afirmação de Korzybski (1941) de que o "o mapa não é o território", uma referência ao entendimento individual que cada pessoa tem – modelo mental -, de acordo com a sua experiência, crenças, cultura, conhecimento e valores. PNL provê um conjunto de métodos e modelos que podem levar a uma maior compreensão do comportamento humano.

Para Tonsey e Mathison (2009), a PNL é representada em um mapa de faces baseada em três principais aspectos que os autores acreditam caracterizar o campo de abordagem. Esses aspectos são identificados como os três "P's":

- Prática (do inglês *Practice*): PNL como um comportamento, ou como uma comunicação prática.
- Filosofia (do inglês *Philosophy*): PNL como um conjunto de idéias e princípios;
- Produto (do inglês *Product*): PNL como um produto que pode ser consumido;

Uma representação das faces da NPL, adaptada de (TOSEY & MATHISON, 2009) é abordada na Figura 1.

Figura 1. Faces da PNL



Fonte: TOSEY & MATHISON, 2009. Adaptada

"Uma característica importante do mapa é que, metaforicamente, representa a PNL como um *iceberg*[...]"(TOSEY & MATHISON, 2009, p. 14). O aspecto mais evidente está acima da linha d'água e os identificados abaixo da superfície são menos óbvios, mas representam a principal essência da PNL.

Em Tosey & Mathison (2007), é apresentada a PNL sob uma perspectiva epistemológica, com princípios científicos que muitas vezes não são apresentados em sua literatura promocional. Os primeiros trabalhos publicados por Bandler e Grinder (BANDLER & GRINDER, 1979) (DILTS et al., 1980) foram baseados nos modelos de Fritz Perls, fundador da Gestáltica, Virginia Satir, pesquisadora em terapia de família, e Milton Erickson, doutor em medicina, mestre em psicologia e hipnoterapeuta de reconhecimento mundial. Conseqüentemente, a visão epistemológica da PNL fortalece sua teoria sobre os processos através dos quais os indivíduos podem perceber, conhecer e aprender. Parte desta teoria foi

explorada nessa pesquisa: a utilização de um sistema preferencial de representação para cognição.

Esta representação interna ou sistema de representação é altamente contextualizado, ou seja, varia de acordo com a situação, esse contexto vai influenciar diretamente o sistema utilizado (EINSPRUCH & FORMAN, 1985). Quando temos um estímulo dos nossos sentidos, inconscientemente filtramos ações sensoriais através de nossas experiências e geralmente nossas emoções ou sentimentos são acessados e estimulados antes do "pensamento". Desta maneira, algumas pessoas, em específicos contextos, podem preferir comunicar e/ou aprender utilizando um ou mais sistemas básicos de cognição (DENT, 1983) (MATTHEWS, 1991) (PETERS et al., 2008) (COLAÇO et al., 2010), que são:

> (1) Visual, que envolve a criação de imagens internas e a utilização de visões ou observações das coisas, incluindo fotos, diagramas, demonstrações, exposições, folhetos, etc;
>
> (2) Auditivo, o que envolve lembranças de sons e transferências de informação através da escuta; e
>
> (3) Cinestésico, que envolve as sensações internas de toque, emoções, experiências físicas, o realizar para entender e a prática.

Pesquisadores que estudam a PNL acreditam que os predicados, ou seja, palavras que uma pessoa escolhe para descrever uma situação – quando elas são especificadas pelo sistema de representação – podem revelar características comportamentais internas. Esses predicados podem nortear uma comunicação eficiente ou simplesmente serem despercebidos.

Um dos grandes problemas da comunicação, seja ela utilizada em uma conversa informal, em uma comunicação textual, ou para expor a funcionalidade de uma classe de software, é a falta de habilidade para despertar o interesse de quem está lendo ou ouvindo. Muitas vezes, quem recebe a mensagem não sabe e não consegue absorver o que está sendo transmitido. Para PNL, uma das maneiras de otimizar a comunicação é identificar o sistema representacional que está sendo utilizado pelo indivíduo e, através de um processo chamado *matching*, utilizar o mesmo sistema para a construção de empatia. O *matching* consiste em identificar os predicados que indicam um sistema representacional e utilizar os mesmos para comunicação, ou utilizar outros predicados, desde que sejam pertencentes ao mesmo sistema

utilizado pelo transmissor da mensagem (DILTS et al., 1980; NIEDERHOFFER and PENNEBAKER, 2002).

Para exemplificar, num processo de *matching*, uma frase como, "você viu a lógica dos algoritmos que lhe mostrei?", uma resposta coerente seria: "ainda não, mas agora que deixei minha mesa com uma visão limpa e arrumada, vou examiná-los com toda a atenção". As palavras sensoriais "viu" e "mostrei" da primeira frase indicam um processamento visual, desta forma, a resposta usou o mesmo sistemas através das palavras sensoriais visuais "visão limpa" e "examiná-los".

Essa pesquisa lidou com a identificação de palavras sensoriais e sistemas representacionais preferenciais de programadores em listas de discussão, analisando essas preferências com as especificidades do desenvolvimento de softwares OSS.

2.2.1 Onde PNL pode ser aplicada?

Para Tosey e Mathison (2009), a PNL trata da comunicação humana, sendo assim suas potenciais aplicações são infinitas. Alguns exemplos podem ser encontrados em revistas específicas, quais sejam "*Rapport*" e "*The model*". A ANLP[11] (do inglês *Association for NLP*) reporta aplicações em diversas áreas, como negócios, educação, saúde, relacionamentos, entre outras. Além de aplicações em organizações que incluem *Astra Zeneca, British Telecom, e Towergate Insurance*. Algumas aplicações típicas serão ilustradas nas seções seguintes.

Aplicações em negócios: São recentes no contexto da PNL. Uma adaptação do modelo de linguagem da PNL para o mercado empresarial, proposta por John Grinder, foi provavelmente o primeiro exemplo desses fenômenos. O "*The Precision Model*" (MCMASTER & GRINDER, 1980) propõe um conteúdo sobre tecnologia da gestão e modelos de liderança.

PNL enquanto ferramenta orientada a resultados: Tosey e Mathison (2009) nos revelam um caso em que gerentes reconheceram que a ênfase em suas reuniões era focada em problemas, de tal modo que não imaginavam o que realmente queriam alcançar ao final das reuniões, proporcionando um ambiente não produtivo. A partir desse cenário, cada reunião foi iniciada discutindo sobre os reais resultados que desejariam alcançar a partir de um problema. (KNIGHT, 2002 apud TOSEY & MATHISON, 2009). Este exemplo ilustra

11 http://www.anlp.org

o princípio da PNL que quando mudamos a linguagem que usamos podemos mudar a maneira como enxergamos uma situação, e também como nós podemos agir diante dela.

Vendas: Uma das principais áreas influenciadas pela PNL. A habilidade de acessar e reconhecer informações simples sobre como algumas pessoas podem ser mais visuais ou auditivas, por exemplo, é caracterizada como um fator importante para a conciliação da linguagem e criação de harmonia no momento da abordagem ou venda de um produto. A PNL também tem sido promovida como uma forma de sucesso em vendas por telefone (ZARRO & BLUM, 1989 apud TOSEY & MATHISON, 2009). A identificação do sistema cognitivo de uma pessoa (visual, auditivo ou cinestésico) e a utilização de padrões de linguagem pode conduzir o vendedor a uma interação facilitadora ao telefone.

Meta-programas: PNL propõe um modelo de preferências psicológicas e comportamentais, com base na forma como as pessoas respondem e no processamento de informações (CHAVERT 1997 apud TOSEY & MATHISON, 2009). São estratégias motivacionais que pessoas usam em determinadas situações. Um simples exemplo de meta-programas é referenciado pelos mesmos autores como "*towards – away from*", isso descreve que uma pessoa pode ser motivada pelo desejo de evitar uma consenquência indesejável (*away from*), ou ser atraída em direção (*towards*) a resultados estimulados por seus sentidos e imaginação.

Aplicação na educação: NPL é considerada uma área pouco explorada, especialmente em relação a sua aplicação na educação, mas atualmente há um grande crescimento de seu uso na esfera educacional. "É consenso geral a importância da interação e comunicação na relação entre o ensino e aprendizagem" (EVIDENCE FOR EDUCATION, 2011, tradução nossa). Dezenas de pesquisas e estudos de caso descritos por Carey et al. (2010) demonstram que professores reconhecem a potencialidade da aplicação de PNL, em relação a linguagem e comunicação, para a relação ensino e aprendizagem. O *CfBT education Trust*[12], localizada no Reino Unido, oferece periodicamente cursos de aperfeiçoamento em PNL, inclusive cursos de pós-graduação. Desde 2004, milhares de professores concluíram alguma formação em PNL. Segundo Tosey e Mathison (2009), pesquisas posteriores têm mostrado que isso teve um impacto positivo no desenvolvimento de professores e profissionais da educação.

12 Website official: http://www.cfbt.com/

Treinamento: O uso de PNL em diversos treinamentos nas organizações vem crescendo ao longo dos anos. O CfBT e ANLP vem contribuído para esse crescimento com a formação de diversos profissionais na área de treinamento motivacional com foco em negócios. Hayes (2006 apud TOSEY & MATHISON, 2009) afirma que a PNL possui um excelente conjunto de ferramentas para ajudar as pessoas a atingirem seus objetivos, assim como técnicas que podem ser usadas em diferentes treinamentos. Essas estratégias podem ajudar gerentes e executivos a terem uma visão mais consciente das diversas dimensões acerca da forma de pensar e agir.

2.2.2 Uso da linguagem para descoberta de traços de personalidade

As palavras que usamos diariamente em nosso cotidiano refletem os nossos sentidos: o que estamos pensando, como estamos nos sentindo, como estamos nos relacionando, ou seja, quem somos. A maneira como as pessoas escrevem e se expressão transmite pistas linguísticas acerca das características pessoais de personalidade.

De acordo com Tausczik e Pennebaker (2010, p. 2, tradução nossa), "[...] a linguagem é a maneira mais comum e confiável das pessoas traduzirem seus pensamentos e emoções internas de forma que os outros possam entender [...]". A análise de textos pode indicar o uso de padrões particulares sobre predições e também refletir estilos de personalidades. Como enfatizou Pennebaker e Graybeal (2001, p 2-4, tradução nossa):

> Através da análise de diferentes tipos de amostras de texto - tanto de fontes escritas e faladas - é possível explorar o uso da linguagem em uma variedade de situações e, em seguida, relatar seu uso para muitos aspectos que envolvem experiências, como o funcionamento do canal cognitivo, personalidade e vida social. [...] Os indivíduos têm maneiras únicas em seus estilos comportamentais, como por exemplo, a maneira de andar, vestir-se ou sorrir. Isto permite que eles tenham características individuais ao se expressar em textos. Esse estilo de linguagem, então, poderia ser considerado uma sólida diferença individual ou um estilo de personalidade. [...] O uso da linguagem reflete alguns mecanismos cognitivos básicos que podem ser mais reveladores sobre a natureza humana que alguns tradicionais métodos usados na psicologia [...]

Kahn, et al. (2007) identificou que palavras como *love, nice, sweet* são usadas em textos quando o escritor está abordando eventos positivos. Ao mesmo tempo, palavras como *hurt, ugly, nasty* são usadas para representar emoções negativas. Pesquisas sugerem que contagem de palavras em textos livres identifica, precisamente, emoções individuais. (KAHN, et al 2007; CHUNG and PENNEBAKER, 2008; NGWENYA, MILLS, KINGSTON , 20??).

Muitas pesquisas assumem que linguagens são contextuais. Assim, para a descoberta de conhecimento, precisamos de uma análise qualitativa e interpretada por parte de

um analista humano. Uma alternativa para esse cenário é que as palavras podem ser contadas e analisadas estatisticamente. Tais resultados podem ser vistos nas pesquisas de: Tausczik e Pennebaker (2010), Kahn (2007), Ireland et al. (2010), Newman et al. (2008), Gill e Oberlander (2006) e Ngwenya, Mills e Kingston, (20??), que apresentam estudos sobre o uso de ferramentas de contagem de palavras em amostras de texto com a finalidade de analisar características de personalidade.

Essa estratégia é baseada na suposição de que as palavras que as pessoas utilizam em seus textos ou falas imprimem informações psicológicas que vão além do seu significado literal e independente do seu contexto semântico. (PENNEBAKER et al., 2003)

Oportunamente, classes gramaticais foram divididas em categorias que refletem o que as pessoas querem dizer, o que se preconizou chamar de *content words*, e como se comunicam, *style words*. Sabe-se que "[...] *content* e *style words* tendem a ser processadas na mente de formas muito diferentes [...]" (MILLER, 1995 apud TAUSCZIK e PENNEBAKER, 2010, p. 5, tradução nossa).

Content words são compostas por verbos, adjetivos, substantivos e advérbios. Por exemplo, na frase "*This might be a brilliant algoritm*" temos as *content words* "*brilliant*" e "*algoritm*". *Style words* são compostas por pronomes, preposições, artigos, conjunções e verbos auxiliares e interligam o conteúdo do texto. "*This*", "*might*", "*be*", "*a*" representam as *Style words* da frase acima. . (TAUSCZIK and PENNEBAKER, 2010). "[...] No idioma Inglês, existem aproximadamente 200 *style words*, porém essas representam mais da metade das palavras em uso." (NEWMAN, et al., 2008, p. 6, tradução nossa)

Abaixo é ilustrado o exemplo abordado por Oberlander e Gill (2006, p. 3. tradução nossa), que apresenta um estudo sobre a comparação de diferenças individuais em comunidades de e-mails. O exemplo traz pequenos trechos de e-mail de diferentes autores que abordam suas recentes experiências profissionais.

1. *Hi, I'm just back at uni since yesterday, I'm finishing my project proposals. It's going OK, but I really don't want to be slogging over it at the weekend.*
2. *Hi. This has been my first full week of work in my new job. Actually, not much has changed because I have the same office and the same computer, and AM doing much the same work.*
3. *Hi, how's your week been? Mine has been OK but not very interesting. This weekend everyone is going skiing except me, which will be great fun …NOT!*

4. *Hi again. OK, so next week I have a few things planned. As I said before I need to start revising - so work is going to be a focus point. I think if I try to do some studying everyday, then I can still have fun at night.*

Os textos são semelhantes, porém os trechos sugerem diferenças no estilo lingüístico, que podem levar a diferenças individuais, qual sejam experiências anteriores, caráter, personalidade ou até mesmo aspectos cognitivos.

Programas de análise de texto computadorizados, baseado em contagem de palavras, representam essa estratégia para reconhecer informações de maneira objetiva e rápida a partir de amostras de textos. Partículas linguísticas sólidas apresentam propriedades psicométricas que estão relacionadas com características psicológicas (GROOM e PENNEBAKER, 2002; RIGBY and HASSAN, 2007).

Quando as primeiras ferramentas de análise de texto computadorizadas surgiram, o objetivo principal foi construir um sistema capaz de ajudar especialistas a mapear processos psicológicos e identificar o conteúdo expressado por pessoas em textos ou discursos. Neste capítulo, apresentaremos sucintamente alguns métodos abordados por Pennebaker et al. (2003) e Ireland et al. (2010):

The General Inquirer. "[...] General Inquirer é considerada a "mãe" das ferramentas de análise de texto computadorizada [...]" (Op. cit., p. 4, tradução nossa). Ela foi projetada para ser uma ferramenta de análise de texto multiuso com base em complexas rotinas de contagem de palavras. (STONE et al., 1966). A *General Inquirer* além de contar palavras, apresenta um conjunto de regras pré-programadas para tentar elucidar o significado correto no texto dependente do contexto.

A ferramenta possibilita a criação de novos dicionários e a especificação de novas regras para auxiliar no entendimento de palavras ambíguas.

Analyzing Emotion-Abstraction Patterns: TAS/C. É um assistente de análise de textos. O TAS/C foca em duas dimensões de linguagem, abstração e emocional. No dicionário de abstração há aproximadamente cerca de 3900 entradas que, em média, representam 4% do texto.

Além de analisar textos sob o prisma das duas dimensões, a ferramenta foi preparada para mensurar habilidades não verbais, caracterizadas nos discursos por concretude, especificidade, clareza e imaginação (MERGENTHALER and BUCCI, 1999 apud PENNEBAKER et al., 2003). Pronomes em terceira pessoa e preposições são identificados e sumarizados para mensurar essas habilidades.

DICTION. Esta ferramenta foi projetada para avaliar discursos políticos. Ela utiliza cinco variáveis estatisticamente independentes: atividade, otimismo, certeza, realismo e semelhança.

Existem cerca de 10 mil palavras associadas a diversas categorias. Uma característica importante dessa ferramenta é a capacidade de aprender com atualizações de textos processados.

***Linguistic Inquiry and Word Count* – LIWC**. O LIWC[13] foi inicialmente desenvolvido por Pennebaker et al. (2001) para descobrir características em textos sobre emoções negativas com o objetivo de ajudar pacientes. Seu dicionário foi criado sob base em técnicas de psicometria.

Posteriormente, seu dicionário foi categorizado em aproximadamente 70 dimensões e mais de 2300 termos. Essas dimensões incluem quatro categorias: linguagem, que abrange artigos, preposição e pronomes; as que envolvem processos psicológicos, tais como, emoções positivas e negativas e aspectos cognitivos; as relativas ao termo, como tempo verbal, movimento e espaço e por fim, as categorias referentes ao conteúdo, que são: sexo, morte, lar e ocupação.

As dimensões do LIWC são divididas hierarquicamente, portanto um termo pode estar inserido em várias categorias em uma estrutura *bottom-up*. Uma nova versão do LIWC implementou uma série de atualizações em seu dicionário. Algumas dimensões foram ligeiramente alteradas, outras atualizadas e algumas excluídas (LIWC, 2007).

Diversos experimentos[14] vêm utilizando LIWC para tentar descobrir informações sobre o comportamento humano através de análise de amostras de textos. LIWC procura e analisa freqüência de palavras isoladas, ou seja, u-grama[15]. Ferramentas de análise de texto mais poderosas são capazes de analisar estruturas de linguagem mais complexas. Estudos começam a utilizar técnicas de contagem de palavras com n-gramas[16] da mesma forma que LIWC sumariza a freqüência de u-gramas. (COLAÇO et al, 2010, OBERLANDER and GILL, 2006)

13 Website oficial http://www.liwc.net

14 Experimentos que utilizaram LIWC. http://homepage.psy.utexas.edu/homepage/faculty/pennebaker/reprints/. Acessado em 05.out.2011

15 Grupos de apenas uma palavra que designam conceito ou entidades de um campo altamente especializado da atividade humana.

16 Grupos de duas ou mais palavras juntas que forma um termo e designam conceito ou entidades de um campo altamente especializado da atividade humana.

.

Biber: Factor Analyzing the English Language. Essa ferramenta de análise de texto foi desenvolvida a partir dos trabalhos de Biber (1988 apud PENNEBAKER et al., 2003). Foi considerada um *framework* útil para analisar variações linguísticas e de gênero em diferentes tipos de textos.

Suas dimensões foram construídas a partir de 23 gêneros literários, 481 textos e aproximadamente um milhão de palavras. Suas análises são inovadoras na medida em que identifica no texto a estrutura do idioma usada nos gêneros.

***Language Style Matching* – LSM**. Recentemente desenvolvida com o objetivo de analisar os relacionamentos entre as pessoas. Tem como base a coordenação verbal inconsciente. "[...] Intuitivamente quando interagimos com os outros, nós nos adaptamos a eles [...]. Quando duas pessoas estão falando, seus comportamentos comunicativos são padronizados e coordenados, como uma dança" (NIEDERHOFFER & PENNEBAKER, 2002, p. 2, tradução nossa).

LSM é uma medida do grau de combinação do estilo de linguagem em conversações, faladas ou escritas, entre duas pessoas, ou seja, mensura o quando o relacionamento é síncrono. Para isso, LSM utiliza *function words* que são independentes do contexto.

