MÉTODO DE APOIO À INOVAÇÃO DO PRODUTO DE SOFTWARE EM USO UTILIZANDO MINERAÇÃO DE OPINIÃO DE USUÁRIOS NA WEB

FRANK MENDES NOBRE

MÉTODO DE APOIO À INOVAÇÃO DO PRODUTO DE SOFTWARE EM USO UTILIZANDO MINERAÇÃO DE OPINIÃO DE USUÁRIOS NA WEB

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Clarindo Isaías Pereira da Silva e Pádua - Co-orientador: Prof. André Luiz Zambalde

Belo Horizonte

21 de junho de 2011

Mendes Nobre, Frank
D1234p Método de apoio à Inovação do Produto de Software em Uso Utilizando Mineração de Opinião de Usuários na Web / Frank Mendes Nobre. — Belo Horizonte, 2011
xxii, 101 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais
Orientador: Prof. Clarindo Isaías Pereira da Silva e Pádua - Co-orientador: Prof. André Luiz Zambalde

1. Qualidade de software. 2. Produto de software. 3. Software em uso. 4. Inovação em software. 5. Mineração Web. I. Título.

CDU 519.6*82.10

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Método de apoio à inovação do produto de software em uso utilizando mineração de opinião de usuários na web

FRANK MENDES NOBRE

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. CLARINDO ISAÍAS P. DA SILVA E PÁDUA - Orientador
Departamento de Ciência da Computação - UFMG

PROF. AHMED ALI ABDALLA ESMIN
Departamento de Ciência da Computação - UFLA

Eduardo M P Figueiredo
PROF. EDUARDO MAGNO LAGES FIGUEIREDO
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 21 de junho de 2011.

À minha filha Fernanda e minha esposa Cassia, amores da minha vida, Dedico.

Agradecimentos

Agradeço a Deus em primeiro lugar, pois ele me fornece forças diariamente para suportar todas as adversidades que se apresentam e me faz compreender que preciso viver com tranqüilidade e simplicidade.

À minha esposa Cassia, que me alertou para este curso de mestrado e forneceu todo o suporte que eu precisei ao longo deste projeto. Senti e sou grato por todo o esforço, amor, paciência e carinho que ela me proporcionou. Sem isto eu não teria chegado até aqui.

À minha filha Fernanda. Por várias vezes deixei de brincar com ela para cuidar dos trabalhos e compromissos acadêmicos relacionados a este trabalho. Acredito que ela sentiu tudo que eu senti. Mas o resultado deste esforço, tenho certeza, será fundamental para o futuro dela.

Ao meu falecido pai (Fernando). Enquanto ele estava entre nós sempre se esforçou e priorizou a educação dos filhos. Sei que ele está orgulhoso de me ver chegar até aqui e saber que não vou parar, pois, ele me ensinou que não preciso ter pressa, mas não posso parar.

À minha mãe Maria Jovita e meus dois irmãos (Fernando Jr. e Frederico) que também me deram muito suporte durante este projeto. Eles cuidaram de outras tarefas muito importantes que eu infelizmente não conseguia cuidar.

Ao professor André Zambalde, pela paciência, ensinamentos, dedicação e responsabilidade com que me orienta desde a época da graduação.

Ao professor Clarindo que prontamente aceitou a responsabilidade de me orientar e dedicou-se a este trabalho. Ele mostrou-se interessado e paciente mesmo quando este projeto ainda não se mostrava muito interessante.

Ao professor Ahmed Esmin pelas boas idéias fornecidas desde o início deste trabalho de pesquisa.

Ao Centro Universitário de Lavras - UNILAVRAS, onde orgulhosamente sou funcionário, por me possibilitar realizar este curso de mestrado. Em especial à professora Christiane Lunkes Argenta e ao professor João Antônio Argenta, que me motivaram e

sempre se mostraram interessados pelo meu trabalho e crescimento profissional.

À equipe do Centro de Atendimento ao Aluno - UNILAVRAS, que durante este curso de mestrado me forneceu muito apoio no nosso trabalho de atendimento a alunos de graduação do UNILAVRAS.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pelos seis meses de bolsa de estudos que me ajudaram muito nas viagens que fiz entre Lavras e Belo Horizonte.

À UFLA e UFMG, que em função de suas parcerias e esforços possibilitaram o meu ingresso neste curso de mestrado.

E, finalmente, agradeço a todos aqueles que de maneira direta o indiretamente contribuíram para realização deste importante projeto em minha vida.

"O mal de quase todos nós é que preferimos ser arruinados pelo elogio a ser salvos pela crítica."
(Norman Vincent Peale)

Resumo

Entende-se como método o caminho ou processo racional para se atingir a solução de determinado problema. O presente trabalho propõe um método para levantamento e avaliação de opinião de usuários do produto de software em uso na Web e discute as potencialidades deste método no apoio ao problema de pesquisa relacionado à inovação - incremental ou radical - do produto de software a partir destas opiniões. O método proposto foi fundamentado nos estudos e integração das áreas de Engenharia, principalmente qualidade, avaliação e manutenção de software; Inovação e inovação do produto de software; e mineração de dados na Web (*Web Mining*). Na seqüência do trabalho este método foi validado através de um estudo de caso, onde os principais objetos de estudo foram o produto de software em uso Windows Vista® e páginas Web recuperadas no fórum de discussão do sítio Clube do Hardware. Os resultados alcançados neste estudo prático foram satisfatórios e indicaram, principalmente, opiniões de usuários que podem levar a melhoria de qualidade e conseqüentemente inovação do produto de software avaliado, mediante analise e decisão gerencial por parte do mantenedor.

Palavras-chave: Qualidade de software, Produto de software, Software em uso, Inovação em software, Mineração Web.

Abstract

A method can be understood as a way or a rational process to reach the solution of a determined problem. This paper proposes a method to verify and evaluate the user's opinion about the software product in use in Web. It discusses the potential of this method to support on the search problem related to inovation - incremental or radical - of the software product when analysing these user's opinion. This method is based in a study that integrates software engineering areas, mainly : software quality, evaluation and maintenance; Inovation and software inovation; and web data mining. This method was evaluated using a case study where have been used the Windows Vista® software product and opinions presented in webpages from the Clube do Hardware forum. The results reached in this case study were satisfactory and they indicated user's opinions that can lead to a quality improvement and, thus, to inovate the software product being evaluated, according analysis and management decision by the maintainer.

Keywords: Software quality, Software product, Software in use, Innovation in software, Web Mining.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Este trabalho encontra-se em um contexto onde a investigação de métodos que auxiliem a inovação de produtos e serviços é uma questão desafiadora e aberta a pesquisas. Consumidores domésticos ou organizações cada vez mais exigem produtos e serviços com qualidade e pressionam as organizações produtoras a buscar inovações neste sentido.

Para suprir essa demanda as empresas procuram métodos que possam propiciar inovação incremental nos produtos e serviços já existentes ou inovação radical que gerem novos produtos ou serviços para comercialização. Em ambas as situações, o objetivo principal é atender às necessidades e expectativas dos clientes [Grützmann et al., 2010].

O fato é que em uma economia cada vez mais globalizada, as organizações produtoras de bens e serviços somente conseguem sobreviver se agirem no sentido de inovação contínua, principalmente observando as necessidades e expectativas manifestadas por consumidores ou usuários[1].

Dentro deste escopo de pesquisa, um produto que merece especial atenção no sentido de inovação é o software[2]. Guerra & Colombo [2008] relatam que o software influencia a vida cotidiana de milhares de pessoas e organizações, sendo que, em alguns casos, é usado para apoiar tarefas simples do cotidiano, mas, em inúmeras situações, é ferramenta de trabalho aplicada à gestão e controle de complexos recursos e atividades organizacionais.

[1] No restante deste trabalho será utilizado o termo "usuários" para se referir também a consumidores e clientes. Em especial, usuário é o indivíduo que usa o produto de software para executar funções específicas, não sendo discutido o nível de conhecimento dele.

[2] Produto de software: software "pronto" disponível ao usuário na forma de pacote ou serviço.

Ainda segundo Guerra & Colombo [2008], ao se observar os últimos 20 anos de história do desenvolvimento de software, a realidade mundial demonstra um aumento significativo no uso de sistemas computacionais. Conseqüentemente, isto implica na existência de uma demanda crescente por novos produtos de software e, também, por melhorias contínuas daqueles que já estão em uso.

As primeiras manifestações relacionadas à busca por inovações em software ocorreram entre os anos de 1960 a 1970, período da crise do software. A essência dessa crise era a dificuldade de se criar e manter produtos de software que atendessem adequadamente às expectativas dos usuários, somando-se à devida preocupação com requisitos de manutenção e especialmente de qualidade. A demanda por um produto de qualidade era crescente, principalmente porque a capacidade do hardware havia evoluído muito, propiciando e induzindo a satisfação dessa demanda.

Em meio aquele cenário de crise e busca pela qualidade surgiu uma área de conhecimento denominada Engenharia de Software (ES), com objetivos relacionados à pesquisa de métodos, ferramentas e procedimentos destinados ao desenvolvimento de softwares de qualidade, que atendam principalmente às necessidades e expectativas de usuários [Koscianski & Soares, 2007].

O *SWEBOK* (*Software Engineering Body of Knowledge*) [Abran & Moore, 2004], guia de referencia para a área de ES, procura descrever, organizar e prover acesso ao corpo de conhecimentos sobre esta área.

Para cobrir esta ampla área de estudo, o *SWEBOK* divide a ES em 11 áreas de conhecimento: Requisitos de software, Projeto, Desenvolvimento, Teste, Manutenção, Gerência de configuração, Qualidade, Gerência de ES, Processo, Ferramentas e Métodos, e Disciplinas relacionada à Engenharia de Software [Abran & Moore, 2004].

Dentre estas áreas, Manutenção e Qualidade de Software possuem especial relevância em inovações de software em uso. Pois, Manutenção de Software visa principalmente à correção de erros e aprimoramentos de software em uso, propiciando uma inovação incremental nesse produto. E a área de Qualidade de Software está concentrada no acompanhamento e controle deste produto durante todo o seu ciclo de vida, desde o projeto inicial, passando pela produção, até a entrega e manutenção [ISO/IEC 9126-1, 2001; ISO/IEC 14764, 2006; ISO/IEC 12207, 2008]. Neste sentido, pode-se afirmar que se incluem como objetivos indiretos destas duas áreas abordagens relacionadas à inovação de produtos de software, sejam nas formas incremental e ou radical, de maneira a atender às necessidades e expectativas dos usuários.

De acordo com Lippoldt & Stryszowski [2009], as inovações no setor de software são freqüentemente impulsionadas pelas necessidades e expectativas dos usuários. De forma similar, Pfleeger & Atlee [2006] relatam que usuários são *experts* sobre como um produto de software se comporta, quais as características são mais utilizadas e quais aspectos dele precisam evoluir (melhorar) e ou mudar radicalmente.

Diante disto, é observado que a participação de usuários em processos de inovação do produto de software é importante. Seja em manutenção, que muitas vezes demandam inclusão de novas funcionalidades no produto de software em uso, ou em ações que visem a melhoria de qualidade, pode surgir a necessidade de criação de um novo produto. Nestas duas situações, o mantenedor[3] pode observar e decidir sobre aceitar ou não opiniões[4] de usuários para inovar o produto de software.

De acordo com Von Hippel [1986] e Tapscott & Williams [2007] são vários os episódios de criatividade guiada por opiniões de usuários na história das inovações. Entretanto, em muitos casos, até mesmo quando pareciam promissoras, as inovações propostas por usuários não foram implementadas, principalmente pela rigidez dos processos internos, ausência de ferramentas e métodos e ou centralidade excessiva nas organizações.

Contudo, hoje, pode-se afirmar que este cenário encontra-se em efetivo processo de mudança, especialmente no contexto dos produtos de software. A utilização intensiva pelos usuários da ferramenta *Word Wide Web* (Web) é um aspecto significativo. Um palco para criar comunidades, fóruns, redes e, enfim, coletar opiniões que podem levar a pequenas alterações (melhorias) ou mudanças radicais nos produtos. Em outras palavras, as organizações produtoras de software podem coletar opiniões de usuários no ambiente Web para promover a inovação.

É claro que a coleta de opiniões manifestadas por usuários de software, particularmente visando à busca de melhorias, qualidade e inovação, pode ser realizada através de pesquisas de mercado, grupos de foco ou outros instrumentos que visam questionar diretamente o usuário sejam elas puramente manuais ou utilizando a Web [Grützmann et al., 2010].

[3]Neste trabalho de pesquisa o termo mantenedor se refere a organizações produtoras e mantenedoras do produto de software.

[4]O termo "opinião" neste trabalho tem o sentido de crítica, apreciação ou análise de um fato, de uma situação.

Por exemplo, poderiam ser utilizados técnicos para realizar o trabalho de coleta de opiniões a respeito do produto de software em uso e, em seguida, especialistas que fizessem uma avaliação e seleção dessas opiniões, de forma a obter informações a respeito do produto em investigação. O resultado seria, então, apresentado ao mantenedor, o qual tomaria a decisão de realizar as melhorias (inovações incrementais) ou mudanças (inovações radicais) necessárias ao produto. Entretanto, a abordagem manual sugere dois problemas potenciais, o tempo e a quantidade de recursos demandados para cumprir uma tarefa de coleta de opiniões - particularmente se considerarmos um produto de software de alcance global.