> [...] embora dois amigos que trabalham em uma construtora e em uma pedreira, respectivamente, usariam diferentes content words durante uma conversa sobre o dia de trabalho de cada um, pesquisas sugerem que suas function words seriam semelhantes na medida em que eles se gostam e entendem um ao outro. (GONZALES et al., 2010; PICKERING and GARROD, 2004 apud IRELAND et al., 2010, p. 2, tradução nossa)

Programas tradicionais de análise de textos não têm subsídios para identificar automaticamente os significados adjacentes dos termos dentro de um determinado contexto. Portanto, pode categorizar erroneamente alguns tipos de palavras. Newman et al. (2008) defende que a alternativa para esse problema encontra-se no uso da estratégia de contagem de palavras com o emprego de técnicas estatísticas. Nesse cenário, ferramentas de contagem estão propensas a erros, mas a incidência de uma identificação errônea – devido ao contexto ou outro significado semântico – em grandes amostragens de texto é muito baixa. (Op. cit.).

Uma simples contagem de palavras não é condição única para o entendimento de um texto, sobretudo de traços psicológicos de indivíduos. "[...] O poder de ferramentas de contagem de palavras está no seu dicionário categorizado [...]" (RIGBY e HASSAN, 2007, p. 2, tradução nossa).

Por exemplo (Op. cit., p. 2), as duas frases abaixo nos trazem conotações diferenciadas acerca da confiabilidade no contexto específico de engenharia de Software:

1. *I sent these patches in a couple weeks ago. They maybe of interest, maybe not.*
2. *This patch fixes the important half of the bug.*

Na frase 1, o termo "*maybe*" é associado a incerteza do autor acerca da importância dos *patches*. Na segunda frase, a confiança do autor é associada à palavra "*important*". Ferramentas tradicionais de contagem de palavras utilizam a estratégia que busca sumarizar termos u-grama contidos em seus dicionários e assumem que palavras transmitem informações sobre traços de personalidade independentes do contexto semântico. O dicionário é agrupado dentro de linguística básica e dimensões psicométricas e cada termo pode ser associado a uma ou mais categorias.

Para Rigby e Hassan (2007), no exemplo acima, os termos "*patch*" e "*bug*" não estão presentes no dicionário, portanto, mesmo que o contexto das frases estivesse inserido em outro domínio o resultado seria o mesmo. Para uma análise textual mais aprofundada, o dicionário deve estar diretamente associado ao contexto analisado.

Deste modo, acredita-se que o poder do dicionário é incrementado quando há a contabilização de conceitos do domínio (contextualização) combinados a palavras sensoriais (COLAÇO et al. 2010). Esse aspecto será abordado com mais detalhes na seção 4.

2.3 Aspectos de Gerenciamento de Projetos de Software

Um projeto é um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo (PMI, 2008). Em se tratando de projeto de software o produto final é um software. Uma característica evidente em projetos de software é a sua unicidade. Mesmo seguindo metodologias e procedimentos de desenvolvimento de softwares, um projeto de software é único, ou seja, tem suas características individuais, como escopos diferentes e equipes com perfis diferentes.

Ainda de acordo com PMI (2008), o gerenciamento de projetos é a aplicação de conhecimento, habilidades, ferramentas e técnicas às atividades do projeto a fim de atender aos seus requisitos. Devem ser consideradas e gerenciadas diferentes áreas de conhecimento no projeto: escopo, tempo, custo, qualidade, recursos humanos, comunicações, riscos, aquisições e integração.

Neste ínterim, gerenciar a equipe do projeto de software e os recursos utilizados é uma das principais funções do gerenciamento de projeto. Isso significa que o gerente é responsável por acompanhar o desempenho de membros da equipe, fornecer *feedback*, alocar desenvolvedores em atividades específicas e coordenar mudanças para otimizar o desenvolvimento do software. Neste contexto, informações de alto nível são imprescindíveis para o gerente de projeto, possibilitando uma visão mais tática dos problemas de forma que possam ser implementadas ações mais precisas, corretivas ou de melhorias, para conduzir o projeto condizente com o caminho planejado ou levando-o a caminhos mais produtivos (KAWANO, 2009).

Na área de engenharia de software, as atividades que mais consomem recursos são as atividades de projeto, codificação e teste. Nesse ponto, gerentes de projetos de softwares precisam tomar decisões que, em última análise, afetarão o sucesso da implementação do software e a facilidade com que este será mantido. Essas decisões são tomadas durante o projeto e são fundamentais na fase de desenvolvimento e manutenção (PRESSMAN, 1995).

Segundo Witte et at. (2008), a tarefa de manutenção de software, muitas vezes também referenciada como a evolução do mesmo, constitui, de modo geral, uma parcela significativa que ocorre durante o ciclo de vida deste. Quanto mais maduro o software, mais complexo para mantê-lo. Manutenção de software é uma tarefa difícil e complicada, devido a diferentes representações e inter-relações que existem entre artefatos de software, recursos de conhecimento e equipe de desenvolvimento.

Informações contidas em fontes de dados não estruturadas resultantes dessas fases são importantes para o apoio a gerentes de software no desempenho de suas atividades. Frequentemente é complexo extrair essas informações dessas fontes de dados, dessa maneira acaba por não serem visualizadas e utilizadas no processo de gerenciamento de projeto de software.

A implementação de melhorias em processos de desenvolvimento de software é um fenômeno sociocultural, ou seja, ultrapassa fatores tecnológicos, como por exemplo, deve-se considerar o comportamento social e as características individuais dos membros da equipe (MONTONI & ROCHA, 2011). Destarte, conhecer as preferências representacionais dos desenvolvedores de software pode ser considerado uma informação estratégica. Dessa maneira, desenvolvedores podem ser alocados em tarefas mais próximas ao seu Sistema

Preferencial e/ou agrupados de acordo com o SRP da equipe, criando uma empatia e aumentando o grau de produtividade do projeto de software. A ferramenta mais importante no desenvolvimento de software são as pessoas e o sucesso ou fracasso de uma equipe em um projeto de software depende principalmente da interação entre os membros.

Em tarefas de desenvolvimento ou manutenção de um software, desenvolvedores utilizam diferentes recursos e canais cognitivos, tais como:

(1) Visual: diagramas, fluxogramas, modelos gráficos, imagens representando classes e figuras de linguagem representando discussões do dia a dia.

(2) Auditivo: há discussões com colegas da equipe sobre as descrições de um diagrama de caso de uso, por exemplo.

(3) Cinestésico: utilização de um passo a passo para o aprendizado de uma nova tecnologia ou a compreensão de um novo escopo de um projeto.

Nesse contexto, um programador pode preferir uma estratégia VAC (Visual, Auditivo,Cinestésico) para compreender uma classe, ou seja, ele pode preferir ver vários diagramas (V) sobre a classe, seguindo com uma narrativa (A) do funcionamento da classe e somente então ter uma experiência prática (C) com a mesma. Neste caso específico, diríamos que o desenvolvedor tem uma estratégia equilibrada. Em uma situação diferente, o mesmo desenvolvedor pode preferir uma estratégia Visual, através de análise de fluxograma, para o entendimento global do escopo de um software.

Colaço (2011) discorre que o desempenho dos programadores no processo de representação, cognição e comunicação é o principal fator de sucesso no desenvolvimento de softwares. A equipe depende, em alta proporção, das interações e colaborações entre os membros, a qual é influenciada pelas características da personalidade de cada um.

Em síntese, predizer informações acerca dessas características, analisando o impacto verbal e não-verbal sobre o comportamento do indivíduo em cada contexto, pode resultar em informações de apoio a decisão de alto nível. Por exemplo, as estratégias descobertas podem nortear qualquer comunicação no contexto de desenvolvimento de software.

3 TÉCNICAS DE MINERAÇÃO DE TEXTO E ONTOLOGIA

3.1 Mineração de Texto

Em pleno século XXI, a informação é vista como parte sutil do capital e consiste em um dos bens mais valiosos dentro de uma organização. Dispor da informação correta no momento oportuno vem sendo caracterizado como um imprescindível diferencial para gestores de empresas.

A coleta intensiva de dados não estruturados através de atividades diárias como relatórios de sistemas, listas de discussão, questionários eletrônicos, código fonte de sistemas e *log* de erros provê informações valiosas que podem se revelar como uma fonte promissora para a mineração de dados, em específico a mineração de texto. Contudo, dispor de uma mina de dados não é garantia de um diferencial competitivo, conforme sintetiza Magalhães (2008):

> [...] uma vasta quantidade informacional não assegura uma posição de destaque no ranking das melhores organizações – além de possuir a informação, é necessário dispor de mecanismos que facilitem esse processo de recuperação, objeto de estudo da área conhecida como *Information Retrieval*, em português, recuperação da informação. (Magalhães, 2008. p. 39)

Uma definição de mineração de dados, adaptada de SAMPAIO (2009), infere que se trata de uma tecnologia que visa extrair automaticamente conhecimento útil, confiável e não trivial de uma base de dados. De forma análoga à mineração de dados, a mineração de texto procura descobrir conhecimentos úteis em 'minas' de dados textuais, ou seja, não estruturadas. Desta forma, apresenta-se como uma forma de garimpar grandes bases em busca da informação necessária para o processo de tomada de decisão.

A mineração de texto vem sendo uma alternativa importante para a descoberta de padrões ocultos em diversas áreas do conhecimento, entre elas a de engenharia de software (COLAÇO et al., 2010; SANTOS, 2009; WITTE et al., 2008; RIGBY & HASSAN, 2007; KO et al., 2006; GAIZAUSKAS et al., 2005). Entretanto, minerar dados na forma de linguagem natural não é uma tarefa trivial, textos são escritos e organizados de forma livre e na maioria das vezes as informações não estão disponíveis em apenas uma fonte de dados. Técnicas refinadas de mineração e recuperação da informação combinadas com métodos estatísticos e um dicionário especializado são comuns na construção de ferramentas de mineração de texto (KONCHADY, 2006), retratando um verdadeiro processo de mineração.

3.1.1 Origem da mineração de texto

Embora o termo 'Mineração de texto' seja relativamente novo, esta área está ligada diretamente a pesquisas em Recuperação da Informação - RI. No contexto de RI, as informações podem ser oriundas de textos, imagens, áudio, vídeo e outros objetos multimídia (BALINSKI, 2002). Nos últimos anos, houve um crescimento em pesquisas relacionadas à recuperação da informação em virtude da disseminação dessas informações digitais (SEBASTIANI, 1999 apud MAGALHÃES, 2008).

3.1.2 Recuperação da Informação

Segundo Konchdy (2006), o termo 'recuperação da informação' originou-se historicamente na década de 60, quando grandes sistemas foram desenvolvidos para *mainframes* com o objetivo de tratarem coleções de documentos não estruturados. Nos anos 80, esses sistemas ganharam interfaces intermediárias para PC's com o propósito de viabilizar pesquisas e a recuperação de dados. Eram pesquisas baseadas em palavras chave, ainda hoje utilizadas em alguns 'motores' de busca na web. Em meandros da década de 90, os esforços no desenvolvimento de sistemas de RI foram focados em desempenho, conectividade e em pequenas melhorias na utilização de processamento de linguagem natural.

A utilização de palavras chave em questões simples é suficiente para a recuperação de informação, mas quando se tem questões mais complexas, o uso, apenas, desse recurso não é condição para se ter respostas satisfatórias. Portanto, a utilização crescente de Processamento de Linguagem Natural – PLN vem crescendo em ferramentas de recuperação de informação textual. Em sistemas de recuperação de informação que não utilizam PLN temos interrogativos como 'quem', 'quando' e 'o que' são ignoradas e somente *content words*[17] são utilizadas (KONCHADY, 2006).

A combinação de técnicas de recuperação de informação, métodos estatísticos e PLN compõem as novas ferramentas de recuperação de informação que são capazes de encontrar respostas difíceis ou impossíveis de serem encontradas em sistemas tradicionais de recuperação de informação.

O processo de recuperação de informação assemelha-se ao processo de consultas em Sistemas de Gerenciamento de Banco de Dados – SGBD, mas na contra mão a essa afirmação existe uma grande diferença entre esses processos. Isso se deve porque um SGBD

17 Uma abordagem detalhada sobre *content word* será vista na sessão 2.3

funciona sobre uma base de dados bem estruturada, ou seja, entidades e relacionamentos originários da teoria matemática de conjuntos. Em um sistema de recuperação de informação, a fonte de busca é constituída por bases de dados não estruturadas e a consulta geralmente utiliza modelos probabilísticos.

3.1.3 Áreas de aplicações com mineração de dados

Ainda que problemas com processamento automático e identificação de linguagem natural pareçam complexos, muitos métodos oriundos de procedimentos estatísticos, inteligência artificial e recuperação da informação vêm sendo utilizados para construção de soluções úteis para mineração de texto. Abaixo são ilustradas algumas aplicações de mineração de texto, sintetizadas por Konchady (2006):

Pesquisas: questões com respostas de múltipla escolha são rápidas e simples de tabular e sumarizar, contudo questões abertas que aceitam respostas com linguagem natural são difíceis de processar. Tabular e interpretar manualmente uma base de dados com milhares de questões abertas não é uma tarefa trivial. O uso de técnicas de mineração textual é imprescindível para situações como essas. Por exemplo, se uma pergunta for em relação a algum tipo de mudança em algum tipo de processo na organização, as respostas podem ser categorizadas automaticamente em positiva, negativa ou neutra.

Gerenciamento do relacionamento com o cliente: é muito importante para as organizações conhecer e entender seus clientes. O retorno da satisfação em relação a produtos e/ou serviços é recebido de várias formas, o cliente pode enviar um e-mail, ligar para a central de suporte técnico ou postar uma mensagem no site da empresa. Todas essas informações em linguagem natural podem ser armazenadas e analisadas com diferentes objetivos.

Suporte técnico: se uma mensagem de um cliente retrata uma dúvida ou problema que não pode ser resolvido no primeiro nível de atendimento, a mensagem é redirecionada para um atendente especialista. Um sistema de mineração de texto, contendo conhecimento de especialistas, pode redirecionar a mensagem para o atendente apropriado, ou ainda responder a mensagem com respostas automática.

Recrutamento on-line: métodos tradicionais de anúncios de vagas de emprego e recrutamento foram transformados em procedimentos executados na web. Atualmente, muitos dos processos de recrutamento que antecedem à entrevista presencial são realizados através de e-mails ou postados em *web sites*. Algumas empresas usam bases de conhecimentos e

ferramentas de mineração de texto para executarem *matching*[18] entre candidatos e vagas disponíveis. A primeira etapa desse processo consiste na extração automática de informações, seguido do cruzamento de dados e da categorização dos candidatos.

Medicina: a prática da medicina gera e consome um grande volume de informação. Estudos, relatórios clínicos, registros de hospitais, anotações de médicos constituem ricas fontes de informações. A maioria dessas informações está na forma de texto. Resultados acerca da mineração de texto no âmbito da medicina podem se vistos na pesquisa de Moreira et al. (2009).

Monitoramento de opinião pública: a opinião das pessoas sobre determinados assuntos é de grande interesse de políticos e cientistas sociais. Métodos tradicionais de recuperação e processamentos dessas informações são imprecisos e lentos. Uma grande fonte dessas informações são as redes sociais. Ferramentas de mineração de texto podem ajudar na análise automática de informação e revelar sentimentos positivos ou negativos sobre diversas questões.

Apoio a engenharia de software: conhecer características de projetos de softwares e de desenvolvedores é um recurso que pode ajudar a muitos gerentes de projetos na tomada de decisão. Atualmente, os métodos e metodologias de desenvolvimento de software geram ricas bases de dados textuais com importantes informações sobre o processo de desenvolvimento. É possível recuperar esses dados e transformá-los em informação valiosa para o aperfeiçoamento desse processo. Resultados podem ser vistos nos trabalhos: Colaço et al. (2010), Gegick et al. (2010), Nia et al. (2010), Santos et. Al (2009), Witte et al. (2008), Rigby e Hassan (2007), Ko et at. (2006).

3.1.4 Entendendo a mineração de texto

Compreender textos de forma automática não é um tarefa simples para sistemas de mineração de texto. Para analisar textos não estruturados, muitas ferramentas de mineração textual utilizam algum tipo de conhecimento prévio e processamento de linguagem natural. Linguagem natural é diferente, por exemplo, de linguagens de programação, que de modo geral, somente pode ser escrita de uma única forma (sintaxe). Todavia, linguagem natural é mais flexível, dessa maneira novas expressões e regras surgem frequentemente, assim

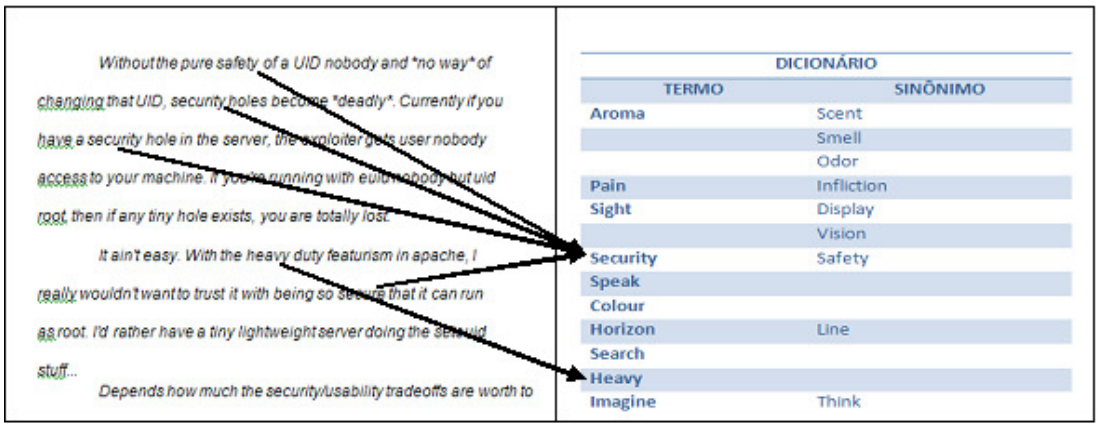
18 Processo de combinação de determinados níveis de características

construir uma ferramenta que alcance todas as regras gramaticais é praticamente impossível (KONCHADY, 2006).

Em linguagem de programação, uma sentença tem apenas um significado, enquanto que em linguagem natural uma sentença pode ter diferentes interpretações. Na sentença, 'Eu vi o servidor leve e limpo, desta maneira o algoritmo poderá brilhar', os termos 'leve' e 'limpo' podem ser interpretado respectivamente, em um contexto geral, como adjetivos que indicam o peso físico e o nível de limpeza do servidor. O verbo 'brilhar' pode ser entendido como a emissão de luzes através do algoritmo. Em uma outra análise e em um contexto específico de engenharia de software, os adjetivos 'leve' e 'limpo' serão interpretados como indicativos de um bom desempenho para o servidor e o verbo 'brilhar' denotará que o algoritmo irá funcionar bem. "[...] Sem nenhum conhecimento prévio acerca do texto analisado, uma única interpretação da sentença não poderá ser feita com precisão. [...]"(Op. cit., p. 7, tradução nossa).

Conforme ilustrado na Figura 2, dicionários e/ou tesauros, representando conhecimentos prévios, podem ser utilizados em ferramentas de mineração para prover soluções capazes de extrair e processar dados em textos de áreas específicas.

Figura 2. Uso de um dicionário na mineração de texto.



Fonte: Autor

3.1.5 Processo de mineração de texto

Para Magalhães (2008), o processo de mineração de texto assemelha-se ao processo de mineração de dados, do mesmo modo dividido em quatro etapas – seleção, pré-processamento, mineração e assimilação – conforme ilustrado na Figura 3. Após o pré-processamento, as etapas que seguem compõem o processo de mineração de dados tradicional.

Figura 3. Mineração de texto

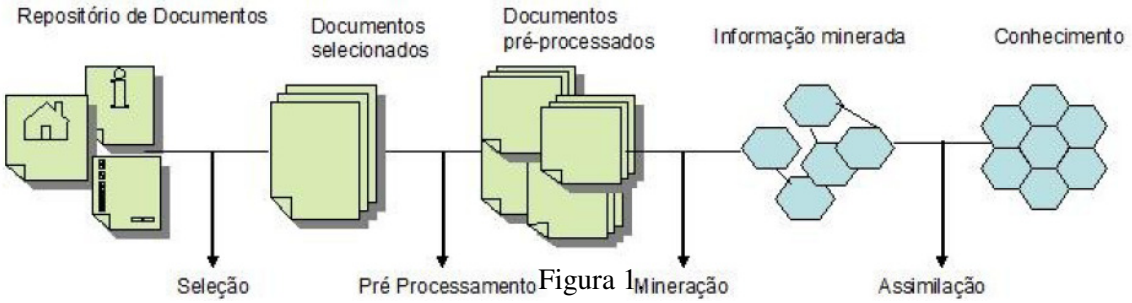


Figura 1

Fonte: (MAGALHÃES, 2008. p. 43)

3.1.5.1 Seleção

Nesta etapa, as bases de dados a serem utilizadas para a mineração deverão ser escolhidas. O objetivo da mineração deverá ser levado em consideração para a seleção.

3.1.5.2 Pré-processamento

Fontes de dados não estruturadas, por muitas vezes, necessitam ser formatadas, adaptadas e integradas. Os dados devem ser transformados em um padrão que sirva de entrada para algoritmos de mineração. Esta etapa, geralmente, é subdividida em: padronização do formato de texto, remoção das *stopwords* e conversão dos termos em radicais, *stemming*.

A padronização do formato de texto é responsável em padronizar o formato original em um formato contendo apenas texto e remover itens, como imagens, formatação ou qualquer outro item não textual.

As *stopwords* são palavras com elevada frequência na maioria dos textos, como pronomes, preposições, artigos, conjunções e advérbios. Esses termos não revelam pistas acerca do conteúdo do texto analisado. As *stopwords* podem ser removidas para melhorar a performance no processo de mineração.

Não existe exatamente uma coleção de palavras que compõem a lista de *stopwords* porque ela é dependente da categoria do texto a ser minerado. No Quadro 1, é ilustrado uma lista de *stopwords* proposta por Konchady (2006) que aborda uma base universal de palavras na língua inglesa. Contudo, uma única lista de *stopword* não é apropriada para todas as categorias de texto, as listas podem ser customizadas a partir do gênero do texto a ser minerado.

Quadro 1. Lista de stopwords

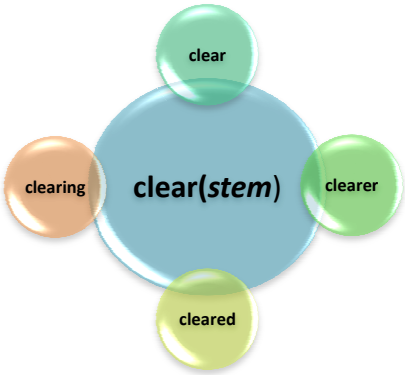
about add ago after all also na and another any are as at be because been before being

between big both but by came can come could did do does due each else end far few for from get got had has have he her here him himself his how if in into is it its just let lie low make many me might more most much must my never no nor not now of off old on only or other our out over per pre put re said same see she should since so some still such take than that the their themthen there these they this those through to too under up use very via want was way we well were what when where which while who will with would yes yet you your

Fonte: KONCHADY, 2006. p 96

Conversão dos termos em radicais, segundo Magalhães (2008, p 49) "[...] é o processo de combinar as formas diferentes de uma palavra numa representação comum, o radical (em inglês, *stem*)". Isso é feito para agrupar os termos que tenham o mesmo significado conceitual, conforme retratado na Figura 4. Este processo auxilia na classificação e filtragem dos textos, haja vista a redução no número de termos que podem ser identificados e pontuados, por conseguinte, se um verbo aparece *x* vezes e seu passado *y* vezes em um texto, sem a conversão dos termos em radicais existiriam duas frequências diferentes para serem contabilizadas, no entanto com a radicalização existirá apenas as ocorrências para seu radical, ou seja *x* + *y* (BALINSKI, 2002).

Figura 4. Conversão de termos em radicais



Fonte: Autor

Há vários algoritmos de radicalização disponíveis na língua inglesa. Um dos algoritmos mais utilizados em pesquisas na área de mineração de texto e mais citado na literatura é o Porter Stemming Algorithm (PORTER, 1980). Segundo Balinski (2002), "Este é um algoritmo simples e muito eficiente para a radicalização de termos. O algoritmo é executado em cinco passos, sendo que cada passo realiza uma transformação sobre o termo alvo.(BALINSKI, 2002, p. 31)". O Quadro 2 retrata um exemplo de um texto pré-processado, ou seja, padronização do seu formato, remoção das stopwords e conversão dos termos em radicais.

Quadro 2. E-mail pré-processado. Retirado da lista de discussão do projeto Apache.

Texto no seu formato original

```
From: sameer <sameer@c2.org>
Message-Id: <199601010953.BAA05354@infinity.c2.org>
Subject: Re: setuid() again
To: new-httpd@hyperreal.com
Date: Mon, 1 Jan 1996 01:53:31 -0800 (PST)
Reply-To: new-httpd@apache.org

> Ahem.
>
> Apart from the danger of processing the request as root, and the objections
> above, totally safe.

      uh, i hope that last sentence was sarcasm.

      Without the pure safety of a UID nobody and *no way* of
changing that UID, security holes become *deadly*. Currently if you
have a security hole in the server, the exploiter gets user nobody
access to your machine. If you're running with euid nobody but uid
root, then if any tiny hole exists, you are totally lost.

      It ain't easy. With the heavy duty featurism in apache, I
really wouldn't want to trust it with being so secure that it can run
as root. I'd rather have a tiny lightweight server doing the seteuid
stuff...

      Depends how much the security/usability tradeoffs are worth to
you.

--
sameer                                  Voice:   510-601-9777x3
Community ConneXion                     FAX:     510-601-9734
The Internet Privacy Provider                 Dialin:  510-658-6376
http://www.c2.org/ (or login as "guest")            sameer@c2.org
```

Texto pré-processado

```
hope last sentenc sarcasm
without pure safeti
chang uid secur hole becom deadli current
secur hole server exploit get user
access machin run with euid uid
root tini hole exist total lost
ain easi with heavi duti featur apach
realli wouldn trust with secur run
root rather tini lightweight server do seteuid stuff
depend secur usabl tradeoff worth
```

Fonte: Autor

3.1.5.3 Atribuição de pesos

Após os procedimentos que compõem a etapa de pré-processamento, o texto é transformado em uma coleção de termos que serão associados a pesos e podem categorizar o texto. Em geral, a determinação do peso de um termo em um documento pode ser efetuada mediante dois paradigmas (MAGALHÃES, 2008):

- Quanto mais vezes um termo aparece no documento, mais relevante ele é para o tópico do documento.
- Quanto mais vezes um termo ocorre dentre todos os documentos de uma coleção, menos importante ele é para diferenciar os documentos.

Os valores dos pesos podem variar entre 0 ou 1 para pesos booleanos ou valores que se baseiam em técnicas estatísticas relacionado a frequência do termo no documento, pesos numéricos. Os pesos numéricos podem ser representados conforme as medidas a seguir (MANNING et al., 2008):

Frequência do termo (*term frequency – tf*): é a atribuição do número de ocorrência de um termo *t* em um documento *d* ao seu peso. A frequência do termo no documento fornece pistas úteis sobre a sua importância para o documento.

Frequência do documento (*document frequency – df*): número de documento *d* na coleção que contêm pelo menos uma ocorrência do termo *t*. Sabe-se que, estatisticamente, é melhor usar o número de documento que contenha o termo a usar toda a coleção de termos.

Frequência inversa do documento (*inverse document frequency – idf*): Atribuir pesos apenas sob a luz da frequência do termo pode ocasionar um problema crítico: todos os termos são considerados igualmente importantes quando se trata de categorizar um texto. De fato, alguns termos não têm poder de discriminação na determinação de sua relevância. Encontrar o termo '*software'* em um documento não será importante para diferenciá-lo dos demais documentos em uma coleção de documentos de engenharia de software, por exemplo. Para este fim, introduzir um mecanismo para atenuar o efeito de termos com elevada frequência na coleção é essencial para garantir sua relevância da categorização. O Quadro 3 ilustra a equação *idf* que indica a ideia de reduzir o peso *tf* de um termo através de um fator que cresce com sua frequência na coleção de documentos.

Quadro 3. Equação idf.

$idf_t = \log \frac{N}{df_t}$, **onde**:
N representa o total de documentos na coleção.
$\boldsymbol{df_t}$ representa a frequência do termo nos documentos.

Fonte: Autor

Frequência do termo - Frequência inversa do documento (*tf-idf*): é a combinação da frequência de um termo com sua frequência inversa, produzindo um peso composto para cada termo *t* em cada documento *d*.

Em outras palavras, a partir do Quadro 4, inferi-se que $tf - idf_{t,d}$ atribui ao termo *t* um peso no documento *d* que é:

- **Maior** quando o termo t ocorre muitas vezes dentro de um número pequeno de documentos;
- **Baixo** quando o termo t ocorre poucas vezes em um documento, ou ocorre em muitos documentos;
- **Baixíssimo** quando o termo t ocorre em praticamente todos os documentos.

Quadro 4. Equação Tf-idf.

$tf - idf_{t,d} = tf_{t,d}\ x\ idf_t$, **onde**:
$\boldsymbol{tf_{t,d}}$ representa a frequência do termo *t* no documento *d*.
$\boldsymbol{idf_t}$ representa a frequência inversa.

Fonte: Autor

De forma subsequente, os dados transformados resultantes dos procedimentos expostos acima são submetidos a algoritmos de mineração textual a fim de descobrir conhecimentos úteis ou interessantes acerca de padrões, modelos ou regras escondidos nos textos originais.

3.2 NEUROMINER

O NEUROMINER é uma ferramenta de mineração de texto que combina técnicas de análise de dados não estruturados, estatística e predicados sensoriais da PNL e conceitos de engenharia de software, com o objetivo de classificar o Sistema Representacional Preferencial de desenvolvedores de software. Utiliza listas de e-mails de projetos de softwares como fonte de dados e realiza as fases de pré-processamento e extração dos termos de maneira automática, um dos pontos positivos dessa ferramenta. O NEUROMINER é a primeira ferramenta a realizar análise neurolinguística de texto de maneira automática.

O NEUROMINER usa *Linguistic Inquiry and Word Count* (LIWC), similar ao apresentado por Pennebaker et al. (2001), mas especialmente criado para identificar o SRP de desenvolvedores de software em um contexto específico. Foi inspirada no trabalho de Rigby e Hassan (2007), uma análise psicométricas de texto, que é baseada em psicometria e na teoria neurolinguística.

O dicionário criado para o NEUROMINER possui três perfis básicos (Visual, Cinestésico e Auditivo), bem como os predicados sensoriais (BANDLER & GRINDER 1979; DILTS et al. 1980) correspondentes a cada um deles na línguas inglesa.

O NEUROMINER extrai automaticamente, a partir de seu dicionário neurolinguístico, os termos da amostra de texto investigada e os armazenam em uma estrutura de dados multidimensional, que será apresentada na próxima Seção.

3.2.1 Modelo Multidimensional

O *Data Mart* do NEUROMINER foi implementado na mesma arquitetura de *Data Warehouse* (DW) desenvolvida anteriormente para mineração de repositórios de software (COLAÇO et al. 2009). Assim, as dimensões LIWC que descrevem os perfis neurolinguísticos e os conceitos de engenharia de software foram modeladas como uma única dimensão (*Profile_Dim*) no esquema estrela (KIMBALL et al. 1998; COLAÇO Jr. 2004). Essa dimensão está associada a uma dimensão dicionário (*Dictionary_Dim*) que armazena todos os termos correspondentes. No modelo, um termo pode ser: um predicado sensorial, um conceito da engenharia de software ou uma relação entre um predicado e um conceito.

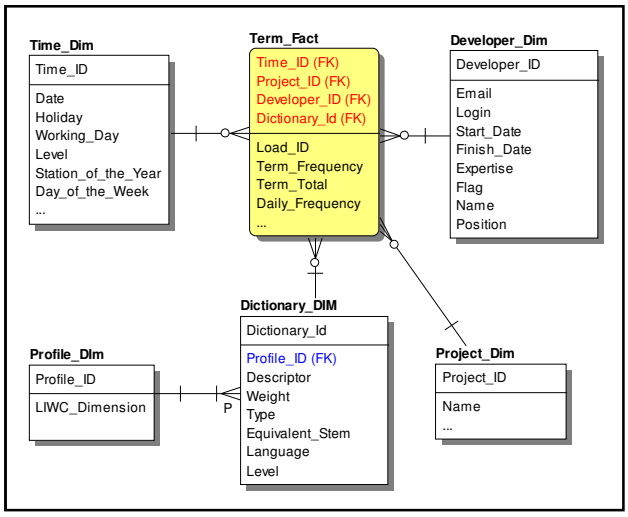
Desta forma, os conceitos da engenharia de software foram marcados como substantivos, elementos centrais, e os predicados sensoriais aptos para esse fim, como palavras adjacentes ou modificadores, vide detalhamento na Seção 4. Por exemplo, o fragmento de texto em inglês "*concrete algorithm*" será considerado uma frase pelo NEUROMINER. Além disso, esse termo será computado como processamento cinestésico, uma vez que seu modificador pertence à dimensão cinestésica. Em outras palavras, ao encontrar um conceito da engenharia de software no texto, o NEUROMINER buscará predicados sensoriais adjacentes.

Todos os modificadores próximos a um conceito são computados, independente do sistema utilizado. Por exemplo, na frase: "*i saw it and ran the algorithm*", são pontuados os sistemas visual e cinestésico, devido à presença dos modificadores "*see*" e "*run*", pertencentes aos sistemas visual e cinestésico, respectivamente (COLAÇO et al., 2010).

A tabela de fatos (*Term_Fact*), em um nível de granularidade mensal, armazena os escores calculados para os termos encontrados, bem como dados atômicos tal como a freqüência do termo, utilizados para cálculo destes escores. A Figura 5 apresenta a tabela *Term_Fact* e parte do modelo utilizado.

A Tabela 1 descreve as principais dimensões do modelo. Na Tabela 2, são detalhados os atributos da tabela de fatos *Term_Fact*. A próxima seção explica o cálculo destes atributos.

Figura 5. Modelagem Multidimensional para as dimensões LIWC



Fonte: Autor

3.2.2 ETL dos Emails

As atividades de extração, transformação e carga (ETL) da ferramenta foram desenvolvidas a partir de etapas tradicionais de ETL e de recuperação de informação textual [Colaço Jr. 2004; Witte et al. 2008].

Tabela 1. Dimensões

Dimensão	Descrição
Profile_Dim	Esta dimensão descreve os perfis neurolinguísticos e os conceitos da Engenharia de Software (Dimensões LIWC).
Dictionary_Dim	Representa os temos correspondentes às dimensões LIWC. No modelo, um termo pode ser: um predicado sensorial, um conceito da Engenharia de Software ou uma relação entre um predicado e um conceito.
Time_Dim	Possui campos tais como descrição da data, dia útil e semestres. Importante para armazenar o que uma função de banco de dados não pode retornar e também para melhorar o desempenho de consultas.
Developer_Dim	Descreve características dos desenvolvedores, tais como nome (*Name*), competência (*Expertise*) e função (*Position*).
Project_Dim	Descrição dos projetos aos quais os desenvolvedores estão alocados.

Fonte: Autor

Tabela 2. Term_Fact

Atributo	Descrição
daily_frequency	Número de dias que o termo aparece dividido pelo núm. de dias nos quais houve envio de emails.
term_frequency	Freqüência do termo no mês
term_frequency_total	Freqüência acumulada do termo
term_number	Quantidade total do termo no mês
weight_total	Peso atual do termo (*score)*
term_total	Quantidade total do termo (acumulado)
days_number	Números de dias que contém o termo

Fonte: Autor

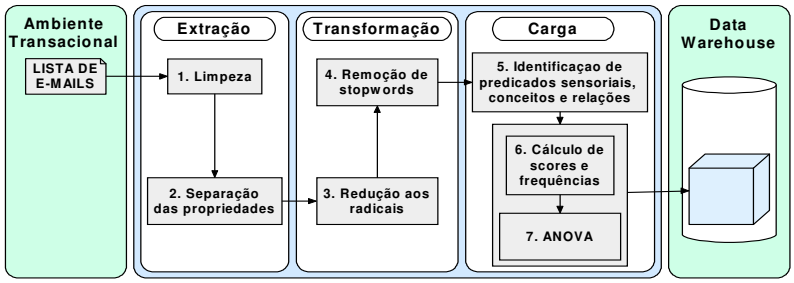
A Figura 6 ilustra a cadeia básica do processo. Um arquivo texto contendo lotes de e-mails é submetido ao processo de extração. No passo 1, o extrator remove cabeçalhos, rodapés, assinaturas, códigos de softwares e anexos de maneira automática, retornando um texto limpo e estruturado para a separação das propriedades: remetente, assunto, data e mensagem (passo 2). Além disto, apenas o texto do remetente é considerado, sendo removido o texto de outrem, geralmente anexado a uma resposta (RIGBY & HASSAN, 2007).

Na etapa de transformação (passos 3 e 4), os conteúdos das mensagens são refinados através da remoção de *stopwords* e da redução das palavras aos seus radicais (ver seção 3.1.5.2) [Porter 1980; Witte et al. 2008]. Na última etapa, os termos são detectados (passo 5) e, em seguida, no passo 6, as freqüências e *scores*[19] dos termos encontrados, tanto mensais quanto acumulados[20], são calculados e armazenados na tabela de fato *Term_Fact* (vide Figura 5), para um maior detalhamento dos cálculos de *score* ler Colaço et al. (2010). Por fim, um agregado com dados estatísticos descritivos dos *scores* é carregado, para servir de base para um teste de hipótese ANOVA (passo 7). Um sistema representacional só é classificado como o mais usado por um desenvolvedor, se, e somente se, considerando-se um nível de significância de cinco por cento, houver evidências estatísticas que as médias de *scores* são diferentes. Vale ressaltar que as classificações do NEUROMINER não são estanques, ou seja, ninguém é Visual, Auditivo ou Cinestésico, o objetivo é identificar o sistema de representação mais usado por um programador, dentro de um contexto, e o percentual de uso dos demais. Um indivíduo pode variar o SRP, dependendo do momento e do contexto (COLAÇO et al. 2010).

19 O *score* de um predicado sensorial *j* é dado por *(tf(j)+df(j)) x b*, onde tf é freqüência do predicado nos emails, *df* é freqüência diária (vide Quadro 3) e *b* é um bônus cujo valor é diferente de 1 apenas para frases.

20 O acúmulo de todos os emails enviados é considerado um grande texto escrito pelo desenvolvedor. Considerado um inovação do NEUROMINER.

Figura 6. Módulo ETL do NEUROMINER



Fonte: Autor

3.3 Ontologia

O principal componente de um sistema baseado em conhecimento é a sua base. A base de conhecimento é um conjunto de representação de fatos e regras sobre o domínio específico (SILVA, 2006). Nos últimos anos, estudos apontam consideráveis progressos na área de engenharia de conhecimento, que correspondem ao processo de construção da base de conhecimento, e desempenha papel importante no compartilhamento e reutilização de informação. Esses processos levaram as ontologias a serem largamente usadas em aplicações relacionadas ao gerenciamento do conhecimento, processamento de linguagem natural, engenharia de software, recuperação da informação, integração de informação e web semântica.