A abordagem direta utilizando a Web como, por exemplo, uso de questionários enviados por e-mail ou questionários disponibilizados para preenchimento em uma página Web, reduz a quantidade de recursos demandados. No entanto, enfrenta problemas relacionados ao tempo e mesmo à disponibilidade ou à boa vontade dos usuários em responderem às questões em estudo. Segundo Gamon et al. [2005], nem sempre as pessoas estão dispostas a dar o retorno explícito requerido nestas pesquisas.

A coleta indireta via Web (fóruns, blogs, microblogs, redes sociais, entre outros) de opiniões manifestadas por usuários sobre produtos em uso é uma alternativa atual e em evolução [Lo & Potdar, 2009; Shandilya & Jain, 2009]. Neste contexto, uma das tarefas que se revela importante diz respeito proposição de métodos e ou práticas de mineração dos dados representativos destas opiniões e a transformação dos mesmos em informações estratégicas que auxiliem os mantenedores de software a realizar melhorias e ou mudanças, alcançando inovações.

Então, diante da importância do produto de software para a sociedade e organizações, da relevância de opiniões de usuários para melhoria de qualidade desse produto e os potenciais problemas a serem enfrentados no processo de levantamento automatizado dessas opiniões na Web, a motivação deste trabalho esta relacionada a necessidade de pesquisa e proposição de métodos e técnicas capazes de coletar e tratar as opiniões de usuários disponíveis em páginas da Web, a fim de auxiliar os processos de melhoria e qualidade visando à inovação do produto de software em uso.

1.1 Objetivos e escopo

O objetivo geral deste trabalho é propor um método para levantamento e avaliação de opiniões de usuários disponíveis na Web sobre determinado produto de software e discutir as potencialidades deste método no apoio aos processos de inovação incremental ou radical deste produto.

Entende-se como método o caminho ou processo racional para se atingir a solução de determinado problema, no caso o levantamento e avaliação das opiniões dos usuários visando propor inovação ao produto de software. Inovação é a transformação das idéias e ou opiniões dos usuários em melhorias, novas funcionalidades ou mudanças potenciais no produto de software em uso. Segundo Rose [2010], inovação incremental diz respeito a atividades que envolvem manutenção e ou customização do produto (pequenas modificações e melhorias). Um tipo de inovação associado a organizações com forte orientação para o mercado e que procuram entregar aos seus clientes (usuários) aquilo que eles procuram no imediato. Por outro lado, inovação radical refere-se, por exemplo, à inclusão de funcionalidades avançadas, criação de versão superior ou alterações de plataforma. Um tipo de inovação associado a maiores riscos, maiores retornos e, muitas vezes, orientada para novas tecnologias/produtos e menos para o cliente.

Como objetivos específicos, associados a este trabalho tem-se:

- O estudo e integração nas áreas de Engenharia da qualidade, avaliação e manutenção de software; Inovação e inovação do produto de software; e mineração de dados na Web (*Web Mining*);
- A proposição do método para levantamento e avaliação de opiniões de usuários de produtos de software em uso, tendo como repositório de opiniões páginas Web (fóruns, blogs, redes sociais, entre outros);
- A validação do método via estudo de caso: seleção de um produto de software em uso, levantamento de páginas Web com opiniões sobre este produto e aplicação do método proposto; e
- A avaliação dos resultados obtidos, considerando discussões e conclusões relacionadas as potencialidades e limitações do método proposto no apoio aos processos de inovação incremental ou radical de produtos de software.

Desta forma, é esperado que esta pesquisa apresente um método validado por um estudo de caso, e que contribua para a inovação incremental do produto de software em uso, ou contribua na concepção de novos produtos e ou novas formulações de negócios.

1.2 Organização do trabalho

Visando atender aos objetivos propostos e contribuir para os estudos envolvendo Engenharia e produto de software, mineração de dados Web e inovação, o presente texto está organizado da seguinte maneira:

- No Capítulo 2 tem-se o referencial teórico, com a descrição de conceitos e técnicas envolvendo os temas engenharia, principalmente qualidade, avaliação e manutenção de software; inovação e inovação de produtos de software; mineração de dados na Web; e, por fim, trabalhos que relacionam os temas inovação do produto de software e mineração de dados;

- O Capítulo 3 busca inicialmente a classificação teórico-metodológica da pesquisa. Em seguida apresenta os procedimentos que viabilizaram a criação do método para levantamento e avaliação de opiniões de usuários de produtos de software em uso;

- No Capítulo 4, o método para levantamento e avaliação de opiniões de usuários de software em uso na Web é apresentando utilizando diagrama de atividades da UML;

- O Capítulo 5, descreve o estudo de caso visando a validação do método proposto. Os resultados alcançados são avaliados e discutidos no contexto de inovação do produto de software em uso;

- Finalmente, no Capitulo 6, tem-se as considerações finais sobre o trabalho de pesquisa e a proposição de trabalhos futuros.

Capítulo 2

Referencial Teórico

Este capítulo é composto de quatro seções. A primeira seção apresenta uma revisão sobre temas Engenharia de Software (ES), qualidade de software e manutenção de produtos de software. O objetivo é descrever alguns dos fundamentos que aproximam esses temas com a inovação de produto de software. Conforme mencionado no capítulo de introdução, qualidade e manutenção de software encontram-se relacionados à inovação incremental. E, num contexto mais amplo, estas e as demais áreas da ES somam-se ao processo de inovação do produto de software em sua abordagem radical.

Na segunda seção são referenciados os conceitos relacionados à inovação e inovação de produto de software. Trata-se de uma revisão e contribuição desta pesquisa ao tema inovação de produto de software.

Na terceira seção deste capítulo tem-se a abordagem relacionada a técnicas e métodos para mineração de dados. Estas ferramentas tecnológicas compõem a base do método proposto neste trabalho, ou seja, para levantar as opiniões de usuários do produto de software em uso na Web, de forma a fornecer subsídios para propor inovações, utiliza-se a Mineração de dados Web (*Web Mining*).

Finalmente, na quarta seção são apresentados trabalhos que possuem relação com os temas principais desta pesquisa: inovação do produto de software e mineração de dados; como forma de fundamentação através de uma analise comparativa de outras abordagens.

2.1 Engenharia, qualidade e manutenção de software

O termo Engenharia de Software (ES) foi introduzido no ano de 1968 durante uma conferência de mesmo nome promovida pelo comitê científico da OTAN - Organização do Tratado do Atlântico Norte, cujo objetivo era discutir vários dos problemas de organizações e usuários com o produto de software [Naur & Randell, 1968].

De acordo com Koscianski & Soares [2007], a conferência promovida pela OTAN era a primeira reação à crise do software, que aconteceu naquela época. A essência dessa crise era a dificuldade que os programadores enfrentavam para criar softwares de qualidade, que já era demandada por consumidores. Em meio a esse cenário surgia a área de Engenharia de Software.

Conforme *SWEBOK* (*Software Engineering Body of Knowledge*) [Abran & Moore, 2004], um dos principais guias relacionados a Engenharia de Software, esta área do conhecimento visa a aplicação de uma abordagem sistemática, disciplinada e quantificada para o desenvolvimento, operação e manutenção de software. Para cobrir uma área de estudo tão ampla o guia divide a ES em onze sub-áreas, a saber:

I. Requisitos de Software - identificação, análise, especificação e validação dos requisitos de software;

II. Projeto de Software - processo de definição da arquitetura, componentes, interfaces e outras características de um sistema ou componente;

III. Desenvolvimento de Software - criação de normas de trabalho, na construção de software, verificação, testes de unidades, testes de integração e depuração;

IV. Teste de Software - verificação experimental da qualidade do produto em teste, considerando o contexto onde este será utilizado;

V. Manutenção de Software - suporte ao produto de software, durante o seu desenvolvimento e após início do seu uso;

VI. Gerência de Configuração de Software - identificação da configuração de um sistema em pontos diferentes no tempo, com a finalidade de controlar sistematicamente alterações de configuração;

VII. Gerência de Engenharia de Software - gestão de projetos de desenvolvimento de software;

VIII. Processo de Desenvolvimento de Software - criação, modificação, definição, avaliação e medição do processo de desenvolvimento de software;

IX. Métodos e Ferramentas de Engenharia de Software - processo do ciclo de vida do software;

X. Qualidade de Software - qualidade na produção e manutenção de um produto de software; e,

XI. Disciplinas relacionadas à Engenharia de Software, tais como gerenciamento de projeto, gerenciamento de qualidade, dentre outras.

Dentre estas sub-áreas, Manutenção e Qualidade de Software possuem especial relevância em inovações do produto de software em uso. A Manutenção de Software visa principalmente à correção de erros e aprimoramentos de software em uso, propiciando uma inovação incremental. A sub-área de Qualidade de Software, por sua vez, está concentrada no acompanhamento e controle de qualidade do produto durante todo o seu ciclo de vida, desde o projeto inicial, passando pela produção, até a entrega e manutenção; ocupando-se, portanto, com o desenvolvimento de novos produtos de software (inovação radical) e manutenção (inovação incremental) de maneira a atender às necessidades e expectativas dos usuários.

As normas de Qualidade e Avaliação do Produto de Software propiciam um conjunto de conhecimentos, métodos e técnicas que viabilizam a avaliação de software com vistas, por exemplo, a inovar este produto [ISO/IEC 9126-1, 2001; ISO/IEC 14764, 2006; ISO/IEC 12207, 2008].

2.1.1 Qualidade de software

De acordo com Pressman [2006] a Qualidade de Software é a satisfação de requisitos funcionais e de desempenho explicitamente declarados, normas de desenvolvimento explicitamente documentadas e características implícitas que são esperadas em todo produto de software desenvolvido profissionalmente.

A qualidade do produto de software é vista como sendo o conjunto de características que devem ser alcançadas em um determinado nível, para que o produto atenda as necessidades dos seus usuários. E por meio desse conjunto de características a qualidade do produto pode ser especificada e avaliada [Maldonado et al., 2001].

A definição de características de qualidade apropriadas, levando em consideração o uso pretendido do produto de software, assim como sua avaliação, constitui a base para o desenvolvimento de produtos de software de qualidade. Para alcançar tal objetivo, o conjunto de normas ISO/IEC 9126-1 [2001], ISO/IEC 9126-2 [2003], ISO/IEC 9126-3 [2003] e ISO/IEC 9126-4 [2004], que dizem respeito à Qualidade de Software, fornecem uma base valiosa para medidas e uma extensa lista de verificação para avaliar a qualidade do produto de software.

De acordo com Guerra & Colombo [2008], avaliar a qualidade de produtos de software é uma alternativa que tanto produtores quanto consumidores podem utilizar para melhorar a qualidade desse produto. Para realizar uma avaliação efetiva é necessário utilizar um modelo válido que permita estabelecer e avaliar requisitos de qualidade. Este objetivo pode ser alcançado utilizando as normas ISO/IEC 9126 em conjunto com as normas ISO/IEC 14598, que cobrem o assunto sobre avaliação do produto de software.

A ISO/IEC 9126 é um conjunto de normas sobre qualidade do produto de software composta de quatro partes, detalhadas a seguir.

***ISO/IEC 9126* - Modelo de Qualidade:** Essa parte apresenta um modelo de qualidade com as características e sub-características que podem ser utilizadas para a avaliação de um produto de software. Ela é dividida em duas partes:

Qualidade interna e qualidade externa: especifica seis características para qualidade interna e externa, as quais são por sua vez divididas em sub-características. Estas sub-características são manifestadas externamente, quando o software é utilizado como parte de um sistema computacional, e são resultantes de atributos internos do software. As seis características são apresentadas na Figura 2.1[1] com suas respectivas sub-características, conforme descrito na ISO/IEC 9126-1 [2001].

[1]Esta figura foi extraída da ISO/IEC 9126-1 [2001] e adaptada (o texto foi traduzido para a língua portuguesa) para apresentação neste trabalho de pesquisa.

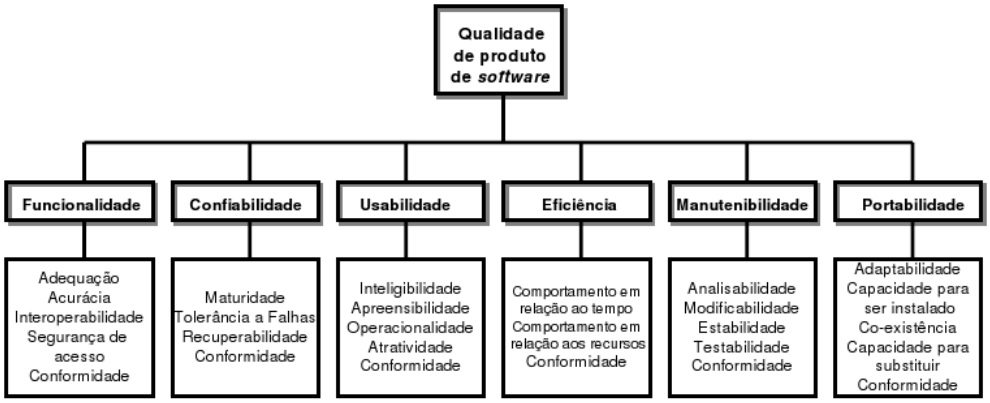

Figura 2.1. Modelo de qualidade para qualidade externa e interna [ISO/IEC 9126-1, 2001]

A seguir, as seis características conforme definidas na norma ISO/IEC 9126-1 [2001] são descritas:

- Funcionalidade: é a capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições específicas.
- Confiabilidade: é a capacidade do produto de software de manter um nível de desempenho especificado, quando usado em condições específicas.
- Usabilidade: é a capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições específicas.
- Eficiência: é a capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições específicas.
- Manutenibilidade: é a capacidade do produto de software de ser modificado, por exemplo, necessidade de correções, melhorias ou adaptações do software devido a mudanças no ambiente e seus requisitos ou especificações funcionais.
- Portabilidade: é a capacidade do produto de software de ser transferido e adaptado em outro ambiente.