Segundo Coral et al. (2006), a web semântica contribuiu largamente para a evolução das ontologias no contexto semântico. Para Zavaglia, "[...] ontologias têm sido utilizadas para a representação de informações que veiculem um entendimento semântico comum de situações variadas do mundo real" (ZAVAGLIA et al., 2007, p. 2). A existência de especificações de regras, através dos relacionamentos, nas ontologias é um fator importante para tornar a semântica mais precisa e sem ambiguidade. "[...] Além disso, ontologias favorecem a inferência pela representação explícita do conhecimento e propiciam compreensão mais apurada do domínio abordado (SILVA, 2006, p. 44)".

Diante do grau de gerenciamento do conhecimento proporcionado pelas ontologias uma das principais utilidades do seu uso é a padronização no compartilhamento de informações, principalmente na padronização de comunicações. Pessoas e computadores não podem compartilhar conhecimento se não falam a mesma língua. Assim, ontologias podem ajudar a padronizar a comunicação entre humanos-humanos e humanos-computadores (WONGTHONGTHAM et al., 2009). Pode-se pensar no dia em que as pessoas poderão

realizar requisições ao computador utilizando linguagem natural e receberão resposta da mesma maneira que outro ser humano a daria.

3.3.1 Ontologias: Definições e conceitos básicos

O termo ‘ontologia’ tem origem no grego ‘ontos’, que significa ser, e ‘logos’, que significa palavra. Surgiu no século XIX, introduzido por Aristóteles. Gruber (1995) apresenta uma das mais conhecidas e compartilhadas definições de ontologia: “Uma ontologia define um domínio ou, mais formalmente, especifica uma conceitualização acerca dele”. Uma conceitualização é uma visão abstrata e sucinta do domínio real que se deseja representar em determinado contexto ou área, que corresponde a uma coleção de classes, objetos, conceitos, relações, funções e os axiomas formais que restringem o uso ou combinação dessas entidades.

As comunidades de Inteligência Artificial – IA e engenharia de conhecimento adotaram esse termo para representar os conceitos e seus relacionamentos em uma área do conhecimento ou para construir uma representação desses (SEMPREBOM, 2007). Ainda nesse contexto, ontologia pode ser definida como "[...] uma especificação formal e explícita de uma conceituação compartilhada [...] (ZAVAGLIA et al., 2007, p.3)". Studer (98 apud SILVA, 2006) explica essa definição como segue:

- Uma conceitualização refere-se a um modelo abstrato de algum fenômeno no mundo que identifica conceitos relevantes desse fenômeno;
- No que diz respeito a explícito, os tipos de conceitos usados e suas restrições estão definidos de forma clara, explicitamente, como conceitos, propriedades, relações, funções, restrições, axiomas e terminologia;
- Formal refere-se ao fato de que a ontologia é compreensível para agentes e sistemas;
- Compartilhada reflete o fato de que uma ontologia captura conhecimento consensual, isto é, ela não é privada a um indivíduo, mas aceita por um grupo.

Para Wongthongtham et al. (2009), ontologia, na área de ciência da computação, representa um esforço para formular uma estrutura conceitual exaustiva, rigorosa e formal dentro de um domínio. Portanto, devem ser interpretadas por humanos e principalmente por computadores, devido à estrutura formal aplicada.

O formalismo empregado nas ontologias define termos e relações que compõem o domínio de uma área específica e utiliza componentes básicos como: conceitos, relações,

funções, axiomas e instâncias, que serão conceituadas abaixo segundo Gruber (1995) e Silva (2006):

• **Conceitos**: são uma representação usada em amplo sentido. Um conceito pode ser abstrato ou concreto, elementar ou composto, real ou fictício. Em suma, conceito pode ser qualquer coisa sobre algo que é dito e, portanto, algo como uma tarefa, uma função, uma ação, uma estratégia, raciocínio, processo, etc.

• **Relações**: representam um tipo de associação entre conceitos do domínio. Elas são formalmente definidas como qualquer subconjunto de um produto de um conjunto. Assim: *R: C1 x C2 x ... x Cn*. Ontologias normalmente contêm relacionamentos binários, o primeiro argumento é conhecido como domínio da relação, e o segundo como o *range*. Um exemplo de relacionamento entre os conceitos *Autor* e *Livro* é o relacionamento *autor-de*. Assim teremos como domínio o *Autor* e como *range* o *Livro*.

• **Funções**: são casos especiais de relações, onde um conjunto de elementos tem uma relação única com outro elemento. Um exemplo é *Mãe-de*.

• **Axiomas**: servem para modelar sentenças que são sempre verdadeiras. Um axioma no domínio específico da língua inglesa seria afirmar que não é possível combinar um substantivo antes de um adjetivo em uma sentença simples.

• **Instâncias**: são usadas para representar elementos em uma ontologia. O termo '*Algorithm'* pode ser uma instância do conceito de engenharia de software.

Ainda neste sentido, de acordo com Maedche (2002 apud BREITMAN & LEITE, 2003), uma ontologia pode ser descrita em 5 *tuplas* incluindo os elementos: conceito, relações, hierarquia, funções e axiomas que são definidos no Quadro 5.

Quadro 5. Descrição da estrutura de uma ontologia

$O := \{C, \mathcal{R}, \mathcal{H}^c, rel, \mathcal{A}^o\}$, **onde**: $\boldsymbol{C}$ *representa o Conceito*; $\boldsymbol{\mathcal{R}}$ *representa a Relação*; $\boldsymbol{\mathcal{H}^c}$ *representa Hierarquia* $\boldsymbol{rel}$ *representa a Função* $\boldsymbol{\mathcal{A}^o}$*representa o Axioma*

Fonte: Autor

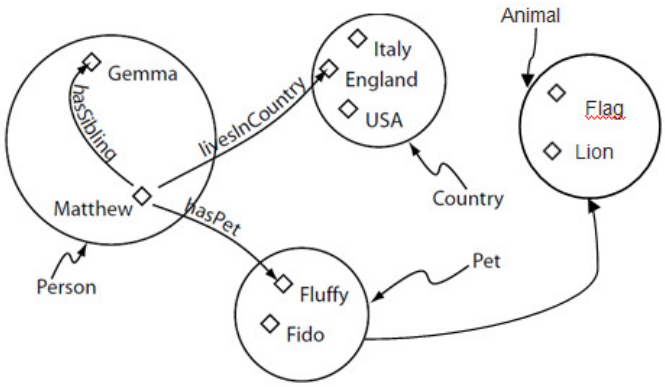
Para esse autor, uma hierarquia de conceito, $\mathcal{H}^{\mathrm{c}}$: $\mathcal{H}^{\mathrm{c}}$ é um relacionamento direto $\mathcal{H}^{\mathrm{c}} \subseteq$ C x C que é chamado de taxonomia. $\mathcal{H}^{\mathrm{c}}(C1, C2)$ significa que *C1* é um subconjunto de *C2*. A Figura 7 apresenta o uso da estrutura ontológica através do exemplo adaptado de Drummond et al. (2009) que representa um domínio de convivência. Nessa representação

existem três conceitos, *Person* ($C1$), *Country*($C2$), *Pet*($C3$) e *Animal*($C4$), uma hierarquia $\mathcal{H}^c$ e três relacionamentos, *hasSibling*($W1$), *liveinCountry*($W2$) e *hasPet*($W3$). Formalmente pode-se representar a estrutura ontológica como: $C := \{C1, C2, C3, C4\}$, $\mathcal{H}^c(C3, C4)$, $\mathcal{R} := \{W1, W2, W3\}$, $W1(C1, C1), W2(C1, C2)$ *e* $W3(C1, C3)$.

3.3.2 Classificação de ontologias

Podem existir diferentes classificações de ontologias de acordo com o nível de generalidade. Quatro tipos de ontologias serão descritas abaixo, conforme Guarino (1998 apud KAWANO, 2009; SILVA, 2006), e cujas hierarquias estão ilustradas na Figura 8.

Figura 7. Representação da estrutura ontológica



Fonte: DRUMMOND et al. 2009, p. 13, adaptado

Ontologias de alto-nível: descrevem conceitos muito gerais como espaço, tempo, matéria, objeto, evento, ação, etc. Esses conceitos são independentes de um problema particular ou domínio. Portanto, é razoável, pelo menos em teoria, ter uma ontologia de alto-nível compartilhada por grandes comunidades de usuários;

Ontologias de domínio: descrevem o vocabulário relacionado a um domínio específico através da especialização de conceitos introduzidos nas ontologias de alto-nível. Por exemplo, medicina, direito e engenharia de software;

Ontologias de tarefa: descrevem um vocabulário relacionado a uma tarefa ou atividade genérica, ou atividade através da especialização de conceitos presentes na ontologia de alto-nível.

Ontologias de aplicação: essas ontologias descrevem conceitos tanto das ontologias de domínio quanto das de tarefas. São as mais específicas por serem utilizadas de

Figura 8. Tipos de ontologias de acordo com o nível de generalidade

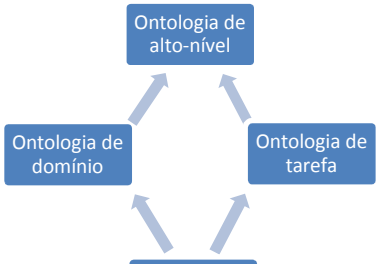


Figura 2.

Fonte: GUARINO, 1998 apud KAWANO, 2009

As ontologias possuem características e componentes básicos comuns - classes, relações, axiomas e instâncias - e não apresentam sempre a mesma estrutura, mas mesmo apresentando propriedades distintas é possível identificar diferenças entre elas. A generalidade é usada para classificar os tipos de ontologias apresentadas nesta seção. Outros tipos de classificação são descritas na Tabela 3, que faz um levantamento sucinto dos tipos de ontologias existentes, quais sejam, relacionando-as a sua função (MIZOGUCHI et al., 1995 apud ALMEIDA & BAX, 2003), ao grau de formalismo de seu vocabulário (USCHOLD & GRUNINGER, 1996 apud ALMEIDA & BAX, 2003), a sua aplicação (JASPER & USCHOLD, 1999 apud ALMEIDA & BAX, 2003) e à estrutura e conteúdo da conceitualização (VAN-HEIJIST et al., 1997 apud ALMEIDA & BAX, 2003).

Tabela 3. Tipos de ontologias

Abordagem quanto à função	
Classificação	**Descrição**
Ontologias de domínio	Reutilização no domínio fornece vocábulos sobre conceitos, seus relacionamentos, sobre atividades e regras que os governam.
Ontologias de tarefa	Fornecem um vocábulo sistematizado de termos, especificando tarefas que podem ou não estar no mesmo domínio
Ontologias gerais	Incluem um vocabulário relacionado a coisas, eventos, tempo, espaço, casualidade, comportamento, funções, etc.
Abordagem quanto ao grau de formalismo	
Ontologias altamente formais	Expressa livremente em linguagem natural
Ontologias semi-formais	Expressa livremente em linguagem natural de forma restrita e estruturada
Ontologias simi-formais	Expressa em uma linguagem artificial definida formalmente
Ontologias rigorosamente formais	Os termos são definidos com semântica formal, teoremas e provas
Abordagem quanto à aplicação	
Ontologias de autoria neutra	Um aplicativo é escrito em uma única língua e depois convertido para uso em diversos sistemas, reutilizando-se as informações.

Ontologia como especificação	Cria-se uma ontologia para um domínio, a qual é usada para documentação e manutenção no desenvolvimento de *softwares*.
Ontologias de acesso comum à informação	Quando um vocabulário é inacessível, a ontologia torna a informação facilmente compreendida, proporcionando conhecimento compartilhado dos termos.
Abordagem quanto ao conteúdo	
Ontologias terminológicas	Especificam termos que serão usados para representar o conhecimento em um domínio (léxicos).
Ontologias de informação	Especificam a estrutura de registros de banco de dados (esquemas de banco de dados).
Ontologias de modelagem do conhecimento	Especificam conceitualizações do conhecimento, têm uma estrutura interna semanticamente rica e são refinadas para uso no domínio de conhecimento que descreveram.
Ontologias de aplicação	Contêm as definições necessárias para modelar o conhecimento em uma aplicação.
Ontologias genéricas	Similares às ontologias de domínio, mas os conceitos que as definem são considerados genéricos e comuns a vários campi.
Ontologias de representação	Explicam as conceitualizações que estão por trás dos formalismos de representação do conhecimento.

Fonte: ALMEIDA & BAX, 2003, p. 4, adaptada.

3.3.3 Utilização de ontologias

O uso de ontologias vem crescendo em grande escala em diversas áreas do conhecimento, visto que "[...] vários campos de pesquisa têm reconhecido a importância das ontologias para modelagem, representação e reutilização do conhecimento [...]" (SILVA, 2006, p. 31), principalmente em áreas que buscam desenvolver um dicionário contendo os conceitos relativos a um domínio de aplicação específico.

Um grande volume de pesquisas sobre o tema sugere a importância e utilidade das ontologias na tarefa de organizar informações dentro de um determinado domínio do conhecimento (SICILIA et al., 2009; WONGTHONGTHAM et al., 2009; WITTE et al, 2008; ZAVAGLIA et al., 2007; SILVA, 2006;). Ontologias vêm sendo associadas a diversas aplicações de diferentes áreas com o objetivo de representar e facilitar o gerenciamento do conhecimento.

Na web semântica a informação possui um significado bem definido, permitindo assim que computadores e pessoas trabalhem em cooperação. Essa cooperação pode ser alcançada através do uso de soluções de compartilhamento do conhecimento. Dessa forma,

ontologias definem formalmente conceitos e relacionamentos entre termos, tornando-se uma imprescindível chave para essa área.

Em comércio eletrônico, a estrutura ontológica oferece informações que podem ser utilizadas para unificar e integrar semanticamente a comunicação entre diversos serviços através de um formato padrão e único. Dessa forma, aplicações comerciais podem comunicar-se com mais precisão e rapidez (ZAVAGLIA et al., 2007). Ainda no contexto web, ontologias podem ser usadas em motores de busca, servindo de suporte semântico. As respostas às consultas realizadas através de palavra-chave, além de retornarem páginas que contenham o termo pesquisado, podem oferecer outras páginas com informações associadas ao conceito utilizado.

Na área de engenharia de software, estudos vêm fazendo uso de ontologias para descobrir informações ocultas em projetos de softwares ou proporcionar melhorias em procedimentos como proposto por Witte et al. (2008) e Wongthongtham et al. (2009). Documentos escritos em linguagem natural e código fonte de software constituem uma importante parcela dos artefatos produzidos durante o ciclo de vida de engenharia de software. Witte et al. (2008), apresenta um sistema de mineração de textos capaz de popular uma ontologia com informações extraídas desses documentos, permitindo o cruzamento de artefatos de software e código fonte. Wongthongtham et al. (2009) propõe um projeto de uma ontologia[21] que facilite o compartilhamento dos conceitos de projetos e engenharia de software para todos os membros da equipe que estão separadas geograficamente.

Uma área bastante promissora é a que faz uso de processamento de linguagem natural. No campo de ação da representação formal do conhecimento, "[...] a ontologia pressupõe um enlace entre os símbolos da linguagem natural e as entidades do mundo real que ela representa" (ZAVAGLIA, 2005, p 2).

As ontologias podem proporcionar melhorias no processo de recuperação da informação, organizando o conteúdo das fontes de dados que compõem o domínio em questão. Para Palmeria & Freitas (2007), a união entre ontologias e sistemas capazes de inferir sobre a relevância da informação pode resultar em um entendimento mais detalhado e preciso do conhecimento, influenciando em uma classificação textual eficiente. Além de proporcionar uma prática troca do domínio explorado. Ainda segundo os mesmos autores, "[...] uma ontologia bem detalhada a respeito de um dado domínio leva a resultados de classificação de

21 Disponível em www.seontology.org

texto bastante promissores, mesmo com a aplicação de técnicas simples, como um pequeno conjunto de regras de produção" (PALMEIRA & FREITAS, 2007, p.1).

Zavaglia (2005) discorre que no campo do processamento de linguagem natural, principalmente em sistemas de bases de conhecimento lexical, é evidente que a inclusão de repositórios ontológicos é essencial para a representação do conhecimento. Uma ontologia pode servir como um importante repositório semântico para a representação do conhecimento, formando um vocabulário estruturado e organizado, para que seja possível resgatar o significado de um item léxico de forma unívoca. Além disso, à medida que o vocabulário do domínio e a ontologia são refinados a classificação torna-se mais eficaz.

Vários projetos e aplicações vêm fazendo uso de ontologias para a representação do conhecimento. Segue sucintas descrições, adaptada de Almeida & Bax (2003), da utilização de ontologias em projetos relacionados ao processamento de linguagem natural:

Oncolerm: Facilita a tradução de textos médicos sobre oncologia mediante uma ontologia baseada em textos especializados e dicionários médicos.

Gazelle: Traduz textos japoneses, árabes, e espanhóis para o inglês. Inclui o processamento e análises semânticas das línguas, geração de sentenças em inglês.

KMPL: Desenvolve gramáticas e gera linguagens naturais; oferece um ambiente de desenvolvimento gráfico para a construção, manutenção e uso de gramáticas para diversas línguas. Gera textos em espanhol no domínio da química e utiliza linguagem natural para responder à consulta sobre grupos, elementos e propriedades químicas. Utiliza uma ontologia da química, uma ontologia lingüística e uma gramática em espanhol.

Penman: Gera sentenças em linguagem natural a partir de uma entrada não linguística, aceita várias notações de entrada e foi projetado para uso por pessoas com vários graus de sofisticação linguística e computacional.

Techdoc: Gera documentos técnicos multilinguísticos (inglês, alemão e francês) a partir de uma representação independente construída pela análise comparativa de partes dos manuais técnicos e linguísticos disponíveis em diferentes línguas.

Além disso, de modo geral, uma ontologia lida com conceitos e especificamente - em contextos específicos - com signos linguísticos, dada a forma como os termos são relacionados e utilizados, como é o caso do WordNet[22] . Para França (2009, p. 3): "Considerar uma ontologia um objeto linguístico não depende da natureza da própria ontologia, mas do

22 http://wordnet.princeton.edu/

propósito para que é construída e da forma como olhamos para os termos com os quais queremos construir uma ontologia."

A mesma autora retrata um exemplo prático com uma abordagem a WordNet. Nesse exemplo a entrada '*cat*' é apresentada com vários significados e categorizada gramaticalmente em substantivo, tendo oito significados, e verbo, tendo dois significados. São apresentados, ainda, exemplos e sinônimos, sendo comum essa abordagem em dicionários de língua. Porém, o que faz da WordNet uma ontologia, segundo a autora, são as relações semânticas entre os termos.

3.3.4 Engenharia da Ontologia

O processo de construção de ontologias não é constituído por procedimentos rígidos e sistematizados. Segundo Pérez (1999 apud SILVA, 2006, p. 36) "o processo de construção de ontologias assemelha-se mais a uma arte do que a uma atividade de engenharia [...]". Mesmo fazendo uso de uma metodologia, as fases pré-estabelecidas podem não ser seguidas de forma absoluta, fazendo com que os desenvolvedores possam ser flexíveis e adotar critérios pessoais no processo de construção da ontologia. Além disso, de acordo com Noy & McGuinnes (2001), não há uma maneira ou metodologia definida e correta para criar uma ontologia, existem diversas maneiras de as mesmas serem desenvolvidas. Contudo, algumas regras fundamentais podem ser seguidas com o propósito de ajudar a tomar decisões de projeto (Op. cit. Tradução nossa):

- Não há uma maneira correta de modelar um domínio, sempre há alternativas viáveis. A melhor solução quase sempre depende da aplicação;
- Desenvolvimento de uma ontologia é necessariamente um processo interativo;
- Conceitos e relações, na ontologia, devem ser relacionados diretamente a objetos (físicos ou lógicos) em seu domínio de interesse.

Diante da complexidade em formalizar uma ontologia, independente do domínio observado, Gruber (1995) propõe alguns critérios que foram comprovadamente úteis no desenvolvimento de ontologias. Ainda segundo o mesmo autor, esses princípios são critérios objetivos para orientar e avaliar projetos de ontologias (GRUBER, 1995):

Clareza: A ontologia deve efetivamente comunicar o significado pretendido dos termos definidos. As definições devem ser formalmente definidas, assim independentes de contextos sociais ou computacionais. As definições devem ser objetivas e documentadas em linguagem natural. É preferível se ter uma definição completa a uma definição parcial.

Coerência: Uma ontologia deve ser coerente, isto é, ela deve ratificar inferências que sejam consistentes com as definições. Se uma sentença que pode ser inferida de um axioma contrariar uma definição ou um exemplo definido informalmente, ou seja, em linguagem natural, então a ontologia é incoerente.

Extensibilidade: Uma ontologia deve ser capaz de definir novos termos para usos especiais baseados em um vocabulário já existente, de um modo que não requeira a revisão de definições existentes.

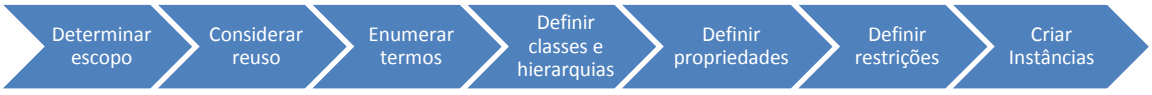
Codificação mínima: A conceitualização deve ser especificada em nível de conhecimento sem dependência de uma codificação ou símbolos particulares.

Mínimo compromisso ontológico: uma ontologia deve exigir o mínimo compromisso ontológico, suficiente para atender o compartilhamento do conhecimento, isto é, o compromisso ontológico deve ser baseado no uso consistente do vocábulo, assim deve ser minimizado, a fim de definir apenas os termos que são essenciais para a comunicação do conhecimento consistente com a teoria do domínio observado.

Para a construção da ontologia que será apresentada neste trabalho, foi utilizada a metodologia 101, desenvolvida por Noy e McGuinness (2001), que tem o fito de ser interativa, ou seja, a ontologia é revisada e refinada durante todo o processo. "Após definirmos uma versão inicial da ontologia, podemos avaliá-la e depurá-la usando-a em aplicações ou métodos de resolução de problemas ou discutir com especialistas na área, ou ambos. [...]" (Op. cit., p. 4, tradução nossa). Essa abordagem metodológica destaca um possível roteiro de passos a serem observados na construção de uma ontologia.

A metodologia 101 é composta por sete passos, conforme ilustrados na Figura 9 e descritos abaixo (NOY, 2005):

Figura 9. Sequência de passos da metodologia 101.



Fonte: NOY, 2005, p. 6, adaptada

Passo 1 – Determinar o escopo: Consiste em delimitar o espaço de abrangência dos termos do domínio observado. Para iniciar este processo, são sugeridas algumas perguntas básicas que poderão ser aplicadas:

- Qual é o domínio que a ontologia cobrirá?
- Qual será o uso da ontologia?

- Para que tipos de perguntas a ontologia deverá fornecer respostas?
- Quem vai utilizar e manter a ontologia?

As respostas para essas perguntas podem mudar durante o processo de criação da ontologia, mas a qualquer momento podem ajudar a delimitar o escopo do modelo. Uma maneira prática de se determinar o escopo de uma ontologia é identificar uma série de perguntas que a base de conhecimento de uma ontologia deve ser capaz de responder.

Passo 2 – Considerar reuso: Este passo tem como objetivo considerar as estruturas ontológicas e suas bases de conhecimento e analisar seu uso para utilizá-la ou aperfeiçoá-la para o domínio específico. Outro sentido para este passo é observar se o novo sistema interage com outras aplicações que já têm ontologias, assim o reuso de ontologias pode ser um requisito. Existem repositórios públicos de ontologias reusáveis como DAML[23], Ontolingua[24] e OntoLP[25].

Passo 3 – Enumerar termos: Neste passo, os termos mais importantes do domínio observado devem ser enumerados. Para Silva (2006, p. 36), "Este passo constitui um ponto oneroso para o desenvolvimento da ontologia, por requerer aquisição de uma quantidade significativa de conhecimento do domínio abordado [...]". Inicialmente, é importante listar os termos sem a preocupação com a sobreposição entre os conceitos ou qualquer propriedade.

Passo 4 – Definir classes e hierarquia de classes: Aqui, os termos selecionados são inseridos em uma hierarquia, podendo seguir uma das três abordagens:

- ***Top-down***: Inicialmente, os conceitos mais genéricos do domínio são definidos e categorizados e posteriormente é feita a especializações desses conceitos.
- ***Botton-up***: As classes mais específicas são definidas e posteriormente agrupadas em classes mais gerais.
- ***Combination***: Esta abordagem é formada pela combinação das duas anteriores. São definidos os conceitos que mais se destacam primeiro e depois esses conceitos são generalizados ou especializados adequadamente.

Os termos gerados no passo 3 serão candidatos a classes e serão organizados em uma taxonomia. De tal modo, toda instância de uma classe mais específica será uma instância da classe mais genérica, ou seja, se a classe *A* é uma superclasse de *B*, então toda instância de

23 http://www.daml.org/ontologies/

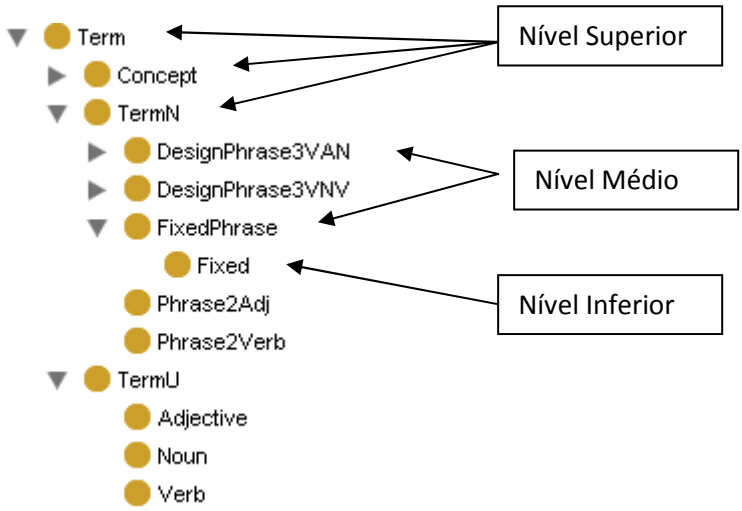
24 http://www.ksl.stanford.edu/software/ontolingua/

25 http://www.inf.pucrs.br/~ontolp/downloads.php

B é também uma instância de *A*. A ontologia apresentada nesta dissertação utilizou da combinação das abordagens *top-down* e *botton-up* na criação da taxonomia hierárquia. Assim, é possível visualizar na Figura 10 que a classe *Term* é o conceito de nível mais alto, e a partir dele as classes de níveis mais baixos foram especificadas. Enquanto que as classes *Adjective*, *Noun* e *Verb* foram definidas e somente depois vinculada ao conceito *TermU*.

Não existe um método melhor que outro, a abordagem a ser utilizada no desenvolvimento de uma ontologia depende da visão pessoal do desenvolvedor diante do domínio. Se o desenvolvedor tem uma visão geral da taxonomia, a abordagem *top-down* é apropriada, mas se o desenvolvedor tende a visualizar o domínio através de exemplos mais específico, pode funcionar melhor. Para Rosch (1978 apud NOY & MCGUINNESS, 2001) a combinação dessas duas abordagens é frequentemente usada por muitos desenvolvedores, visto que os conceitos "*middle*" tendem a ser mais descritivos.

Figura 10. Diferentes níveis de taxonomia



Fonte: Autor

Passo 5 – Definir as propriedades (slots) das classes: Após a definição das classes e taxonomia, suas características precisam ser atribuídas, visto que uma classe sozinha não apresenta nenhuma informação. As classes normalmente apresentam atributos que representam suas características, ou seja, a estrutura interna da idéia. As propriedades são definidas a partir da listagem de termos resultantes do passo 3, os termos que não foram nomeados como classes provavelmente serão eleitos propriedade de uma ou mais classes. Uma classe herda as propriedades de sua classe superior.

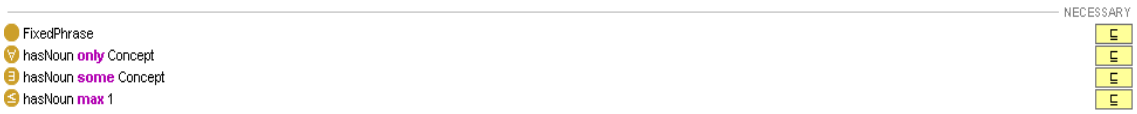
Passo 6 – Definir restrições das propriedades (facetas): As restrições são utilizadas para restringir os indivíduos de uma classe. As propriedades podem ter diferentes

restrições de valores, cardinalidade e tipos de dados que podem ser definidos como numéricos, lógico, texto ou até mesmo como tipo de outra classe. Por exemplo, uma propriedade pode receber como valor somente dados textuais, outra pode receber múltiplos valores e esses valores podem ser instâncias de outra classe. Abaixo, serão descritas algumas restrições implementadas na ferramenta Protégé:

Restrição de cardinalidade: Define a quantidade de valores que uma propriedade pode receber. Essa definição pode ser definida em 'pelo menos', ou 'no máximo' ou 'exatamente' um número possível de valores associados à propriedade. Uma restrição de cardinalidade 'pelo menos' significa que existirá no mínimo um valor associado à propriedade. Para estabelecer um valor máximo relacionado à propriedade, uma restrição de cardinalidade máxima, 'no máximo', é utilizada. Uma restrição de cardinalidade, 'exatamente', especifica o número preciso que será relacionado à propriedade. Na Figura 11 é possível visualizar a implementação da restrição 'no máximo', a qual define o máximo de uma instância para a propriedade *hasNoun*.

Restrições existenciais: Especifica a existência de pelo menos um relacionamento de um indivíduo com outro indivíduo, o qual é membro de uma classe especificada (*filler*). Na Figura 12, é apresentada a janela de criação de restrições da ferramenta Protégé, onde é possível selecionar a propriedade que receberá a restrição (*Create Restriction*), o tipo da restrição (*Restriction*) e o *Filler*. Por exemplo, ainda na Figura 11 é possível notar a restrição *hasNoun some Concept* que descreve todos os indivíduos que têm pelo menos um relacionamento com um indivíduo membro da classe *Concept*, através da propriedade *hasNoun*, ou seja, os indivíduos que tem pelo menos um conceito.

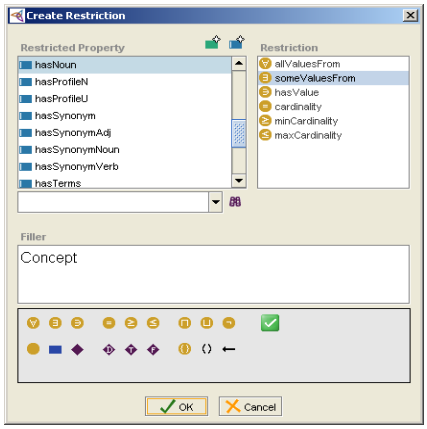
Figura 11. Implementação de restrições – Ferramenta Protégé



Fonte: Autor

Restrições universais: Especifica a existência do relacionamento unicamente com indivíduos membros de uma classe especificada. Na Figura 11, a restrição universal *hasNoun only Concept* descreve os indivíduos cuja os relacionamentos *hasNoun* só podem ocorrer com membros da classe *Concept*. Dessa maneira, as restrições *hasNoun some Concept* e *hasNoun only Concept* restringem o indivíduo a ter pelo menos e somente um relacionamento *hasNoun* com indivíduos membros da classe *Concept*.

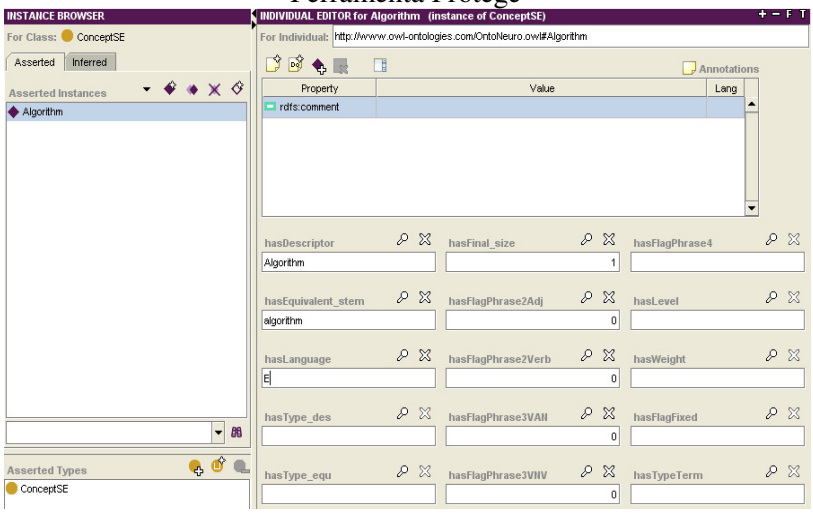
Figura 12. Atribuição do Filler a restrição existencial – Ferramenta Protégé



Fonte: Autor

Passo 7 – Criar instâncias: As instâncias ou indivíduos em ontologias são representações das referências das classes. Neste último passo, é possível criar instâncias individuais das classes na hierarquia. No momento da criação das instâncias, as propriedades (*slots*) devem ser preenchidas. Por exemplo, pode-se criar uma instância individual de *ConceptSE* para representar um específico termo da classe conceito de engenharia de software. A Figura 13 ilustra o preenchimento de algumas propriedades no momento da criação da instância '*Algorithm*'.

Figura 13. Definição da instância Algorithm e algumas propriedades para a classe ConceptSE – Ferramenta Protégé



Fonte: Autor

4 ONTOLOGIA PARA O DOMÍNIO DA NEUROLINGUÍSTICA

Nesta seção, será apresentada uma ontologia que contém informações sobre padrões linguísticos em inglês, compondo as relações hierárquicas entre predicados sensoriais (vide Seção 2.3) e termos da engenharia de software. O propósito geral desta ontologia é servir como repositório para a representação do conhecimento neurolinguístico, constituindo um vocabulário estruturado de predicados sensoriais, conceitos de engenharia de software e uma estrutura de padrões lexicais, capaz de ser empregado em ferramentas de extração e classificação de texto, como o Neurominer.

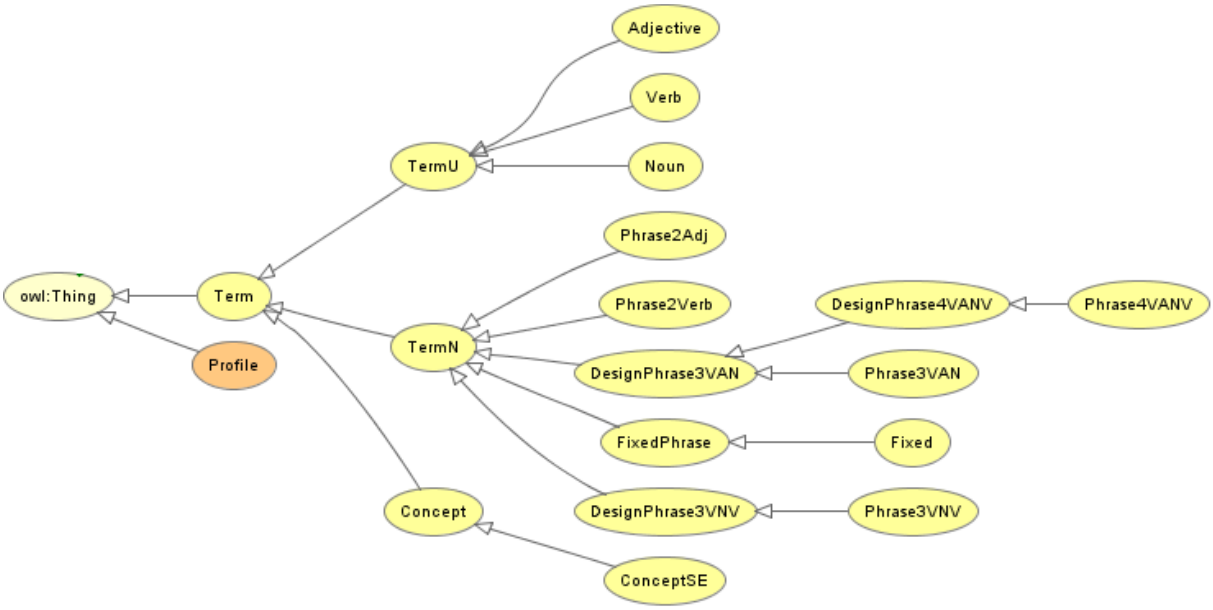
Seu uso não está restrito ao domínio da engenharia de software, podendo ser utilizada em outras áreas do conhecimento. Os conceitos específicos, *ConceptSE*, podem ser substituídos por conceitos de outro domínio, possibilitando que os padrões sirvam de base para uma nova combinação e o reuso do conhecimento armazenado.

Os padrões definidos na ontologia foram criados levando em consideração seus significados léxicos, ao mesmo tempo em que foram estabelecidas relações entre termos gramaticais – verbos, adjetivos, substantivos – e conceito de engenharia de software, constituindo uma hierarquia conceitual, vide Figura 14. Dessa forma, foi especificada, em linguagem formal, a maneira como esses termos podem ser combinados com o propósito de gerar uma base de conhecimento neurolinguístico, capaz de auxiliar na análise de SPR's no domínio da engenharia de software. Além disso, segundo Silva (2006, p. 45), "[...] ontologias favorecem a inferência pela representação explícita do conhecimento e propiciam compreensão mais apurada do domínio abordado."

As classes de padrões de combinação, como por exemplo, *DesignPhrase3VAN* e *DesignPhrase3VNV*, foram definidas e relacionadas levando em consideração as estruturas gramaticais do inglês e são responsáveis pelos padrões que serão utilizados pelo NEUROMINER para combinar predicados sensoriais e termos da engenharia de software, gerando uma base de conhecimento com um poder contextual maior. Os conceitos de engenharia de softwares são representados pela classe *ConceptSE*. Por exemplo, a frase '*I see the brilliant algorithm*' corresponde à combinação da instância '*See Brilliant*' da classe *DesignPhrase3VAN* e o conceito '*Algorithm*'. Essa classe representa um padrão formado por um verbo, um adjetivo e conceitos. O padrão '*See Brilliant*' deverá ser combinado com todos os conceitos existentes na classe *ConceptSE*, resultando frases como: '*I see the brilliant activity diagram*', '*I see the brilliant source code*', '*I see the brilliant server*', etc. No

apêndice A do presente documento estão relacionados os conceitos de engenharia de softwares utilizados dessa combinação.

Figura 14. Hierarquia da ontologia



Fonte: Autor

Existem frases que identificam fortemente um comportamento neurolinguístico, independentemente do padrão de combinação. Como por exemplo, '*I feel good about the problem*', que é associado a um Sistema de Representação Cinestésico. Essas frases são representadas pela classe *FixedPhrase* e, também, são combinadas com *ConpectSE* através da substituição dos substantivos por conceitos de engenharia de software. Utilizando o mesmo exemplo, essa frase fixa resultará outras frases como: '*I feel good about the algorithm*', '*I feel good about the activity diagram*', '*I feel good about the source code*', '*I feel good about the server*', etc.

Além das combinações descritas acima entre padrões, frase fixa e conceitos, é possível combinar cada predicado sensorial com seus sinônimos. Portanto, considerando os termos '*Great*', '*Better*' como sinônimos do adjetivo '*Good*', teremos a partir do exemplo já citado acima:

- **Sinônimo *Great*:** *'I feel great about the algorithm', 'I feel great about the activity diagram', 'I feel great about the source code, 'I feel great about the server'.*

- **Sinônimo *Better***: *'I feel better about the algorithm', 'I feel better about the activity diagram', 'I feel better about the source code, 'I feel better about the server'*.