Qualidade em uso: este modelo especifica quatro características de qualidade em uso - efetividade, produtividade, segurança e satisfação - não descendo ao nível de sub-características. A qualidade em uso pode ser vista como a capacidade do produto de software de permitir que usuários especificados atinjam metas especificadas com efetividade, produtividade, segurança e satisfação em contextos de uso específicos. Estas quatro características que representam a qualidade em uso são apresentadas na Figura 2.2[2] e descritas na seqüência conforme ISO/IEC 9126-1 [2001].

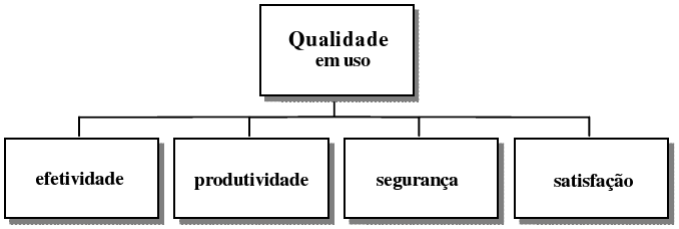

Figura 2.2. Modelo de qualidade para qualidade em uso [ISO/IEC 9126-1, 2001]

- Efetividade: é a capacidade do produto de software de permitir ao usuário alcançar objetivos específicos com precisão e de maneira completa em um contexto de uso específico.
- Produtividade: é a capacidade do produto de software de permitir ao usuário o emprego de quantidade apropriada de recursos, tais como tempo para executar uma tarefa, esforço, materiais ou custos financeiros, em relação à eficácia obtida em um contexto de uso específico.
- Segurança: é a capacidade do produto de software de apresentar níveis aceitáveis de riscos, decorrentes de deficiências na funcionalidade, que podem provocar danos às pessoas, negócios, software, propriedades ou ao ambiente, em um contexto de uso específico.
- Satisfação: é a capacidade do produto de software de alcançar a satisfação dos usuários em um contexto de uso específico.

***ISO/IEC* 9126-2 - Métricas Externas:** apresenta métricas que podem ser aplicadas para avaliar o produto de software por meio de medidas baseadas nos requisitos do usuário. Essas métricas permitem aos usuários, avaliadores, testadores e desenvolvedores a capacidade de avaliar a qualidade do produto durante a fase de testes ou operação [ISO/IEC 9126-2, 2003].

[2]Esta figura foi extraída da ISO/IEC 9126-1 [2001] e adaptada (o texto foi traduzido para a língua portuguesa) para apresentação neste trabalho de pesquisa.

***ISO/IEC* 9126-3 - Métricas Internas:** apresenta métricas que podem ser aplicadas para avaliar o produto de software durante seu estágio de desenvolvimento. O propósito principal dessas métricas é assegurar a qualidade externa e a qualidade em uso do produto de software. Elas permitem aos usuários, avaliadores, testadores e desenvolvedores a capacidade de avaliar a qualidade do produto antes de ele ser disponibilizado para uso [ISO/IEC 9126-3, 2003].

***ISO/IEC* 9126-4 - Métricas de Qualidade em Uso:** apresenta métricas que podem ser aplicadas para avaliar quanto o produto de software atende ao usuário durante o seu uso. O objetivo é constatar o quanto um produto auxilia um usuário específico para alcançar metas especificadas como efetividade, produtividade, segurança e satisfação. Quando se avalia a qualidade em uso se valida a qualidade do produto de software em um cenário específico de tarefas do usuário [ISO/IEC 9126-4, 2004]. Ademais, a ISO/IEC 9126-4 [2004] apresenta em seu anexo E, em caráter informativo, uma sugestão de processo que pode ser adotado para avaliação de qualidade em uso. Esse processo possui a seguinte seqüência: i) Estabelecer os requisitos de avaliação; ii) Especificar a avaliação; iii) Projetar a avaliação, e; iv) Executar a avaliação.

O conjunto de normas ISO/IEC 14598 foi criado para uso de desenvolvedores, adquirentes e outros interessados em avaliar o produto de software, que tenham necessidade de tomar alguma decisão. Por exemplo, os resultados de uma avaliação, gerados a partir do uso destas normas, podem ser usados para fazer melhorias no produto de software, sob o ponto de vista do fabricante, ou decidir positivamente ou negativamente pela compra do produto, sob o ponto de vista do comprador.

A norma ISO/IEC 14598-1 [1999] apresenta um framework que fundamenta o processo de avaliação, baseado em um modelo de qualidade, aplicável a qualquer produto de software, Figura 2.3[3].

[3]Esta figura foi extraída da ISO/IEC 14598-1 [1999] e adaptada (o texto foi traduzido para a língua portuguesa) para apresentação neste trabalho de pesquisa.

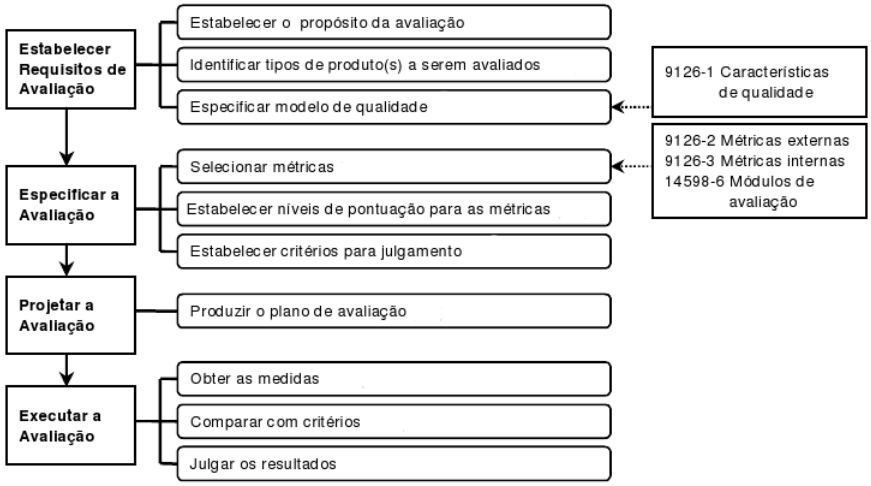

Figura 2.3. Processo de avaliação aplicável a qualquer produto de software [ISO/IEC 14598-1, 1999]

O primeiro passo deste processo consiste em estabelecer os requisitos para a avaliação. Para isto, pode-se estabelecer o propósito da avaliação, identificar o produto a ser avaliado e especificar o modelo de qualidade que será utilizado.

O propósito da avaliação pode ser, por exemplo, apoiar o desenvolvimento, decidir pela aquisição ou não do produto, assegurar que o produto ainda fornece a qualidade requerida aos seus usuários, fornecer retorno aos responsáveis pela manutenção sobre qualquer necessidade de alteração, validar se os requisitos de manutenibilidade e portabilidade são atingidos, decidir quando aprimorar ou substituir o produto.

A decisão sobre o produto de software a ser avaliado está diretamente conectada com propósito do processo de avaliação. Em geral, pode-se selecionar o produto em qualquer fase do seu ciclo de vida. No entanto, quando se trata de um produto finalizado é importante definir um escopo para o processo, optando por avaliar todo o produto de software ou, eventualmente, alguns de seus componentes.

A última parte do primeiro passo do processo de avaliação da Figura 2.3 é especificar o modelo de qualidade que será usado no processo de avaliação. A norma ISO/IEC 14598-1 [1999] prescreve o uso de qualquer modelo de qualidade. No entanto, recomenda utilizar o modelo de qualidade de software apresentado na norma ISO/IEC 9126-1 [2001], por se tratar de um modelo já conhecido e validado.

O segundo passo apresentado no framework da Figura 2.3 é especificar a avaliação. Esse passo envolve a seleção de métricas, o estabelecimento de níveis de pontuação para essas métricas e critérios para julgamento das mesmas. A seleção de métricas pode ser baseada nos requisitos de qualidade considerados fundamentais. Esses recebem níveis de pontuação e sub-sequentemente o avaliador e o requisitante da avaliação estabelecem juntos os critérios para a interpretação dos resultados das medições, para obter conclusões sobre a qualidade do produto, a partir de conjunto de valores obtidos da aplicação de métricas.

O terceiro passo do framework consiste em projetar a avaliação. Neste passo, é criado o plano da avaliação, que consiste em descrever os métodos e o cronograma das ações do avaliador. A execução da avaliação é o último passo do processo apresentado na Figura 2.3. Esse passo consiste em obter as medidas (coletar os dados), comparar com os critérios de julgamento definidos no segundo passo e julgar os resultados.

O resultado do julgamento pode passar por uma decisão gerencial para enfim obter um resultado final do processo de avaliação. Essa decisão será tomada, baseada em critérios gerenciais que levam em consideração decisões estratégicas organizacionais, tais como custo e adaptações que serão necessários. Por outro lado, o avaliador pode ser o responsável por emitir parecer sobre o resultado final, mas isso precisa ser especificado no processo de avaliação.

O conjunto de normas que cobrem os estudos sobre Qualidade de Software e Avaliação do Produto de Software possibilitam juntos, uma visão sobre como avaliar o produto de software baseando-se em um modelo de qualidade, em qualquer fase do ciclo de vida deste produto [ISO/IEC 9126-1, 2001; ISO/IEC 14598-1, 1999; ISO/IEC 12207, 2008].

De acordo com a norma ISO/IEC 9126-1 [2001], avaliar e melhorar o processo de desenvolvimento de software é um meio para melhorar a qualidade do produto de software, e avaliar e melhorar a qualidade deste produto é um meio de melhorar a qualidade em uso. Analogamente, avaliar a qualidade em uso pode fornecer experiência para melhorar e ou inovar o produto e avaliar o produto pode fornecer experiência para melhorar o processo.

> Cabe observar aqui a importância da participação dos usuários nos processos de avaliação e busca da melhoria da qualidade do produto de software. Conforme Pfleeger & Atlee [2006], usuários, em geral, são *experts* sobre como este produto trabalha, quais as características são mais utilizadas e quais aspectos do produto precisam melhorar.

Por outro lado, Guerra & Colombo [2008] afirmam que o mantenedor de software pode utilizar os resultados da avaliação de seu produto, realizada junto aos usuários, para identificar necessidades de melhoria e ou inovação. Em seguida, após identificar estas necessidades, o mantenedor pode decidir, por exemplo, pela manutenção do produto, corrigindo e aprimorando o software. Diante disto, observa-se que Manutenção de Software também é um campo de estudo importante para fundamentar pesquisas sobre inovação do produto de software.

2.1.2 Manutenção de software

A norma ISO/IEC 14764 [2006] descreve manutenção como um processo que se inicia por meio de requisição de modificação, que se origina de usuários de software. Esta requisição pode gerar ações como alterações no código fonte e documentação associada em função de problemas e/ou necessidade de aprimoramentos/melhorias do produto de software.

O *SWEBOK* (*Software Engineering Body of Knowledge*) [Abran & Moore, 2004], afirma que as ações de manutenção podem ser utilizadas no produto de software para corrigir falhas, melhorar o projeto, implementar aprimoramentos (inovações incrementais), viabilizar conexões com outros softwares, adaptá-lo a mudanças ou novos ambientes, migrar softwares legados e para dar descontinuidade a softwares implantados e em funcionamento.

Para organizar essa ampla quantidade de ações de manutenção de software as requisições de modificações originadas de usuários são divididas em duas categorias e quatro tipos de manutenção.

Em estudos iniciais sobre o assunto Lientz & Swanson [1981] definiram três tipos de manutenção: corretiva, adaptativa e perfectiva. Mais tarde, a ISO/IEC 14764 [2006] atualizou essa lista incluindo a manutenção preventiva. E categorizaram os tipos adaptativos e perfectivos como ações de manutenção que visam aprimoramentos ou melhorias do produto de software, e os tipos corretivos e preventivos como categoria que objetiva correção desse produto.

Essas categorias e tipos de manutenção são apresentados na Figura 2.4[4] e descritos na seqüência, conforme está na ISO/IEC 14764 [2006].

[4]Esta figura foi extraída da ISO/IEC 14764 [2006] e adaptada (o texto foi traduzido para a língua portuguesa) para apresentação neste trabalho de pesquisa.

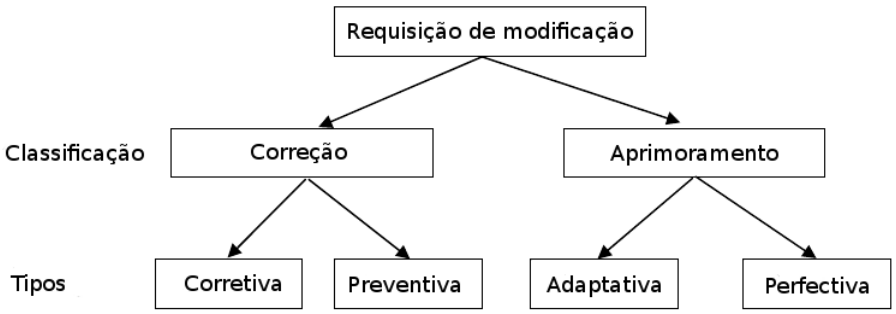

Figura 2.4. Categorias e tipos de manutenção de software [ISO/IEC 14764, 2006]

- Manutenção corretiva - alteração reativa de um produto de software realizada após sua entrega, cujo objetivo é corrigir problemas descobertos;
- Manutenção preventiva - modificação de um produto de software após a entrega, com a finalidade de detectar e corrigir falhas latentes no produto, antes que se tornem falhas efetivas. Essa manutenção é mais freqüentemente utilizada em produtos de software onde segurança é uma questão crítica;
- Manutenção adaptativa - modificação de um produto de software realizada após sua entrega, com objetivo de mantê-lo utilizável ao ocorrer alguma alteração ou mesmo mudança de ambiente;
- Manutenção perfectiva - manutenção que busca a perfeição do produto de software. Trata-se de modificar esse produto após a entrega, com o objetivo de melhorar o desempenho ou prevenir manutenção corretiva.

Em geral, observa-se que as ações de manutenção, que normalmente se iniciam em função de requisições manifestadas por usuários, objetivam melhorar a qualidade do produto de software, tanto ao corrigir falhas/erros quanto por necessidade de aprimoramentos e ou melhorias.

Nesse sentido, Godfrey & German [2008] relatam em seu trabalho que manutenção é uma forma de inovação de software, uma vez que visa à melhoria incremental do produto em uso. Segundo os autores, a mudança é uma característica essencial dos produtos de software e estes devem responder às exigências relacionadas a tecnologias, cultura e ambientes com a evolução. Esta evolução pode significar inovação originária, por exemplo, dos processos de manutenção e customização.

Por exemplo, ações de manutenção como alteração no projeto do produto de software com a finalidade de corrigir erros, adaptação deste produto para utilização em novos ambientes, adição de novas características ou mesmo a melhoria do projeto interno para facilitar gerenciamento e modificação, geram inovação incremental neste produto. E esta inovação impulsiona a mudança no software, pois surge um sistema com qualidade melhorada que evoluiu a partir do antigo sistema. Quando o conjunto de alterações se torna importante deve-se buscar uma nova visão do produto, ou seja, inovação radical [Godfrey & German, 2008].

De acordo com Higgins [2007], os temas manutenção de software e inovação são mais estreitamente relacionados do que se é comumente aceito. Segundo o autor, processos de inovação implicam a configuração de usuários, comunidades e artefatos através dos trabalhos de manutenção e customização, manifestos talvez como bricolagem, ou seja, atividades próprias (faça você mesmo) de usuários e comunidades que podem gerar inovações no produto. "Se esta interpretação alternativa de inovação é aceita, implica uma sensibilidade renovada para a pesquisa e o método alinhados com as definições de inovação, enfatizando as interpretações dos sujeitos, linguagem, percepção, comportamento e até mesmo a cultura. Isto tem implicações para o desenvolvimento de uma compreensão mais profunda e íntima em torno dos processos de desenvolvimento de software".

2.2 Inovação do produto de software

Buscando esclarecer à proposição de que qualidade, avaliação e manutenção de software podem levar a inovação deste produto, torna-se necessário a revisão de alguns dos principais conceitos relacionados a esta proposição, que são: inovação, formas de inovação e inovação de produto. É o que se pretende nesta seção sobre inovação de produto de software.

De acordo com Freire [2002], inovação é o processo de concepção e introdução de algo novo no contexto de produtos, organizações e mercado. Schumpeter [1942] detalhou este processo de concepção e introdução de novidades de cinco formas: i) Lançamento de novos produtos ou melhoria em produtos existentes; ii) Novas formas de produção através de inovações em processos, que normalmente têm impacto significativo nos custos de produção; iii) Abertura de novos mercados consumidores; iv) Novas fontes de suprimentos de matéria prima; e, v) Mudanças organizacionais.

As formas de inovação apresentadas por Schumpeter [1942] são relevantes para tratar principalmente da gestão tecnológica e organizacional, pois são direcionadas para a busca de melhoria de qualidade de produtos e/ou processos visando o aumento da competitividade de empresas no mercado [Tigre, 2006].

Numa abordagem popular, inovação significa novidade e renovação - idéias e ou invenções que chegam à realidade organizacional ou de mercado gerando valor técnico, social e ou econômico.

Sob o ponto de vista tecnológico e organizacional, segundo o Manual de Oslo [OECD, 1997] pode-se caracterizar as inovações nos seguintes tipos: i) Implantação de um novo processo ou a melhoria de um processo que já existe; ii) Um novo método organizacional nas práticas de negócios, na organização do local de trabalho ou nas relações externas; iii) Um novo método de marketing; ou, iv) Implementação de um produto (bem ou serviço) novo ou significativamente melhorado.

As inovações em processos representam a adoção de novos ou significativos métodos de produção ou distribuição por parte das organizações. E inovações organizacionais, por sua vez, referem-se à criação de novas estratégias organizacionais, tais como mudanças nas regras de negócio, na organização do local de trabalho ou nas relações externas da organização com seus parceiros de negócio - cliente e fornecedores.

As inovações em marketing envolvem a implementação de novos métodos de marketing, incluindo modificações no desenho do produto e/ou embalagem, na promoção do produto e sua colocação no mercado, e em métodos de estabelecimento de preços de bens e de serviços.

Finalmente, as inovações em produtos dizem respeito à concepção e colocação no mercado de um novo produto ou serviço com características melhoradas. Pode-se afirmar, preliminarmente, que as inovações em produto de software se enquadram no contexto das inovações de produto e ou serviço, uma vez que o software tem características que o enquadram no contexto não só de produtos, mas também de serviços.

Estes tipos de inovação - processos, organizações, marketing ou produtos - podem ser categorizados quanto ao grau de evolução, sendo considerada inovação incremental melhorias onde se mantém a mesma base tecnológica e inovação radical a criação de novidades sustentadas por bases tecnológicas diferentes e inovadoras [Rose, 2010].

Ainda segundo Rose [2010], na inovação incremental pode-se observar mudanças contínuas que visam possibilitar melhorias nas características ou funcionalidades de uma mesma plataforma tecnológica de produtos, por exemplo. Em outro patamar, as mudanças radicais da plataforma tecnológica levando à ruptura das trajetórias existentes e iniciando uma nova direção tecnológica configuram-se como uma inovação radical do produto.

O Manual de Oslo [OECD, 1997] ressalta ainda que dois aspectos interessantes relacionados à inovação, seja ela incremental ou radical, precisam ser observados. O primeiro deles é que inovação, por exemplo, a de produto, só é caracterizada como inovação quando o produto melhorado ou novo é criado e disponibilizado para consumo e uso. E o segundo aspecto é que a busca por inovações podem envolver a aquisição e a adaptação de softwares.

Neste segundo aspecto apontado pelo Manual de Oslo é interessante observar que o software além do fato de ser o alvo do processo de inovação em si, também constitui ferramenta tecnológica utilizada para propiciar inovações incrementais ou radicais em processos, organizações, marketing ou produtos, sendo relativamente comuns investigações e pesquisas que indicam a presença de softwares apoiando estes tipos de inovações. Considerando o primeiro aspecto, inovação do produto de software, pode-se afirmar que são escassas ou quase inexistentes investigações e contribuições a respeito do tema.

Segundo Lippoldt & Stryszowski [2009] e Rose [2010], inovação do produto de software, tema de estudo desta pesquisa, pode ser vista com a busca da melhoria em características ou funcionalidades de gerações anteriores deste produto - inovação incremental - ou a criação de um novo produto que será disponibilizado para consumo e uso no mercado - inovação radical.

Em geral, processos de inovação em software podem ser implementados e empregados de forma livre pelas organizações. Entretanto, Lippoldt & Stryszowski [2009], comentam sobre dois modelos de processos de inovação do produto de software que são complementares: o modelo tradicional, também chamado de linear, e o modelo colaborativo.

No modelo linear ocorre uma seqüência de eventos onde os inovadores podem ser vistos como agentes economicamente racionais motivados por um conjunto específico de metas. Na tentativa de atingir essas metas os inovadores empreendem inicialmente um processo de reconhecimento do potencial de inovação, seguida por escopo das idéias, pesquisa, desenvolvimento de protótipos, testes, introdução do novo produto ao mercado e por fim a comercialização.

Conforme Coelho [2006] algumas organizações de software hoje já se encontram suficiente maduras, capitalizadas e com um bom conjunto de processos "lineares" de desenvolvimento de novos produtos e ou inovações que envolvem questões como: análise do mercado (tamanho, oportunidade, geografia); análise de segmento alvo; análise das necessidades do cliente; modelo de negócio; tecnologias vigentes, possíveis arquiteturas e questões como patentes e legislação.

Entretanto, este não é o procedimento comum ou padrão das empresas de software. Os modos e formas muitas vezes são diferentes dos processos lineares. Segundo Coelho [2006] é comum encontrar, por exemplo, o desenvolvimento baseado em personalizações (também conhecido como customizações). Isto é, o desenvolvimento voltado à demanda dos clientes atuais, que pedem personalizações para diminuir deficiências do software ou para adaptá-lo melhor ao seu negócio.

Assim, a abordagem ou modelo aqui denominado colaborativo [Chesbrough, 2003; Tapscott & Williams, 2007] pode ser utilizado para reforçar o processo de inovação linear. Por exemplo, há casos em que um desenvolvedor de software pretende alavancar os esforços de inovação buscando conhecimento externo e ou trazendo capacidades para dentro do processo de inovação de produto de software. Alguns exemplos mais comuns incluem:

- Contratação de mão-de-obra qualificada para trabalhar em pesquisas relacionadas a inovação do produto de software;
- Parcerias ou contratos de pesquisas com universidades que gerem a criação de laboratórios e ambientes de alta tecnologia para realizar pesquisa em inovação;
- Alianças entre organizações (industriais, institutos, entre outras) para troca de conhecimentos que podem gerar inovação;
- Prospecção de novas idéias vindas de indivíduos ou comunidades de software.

Cabe ressaltar que Tapscott & Williams [2007] cunharam os termos como "peering, ideágoras, prossumers e novos alexandrinos" basicamente referindo-se a inovação originária de colaboradores, sejam eles clientes, consumidores ou usuários.

Especificamente com relação à inovação de produto de software, no estudo realizado por Lippoldt & Stryszowski [2009] desenvolvedores[5] de sistemas apontaram dezesseis fatores de alta importância para inovação deste produto. Dentre os cinco mais altos índices três estão relacionados ao envolvimento de usuários ou consumidores, a saber: requisitos de consumidores a respeito de qualidade, requisitos de consumidores a respeito de custos e requisitos de consumidores a respeito de segurança.

Ainda segundo Lippoldt & Stryszowski [2009], a inovação do produto de software pode ser, de fato, direcionada para necessidades e expectativas de usuários. Um fato relevante relacionado a isto acontece durante o desenvolvimento de software, quando freqüentemente projetistas e desenvolvedores, solicitam e observam opiniões de usuários ou consumidores a respeito do produto em desenvolvimento.

[5] Os desenvolvedores podiam indicar múltiplos fatores simultaneamente.

A inovação de produtos a partir da prospecção de idéias vindas de usuários ou comunidades de usuários está dentro do conjunto de estratégias de inovação conhecida na literatura por Inovação Aberta (*Open Innovation*) [Chesbrough, 2003].

Nesta estratégia pressupõe-se que as organizações produtoras podem utilizar fontes externas e internas de idéias a fim de aumentar sua competitividade na geração de novas tecnologias. Por exemplo, no sítio Web do projeto FIAT MIO[6], a FIAT® - empresa do setor automobilístico - esta aberta a depoimentos (comentários, opiniões, sugestões) de consumidores para concepção de um novo veículo.

A relevância desta participação de usuários no processo de inovação de produtos foi ressaltada por Von Hippel [1986]. Segundo este autor a maior parte das melhores idéias para melhorar a qualidade de um produto existente ou mesmo criar um novo produto origina-se através da contribuição dos consumidores ou usuários finais.

No setor de software o cenário não é diferente daquele adotado atualmente no setor automobilístico e entre outros setores. É possível observar que cada vez mais usuários - indivíduos, empresas e governos - têm um crescente papel de participação na condução de inovação do produto de software. Os conhecimentos e as experiências de usuários, adquiridas durante o uso dos sistemas, se transformam em necessidades e emergem como uma fonte promissora de entrada de informação útil no processo de inovação do produto de software [Lippoldt & Stryszowski, 2009].

Sendo as opiniões de usuários uma fonte de informação relevante para inovação do produto de software, entende-se como fator importante estudar formas de como coletar e analisar essas opiniões.

De acordo com Grützmann et al. [2010] as principais técnicas para coleta de dados de usuários ou consumidores de produtos envolvem pesquisas de mercado (questionários diretos ou via Web), observações técnicas e ou experiências práticas com grupos de foco. Estes e outros instrumentos visam levantar diretamente comentários, opiniões ou sugestões de usuários ou consumidores. Estas opiniões, sob a forma de dados, são posteriormente pré-processadas em busca de informações úteis para inovação de produtos.