As classes *Verb, Noun* e *Adjective* representam, respectivamente, verbos, adjetivos e substantivos da língua inglesa. Enquanto que a classe *Profile* representa os SRP's Visual, Auditivo e Cinestésicos. As instâncias das classes *Verb, Noun* e *Adjective* estão associadas por restrições existênciais, através das propriedades *hasProfileU* e *hasProfileN,* à classe *Profile* e são identificadas como modificadores dos termos, enquanto que as instâncias da classe *ConceptSE* são identificadas como termo centrais (*head*), conforme Tabela 4. Desta maneira, frases ou termos N-gramas formados pela combinação de instâncias modificadoras e conceitos são mais contextualizadas, recebendo, assim, um bônus multiplicador ao seu peso (Colaço Jr. et al. 2010).

Tabela 4. Relação entre as instâncias.

Profile	Termo	*Tag*
Visual	*brilliant*	Mod
Auditive	*dissonant*	Mod
Kinesthetic	*concrete*	Mod
Concepts	*algorithm*	Head

Fonte: Colaço Jr. et al. 2010, p 2, adaptada

A metodologia empregada na construção da ontologia foi a Metodologia 101, detalhada na Seção 3.3.4, e a ferramenta de edição utilizada foi o Protégé 3.4, acompanhada do pluging OWLViz que consiste em um gerador de gráficos com instâncias, herança e outros tipos de relacionamentos.

Como mecanismo de inferência foi utilizado o RacerPro 1.9[26], baseado em lógica descritiva, para verificar se as declarações e as definições da ontologia são mutuamente consistentes entre si. É possível ainda, inferir automaticamente a hierarquia de classes da ontologia. Outro serviço padrão oferecido pelo mecanismo de inferência é o de *consistency checking* (*verificação da consistência*). Baseado na descrição (condições) de uma classe, o mecanismo de inferência pode verificar se é possível, ou não, que uma classe possua instâncias (DRUMMOND et al., 2009).

26 Disponível em: http://www.racer-systems.com

4.1 Aplicação da metodologia 101

Passo 1 – Determinar o escopo: Algumas dificuldades foram encontradas no processo de construção da ontologia. A primeira delas foi a definição do escopo, proporcionado pela falta de padronização e hierarquia do antigo dicionário que compunha o Neurominer. Assim buscou-se responder as perguntas que seguem a fim de determinar o escopo e domínio da ontologia:

- *Qual é o domínio que a ontologia cobrirá?*

  A ontologia abordará conceitos relacionados a padrões linguísticos em inglês, a fim de relacionar hierarquicamente predicados sensoriais e termos de um domínio específico.

- *Qual será o uso da ontologia?*

  A ontologia será utilizada como ferramenta de apoio à construção de um dicionário neurolinguístico contextualizado e também servirá de base de conhecimento neurolinguístico.

- *Para que tipos de perguntas a ontologia deverá fornecer respostas?*

  Quais as relações entre predicados sensoriais e conceitos de um domínio específico?
  Quais predicados sensoriais devem ser combinados com conceitos de um domínio específico?
  Quais as combinações possíveis entre as classes gramaticais e conceitos de um domínio específico?
  Quais os pesos dos termos gerados a partir dos padrões pré-definidos na ontologia?

- *Quem vai utilizar e manter a ontologia?*

  A ontologia será utilizada pelo módulo gerenciador do dicionário da ferramenta NEUROMINER com a finalidade de implementar melhorias em seu dicionário. A manutenção ocorrerá à medida que surgirem novos padrões e/ou a necessidade de inserção de novas classes gramaticais. Apesar dos aspectos positivos das ontologias, algumas limitações podem estar presentes em função, por exemplo, das características do domínio. Neste caso, poderá haver

mudanças significativas no comportamento linguístico que propiciam criações e combinações de novos termos, principalmente na área de engenharia de software, por ser uma área muito dinâmica.

Passo 2 – Considerar reuso: Foi reutilizado os conceitos de engenharia de softwares da ontologia proposta por Wongthongtham et al. (2009), que possibilita o compartilhamento dos conceitos de projetos e processo de software. Para a ontologia proposta, devido à especificidade, por ser uma ontologia de aplicação, não foi encontrada referências nos repositórios ontológicos já referenciados que atendessem aos propósitos deste trabalho e que pudessem ser reutilizadas. A ontologia desenvolvida estará sendo disponibilizada no repositório ontológico OntoLP.

Passo 3 – Enumerar termos: Os termos foram inicialmente enumerados buscando refletir o cenário real léxico do contexto linguístico estudado.

Passo 4 – Definir classes e hierarquia de classes: Para a definição das classes e suas hierarquias, conforme já mencionado, as abordagens *Top-down* e *Botton-up* foram utilizadas de forma concomitante. A lista resultante do passo anterior serviu de inspiração para esse passo. Cada classe foi avaliada individualmente a fim de verificar a existência de especialização ou generalização, ou seja, superclasse e subclasse, conforme é descrito na Tabela 5.

Tabela 5. Classes da ontologia proposta

Classes	Descrição
Term	Classe principal da ontologia
Profile	Representa as categorias dos SRP's dos desenvolvedores (Visual, Auditivo e Cinestésico)
Concept	Representa os conceitos que serão combinados com os predicados sensoriais.
ConceptSE	Subclasse de Concept e representa os conceitos de Engenharia de Software.
TermU	Essa classe reúne os termos U-grama, ou seja, os termos formados apenas por uma palavra. Ex: *brilliant, clear, feel*
Verb	Os verbos são representados por essa classe U-grama. Subclasse de TermU.
Adjective	Essa classe representa os adjetivos e é uma subclasse de TermU.
Noun	Classe que reúne os substantivos U-grama e é uma subclasse de TermU.
TermN	Essa classe representa os termos N-grama, ou seja, os termos formados por mais de uma palavra. Ex: *brilliant server, clear code, feel the problem.*As classes abaixo são subclasse de TermN.
FixedPhrase	Classe que concebe as frases fixas. Essas frases identificam fortemente um comportamento neurolinguístico. São subclasses de TermN.

Fixed	Reúne os termos N-gramas resultantes da combinação das frases fixas com os conceitos de engenharia de software. A combinação acontece através da substituição dos substantivos pelos conceitos.
DesignPhrase3VNV	Essa classe recebe os padrões pré-definidos de combinação de verbos com os conceitos do domínio. Os padrões dessa classe são formados por um verbo, um substantivo (que será futuramente substituído por todos os conceitos existentes) e outro verbo.
Phrase3VNV	Reúne os termos resultantes da combinação entre os padrões da classe DesignPhrase3VNV e os conceitos de engenharia de software.
DesignPhrase3VAN	Representa os padrões pré-definidos que são formados pela combinação entre um verbo, um adjetivo e um substantivo (que será futuramente substituído por todos os conceitos existentes).
DesignPhrase4VANV	Representa os padrões pré-definidos que são formados pela combinação entre um verbo, um adjetivo, um substantivo (que será futuramente substituído por todos os conceitos existentes) e outro verbo.
Phrase4VANV	Reúne os termos N-gramas resultantes da combinação dos padrões das instâncias da classe DesignPhrase4VANV com os conceitos de engenharia de software. A combinação acontece através da substituição dos substantivos pelos conceitos.
Phrase3VAN	Classe que recebe os termos N-gramas resultantes da combinação dos padrões das instâncias da classe DesignPhrase3VAN com os conceitos de engenharia de software. A combinação acontece através da substituição dos substantivos pelos conceitos.
Phrase2Verb	Classe que representa as instâncias N-gramas resultantes da combinação de todos os verbos com todos os conceitos de engenharia de software.
Phrase2Adj	Classe que representa as instâncias N-gramas resultantes da combinação de todos os adjetivos com todos os conceitos de engenharia de software.

Fonte: Autor

Passo 5 – Definir as propriedades (*slots*) das classes: Propriedades foram definidas com o propósito de estabelecer regras entre os relacionamentos dos termos léxicos e representar as características individuais de cada termo, na Tabela 6 são ilustradas essas propriedades. Frases formadas por conceitos da engenharia de software, caracterizadas ou adjetivadas por termos classificados como sensoriais na ontologia, são consideradas como perfil do predicado sensorial caracterizador e são pontuadas com um peso mais alto.

Passo 6 – Definir restrições das propriedades (facetas): Foram criadas propriedades para descrever os relacionamentos e as cardinalidades entre as classes. Como por exemplo a restrição de cardinalidade entre a classe de padrão *DesignPhrase3VAN* e a classe *Adjective*.

Tabela 6. Propriedades da ontologia proposta

Propriedades	Descrição
hasSynonym	Relacionam uma instância com seus sinônimos.
hasTerms	Propriedade de agrupamento.
hasAdjective	Relaciona um adjetivo a padrões.
hasNoun	Relaciona um substantivo a padrões.
hasVerb	Relaciona um verbo a padrões.
hasVerb2	Relaciona um segundo verbo a padrões.
hasProfileU	Relaciona um profile a um termo U-grama.
hasProfileN	Relaciona um profile a um termo N-grama.
hasDescriptor	Representam o texto descritor do termo.
hasEquivalent_stem	Representa o texto do termo radicalizado cada termo
hasFinal_size	Representa o tamanho final do termo após da radicalização
hasFlagConcept	Indica se o termo já foi combinado com os conceitos.
hasFlagFixed	Indica se o conceito já foi combinado com as frases fixas.
hasFlagPhrase2Adj	Indica se o conceito já foi combinado com as instâncias Adj + Noun.
hasFlagPhrase2Verb	Indica se o conceito já foi combinado com as instâncias Verb + Noun
hasFlagPhrase3VAN	Indica se o Conceito já foi combinado com o padrão DesignPhrase3VAN
hasFlagPhrase3VNV	Indica se o Conceito já foi combinado com o padrão DesignPhrase3VAV
hasFlagPhrase4	Indica se o conceito já foi combinado com as instâncias formadas através do padrão DesignPhrase4VANV.
hasLanguage	Indica a idioma do termo.
hasLevel	Indica se o termo é principal ou um sinônimo.
hasType_des	Indica o tipo do termo antes da radicalização - N - N-grama e U - U-grama
hasType_equ	Indica o tipo do termo depois da radicalização - N - N-grama e U - U-grama
hasWeight	Representa o peso do termo.

Fonte: Autor

Passo 7 – Criar instâncias: A base de conhecimento resultante das instâncias das classes da ontologia é formada por termos U-grama e N-grama. As instâncias das classes criadas a partir das sub-classes de *TermU* fazem referências a termos U-grama. Cada instância pode ser vinculada a um ou mais sinônimos. Os termos N-grama são instâncias das sub-classes de *TermN*. As instâncias das classes dos termos U-Grama e da classe *FixedPhrase* foram extraídas do antigo dicionário do Neurominer e através da experiência de outros

trabalhos no campo da psicologia (COLAÇO, 2011, KAHN et al., 2007, TOSEY & MATHISON, 2009). Para as instâncias dos conceitos de engenharia de software, *ConceptSE*, foram utilizadas as definições das classes da ontologia proposta por Wongthongtham et al. (2009), que se baseia em vários domínios de programação, incluindo linguagens de programação, algoritmos, estrutura de dados e padrões de projeto. Cada instância das classes derivadas de *TermU* e *TermN* é associada através de propriedades de objeto à classe *Profile*, seguindo as regras abaixo:

- Instâncias das subclasses de *TermU* e *FixedPhrase* são vinculadas a apenas um *profile*. Os sinônimos desses termos recebem o mesmo *profile* do termo principal;
- Se a instância de um padrão for similar[27] a alguma instância de *FixedPhrase*, ela receberá o mesmo *profile* da frase, como por exemplo:

  - A frase fixa '*I feel good about the problem*' é associada ao *profile* Cinestésico. No momento da criação, por exemplo, da instância '*feel good*' é verificado que esse padrão é similar à frase fixa descrita acima, portanto o *profile* associado ao padrão será o mesmo da frase, ou seja, Cinestésico.
- Se a instância do padrão não for similar a nenhuma frase fixa, é testado se os termos que compõem o padrão pertencem ao mesmo *profile*. Em caso positivo, um único *profile* é associado ao padrão. Caso cada termo do padrão esteja associado a *profiles* diferentes, o padrão receberá os *profiles* correspondentes a cada termo que compõem o padrão, como por exemplo:

  - Os termos '*See*', '*Brilliant*' e '*Imagine*' são associados ao *profile* Visual, enquanto que o termo '*Pleasant*' é associado ao *profile* Cinestésico. Portanto, no momento da criação da instância dos padrões '*See Brilliant*' e '*Imagine Pleasant*', não é verificada a similaridade dos padrões com nenhuma frase fixa, deste modo, o primeiro padrão será associado ao *profile* Visual, pois os dois termos que compõem esse padrão são associados ao mesmo *profile*, ou seja, Visual. Para o segundo padrão, serão associados os *profiles* Visual e Cinestésico, pois os termos que compõem esse padrão são associados a diferentes *profiles*.

27 Utilizou-se a distância de Levenstein para verificar a similaridade entre os termos. Disponível em http://lucene.apache.org

Para determinar os pesos de cada instância da ontologia, foi levado em consideração a relação de termos sensoriais e os termos dos domínios específicos. Dessa maneira, termos N-Grama contextualizados têm um peso mais forte que predicado, termos U-Grama, frases fixas e padrões, inovação proposta por esta ontologia. Cada instância recebe um peso distinto, conforme segue abaixo:

- Instâncias que representam os predicados sensoriais: Peso = 1,0;
- Instâncias das frases fixas: Peso = 2,0;
- Instâncias dos padrões: Peso = 2,0;
- Instâncias resultantes da combinação das frases fixas com conceito: Peso = 3,0;
- Instâncias resultantes da combinação de predicado e conceito: Peso = 1,5;
- Instâncias resultantes da combinação dos padrões e conceito: Peso = 3,0.

Na identificação e sumarização dos *scores*, os pesos dos termos identificados no texto são divididos entre os *profiles* associados aos mesmos. Como por exemplo, no padrão '*See Brilliant'* serão sumarizados 2,0 (dois) pontos ao cálculo do *score* Visual, pois o padrão é formado apenas por termo de profiles visuais. Para o padrão '*Imagine Pleasant*', será sumarizado 1,0 ponto ao cálculo do *scrore* Visual e 1,0 ponto ao *scrore* Cinestésico, pois o padrão é formado por termos associados a *profiles* Visual e Cinestésico.

5 EXPERIMENTOS E RESULTADOS DA APLICAÇÃO DA ONTOLOGIA NO DICIONÁRIO NEUROLINGUÍSTICO

Com o objetivo de avaliar a influência da ontologia no dicionário neurolinguístico e na ferramenta, um experimento, dividido em duas etapas, foi executado na amostra de dados: lista de e-mails do Projeto *Apache* no período de 1996 a 2005 (35.483 mensagens). Na primeira (COLAÇO et al, 2010), utilizou-se um dicionário simples para identificar SRP's dos quatro desenvolvedores que mais contribuíram no Projeto (*Top-committers*), bem como um *cluster* contendo as mensagens dos demais colaboradores do projeto (25.121 mensagens). Esses são os mesmos desenvolvedores estudados por Rigby e Hassan (2007). Na segunda etapa, foi utilizado o novo dicionário gerado a partir da ontologia proposta na identificação dos SRP's dos mesmos desenvolvedores. Por fim, foram utilizadas técnicas estatísticas para validar diferenças significativas entre os resultados.

O dicionário utilizado no primeiro experimento possui cerca de 3.618 termos, entre predicados sensoriais, termos de engenharia de software e combinações. A combinação entre os modificadores e elementos centrais foi feita de forma manual e imprecisa, gerando um dicionário em que nem todos os predicados sensoriais foram combinados com os termos de Engenharia de Software disponíveis e os sinônimos dos modificadores não foram considerados. Deste modo, limitando o poder contextual do dicionário.

Para o dicionário gerado a partir da ontologia, os termos foram combinados conforme descrito na Seção 4.1, resultando 255.932 termos, entre predicados sensoriais, conceitos de Engenharia de Software e padrões. Uma inovação da ontologia é a proposta da combinação automática de todos os predicados sensoriais com esses conceitos, utilizando regras definidas e padronizadas para a combinação, alocação do *profile* e cálculo dos pesos de cada termo.

Como foi descrito na Seção 3.2.1, a tabela de fatos (*Term_Fact*), em um nível de granularidade mensal, armazena os *scores* calculados para os termos encontrados. Para o primeiro experimento, o total de registro armazenados nessa tabela foi de 15.260. Enquanto que o segundo armazenou 401.809 registros. Portanto, pode-se discorrer que houve maior identificação dos termos por granularidade mensal.

Para cada desenvolvedor e o *cluster* são calculados a frequência e *scores* mensais e acumulados, conforme seção 3.2.2. O *Cluster* foi minerado por três anos (36 meses). Os

Top-committers foram minerados em todo o período (dez anos), mas somente os meses em que o desenvolvedor enviou pelo menos um e-mail para a lista foram inseridos na base de cálculo.

Na Tabela 7, são sumarizadas as médias mensais dos dois experimentos. A coluna *Período* mostra os anos de maior contribuição de cada *Top Commiter* no Projeto. Para cada Sistema de Representação, é exibida a média mensal de cada desenvolvedor. A coluna ANOVA *p-value* evidencia a diferença estatística das médias em cada experimento. A coluna Classificação exibe a categorização, segundo as médias mensais, da preferência dos Sistemas Representacionais, onde: V – Visual; A – Auditivo; C – Cinestésico.

Tabela 7. Sumarização das médias dos experimentos I e II

EXPERIMENTO I						
Top Commiter	Período	Médias Mensais			ANOVA *p-value*	Classificação
		Visual	Auditivo	Cinestésico		
A	1997 – 1998	2,7274	1,4918	2,9774	,000	CVA
B	1999 – 2000	2,6680	0,8785	2,2514	,000	VCA
C	2001 – 2002	2,3582	2,0162	2,6298	,000	CVA
D	2003 – 2004	0,6063	0,6581	0,4682	,000	AVC
Cluster	-	7,7567	7,3756	8,2279	,000	CVA
EXPERIMENTO II						
Top Commiter	Período)	Médias Mensais			ANOVA *p-value*	Classificação
		Visual	Auditivo	Cinestésico		
A	1997 – 1998	0,7042	0,2748	1,6508	,000	CVA
B	1999 – 2000	0,7738	0,2082	1,5610	,000	CVA
C	2001 – 2002	0,5958	0,2224	1,7838	,000	CVA
D	2003 – 2004	0,4760	0,1575	1,8916	,000	CVA
Cluster	-	0,6450	0,2445	1,7847	,000	CVA

Fonte: Autor

Analisando os *scores* e a classificação dos desenvolvedores, observa-se que os dois experimentos coincidiram na classificação da maioria dos desenvolvedores, tendo 10 classificações semelhantes das 15 possíveis, conforme é ilustrado na Tabela 8.

Tabela 8. Semelhanças na classificação entre os experimentos

Top Commiter	***Classificação Experimento I***	***Classificação Experimento II***	***Classificações Semelhantes***
A	CVA	CVA	CVA
B	VCA	CVA	-
C	CVA	CVA	CVA
D	AVC	CVA	V
Cluster	CVA	CVA	CVA

Fonte: Autor

Para uma análise estatística das médias entre os Experimentos I e II, inicialmente foi verificado que as médias não apresentaram distribuição normal entre os experimentos,

como se pode notar nos testes de normalidade *Kolmogorov-Smirnov* e *Shapiro-Wilk* na Tabela 9. Em um segundo momento, foi executado o teste não paramétrico de *KrusKal-Wallis,* deste modo, observa-se o *p-value* igual a 0,000 para todos os SRP's, evidenciando a existência de diferença significativa de todas as médias para todos os perfis e desenvolvedores entre os experimentos, conforme Tabela 10.