[6]Disponível em: `http://www.fiatmio.cc/pt/ideias/`

Entretanto, algumas dessas alternativas técnicas apontadas por Grützmann et al. [2010] para coleta de opiniões de usuários visando inovação podem ter custo elevado, necessitar envolver contato quase que direto com os clientes (usuários) e às vezes levar muito tempo. Também, estas técnicas que abordam diretamente usuários podem condicionar o resultado final da pesquisa e ou não receber a adequada atenção dos mesmos, e, além disto, apresentar um custo relativamente alto quando aplicadas a produtos que tem alcance global, tal como o produto de software.

Neste contexto, conforme Santos & Brasil [2010], organizações como Adidas®, Google®, Skype®, Procter & Gamble®, Peugeot®, Volvo® e Microsoft® estão cada vez mais procurando envolver usuários no auxílio à geração de inovação em seus leques de produtos e serviços usando técnicas alternativas. Ferramentas tecnológicas como a *World Wide Web* - Web e outras relacionadas estão entre as preferências destas organizações, pois elas apresentam recursos de interação a baixo custo para organização e com grande conveniência para os usuários, além possibilitarem alcance global.

Neste cenário, onde se observa a relevância de opiniões de usuários de software para inovação de produto e a Web como um repositório destas opiniões, a tarefa que se apresenta a este trabalho de pesquisa é o estudo de técnicas para o levantamento destas opiniões na Web ou técnicas relacionadas. Enfim, técnicas que envolvam estas duas áreas de conhecimento, ou seja, Mineração de dados e Engenharia de Software e que possam auxiliar o entendimento e a prática de coleta e extração de opiniões de usuários visando inovação do produto de software.

A seguir apresenta-se uma revisão do tema Mineração de Dados direcionada para Web. E na última seção deste capítulo os trabalhos que relacionam inovação de produto de software e Mineração de Dados são apresentados.

2.3 Mineração de dados na Web

A Mineração de Dados (MD) é o processo de descoberta de conhecimento em banco de dados (*Knowledge Discovery in Databases*). Segundo Rezende [2005], este processo envolve um conjunto de ferramentas tecnológicas capazes de explorar registros de banco de dados em busca de informações que podem auxiliar na tomada de decisão. Estas informações podem ser exploradas e apresentadas por estas ferramentas em formas de regras, agrupamentos, classes, entre outras; que facilitem a visão e a tomada de decisão gerencial.

Ainda segundo Rezende [2005], o processo de MD é composto por uma seqüência de passos que envolvem principalmente definições iniciais, coleta e transformação de dados brutos em um formato adequado a entrada de algoritmos de mineração. A saída desses algoritmos pode ser o conhecimento que se buscava, ou alguma novidade ainda desconhecida, que pode ser utilizada na solução de algum problema. Mais especificamente cada passo desse envolve:

- Definições iniciais: inicialmente é preciso definir as classes de usuários que estarão envolvidos no processo de MD. Estes são principalmente especialistas que detém conhecimento sobre o domínio em análise, analista responsável pela execução dos passos técnicos da MD e usuário final que vai utilizar o conhecimento a ser descoberto. A definição seguinte, que pode ser tomada pelos usuários que participarão do processo, diz respeito aos objetivos e metas a serem alcançadas no processo de MD. Para isto os usuários precisam identificar o problema a ser investigado, estudar o domínio deste problema e utilizar o conhecimento gerado para traçar as metas e objetivos a serem alcançados;

- Coleta de dados: este passo envolve a coleta de dados de uma ou mais fontes, que especificamente na MD são registros de banco de dados;

- Pré-processamento dos dados: neste passo a coleção de dados formada no passo anterior passa pelas técnicas de MD que fazem limpeza, redução de dimensionalidade, indexação e transformação destes dados em um formato adequado a entrada do algoritmo de mineração a ser adotado;

- Extração do conhecimento: neste passo é selecionada a configuração e execução de um ou mais algoritmos de MD para extração do conhecimento que se está buscando nos dados pré-processados;

- Avaliação e aplicação do conhecimento: envolve a avaliação do conhecimento gerado no passo anterior e análise de sua aplicação na solução do problema inicialmente definido pelos usuários que participaram do processo de MD. Para detalhes específicos do processo de MD consultar Rezende [2005].

Este conjunto de ferramentas tecnológicas da MD foram criadas para descoberta de conhecimento em registros de banco de dados principalmente em razão de duas tendências: disponibilidade crescente deste tipo de dado e a demonstrada utilidade destas técnicas para encontrar soluções para problemas nestes tipos de dados [Xie et al., 2009].

Conforme Rezende [2005], posteriormente as tecnologias de MD foram adaptadas para descoberta de conhecimento inovador em documentos de texto, processo conhecido na literatura por Mineração de Texto (MT). A principal peculiaridade da MT está relacionada aos dados a serem processados (documentos de texto), que não possuem uma estrutura como aquela presente em registros de bancos de dados. Isto exige a tarefa adicional de criar uma estrutura para os mesmos.

A Mineração Web (*Web Mining*) é outra variação da MD, que está relacionada a mineração de dados presentes em páginas Web. Este tipo de mineração orientado à Web divide-se em três áreas de estudo: i) Mineração de estrutura Web - processo de descoberta de conhecimento baseada na organização topológica (estrutura) da Web e das referências cruzadas criadas por suas ligações (*links*); ii) Mineração de uso da Web - processo de descoberta de padrões de comportamento de uso da Web a partir dos registros de acesso (*logs*) de usuários; e iii) Mineração de conteúdo Web - processo de descoberta de conhecimento sobre um determinado assunto ou tópico em conteúdos de páginas Web [Rezende, 2005].

O presente trabalho situa-se na sub-área de mineração de conteúdo Web, visto que as opiniões de usuários de software em uso a serem exploradas estão presentes na Web, especificamente são conteúdos presentes em páginas Web, requerendo técnicas de Mineração de Texto para sua extração.

De acordo com Bodendorf & Kaiser [2010] a Mineração de Texto trata da análise de fatos contidos em textos e dentro deste tipo de mineração existe a variação mineração de opinião, que se concentra nas atitudes expressadas nos textos. Ainda segundo estes autores, classificação de sentimentos (*sentiment classification*) é um dos principais campos de pesquisa em mineração de opinião, cujo objetivo é utilizar algoritmos de classificação de textos para classificar documentos de acordo com sentimentos expressados no conteúdo textual [Pang et al., 2002; Santos et al., 2010].

Segundo Pang & Lee [2008], algoritmos de classificação de texto são uma tecnologia fundamental para aplicações de Mineração de Textos em geral. Pois, resolvem-se problemas de Mineração de Textos modelando estes em sub-problemas que podem ser mapeados para problemas de classificação de texto, assim como fez Pang et al. [2002] e Santos et al. [2010].

A afirmação de Pang & Lee [2008] sobre a aplicabilidade de algoritmos de classificação para o problema de Mineração de Texto direcionou os estudos sobre mineração de opiniões de usuários de software em páginas Web neste trabalho de pesquisa.

Em função disto, não se pretende realizar neste trabalho uma revisão completa sobre Mineração de Texto, como aquela apresentada por Carrilho Jr. [2007]. Ao contrário, pretende-se direcionar o estudo para as técnicas de Mineração de Texto suficientes para sustentar os procedimentos adotados no método para levantamento e avaliação de opiniões de usuários de software na Web, a ser proposto.

Então, torna-se tarefa importante investigar sobre coleta de páginas Web e em seguida, identificar quais técnicas de pré-processamento devem ser aplicadas às páginas recuperadas, a fim de colocá-las em uma formatação adequada à entrada do algoritmo de classificação de texto.

O algoritmo de classificação de texto, estudado após as técnicas de pré-processamento, desempenha a tarefa de avaliar e apresentar opiniões colocadas nas páginas Web, especificamente aquelas opiniões de usuários do produto de software que serão apresentadas como insumo no processo de inovação deste produto. Este direcionamento de estudos sobre Mineração de Textos foi fortemente influenciado pelo trabalho de Aranha [2007].

2.3.1 Coleta de páginas Web

De acordo com Soares [2008], o primeiro passo para descoberta de conhecimento em textos é a coleta ou recuperação de documentos para criação de uma coleção. Neste processo, a atenção deve estar voltada para a recuperação de documentos que dizem respeito ao assunto que está sendo pesquisado, para evitar que documentos sem relevância façam parte da coleção a ser formada e prejudiquem as próximas etapas da descoberta de conhecimento.

Em geral, documentos que vão formar a coleção podem ser recuperados principalmente de três fontes: arquivos textuais armazenados em computadores de usuários, tabelas dos mais variados tipos de bancos de dados e páginas Web.

A coleta de páginas Web, fonte de interesse desta pesquisa, é realizada de forma automática através de robôs conhecidos por *crawlers*. Esses robôs são configurados por meio de parâmetros e a partir de uma página inicial (semente) começam a visitar e coletar as demais páginas Web, utilizando as ligações (*links*) entre elas.

Um *crawler* pode recuperar todas as páginas que visita e criar uma coleção completa, ou, coletar somente aquelas páginas que atenderem aos pré-requisitos estabelecidos em sua configuração e criar uma coleção selecionada, sendo este último tipo conhecida por coleta focada.

Um bom exemplo de *crawler* configurável, capaz de fazer coletas focadas, é o software livre *GNU Wget*[7]. Esse *crawler* recupera arquivos na Web utilizando os protocolos mais amplamente utilizados nesse ambiente - *HTTP, HTTPS* e *FTP*. No exemplo abaixo, o *Wget* está configurado para iniciar a coleta completa do endereço Web especificado e guardar no diretório corrente (onde o *Wget* está sendo executado) toda a coleção de páginas recuperadas.

wget -r -np http:www.gnu.orgsoftarewget

O parâmetro "*-r*" garante que a recuperação será recursiva via ligações e o parâmetro "*-np*" não permite que esta recursão ultrapasse do diretório pai de onde se iniciou a coleta. Ademais, pode ser obtida uma lista bem completa de opções de configuração do *Wget* através de sua opção de ajuda. As principais, especificamente aquelas adotadas ao longo deste texto, estão disponíveis no anexo A deste trabalho de pesquisa.

Ao final da execução do exemplo acima o *Wget* terá criado uma coleção completa, fazendo o download da página principal do endereço eletrônico especificado e de todas as páginas que possuem ligações com esta e estão no mesmo diretório e nos diretórios de níveis abaixo. Após isto, a coleção está disponível para execução do próximo passo da mineração: o pré-processamento da coleção formada.

Adicionalmente, é possível observar na Tabela 2.1 outros exemplos de Web *crawlers*, proprietários e de código aberto, consultados na Web[8].

Tabela 2.1. Exemplos de *Web crawlers*

Nome	Comentário
WebFountain	Implementado pela IBM
arachnode.net	Web crawler de código aberto
Aspseek	Escrito sob licença GPL[9]

[7]Disponível em: `http://www.gnu.org/software/wget/`
[8]Disponível em: `http://en.wikipedia.org/wiki/Web_crawler`

2.3.2 Pré-processamento de páginas Web

A função geral desta etapa é prover à coleção de documentos criada no passo anterior uma formatação adequada para o processamento no algoritmo de mineração. Esta tarefa costuma ser bastante trabalhosa, pois não existe uma única técnica capaz de criar uma formatação satisfatória para toda a coleção formada.

Os documentos que compõem a coleção são páginas Web que podem se originar de várias fontes diferentes. Em função disto, podem conter documentos com estruturas bem diferentes entre si, exigindo análises e técnicas específicas para cada caso [Aranha, 2007].

Ainda, segundo Aranha [2007], é importante levar em consideração, também, ao buscar esta formatação adequada, qual será o objetivo da mineração que se pretende executar. Por exemplo, para documentos coletados na Web pode-se fazer uma limpeza visando remover *tags* do conteúdo destes documentos, deixando apenas o texto. Desde que estas estruturas, utilizadas para guardar informações semânticas ou estruturais de documentos Web, não sejam úteis no processo de mineração.

Por outro lado, Benbrahim & Bramer [2009] realizaram um trabalho de mineração utilizando algoritmos de classificação de texto onde as *tags* presentes no conteúdo dos documentos foram exploradas no processo, sendo as mesmas importantes para a tarefa de mineração.

Ademais, Aranha [2007] apresenta uma discussão bem completa sobre técnicas de pré-processamento, uma vez que sua pesquisa propõe justamente um novo modelo neste campo de estudo.

Para alcançar o objetivo deste trabalho, foi suficiente estudar três técnicas de pré-processamento: limpeza, indexação e geração de vetores de característica (*feature vector*). Estas técnicas serão descritas com base no trabalho de Aranha [2007].

São técnicas executadas na seqüência descrita por Aranha [2007] que geram uma representação textual dos documentos coletados na Web, tornando a coleção adequada à entrada do algoritmo de classificação a ser utilizado para a tarefa de mineração de opinião de usuários do produto de software.

Técnicas de limpeza visam remover dados do conteúdo dos documentos que não são considerados úteis para a tarefa de classificação de texto. Um processo de limpeza adequado está atrelado ao conteúdo do documento coletado e ao objetivo de mineração que se pretende implementar, conforme comentado anteriormente.

Para a tarefa de classificação de texto estudada nesta pesquisa foi necessário remover *tags*, pontuações, caracteres e quaisquer outras expressões presentes no conteúdo dos documentos da coleção, consideradas não úteis para o processo de mineração a ser adotado, que vai considerar apenas termos ou palavras.

As tabelas 2.2[10] e 2.3 ilustram o funcionamento de técnicas de limpeza em uma coleção exemplo, hipoteticamente recuperada na Web. Na tabela 2.2 os documentos ainda estão em seu formato original, logo após a coleta Web.

Tabela 2.2. Coleção exemplo de documentos recuperados na Web

Doc.	Conteúdo do *doc.*
1	<**html**><**body**>Pease porridge hot, pease porridge cold.</**body**></**html**>
2	<**html**><**body**><**p**>Pease porridge in the pot,</**p**></**body**></**html**>
3	<**html**><**body**><**h1**>Nine days # old.</**h1**></**body**></**html**>
4	<**html**><**body**>Some like it hot, some like it cold,</**body**></**html**>
5	<**html**><**body**><**p**>Some like % it in the pot!</**p**></**body**></**html**>
6	<**html**><**body**><**h1**>Nine days old.</**h1**></**body**></**html**>

Tabela 2.3. Coleção exemplo após etapa de limpeza

Doc.	Conteúdo do *doc.*
1	Pease porridge hot pease porridge cold
2	Pease porridge in the pot
3	Nine days old
4	Some like it hot some like it cold
5	Some like it in the pot

E a tabela 2.3 apresenta a mesma coleção de documentos após a etapa de limpeza. Observa-se que pontuações (, . !), caracteres (%, #) que se encontram no texto e são considerados não úteis, e *tags HTML* foram completamente removidos do conteúdo de cada documento da coleção exemplo. Restaram-se apenas as palavras ou termos.

Além disto, o documento de número seis foi descartado e a coleção ficou com cinco documentos. Após a limpeza, esse documento tornou-se exatamente igual ao documento número três e portanto não há necessidade de mantê-lo.

A limpeza propicia à coleção de documentos uma formatação adequada para realizar a etapa de indexação, que é o passo seguinte para se criar uma representação textual apropriada à entrada do algoritmo de mineração.

[10] Os exemplos apresentados para abstrair uma coleção de documentos foram extraídos do material de aula do Prof. Nivio Ziviani, e adaptados para ilustrar a etapa de pré-processamento de texto apresentado neste trabalho. Este material está disponível em: `http://homepages.dcc.ufmg.br/~nivio/cursos/ri08/transp/indexing.pdf`

A indexação é uma técnica que organiza o conteúdo dos documentos da coleção facilitando o seu acesso e recuperação. Para isto, é necessário selecionar uma unidade textual e, em seguida, criar uma estrutura de índices para ela, que possibilite rapidez e agilidade no processo de busca, tal como funciona o índice de um livro.

Esta unidade textual são as características (*features*) dos documentos da coleção. Elas podem ser sílabas, termos ou palavras, frases, ou qualquer outra unidade semântica e/ou sintática passível de indexação, utilizada para representar o conteúdo textual de cada documento da coleção.

De acordo com Benbrahim & Bramer [2009], a unidade textual mais amplamente utilizada para indexação do conteúdo de documentos são os termos ou palavras. Esta abordagem é puramente estatística, pois a seqüência em que os termos aparecem em um documento e toda a estrutura do texto é ignorada, ou seja, o documento transforma-se num saco-de-palavras (*bag-of-words*).

Ademais, Benbrahim & Bramer [2009] argumentam que esta abordagem não leva em consideração a semântica da linguagem, mas, em contrapartida é computacionalmente muito eficiente. E apresentam referência a outras abordagens diferentes desta para representação textual.

O mecanismo usual para construção de um índice orientado à termos é o arquivo invertido (*inverted file*), conhecido também por *Inverted index*. Esta estrutura de dados é composta de duas partes:

i. Um vocabulário contendo todos os termos distintos encontrados no conteúdo textual dos documentos da coleção; e,

ii. Para cada termo do vocabulário, uma lista invertida de todos os documentos (identificados por números) onde o termo ocorre.

Nesta segunda parte pode-se, por exemplo, indicar o documento onde o termo ocorre e também a freqüência dele no documento. Dessa maneira, cada posição das listas invertidas é formada pelo par:

$$(d, f_{d,t})$$

Onde d é o documento e f é a freqüência do termo t no documento d [Baeza-Yates & Ribeiro-Neto, 1999].

A Tabela 2.4 apresenta o vocabulário de termos distintos construídos a partir da coleção de documentos utilizada como exemplo, e a Tabela 2.5 é a representação de um arquivo invertido e suas listas, criado a partir do dicionário e dos arquivos da coleção exemplo.

Tabela 2.4. Vocabulário de termos distintos da coleção exemplo

ID do termo	Termo
1	pease
2	porridge
3	in
4	the
5	pot
6	nine
7	days
8	old
9	some
10	like
11	it
12	hot
13	cold

Tabela 2.5. Arquivo invertido criado a partir do vocabulário e coleção exemplo

ID do termo	$(d, f_{d,t})$
1	(1,1) (4,2)
2	(1,1) (4,2)
3	(1,1) (5,1)
4	(1,1) (5,1)
5	(1,1) (5,1)
6	(2,1)
7	(2,1)
8	(2,1)
9	(3,2) (5,1)
10	(3,2) (5,1)
11	(3,2) (5,1)
12	(3,1) (4,1)
13	(3,1) (4,1)

A indexação organiza e provê acesso ao conteúdo textual de cada documento da coleção de forma rápida. Isto facilita o processo de criação dos vetores de características ou *features vectors*, que são a representação textual adequada para entrada da coleção de documentos no algoritmo de mineração.

A representação em forma de vetores de características pode ser o conjunto de todos os termos (que são as características ou *features* adotadas neste trabalho de pesquisa) do documento - representação completa - ou pode ser também um subconjunto de todos os termos do documento - representação parcial.

Consegue-se criar uma representação parcial utilizando técnicas de *feature reduction* - *feature selection* e *feature extraction* - cujo objetivo é reduzir a dimensionalidade da coleção inicial de documentos [Yang & Pedersen, 1997; Forman, 2003; Benbrahim & Bramer, 2009].

Em especial, podem ser destacadas duas técnicas de redução de dimensionalidade relacionadas a mineração baseada em algoritmos de classificação de texto:

- Remoção de *stop-words*: consiste em remover palavras ou termos comuns, que se repetem muito no conteúdo textual dos documentos da coleção (pronomes, conjunções, entre outros);
- Remoção de palavras ou termos que aparecem em dois ou menos documentos da coleção. Estas palavras não são representativas na mineração baseada em classificação de texto [Martins Jr., 2003].

De acordo com Lan et al. [2009], para representação do conteúdo dos documentos da coleção - completa ou parcial - normalmente utiliza-se o modelo espaço vetorial (*Vector Space Model*). Neste modelo cada documento é representado como um vetor de termos no espaço euclidiano t-dimensional, onde cada termo possui um valor (peso - do inglês, *weight*) associado que indica o grau de importância deste no documento. Formalmente isto pode ser escrito como:

$$d=(t_1w_1,\ t_2w_2,\ ...,\ t_kw_k)$$

Em que w_i é um indicador (peso) que representa o quanto o termo t_i contribui na representação do documento d. E k é o tamanho do conjunto de termos t pertencentes à d.

Ainda, segundo Lan et al. [2009], o peso w_i atribuído a cada termo pode ser a medida *tf-idf*, utilizada para representar conteúdo textual de documentos que serão submetidos para processamento em algoritmos de mineração.

O *tf-idf* é uma das medidas mais amplamente utilizadas para se representar o conteúdo textual de documentos. Especificamente, o *tf* representa o número n de vezes que o termo i aparece no documento j, conforme Equação 2.1:

$$tf = n_{ij} \tag{2.1}$$

Já o *idf* é obtido como apresentado na Equação 2.2, dividindo-se o número D de todos os documentos da coleção pelo número de documentos que contêm o termo.

$$idf = \frac{D}{d \supset t_i} \tag{2.2}$$

Por fim, o *tf-idf* é obtido através da Equação 2.3:

$$tfidf = tf * log(idf) = n_{ij} * log\left(\frac{D}{d \supset t_i}\right) \tag{2.3}$$

que representa uma medida estatística utilizada para avaliar o quão importante é cada termo de um documento em uma coleção de documentos. A relevância do termo aumenta proporcionalmente ao número de vezes que ele aparece no documento *tf*, mas isso é compensado pela freqüência dele nos documentos da coleção *idf* [Benbrahim & Bramer, 2009].

A Tabela 2.6[11] apresenta a representação textual da coleção de documentos inicialmente apresentada na Tabela 2.2. Os documentos passaram por técnicas de limpeza, indexação baseada em termos/palavras e por fim por uma transformação em vetores de características, onde se utilizou o peso *tf-idf* para representar o quão significante é cada termo do documento na coleção.

Tabela 2.6. Representação textual da coleção exemplo utilizando o esquema *tf-idf*

Doc.	Termo:Peso
1	1:0.79588001734408 2:0.79588001734408 12:0.39794000867204 13:0.39794000867204
2	1:0.39794000867204 2:0.39794000867204 3:0.39794000867204 4:0.39794000867204 5:0.39794000867204
3	6:0.69897000433602 7:0.69897000433602 8:0.69897000433602
4	9:0.79588001734408 10:0.79588001734408 11:0.79588001734408 12:0.39794000867204 13:0.39794000867204
5	3:0.39794000867204 4:0.39794000867204 5:0.39794000867204 9:0.39794000867204 10:0.39794000867204 11:0.39794000867204

Após as etapas de limpeza, indexação e criação dos vetores de características, a coleção de documentos formada após a coleta Web encontra-se pronta para mineração, que especificamente neste estudo é baseada em algoritmos de classificação de texto.

[11] A representação textual apresentada nesta tabela obedeceu àquela requerida pelo algoritmo de mineração *SVM-Light*, estudado na próxima seção.

2.3.3 Mineração baseada em classificação de texto

De acordo com Sebastiani [2002], a classificação de texto é uma forma de organizar informações que permitem entender e interpretar melhor dados. Para ilustrar este conceito considere, por exemplo, uma coleção onde há documentos com críticas a respeito de um produto de software. Assuma que é tarefa importante separar esses documentos em duas classes: críticas ruins e críticas boas; para alguma tomada de decisão.

Este problema pode ser modelado como um problema de classificação de texto, com duas classes - críticas ruins e críticas boas - a respeito do produto de software em análise. Problemas de classificação como este podem ser formalmente definidos como a seguir:

> **Classificação de texto:** Dado um conjunto D de documentos e um conjunto $C=\{c_1, c_2, c_3, ..c_L\}$ de L classes, com seus respectivos rótulos, o classificador de texto é uma função binária $F : D \times C \rightarrow \{0, 1\}$. Essa função atribui 0 ou 1 para cada par $[d_j, c_p]$, tal que $d_j \in D$ e $c_p \in C$. Se o valor atribuído é 1 diz-se que o documento d_j é membro da classe c_p. Se o valor atribuído é 0, diz-se que o documento d_j não é membro da classe c_p [Sebastiani, 2002].

Ainda segundo Sebastiani [2002] a maioria dos algoritmos adotados para solução do problema de classificação de texto é desenvolvida por pesquisadores da área de Inteligência Artificial, que trabalham especificamente com aprendizagem de máquina.

As pesquisas na área de aprendizagem ocupam-se com o projeto e desenvolvimento de algoritmos capazes de aprender padrões de dados para, em seguida, baseando-se no aprendizado, tomar decisões de forma automática, para dados desconhecidos submetidos para processamento.

Em geral, há três tipos de algoritmos de aprendizagem que desempenham esta função: supervisionados, não-supervisionados e semi-supevisionados [Gonçalves, 2009]. No entanto, esse autor argumenta que para obter uma classificação de texto automática com maior precisão é importante utilizar algoritmos de aprendizagem de máquina supervisionada.

A classificação supervisionada de texto, refere-se à construção de classificadores através de processos indutivos. Em um processo indutivo, também chamado de aprendiz (*learner*), constrói-se automaticamente um classificador para uma classe C_i "observando" as características de um conjunto de documentos.

Estes documentos são previamente e manualmente classificados sob C_i por especialistas no domínio em investigação e compõem o conjunto de treinamento, formalmente definido como:

> **Conjunto de treinamento:** Dado uma sub-coleção D_t contida em D de documentos, uma função de conjunto de treinamento $T : D_t \times C \rightarrow \{0, 1\}$ atribui valor 0 ou 1 para cada par $[d_j, c_p]$, $d_j \in D_t$ e $c_p \in C$, de acordo com o julgamento de especialistas humanos.

A função do conjunto de treinamento T é usada para realizar um ajuste fino no algoritmo de classificação de texto, melhorando a precisão da função de classificação $F : D \times C \rightarrow \{0, 1\}$. Entretanto, deve-se ter o cuidado durante este ajuste para que o algoritmo de classificação não se torne específico demais e reflita apenas as características do conjunto de treinamento, fenômeno conhecido por *overfitting*[12].

A Figura 2.5[13] apresenta a seqüência de etapas necessárias para se realizar uma classificação binária de documentos, utilizando algoritmo de aprendizagem supervisionada.