Tabela 9. Teste de Normalidade (*Kolmogorov-Smirnov* e *Shapiro-Wilk*)

	EXPERIMENTO	*Kolmogorov-Smirnov*[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
SCORE	1	,259	637	,000	,768	637	,000
	2	,208	561	,000	,831	561	,000

a. Lilliefors Significance Correction

Fonte: Autor

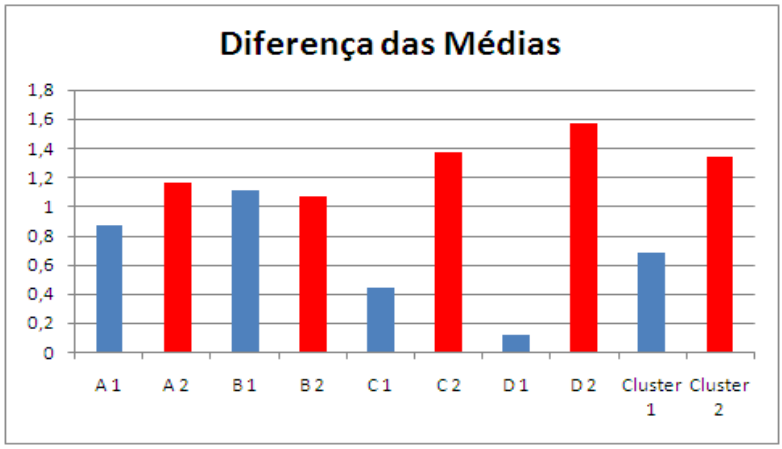
Analisando as diferenças entre as médias de cada experimento ilustradas na Figura 15, observa-se que as distâncias entre as médias dos Sistemas predominantes e as médias dos outros Sistemas são geralmente maiores no experimento II. Esse cenário, possivelmente, nos fornece indícios mais fortes que nos reportam a uma análise mais confiante acerca da classificação do Sistema de Representação Preferencial de cada desenvolvedor. Essa análise confirma a afinidade do domínio analisado, lista de discussão de desenvolvimento de apache, por uma preferência cinestésica.

Considerando a classificação geral do segundo experimento é evidenciado que a maioria dos desenvolvedores possui um Sistema Preferencial cinestésico, certificando que os colaboradores da lista atuam de forma prática no projeto e demonstram serem experientes e menos dependentes de artefatos visuais e auditivos, característica de um Projeto *Open Source*.

Outro aspecto importante é a análise dos termos que mais pontuaram por cada desenvolvedor. Para isto, pode ser realizado um *drill-down*, em conjunto com um *ranking*, os quais dão acesso aos 10 termos N-grama que alcançaram os maiores *scores*. Para essa análise, foi escolhido o desenvolvedor B que representa o programador com menos semelhança na classificação entre os experimentos. Os resultados estão ilustrados nas Figuras 16 e 17, que

representam o *ranking* dos termos N-gramas mais mencionados pelo desenvolvedor B no primeiro e segundo experimento, respectivamente.

Figura 15. Diferença das médias entre os experimentos



Fonte: Autor

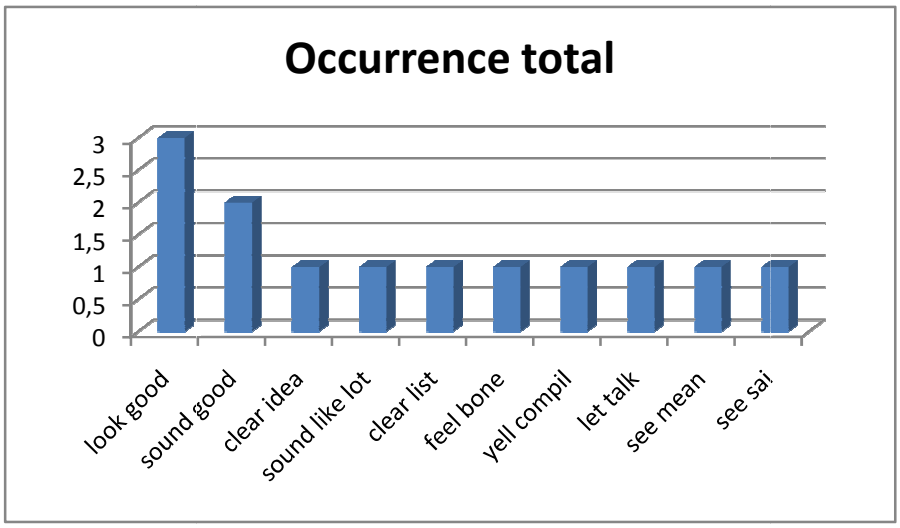
Tabela 10. Teste de KrusKal-Wallis

Experimento	PRS		SCORE
1	Visual	Chi-Square	121,091
		df	3
		Asymp. Sig.	,000
	Auditivo	Chi-Square	92,571
		df	3
		Asymp. Sig.	,000
	Cinestésico	Chi-Square	125,195
		df	3
		Asymp. Sig.	,000
2	Visual	Chi-Square	135,660
		df	3
		Asymp. Sig.	,000
	Auditivo	Chi-Square	120,361
		df	3
		Asymp. Sig.	,000
	Cinestésico	Chi-Square	114,270
		df	3
		Asymp. Sig.	,000

Fonte: Autor

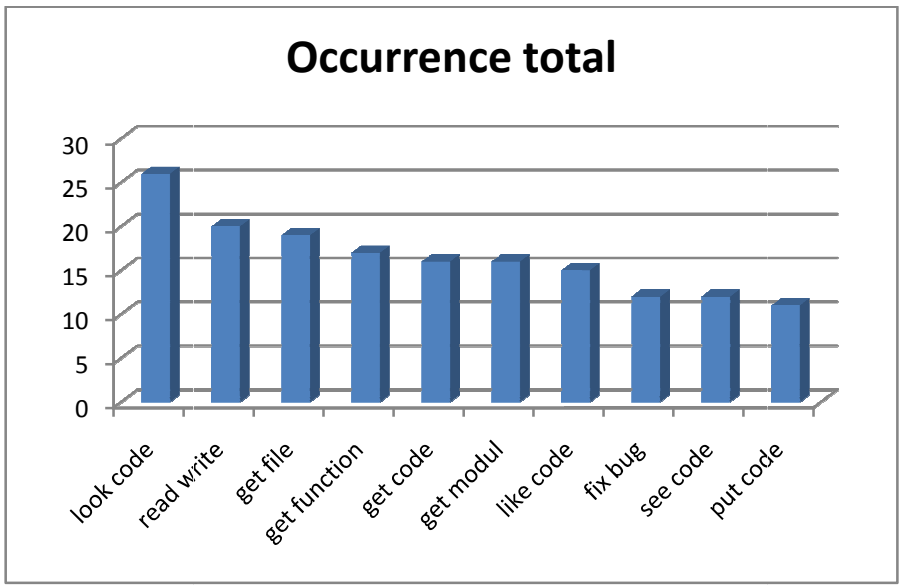
Na Figura 16, é possível comprovar a falta de termos de Engenharia de *Software* entre os 10 termos N-grama mais mencionados pelo Desenvolvedor B. Já na Figura 17, é interessante notar a presença de radicais desses conceitos, tais como *code, file*, *function*, *module e bug* combinados com palavras sensoriais. Indo mais além, também é interessante notar que no segundo experimento, a quantidade de identificação dos termos é bem maior que o primeiro experimento. Portanto, evidenciando uma maior contextualização e eficácia na identificação de temos combinados.

Figura 16. Ranking dos 10 termos mais pontuados pelo Desenvolvedor B no primeiro experimento.



Fonte: Autor

Figura 17. Ranking dos 10 termos mais pontuados pelo Desenvolvedor B no segundo experimento.



Fonte: Autor

Continuando a análise em *drill-down* e partindo dos dados ilustrados na Figura 17, alguns e-mails do Desenvolvedor B foram analisados. No Quadro 6, é possível observar a incidência dos termos (radicalizados) *get code*, *get function*, *look code* e *fix bug*. Esses termos foram pontuados com um poder contextual maior que termos simples U-grama, pois são resultados da combinação de predicado sensorial e conceito de Engenharia de Software. Os mesmos termos não foram identificados no primeiro experimento, isso pode ser um indício do poder do novo dicionário, enfatizando as possíveis melhorias contextuais trazidas pelos padrões linguísticos e de combinação modelados na ontologia.

Quadro 6. E-mails do Desenvolvedor ‘B’

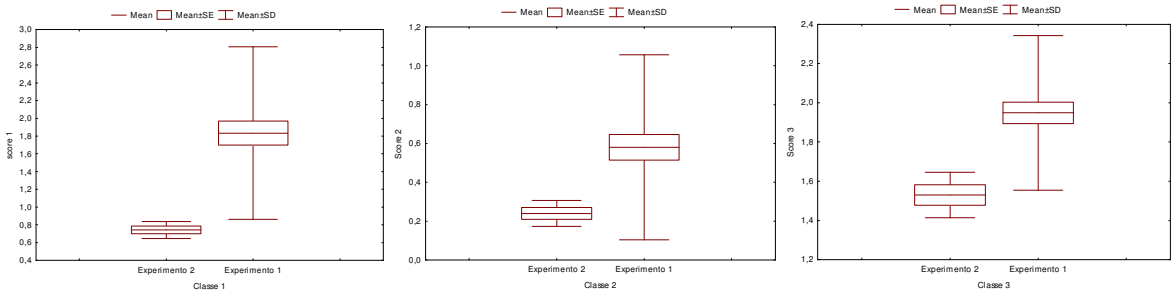
E-mail enviado em *Mon, 11 Oct 1999*
To: new-httpd@apache.org **Subject: Re: Non-threadsafe functions used in threads? (1.3, 2.0)** **I know how to fix this, and I am about to fix them all. With any luck, APR will be completely thread-safe by morning. Please don't ask me to explain tonight. I have been thinking about this since six o'clock, and I won't be able to do anything until I get it coded. There will be something committed to at least begin solving this problem ASAP.**
E-mail enviado em *Mon, 11 Oct 1999*
To: new-httpd@apache.org **Subject: Re: Compiling multiple MPM at the same time.** **[…]** **Prefork can go away, because with a simple compile switch, the pthread MPM functions JUST LIKE the prefork MPM. This doesn't require a userland thread library, it doesn't require ANY thread library. You just compile it the right way, and you don't ever try to create threads. We have the apache-apr tree working as both hybrid and pre-forking servers, and I believe Manoj did the work to get the same function out of mpmt_pthread.**
E-mail enviado em *Tue, 2 Mar 1999*
To: new-httpd@apache.org **Subject: Re: cvs commit: apache-apr/pthreads/src/main http_protocol.c** **[…]** **It has worked in all of my tests, and a very quick look at the code seemed to be okay. I will look closer tonight. I honestly never expected this code to stay here forever. I would rather either use NSPR's read, which has a timeout value, or write an APR function that has a timeout. Both options require that multiple functions be changed, and I don't want to lock us in until a decision has been made. I also wanted a quick solution that would get us moving again, and I had every intention of coming back to it. Hopefully, I'll get to look at it again tonight or tomorrow.** **[…]**
E-mail enviado em *Sun, 13 Feb 2000*
To: new-httpd@apache.org **Subject: Re: What is supposed to happen inside APR when a socket closes** **There is a bug in the ap_send/recv code. Nobody has had a chance recently to fix this bug, or even really investigate what is going on. When an error is return form any system call, APR should return the errno. I will be changing ap_send/recv very soon, because setting the length to -1 is pretty much always wrong. If no byutes are written, then the length should be 0, that's what that variable means.** **[…]**

Fonte: Autor

Ainda com a intenção de analisar os dados provenientes dos resultados dos dois experimentos, as médias mensais de cada desenvolvedor foram ilustradas em um gráfico *Boxplot*. Para essa análise, foram consideradas apenas as médias do desenvolvedor B, vide Figura 18. Através desses gráficos, é possível notar que as médias mensais, no segundo experimento, são mais homogêneas que no primeiro. Deste modo, é possível inferir que a

identificação e o cálculo dos *scores* se deram de maneira mais freqüente no segundo experimento e que os desenvolvedores obedecem a uma normalidade na escrita e em seus SRP's em um período de tempo no contexto de Engenharia de Software.

Figura 18. Gráfico do Boxplot do Desenvolvedor B



Fonte: Autor

6 CONCLUSÕES E TRABALHOS FUTUROS

Esta dissertação abordou questões relacionadas à eficiência e a eficácia de uma ferramenta baseada em mineração de texto e psicometria, objetivando classificar Sistemas de Representação Preferencial de desenvolvedores de software em Projetos *Open Source*. Diante disso, foi proposta uma ontologia com o fito de contribuir para a melhoria do poder contextual do NEUROMINER.

Em resposta à primeira questão da pesquisa, o NEUROMINER representa uma primeira tentativa de usar psicometria neurolinguística para entender desenvolvedores OSS através de lista de e-mails. Descobrir o canal representacional preferencial de um programador pode ajudar a otimizar sua alocação no projeto e melhorar o processo de compreensão de software, estimulando vários canais. Conforme mostra Seções 2.2 e 3.2. Vale ressaltar que o objetivo do NEUROMINER é identificar o sistema de representação mais usado por um programador dentro de um contexto e um indivíduo pode variar o SRP, dependendo do momento e do contexto.

Este trabalho contribuiu de forma direta para a implementação dessa ferramenta, especialmente no desenvolvimento do ETL textual, especificamente para e-mails, e na modelagem de uma ontologia com o propósito de apoiar a criação do novo dicionário neurolinguístico, ver Seção 3.2.2. Como inovação, o dicionário também é composto por conceitos fortes da Engenharia de Software, os quais podem ser caracterizados ou adjetivados por termos sensoriais, formando frases contextuais mais pontuadas no cálculo de scores. Por exemplo, é muito mais proeminente a presença dos termos compostos look code e put code a simplesmente look ou put. Esses aspectos foram detalhados na Seção 4.

Através da aplicação de técnicas de ontologia, foi possível combinar de maneira automática predicados sensoriais, seus sinônimos e termos de Engenharia de Software. Esse foi um ponto chave para obter um dicionário mais contextualizado, aumentando assim o poder contextual da análise neurolinguística em artefatos de software da ferramenta. Desta maneira, em resposta a questão 2 da pesquisa, observa-se a viabilidade no uso de uma ontologia para psicometria utilizada no NEUROMINER. Além disso, a ontologia proporcionou mais flexibilidade à ferramenta para atuar em diferentes áreas de domínios específicos, elevando expressivamente o potencial de mineração e análise da ferramenta. Bastando trocar os conceitos específicos do negócio e recombinar o dicionário neurolinguístico.

A inserção de um predicado sensorial e/ou de um termo de Engenharia de Software não é visto como problema para a estrutura ontológica modelada. Após a inclusão de um novo termo na ontologia, novos padrões de combinação podem ser criados ou simplesmente combinados de forma automática, gerando novas entradas para o dicionário neurolinguístico. Portanto, a inclusão de novos termos é considerada uma tarefa trivial, questão 3.2.

A aplicação de técnicas de mineração de texto presentes no NEUROMINER e a ontologia proposta neste trabalho levaram a resultados promissores, visto que os resultados dispostos na Seção 5 mostram, dentre outros aspectos, consideráveis melhorias no poder contextual do novo dicionário, propondo uma solução eficiente e plausível. Nessa Seção, foi ilustrada a diferença em algumas estratégias preferênciais entre os experimentos I e II, porém é evidênciado as classificações do segundo experimento através da distância entre os scores de cada sistema.

Esses resultados resondem as questões de pesquisa 3.1, 3.3 e 3.4 e são indícios suficientes para acreditar que o uso de ontologias aliadas a ferramentas textuais capazes de extrair, comparar e inferir pode dispor de informações mais contextualizadas acerca de um domínio específico, resultando em uma capacidade mais apurada de mineração.

Duas contribuições gerais são propostas nesta dissertação. A primeira contribuição diz respeito à construção de uma ontologia que modela padrões linguísticos para a combinação neurolinguística de predicados sensoriais e artefatos de um domínio específico. A segunda contribuição diz respeito à descoberta de conhecimento em listas de discussão de projetos de softwares com o objetivo de dispor informação útil para o apoio à decisão de gerentes de projetos.

A ferramenta de mineração de texto e a ontologia apresentadas neste trabalho representam uma nova abordagem para identificar o canal cognitivo preferencial de um desenvolvedor de software, através da utilização de técnicas já consagradas de mineração de texto, PNL e ontologia, conforme descritos nas Seções 2.3, 3.1 e 3.3. A solução proposta possibilita essa identificação em contextos diferentes pela simples troca ou adição de novos domínios, sem exigir praticamente nenhuma alteração no sistema e sem se preocupar em estabelecer novos padrões linguísticos.

Cumpre ressaltar que a ontologia trouxe ganhos significativos para o NEUROMINER, contudo o procedimento de combinação automática dos termos precisa ser customisado com o objetivo de melhorar o ganho de performance. Questão de pesquisa 3.5.

6.1 Trabalhos Futuros

Considerando o estágio atual e os resultados da pesquisa apresentados, algumas perspectivas de trabalhos futuros são discorridas a seguir:

- Construção de uma interface para o povoamento da ontologia;
- Validar a ferramenta e seu dicionário contextualizado em um novo domínio, testando a inclusão de novos conceitos;
- Aprofundar os estudos no campo de Programação Neurolinguística com o propósito de refinar o dicionário existente na ferramenta;
- Remodelar a ontologia adequando aos padrões da língua portuguesa;
- Testar a ferramenta em uma lista de desenvolvedores de software de idioma português e novos projetos;
- Ampliar a entrada do ETL do NEUROMINER para receber textos oriundos de outras fontes de dados, como por exemplo, documentos textuais de software, código fontes, *chats online* e redes sociais.
- Desenvolver um módulo integrador entre os cálculos dos *scores* e a ferramenta;
- Criar uma ontologia para descrever as tarefas de Engenharia de Software, fazendo *matching* entre tarefas e desenvolvedores.
- Incrementar a ontologia para receber as *stopwords* vinculadas ao domínio estudado;
- Implementar sistema de recomendação para indicar processos a partir do perfil da equipe.

Pode-se afirmar que essas perspectivas precisam ser desenvolvidas para que a ferramenta possa ser utilizada por inúmeros usuários em contextos diferentes.

REFERÊNCIAS

ALMEIDA, M. B., BAX, M. P. *Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção*. Ci. Inf., Brasília, v. 32, n. 3, p. 7-20. 2003

BAILENSON & YEE. *Digital chameleons: automatic assimilation of nonverbal gestures in immersive virtual environments.* Psychological Science. v16. 814-819, 2005.

BALINSKI, R. *Filtragem de Informações no Ambiente do Direto*. UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL. Dissertação de mestrado. 2002

BANDLER, R. & GRINDER, J. *Frogs into Princes: Neuro-linguistic Programming*, Moab, Utah: Real People Press, 1979.

BRAGA, C. G., CRUZ, D., A., L., M. *Contribição da psicometria para a avaliação de respostas psicossociais na enfermagem*. Rev Esc Enferm USP; 40:98-104. 2006

BREITMAN, K. K. and LEITE, J. C. S. P. *Ontology as a Requirements Engineering Product.* Proceedings of the 11th IEEE International Requirements Engineering Conference. 2003

CAREY, J., CHURCHES, R., HUTCHINSON, G., JONES, J. and TOSE, P., *Neuro-linguistic programming and learning: teacher case studies on the impact of NLP in education.* CfBT Education Trust. 2010. Disponível em <http://www.cfbt.com/evidenceforeducation/pdf/NLP%20and%20learning%20full%20report.pdf>. Acesso em Out/2011.

CHUNG, C. K. and PENNEBAKER, J. W. *Reveling dimensions of thinking in open-ended self-descriptions: An automaded meaning extraction method for natural language.* Journal of Research in Personality. 2008

COLAÇO Jr, M. *Identificação e Validação do Perfil Neurolinguístico de Programadores Através da Mineração de Repositórios de Engenharia de Software*. Universidade Federal da Bahia, Instituto de Matemática. 2011.

COLAÇO, Jr. M., MENDONÇA, M.M FARIAS, M. A. F. and HENRIQUE, P. *OSS Developers Context-Specific Preferred Representational System: A Initial Neurolinguistic Text Analysis of the Apache Mailing List*, MSR 2010.