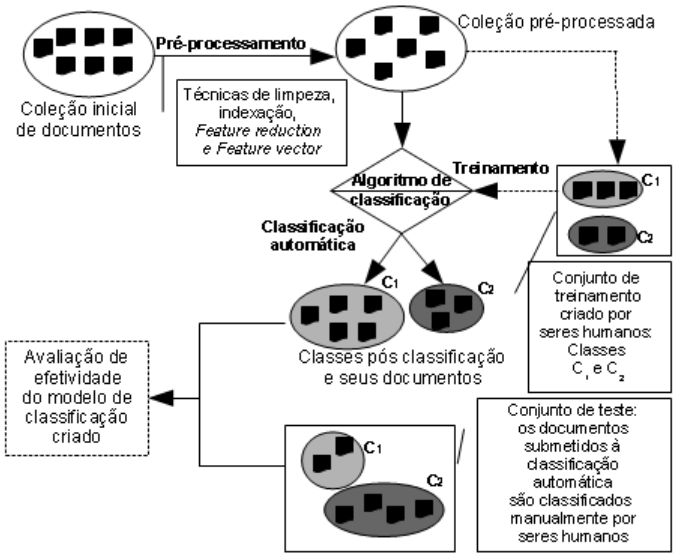

Figura 2.5. Processo supervisionado de classificação de documentos [Gonçalves, 2009]

[12]O modelo ou classificador gerado no treinamento de algoritmos de aprendizagem supervisionada não é capaz de generalizar o que aprendeu, pois passa a ser um mero mecanismo de representação do conjunto de treino.

[13]Esta figura foi criada pelo autor deste trabalho de pesquisa, mas fundamentada na Figura 5.4 apresentada por Gonçalves [2009].

Em seu trabalho, Gonçalves [2009] apresenta detalhadamente o funcionamento dos principais algoritmos de aprendizagem supervisionada utilizados para classificação de textos, que incluem *Decision Trees*, *Nearest Neighbors*, *Rocchio*, *Relevance Feedback*, *Naive Bayes*, *Support Vector Machines* e *Ensemble*.

Em função do escopo relacionado ao tempo de execução deste trabalho de pesquisa, o algoritmo *Support Vector Machines* (*SVM*) foi selecionado para seqüência dos estudos, de forma a propiciar uma base de conhecimento necessária a mineração de opinião de usuários de software em uso em páginas Web.

Optou-se por estudar o *SVM* para a tarefa de mineração, dentre as demais opções apresentadas por Gonçalves [2009], em função dos bons resultados alcançados por este algoritmo na solução de problemas similares aos apresentados nesta pesquisa [Joachims, 1998; Pang et al., 2002; Gonçalves & Quaresma, 2003; Gamon, 2004; Santos et al., 2010]. Além disto, segundo Joachims [1998], o *SVM* é baseado no modelo espaço-vetorial, método amplamente utilizado para representação textual de documentos a serem minerados que foi estudado na seção anterior.

De acordo com Chen et al. [2010], mesmo problemas de classificação de texto que envolvem muitas classes (*Multi-class classification*) que atualmente representam um problema de pesquisa na área de computação, podem ser resolvidos através de classificação binária utilizando o *SVM*. Para isso pode-se, por exemplo, decompor um problema *Multi-class* com n classes em n sub-problemas de classificação binária e solucioná-los um por vez.

A explicação deste algoritmo é fornecida a seguir, e é fortemente baseada no trabalho de Gonçalves [2009].

Suport Vector Machine - SVM

O *SVM* foi introduzido por Cortes & Vapnik [1995], mas quem primeiro utilizou esse algoritmo para solução do problema de classificação de texto foi Joachims [1998].

Este algoritmo utiliza o método espaço-vetorial para a solução do problema de classificação binária de texto. Neste método, os termos ou palavras indexados compõem um espaço t-dimensional onde os documentos são representados como pontos ou vetores no espaço.

Dada esta representação de vetores de termos para os documentos da coleção, o *SVM* encontra uma superfície de decisão, conhecida por hiperplano, que é utilizada para separar da melhor forma possível os documentos em duas classes, C_1 e C_2, por exemplo.

Este hiperplano é aprendido pelo *SVM* no treinamento e divide o espaço em duas regiões. Documentos que pertencem à classe C_1 ficam em uma região, e os que pertencem à classe C_2 ficam em outra região do espaço. E, para cada novo documento d da coleção, o *SVM* computa a posição relativa do mesmo em relação ao hiperplano, e toma a decisão de classificação, colocando d na região do espaço correspondente a C_1 ou C_2.

Em um espaço de duas dimensões, os documentos são separados linearmente, ou seja, o hiperplano é uma linha, como apresenta a Figura 2.6[14]. De todas as possíveis linhas que separam os documentos em duas classes, a linha "s" é aquela que maximiza a distância entre os documentos mais próximos das duas classes e constitui-se o melhor hiperplano de separação ou hiperplano de decisão.

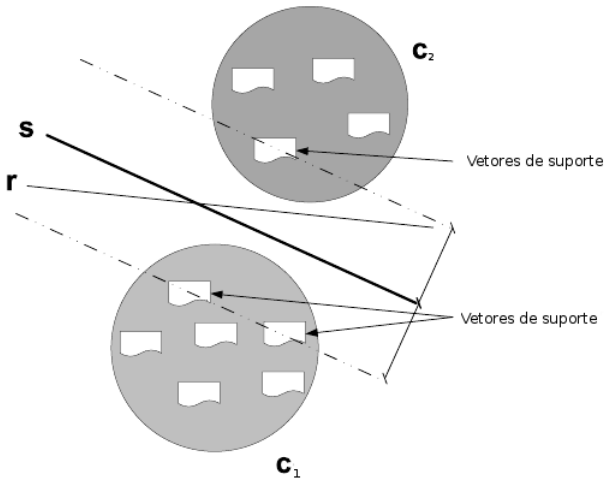

Figura 2.6. Hiperplano de decisão em um espaço bidimensional [Gonçalves, 2009]

Ainda observando a Figura 2.6, vêem-se as linhas paralelas tracejadas que delimitam a região onde é preciso procurar um hiperplano de decisão. Estas linhas são chamadas de hiperplano delimitador.

Documentos que pertencem ao hiperplano delimitador são chamados de vetores de suporte e as linhas que estão neste espaço delimitado são candidatas a hiperplano de decisão. Em geral, as linhas paralelas aos hiperplanos delimitadores são as melhores candidatas.

[14]Esta figura foi extraída de Gonçalves [2009] e adaptada (o texto foi traduzido para a língua portuguesa) para apresentação neste trabalho de pesquisa.

O *SVM* é diretamente aplicável à classificação de texto quando os documentos de uma dada coleção são representados por vetores t-dimensionais ponderados. Segundo Joachims [1998], este algoritmo é apropriado para a classificação de texto, uma vez que trabalha muito bem com grandes dimensões de dados apresentando baixo *overfitting*.

O *SVM-Light*[15] é uma implementação do algoritmo *SVM* disponível para uso. Esta implementação requer que a representação textual de cada linha do documento da coleção a ser treinada/classificada tenha a seguinte estrutura em arquivo:

<line> = <target> <feature>:<value> <feature>:<value> ... <feature>:<value>

Nesta estrutura, a linha do arquivo representa um documento da coleção, onde o *<target>* pode ter o valores {1, -1, 0}. Os valores {1, -1} são as marcações que objetivam distinguir documentos que representam classes opostas. Eles são utilizados para especificar a polaridade dos documentos no conjunto de treinamento. Por exemplo, o valor {1} refere-se a documentos que pertencem à classe C_1 e {-1} à classe C_2.

O valor {0}, por sua vez, é utilizado para representar os documentos ainda desconhecidos que serão classificados pelo algoritmo de forma automática, após o treinamento.

Já o par *<feature>:<value>* representa a característica (que neste trabalho é o termo ou palavra) a ser explorada na mineração e o seu respectivo peso (relevância do termo no documento/coleção, por exemplo o *tf-idf*).

Dois detalhes precisam ser observados ao implementar essa estrutura em arquivo:

- O par *<feature>:<value>* de cada documento da coleção precisa ser criado por ordem crescente do valor da *feature* ou característica. A ordem da *feature* é aquela apontada no vocabulário, por exemplo, aqueles identificadores na primeira coluna da Tabela 2.4;

- É preciso gerar um arquivo, cujo nome precisa ser "*words*", contendo todas as *features* ou características dos documentos da coleção. Cada característica deve estar em uma linha e na ordem em que aparecem no arquivo de representação textual do documento.

Na prática o arquivo "*words*" é similar ao vocabulário criado durante o pré-processamento exemplificado na Tabela 2.4. Porém, este arquivo possui apenas uma coluna, onde se coloca a *feature* ou característica utilizada na abordagem de representação textual.

[15]Disponível em: `http://svmlight.joachims.org/`

Ao observar a Tabela 2.6, onde a característica e o peso (<*feature*>:<*value*>) selecionados foram o termo e o *tf-idf* de cada termo, respectivamente, pode-se constatar que aquela representação atende à requerida pelo *SVM-Light*. Sendo necessário, apenas, definir o campo <*target*> de cada arquivo da coleção, ou seja, definir quais documentos da coleção serão utilizados para treinamento e quais serão classificados após o treino.

Ademais, o sítio do *SVM-Light*, além de oferecer esse algoritmo para download fornece várias informações a respeito do seu uso.

Para finalizar a apresentação completa de um processo mineração baseado em classificação de texto, a Figura 2.5 ilustra também o conceito de avaliação de efetividade do algoritmo adotado. Esta parte do processo de classificação permite avaliar/validar se a mineração alcançou os objetivos planejados, ou seja, se o conhecimento minerado é aquele esperado.

2.3.4 Avaliação e aplicação do conhecimento

A avaliação de mineração baseada em algoritmos supervisionados é feita comparando resultados gerados automaticamente pelo algoritmo com os resultados gerados manualmente por seres humanos.

Especificamente na mineração baseada em classificação, a classificação humana é diferente da classificação gerada automaticamente pelo algoritmo, conforme ilustra a Figura 2.5. Quanto maior é a concordância entre os resultados produzidos automaticamente e aqueles produzidos pelo homem, maior é a precisão do classificador ou modelo de classificação gerado no treinamento do algoritmo.

Na prática, a avaliação do classificador ou modelo de conhecimento gerado no treinamento é comumente realizada sobre o conjunto de exemplos. Rezende [2005] apresenta a Figura 2.7 para ilustrar a idéia desses conjuntos e relata que o mundo real apresenta uma distribuição de exemplos D em um dado domínio, a qual é desconhecida. Ao extrair exemplos do mundo real, formando assim um conjunto de exemplos, obtém-se uma distribuição de exemplo D', a qual é supostamente similar à distribuição D. De modo a estimar uma medida, geralmente a precisão ou o erro, de indutores treinados com base na distribuição D', extraem-se amostras a partir de D', treina-se um indutor com essas amostras e testa-se seu desempenho em exemplos de D' (normalmente com exemplos fora da amostra utilizada para treinamento). Desta maneira, simula-se o processo de amostragem que ocorre no mundo real, assumindo que D' representa o mundo real.

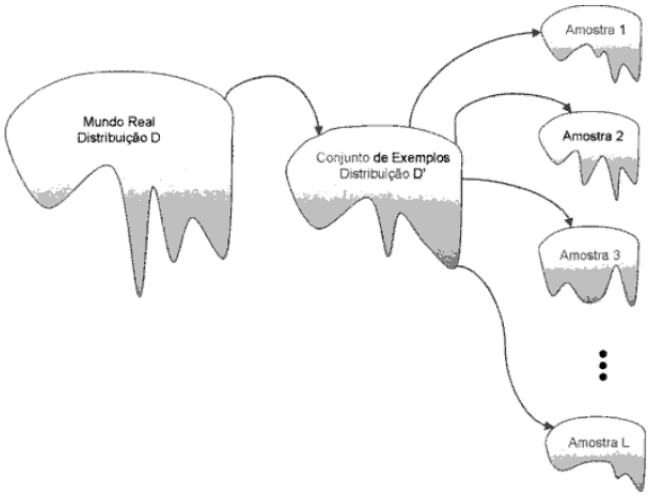

Figura 2.7. Conjunto de exemplos de uma distribuição *D'* [Rezende, 2005]

Ainda segundo Rezende [2005] os métodos mais amplamente utilizados para a prática de avaliação da efetividade de algoritmos de classificação são o *hold-out* e o *cross-validation*.

No *hold-out*, um conjunto teste de tamanho predefinido é aleatoriamente selecionado e o restante dos documentos do conjunto exemplo é utilizado para treinamento e geração do modelo. Uma variação deste método é a estratégia de utilizar 1/3 para o conjunto de teste e 2/3 para o conjunto de treinamento, ambos definidos aleatoriamente. E repetir este processo três vezes sobre o mesmo conjunto exemplo [Linden, 2008].

Já no método *cross-validation*, o conjunto exemplo é dividido em tantos conjuntos quanto o número de realizações da fase de teste e, em cada iteração, um deles é selecionado para teste e os demais formam o conjunto de treinamento.

Uma variação deste método, que também é chamado de *k fold cross-validation*, é o *10-fold cross-validation* [Rezende, 2005]. Neste método, o conjunto exemplo é dividido em dez partes iguais. Treina-se o algoritmo com nove partes e testa-se com uma. Este processo é repetido dez vezes, alternando o conjunto de teste.