COLAÇO, Jr., M, MENDONÇA, M. G. and RODRIGUES, F. *Mining Software Change History in an Industrial Environment.* XXIII Brazilian Symposium on Software Engineering, Fortaleza, Brazil, 2009.

COLAÇO, Jr., M. *Projetando Sistemas de Apoio à Decisão Baseados em Data Warehouse.* Editora Excel Books. 2004

CORAL, C., RUIZ, F., PIATTINI, M., *Ontologies for Software Engineeringand Software Technology*. Springer. 2006

DENT, K. A. *Cognitive Styles: Essence and Origins: Herman A. Witkin and Donald R. Goodenough*. Journal of the American Academy of Psychoanalysis, 11:635-636, 1983.

DILTS, R., GRINDER, J., BANDLER, R. & DELOZIER, J., *Neuro-Linguistic Programming: Volume 1, the Study of the Structure of Subjective Experience Meta Publications*, California. 1980.

DRUMMOND, N., Jupp, S., Moulton, G., Stevens, R. *A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools.* The University Of Manchester. Edition 1.2. 2009.

EINSPRUCH, E. L. and FORMAN, B. D. *Observations concerning Research Literature on Neuro-Linguistic Programming*. Journal of Counseling Psychology, vol. 32, no. 4, 589-596, 1985.

FARIAS, M. A. F., SILVA, R. X., COLAÇO , M. J. , COSTA , E. B., SANTOS, P. H. *Um ambiente visual para mineração de listas de discussão e compreensão da colaboração entre desenvolvedores Open Source*. CONNEPI. 2011.

FRANÇA, P. C. *Os dicionários onomasiológicos e as ontologias computadorizadas. LinguaMÁTICA* - ISSN: 1647-0818 Núm. 2 - Pág. 93-106. 2009.

GAIZAUSKAS, R., HELPPLE, M., SAGGION, H., GREENWOOD, M.A., and HUMPHREYS, K. *'SUPPLE: a practical parser for natural language engineering applications'.* Proc. 9th Int. Workshop on Parsing Technologies (IWPT2005), Vancouver, 2005.

GEGICK, M., ROTELLA, P., XIE, T. *Identifying Security Bug Reports via Text Mining: An Industrial Case Study*. *Mining Software Repositories* – MSR. 2010.

GIL, A. C. *Métodos e técnicas de pesquisa social*. 5ª. ed. São Paulo: Atlas, 1999.

GILL, A. J. and OBERLANDER, J. *Perception of e-mail personality at zero-acquaintance: Extraversion takes care of itself; Neuroticism is a worry*. 2006.

GOULART, R. R. V., LIMA, V. L. S., XAVIER, C. C. *A systematic review of named entity recognition in biomedical texts. Journal of the Brazilian Computer Society*, 2011.

GROOM, C. J. and PENNEBAKER, J. W. Words. *Journal of Research in Personality Volume 36, Issue 6*, Pages 615-621. 2002.

GRUBER, T. R. *Towards Principles for a Design of Ontologies used for knowledge Sharing*. International Journal of Human and Computer Studies, 43. 1995.

IRELAND, M. E., et al. *Language Style Matching Predicts Relationship Initiation and Stability.* Association for Psychological Science. 2010.

KAHN, J. H., TOBIN, R. M., MASSEY, A. E., & ANDERSON, J. A. *Measuring emotional expression with the Linguistic Inquiry and Word Count*. American Journal of Psychology,

120, 263-286. 2007.

KAWANO, V. J. *Desenvolvimento de uma Ontologia para Gerenciamento de Projetos*. Universidade de Brasília. 2009.

KIMBALL, R., REEVES, L., ROSS, M. and THORNTHWAITE, W. *The Data Warehouse Lifecycle Toolkit*, Wiley, 1998.

KO, A. J., MYERS, B. A., and CHAU, D. H. *A Linguistic Analysis of How People Describe Software Problem.* Human-Computer Interaction Institute, 2006.

KONCHADY, M. *Text Mining Applications Programming*. Thomson. Charles River Media. 2006.

KONCHDY, M. *Text Mining Apllication Programming*. Charles River Media. 2006.

KORZYBSKI, A. *Science and Sanity: An introduction to non-Aristotelian systems and general semantics.* Englewood, New Jersey: The International Non-Aristotelian Library Publishing Company. 1941.

MAGALHÃES, C. C. *MinerJur – Uma ferramenta para mineração de bases de jurisprudência*. 2008. Dissertação (Grau de Mestre em Ciências), Universidade Salvador, Salvador-Bahia.

MANNING, C. D., RAGHAVAN, P., HINRICH, S. *Introduction to Information Retrieval*. Cambridge University Press. 2008.

MATTHEWS, D. B. *Learning Styles Research: Implications for Increasing Students in Teacher Education Programs*. Journal of Instructional Psychology, 18 pp. 228-236, 1991.

MCMASTER, M & GRINDER, J. Precision: *A New Approach to Communication Precision Models*, Bonny Doon, CA. 1980.

MINAYO, M. C. de S. (Org.). *Pesquisa social: teoria, método e criatividade*. 4ª. ed. Petrópolis, RJ: Vozes, 1994.

MONTONI, M. A., ROCHA A. R. C. *Uma Investigação sobre os Fatores Críticos de Sucesso em Iniciativas de Melhoria de Processos de Software*. X Simpósio Brasileiro de Qualidade de Software. 2011.

MOREIRA, J. L., CORDEIRO, K. F., CAMPOS, M. L. M. *DoctorOLAP: Ambiente para análise multifacetada de prontuários médicos*. XXIV Simpósio Brasileiro de Banco de Dados. 2009.

NEWMAN, M. L. et al. *Gender Differences in Language Use: An Analysis of 14,000 Text Samples*. Routledge Taylor & Francis Group. 2008.

NGWENYA, N., MILLS, S., KINGSTON, P. *Analysing weblogs of terminally ill patients using the Linguistic Inquiry and Word Count (LIWC) program.*

Stanfford University, 20??.

NIA, R., BIRD, C., DEVANBU, P. D., FILKOV, V. *Validity of Network Analyses in Open Source Projects. Computer Science Department University of California*. MSR 2010.

NIEDERHOFFER, K. G. and PENNEBAKER, J. W. *Language Style Matching in Social Interaction. Journal of language and Social Psychology*. Vol. 21 n. 4. 2002.

NOY, N. F. & MCGUINNES, D. L. *Ontology Development 101: A Guide to Creating Your First Ontology*. Stanford University, Stanford. 2001.

NOY, N. *Ontology Development 101*. Stanford University. 2005. Disponível em http://protege.stanford.edu/conference/2005/slides/T1_Noy_Ontology101.pdf. Protege Conference. Acessado em Nov/2011.

OBERLANDER, J and GILL, A. J. *Language with character: A stratified corpus comparison of individual differences in e-mail communication*. University of Edinburgh. 2006.

PALMEIRA, E, FREITAS, F. *Ontologias detalhadas e classificação de texto: uma união Promissora*. SBC. ENIA. 2007.

PATTISON, D. S., BIRD, C. A. and DEVANBU, P. T. *Talk and work: a preliminary report*. *Proceedings of the MSR*, 2008.

PENNEBAKER, J. W. and GRAYBEAL, A. *Patterns of Natural language Use: Disclosure, Personality and Social Integration. American Psychological Society*, 2001.

PENNEBAKER, J. W., MEHL, M. R. and NIEDERHOFFER, K. G. A. *Psychological Aspects of Natural language Use: Our Words, Our Selves.* Rev. Psychology. 54:547-77. 2003.

PETERS, D. et al. *Preferred 'learning styles' in students studying sports-related programme in higher education in the United Kingdom.* Studies in Higher Education. Vol 33, pages 155 – 166. 2008.

PMI - *Project Management Institute*. *A Guide to the Project Management Body of Knowledge (PMBOK Guide)* - 2000 Edition Experts – Project Management Institute. Disponível em www.pmi.org , acessado Nov/2011.

PORTER, M. F. *An algorithm for suffix stripping*. V. 14, n. 3. 1980. Disponível em: http://telemat.die.unifi.it/book/2001/wchange/download/stem_porter.html. Acesso Out/2011.

PRESSMAN, R. S. *Engenharia de Software*. São Paulo: Makron Books. 1995.

RIGBY, P. and HASSAN, A. *What can OSS mailing lists tell us? A preliminary psychometric text analysis of the Apache developer mailing list*. MSR 2007.

SAMPAIO, M, *Mineração de Dados.* Disponível em: <http://www.dsc.ufcg.edu.br/~sampaio/cursos/2006.2/PosGraduacao/MineracaoDeDados/>, acesso em: 29 de junho de 2009.

SANTOS, F. R. COLAÇO, M. Jr, MENDONÇA, M. *Uma extensão do Microsoft Visual Studio para predição de modificações de software*. 2009.

SEMPREBOM, T., CAMADAFI, M. Y., MENDONÇA, I. *Ontologia e Protégé. Programa de Pós-Graduação em Engenharia de Automação e Sistemas*. Universidade Federal de Santa Catarina (UFSC) - Florianópolis, SC, Brasil. 2007.

SICILIA, M. A., BARRIOCANAL, E. G., ALONSO, S. S. and GARCIA, D. R. *Ontologies of engineering knowledge: general structure and the case of Software Engineering*. The Knowledge Engineering Review, Vol. 24:3, 309–326. 2009.

SILVA, E. P. *Classificação de informação usando ontologias*. Dissertação de Mestrado. Universidade Federal de Alagoas. 2006.

SILVA, E. P. *Ontologias detalhadas e classificação de texto*. SBC ENIA 2007

STONE, P. J, DUNPHY, D. C, SMITH, M. S. OGILVIE, D. M. *The General Inquirer: A Computer Approach to Content Analysis*. Cambridge, MA: MIT Press. 1966.

TAUSCZIK, Y. R. and PENNEBAKER, J. W. *The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. Journal of Language and Social Psychology*. 2010.

TOSEY, P. & MATHISON, J. *Fabulous Creatures of HRD: A Critical Natural History Of Neuro-Linguistic Programming. 8th International Conference on Human Resource Development Research & Practice across Europe*, Oxford Brookes Business School. Available in http://www.nlpresearch.org/. Acessado em Dez/20011. 2007.

TOSEY, P. & MATHISON, J. *Neuro-linguistic Programming and Learning Theory: a response. The Curriculum Journal*. Vol. 14 no.3 pp. 361 – 378, 2003.

TOSEY, P. an MATHISON, J. *Neuro-Linguistic Programming. A Critical Appreciation for Managers and Developers*. Palgrave Macmillan. 2009.

VAN BAAREN, R. B., HOLLAND, R. W., STEENAERT, B., and KNIPPENBERG, A. *Mimicry for money: Behavioral consequences of imitation.* Journal of Experimental Social Psychology 39. 2003.

WAINER, J. *Métodos de pesquisa quantitative e qualitative para a Ciência da Computação*. Atualizações em informática. SBC. Páginas 221-262. Editora PUC Rio. 2007.

WITTE, R., LI, Q., ZHANG, Y. and RILLING, J. *Text mining and software engineering: an integrated source code and document analysis approach*. IET Software, Vol. 2, No. 1, February 2008.

WONGTHONGTHAM, P., CHANG, E., DILLON, T., SOMMERVILLE, I. *Development of a Software engineering Ontology for Multisite Software Development.* TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 21, 2009.

YOUNG, S. M., EDWARDS, H. M., MCDONALD, S. and THOMPSON, J. B . *Personality characteristics in an XP team*. In: HSSE - 2005, USA, ACM Press, 2005.

ZAVAGLIA et al. *Estrutura Ontológica e Unidades Lexicais: uma aplicação computacional no domínio da Ecologia*. V Workshop em Tecnologia da Informação e da linguagem humana. 2007.

ZAVAGLIA, C. *Produção de Ontologias específicas: a modelagem da OntoEco* – Estudos Linguísticos XXXIV, p. 1182-1187. 2005.

APÊNDICE A – LISTA DE TERMOS DA ENGENHARIA DE SOFTWARE

TERMOS		
"Hd"	"Hierarchy"	"Solution"
"Goal"	"Homologation"	"Abstraction"
"Coupling"	"Ide"	"Agility"
"Scope"	"Inception"	"Design"
"Business"	"Interface"	"Source code"
"Algorithm"	"Language"	"Prototype"
"Analysis"	"Library"	"Repository"
"Artifact"	"Logic"	"Recovering"
"Aspect"	"Measurement"	"Fail"
"Base"	"Method"	"Accuracy"
"Backup"	"Metric"	"Help"
"Channel"	"Model"	"Analysis Team"
"Bug"	"Object"	"Design Team"
"Code"	"Orientation"	"Activity Diagrams"
"Component"	"Paradigm"	"Swimlane"
"Conception"	"Polymorphism"	"Transition Activity"
"Construction"	"Procedure"	"Concurrent Transition"
"Data",	"Process"	"Fork Transition"
"Database"	"Program"	"Join Transition"
"Deployment"	"Programmer"	"Class Diagrams"
"Diagram"	"Query"	"Class"
"Documentation"	"Report"	"Class Attribute"
"Elaboration"	"Result"	"Class Operation"
"Encapsulation"	"Screen"	"Class Relationship"
"Environment"	"Software"	"Class Dependency"
"Backup"	"Structure"	"Class Generalisation"
"Channel"	"System"	"Class Agregation"
"Cleaning"	"Template"	"Class Association"
"Client"	"Test"	"Class Corporation"
"Cohesion"	"Use case"	"Use Case Diagrams"
"Communication"	"Allocation"	"Actor"
"Compilation"	"Alteration"	"Use Case Relationship"
"Connection"	"Architecture"	"Testing Team"
"Connector"	"Error"	"Implementation Team"
"Declaration"	"Function"	"Software Engineering
"Dictionary"	"Graphic"	Domain"
"Effectiveness"	"Gui"	"Software Constrution"
"Efficiency"	"Utility"	"Code Documentation"
"Engineer"	"Variable"	"Resource Usages"
"Exception"	"View"	"Use Of Control Structures"
"Firewall"	"Web"	"Construction Languages"
"Hardware"	"Administration"	"Configuration Language"
"Infrastructure"	"Aggregation"	"Programming Language"
"Installation"	"Area"	"Linguistic"
"Interaction"	"Assurance"	"Visual"
"Interpretation"	"Auditing"	"Constrution Testing"

TERMOS		
"Iteration"	"Calculation"	"Reusable Units"
"Layer"	"Complexity"	"Software Design"
"Legibility"	"Composite"	"Software Architecture"
"Link"	"Computer"	"Application Architectures"
"Loop"	"Consistency"	"Data Flow Component"
"Maintenance"	"Constraint"	"Data Store"
"Manual"	"Contract"	"External Entity"
"Message"	"Correction"	"Event Process"
"Metadata"	"Cost"	"Interrupt"
"Module"	"Debug"	"Language-processing
"Necessity"	"Decision"	Systems"
"Package"	"Defect"	"Data Flow Diagrams"
"Performance"	"Depuration"	"Repository Models"
"Precision"	"Disk"	"Interacted Component"
"Product"	"Effort"	"Architectural Design"
"Protocol"	"Estimate" ,	"Control System"
"Proxy"	"Evaluation"	"Routine"
"Range"	"Expertise"	"Manager Models"
"Rapidity"	"Facility"	"Central Controller"
"Release"	"Improvement"	"Managed Process"
"Reliability"	"Independence"	"Broadcast Models"
"Remarks" ,	"Index"	"Event-message Handler"
"Requirements"	"Information"	"Systems Organisation"
"Risk"	"Integration"	"Client-server Models"
"Security"	"Linkedit"	"Layered Models"
"Select"	"Login"	"Distributed Systems
"Semantic"	"Memory"	Architectures"
"Server"	"Modification"	"Distributed Client-server
"Service"	"Monitoring"	Architectures"
"Specification"	"Navigability"	"Distributed Client"
"Speed"	"Net"	"Distributed Network"
"Sql"	"Optimization"	"Distributed Server"
"Strategy"	"Ordering"	"Distributed Object
"Survey"	"Partitiony"	Architectures"
"Syntax"	"Processor"	"Object Request Broker"
"Technique"	"Quality"	"COM"
"Tier"	"Redundancy"	"CORBA"
"Tool"	"Refactoring"	"Peer"
"User"	"Response"	"Service"
"Support"	"Restorability"	"Service Provider"
"Tolerance"	"Reusability"	"Processor"
"Traceability"	"Reuse"	"Swimlane"
"Traffic"	"Revision"	"Collaborative Diagrams"
"Training"	"Rewrite"	"Collaborative Message"
"Trigger"	"Role"	"Condition Message"
"Type"	"Rollback"	"Loop Message"
"Update"	"Rule"	"Decision Action"
"Usability"	"Schedule"	"Decision Condition"

TERMOS		
"Validation"	"Size"	"Decision Rule"
"Verification"	"Storage"	"Flowcharts"
"Array"	"Strength"	"Data Input"
"Checkin"	"Source"	"Data Output"
"Collection"	"Style"	"Processing Step"
"Edit"	"Table"	"Start Relationship"
"File"	"Writing"	"Stop Relationship"
"Guidelines"	"Activity"	"Program Design Language"
"Idea"	"Development"	"Sequence Diagrams"
"Interoperability"	"Framework"	"Timeline"
"Line"	"Implementation"	"State"
"List"	"Methodology"	"Transition State"
"Option"	"Pattern"	"Component Diagrams"
"Parameter"	"Distribution"	"Component Dependency"
"Pointer"	"Handbook"	"Component Aggregation"
"Portability"	"Management"	"Component Association"
"Reference"	"Network"	"Component Composition"
"Return"	"Version"	"Deployment Diagrams"
"Script"	"Commit"	"Entity"
"Self-sufficiency"	"Department"	"Entity Attribute"
"Set"	"Reutilization"	"Entity Relationship"
"Scenarios"	"Site"	"Jackson Action"
"Viewpoints"	"Technology"	"Jackson Entity"
"Interactor Viewpoints"	"Transaction"	"Object Diagrams"
"Requirements Sources"	"Analyst"	"Object"
"Domain knowledge"	"Codification"	"Object Attribute"
"Operation Environment"	"Project"	"Object Relationship"
"Stake Holder"	"Specification-based	"Object Dependency"
"Software Testing"	Techniques"	"Object Structural"
"Test Execution"	"Formal Specifications	"Object Aggregation"
"Test Log"	Testing"	"Object Association"
"Test Planning"	"Random Testing"	"Object Composition"
"Test Level"	"Robustness Testing"	"Use Case Diagrams"
"Integration Testing"	"Operation Profile"	"Software Requirements"
"System Testing"	"Failure"	"Requirements"
"Unit Testing"	"Testing Issues"	"Emergent Properties"
"Test Techniques"	"Test Criteria"	"Functional Requirements"
"Component-based testing"	"Testability"	"Privacy Requirements"
"GUI Testing"	"Testing Objective"	"Safety Requirements"
"Object-oriented Testing"	"Acceptance Testing"	"Space Requirements"
"Code-based Techniques"	"Configuration Testing"	"Standards Requirements"
"FlowGraph"	"Conformace Testing"	"Usability Requirements"
"Error Guessing"	"Instalation Testing"	"Process Requirements"
"Mutation Testing"	"Performance Testing"	"Product Requirements"
"Adhoc Testing"	"Recovery Testing"	"Quantifiable Requirements"
"Exploratory Testing"	"Regression Testing"	"System Requirements"
"Debuggers"	"Stress Testing"	"Users Requirements"
"Editors"	"Usability Testing"	"Conceptual Modeling"

TERMOS		
"Interpreters" "Software Testing Tools" "Performance Analysis Tools" "Customers" "Market Analysts" "Software Engineers"	"Software Tools" "Software Construction Tools" "Compilers" "Software Design Tools" "Closed Interviews" "Open Interviews" "Prototypes" "Interviews"	"Requirements Elicitation" "Elicitation Techniques" "Ethnography" "Facilitated Meeting"