Tanto no *hold-out* como no *cross-validation* o conjunto de teste é utilizado para estimar a qualidade média do modelo gerado no treinamento. Para isto utilizam-se métricas, tais como *Precision* (Precisão), *Recall* (Abrangência) e *Accuracy*, apresentadas nas equações a seguir:

$$Precision = \frac{tp}{tp + fp} \tag{2.4}$$

$$Recall = \frac{tp}{tp + fn} \tag{2.5}$$

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{2.6}$$

A Tabela 2.7 apresenta as definições de cada uma das variáveis presentes nas equações 2.4, 2.5 e 2.6 no contexto de classificação de documentos, permitido entender a explicação de cada uma destas equações que acontece na seqüência.

Tabela 2.7. Definições das variáveis presentes nas equações 2.4, 2.5 e 2.6

Variável	Definição
tp	Documento predito como positivo e é um exemplo positivo
fp	Documento predito como positivo mas é um exemplo negativo
tn	Documento predito como negativo e é um exemplo negativo
fn	Documento predito como negativo mas é um exemplo positivo

A *Precision*, Equação 2.4, mede o percentual das previsões positivas que são corretas para uma categoria C_i, levando em consideração os documentos que foram preditos como positivos que na verdade eram negativos (*fp*, na Tabela 2.7).

O *Recall*, Equação 2.5, mede o percentual de documentos classificados como positivos que foram previstos como positivos para uma categoria C_i, levando em consideração os documentos preditos como negativos que na verdade eram positivos (*fn*, Tabela 2.7).

E, finalmente, a *Accuracy* mede o percentual de previsões que foram corretamente classificados sob C_i. Esta é a métrica normalmente utilizada para avaliar se o modelo ou classificador criado para C_i é válido ou precisa ser revisto. Por exemplo, Martins Jr. [2003] utilizou esta métrica para avaliar os modelos criados com o *SVM-Light* para classificar páginas Internet em: documentos que contém descrição de produtos a venda *versus* documentos que não possuem essa informação.

Especificamente para esta categoria, Martins Jr. [2003] relata que o *SVM-Light* alcançou resultados de *Accuracy* que variaram entre 76,92% à 100% nos vários testes realizados.

É interessante observar que a implementação do *SVM-Light* apresenta na saída, após a classificação, as métricas *Precision*, *Recall* e *Accuracy*, do conjunto que está sendo testado. Para isto, a coleção a ser classificada deve estar classificada manualmente pelos especialistas no domínio em análise, ou seja, o campo <*target*> precisa ser definido com o valor "-1" ou "1" e não "0".

Mediante esta situação, o *SVM-Light* faz a classificação automática, compara-a com a classificação humana feita pelos especialistas e apresenta os valores das métricas *Precision*, *Recall* e *Accuracy* na saída padrão, permitindo a avaliação dos resultados.

Ainda segundo Martins Jr. [2003], esta fase de avaliação do conhecimento ou resultados alcançados é uma oportunidade para detectar possíveis erros em alguma etapa do processo de mineração. Ou mesmo estudar se a aplicação de outras técnicas poderia fornecer melhores resultados.

Se fatos como estes são verdadeiros, os profissionais que conduzem o processo de mineração precisam retornar às etapas anteriores - Coleta, Pré-processamento e Treinamento/Classificação - e realizarem as intervenções necessárias. Em seguida, realizar novos testes a partir de conjuntos exemplos e analisar os resultados gerados.

Este processo de análise do conhecimento, ou resultados gerados, é cíclico, e visa alcançar melhores resultados. Ao finalizá-lo, entende-se que os resultados alcançados na mineração são válidos e podem ser utilizados para aplicação prática.

2.4 Trabalhos relacionados

Esta seção tem como objetivo apresentar o histórico e a evolução de uso da ferramenta tecnológica Mineração de Dados no auxílio a tarefas ou problemas da área de Engenharia de Software (ES), principalmente aquelas relacionadas a inovação do produto de software envolvendo contribuição de usuários destes produtos.

De acordo com Hassan & Xie [2010b], a ferramenta tecnológica Mineração de Dados tem sido muito utilizada para auxílio a tarefas, problemas ou busca de conhecimentos ainda não revelados em dados da área de Engenharia de Software, tais como aqueles relacionados a inovação do produto de software. Assim, os autores denominam de *Software Intelligence* (SI) o uso de Mineração em Dados no contexto da Engenharia de Software (ES), tema de pesquisa crescente nos últimos anos.

Software intelligence tem a mesma finalidade do *Business Inteligence* (BI), só que voltado para o suporte às tomadas de decisões relacionadas a software. Enquanto no BI as organizações utilizam ferramentas de suporte à decisão para obter melhores resultados nos negócios, SI oferece aos mantenedores de produtos de software informações atualizadas e pertinentes sobre este produto, visando apoiar as decisões diárias que precisam ser tomadas no sentido de inovação do mesmo.

Nesta linha de estudos, Xie et al. [2009] criaram uma metodologia de Mineração de dados, ilustrada na Figura 2.8[16], cujo objetivo é auxiliar tarefas de ES tais como aquelas relacionadas a inovação incremental ou radical do produto de software.

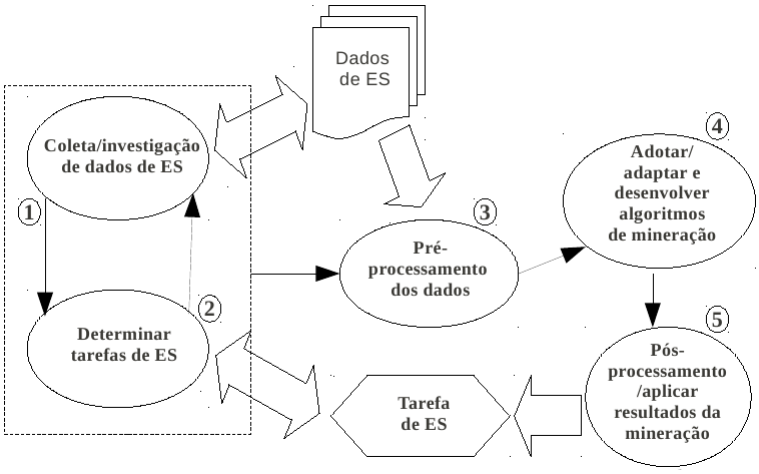

Figura 2.8. Metodologia para mineração de dados de Engenharia de Software [Xie et al., 2009]

Os dois primeiros passos desta metodologia envolvem adotar uma abordagem direcionada para os dados - quando se sabe quais dados de ES minerar - ou uma abordagem direcionada para a tarefa ou problema - quando se sabe qual problema dentro da ES deseja-se auxiliar - para aplicação da mineração de dados.

Para Xie et al. [2009], na prática, pode-se trabalhar com as duas abordagens simultaneamente, ou seja, definir o problema de ES que se pretende abordar e iniciar a coleta de dados onde se pretende buscar soluções para este problema. Os três passos seguintes desta metodologia são o pré-processamento dos dados, a adoção/adaptação/desenvolvimento do algoritmo de mineração e por fim a proposta de aplicação dos resultados encontrados na mineração à tarefa de ES a ser auxiliada.

A etapa de pré-processamento envolve extrair dados relevantes dos dados brutos coletados na etapa inicial. As técnicas aplicadas nesta etapa visam selecionar aqueles dados considerados úteis para as próximas etapas e descartar os demais. A saída dessa etapa é a entrada para o algoritmo de mineração.

De um modo geral, os algoritmos de mineração utilizados para processamento dos dados originados da etapa anterior estão em quatro categorias principais:

- Mineração de padrões freqüentes - os algoritmos dessa categoria visam encontrar ocorrências de padrões comuns em dados;

[16]Esta figura foi extraída de Xie et al. [2009] e adaptada (retirou-se as cores da figura e o texto foi traduzido para a língua portuguesa) para apresentação neste trabalho de pesquisa.

- Casamento de padrão - esses objetivam encontrar instâncias de dados para um determinado padrão;
- Agrupamento ou *clustering* - esses algoritmos agrupam dados semelhantes; e,
- Classificação - categoria de algoritmos que fazem predição de rótulos de dados baseando-se em dados já rotulados.

O passo final desta metodologia visa transformar o resultado encontrado na saída do algoritmo de mineração em um formato apropriado para auxílio ao problema de ES definido inicialmente.

Uma parte considerável do trabalho apresentado por Xie et al. [2009] concentra-se em estudar os algoritmos de mineração utilizados na metodologia apresentada na Figura 2.8. Em especial, esses autores relatam os tipos mais comuns de algoritmos para Mineração de Texto (MT), que incluem agrupamento (*clustering*), casamento de padrões (*pattern matching*) e classificação de texto.

Por fim, Xie et al. [2009] relatam a implementação de uma aplicação prática a partir da metodologia ilustrada na Figura 2.8 , realizada por Wang et al. [2008]. Esta aplicação é fundamentada em técnicas de mineração de textos para auxiliar o problema de detecção de relatórios de *bugs*[17] duplicados. Os dados para os testes experimentais foram coletados no *Firefox bug repository*, repositório que recebe relatórios de *bugs* de usuários deste produto. O funcionamento desta aplicação dá-se quando um novo relatório, ainda desconhecido, é enviado por usuários do software em uso. Este relatório é recebido e sua similaridade é comparada com os relatórios anteriormente recebidos e já analisados.

Um novo relatório que possui alta similaridade com outro já recebido e analisado é considerado como duplicado e é descartado. Segundo Wang et al. [2008], a aplicação conseguiu valores de *Accuracy* que variaram de 67% a 93%, quando decidiu automaticamente se os novos relatórios eram duplicados ou não.

Os autores ressaltam dois pontos importantes relacionados a participação de usuários na aplicação de detecção de relatórios de *bugs* duplicados. Primeiro, os usuários revelam problemas no software em uso e essa informação pode ser utilizada como insumo para propiciar inovação de produto. Segundo, mantenedores têm a oportunidade de inovar a partir de requisitos de usuários, quando estes revelam problemas no produto - requisitos que refletem necessidades e expectativas.

[17] Falha ou problema descoberto no produto de software.

Pode-se afirmar que a evolução do uso de mineração de dados para auxílio a problemas de ES deu-se principalmente a partir da Mineração de Repositórios de Software (*Mining Software Repositories*), tais como o *Firefox bug repository* utilizado por Wang et al. [2008].

De acordo com Hassan & Xie [2010b], os principais trabalhos de mineração de dados voltados para auxílio a tarefas ou problemas de ES, surgiram nos últimos quatro anos a partir de mineração em repositórios de software. Diante de uma quantidade significativa de dados e com visão de inovação, um grupo de pesquisadores iniciou pesquisas sobre o uso da ferramenta tecnológica Mineração de Dados nestes repositórios [Kagdi et al., 2007; Hassan, 2008; Godfrey et al., 2009; Hassan & Xie, 2010a; Xie, 2010]. Estes trabalhos exploram estes dados em busca de soluções para problemas ou conhecimentos inovadores que forneçam suporte às organizações produtoras e/ou mantenedoras no desenvolvimento e/ou manutenção de software. É justamente este suporte que Hassan & Xie [2010b] propõem chamar de *Software Intelligence* - SI.

Motivados por pesquisas como estas, Santos et al. [2010] apresentam a descrição da aplicação de tecnologias de mineração de dados visando verificar o quanto os usuários do produto de software Windows 7® estavam expressando seus sentimentos e opiniões nos comentários de mensagens postadas no *Twitter*[18]. Foram utilizadas técnicas para coleta, limpeza, indexação e transformação dos *tweets*[19], que em seguida foram submetidos à entrada do algoritmo de mineração *SVM-Light*. O objetivo desta mineração foi realizar uma análise de sentimento dos *tweets* separando-os em: *tweets* com sentimentos de usuários a respeito do Windows *Seven*® *versus tweets* que não possuíam estes sentimentos de usuários.

No presente trabalho procurou-se caminhar com base nos estudos de Santos et al. [2010], mas com uma abordagem direcionada a fóruns de discussão. Assim, nos capítulos sub-sequentes apresenta-se a metodologia de pesquisa adotada, o método proposto, o estudo de caso aplicando este método e as considerações finais deste trabalho de pesquisa.

[18]Disponível em: `http://twitter.com/`

[19]As mensagens postadas por usuários recebem o nome de *tweets* na terminologia padrão do *Twitter*.

Capítulo 3

Metodologia

Neste capítulo, inicialmente buscou-se definir a presente investigação quanto ao seu tipo, de acordo com classificações relacionadas a metodologia da pesquisa especificadas na literatura. Na seqüência tem-se a descrição dos procedimentos metodológicos relacionados ao levantamento e avaliação de opiniões de usuários do produto de software na Web, fundamentados no referencial teórico.

3.1 Tipo de pesquisa

Para Gil [1991], o objetivo principal de trabalhos de pesquisa como este é encontrar respostas para problemas do mundo real, mediante o emprego de procedimentos científicos racionais e sistemáticos.

De acordo com Jung [2004], trabalhos de pesquisa científica podem ser classificados quanto à sua natureza, quanto aos seus objetivos e seus procedimentos.

A presente pesquisa classifica-se como tecnológica quanto à sua natureza, uma vez que aplica conhecimentos já existentes de Engenharia de Software, inovação de produto de software e mineração Web, visando encontrar informações que sirvam de insumo para a inovação de produtos de software em uso.

Quanto aos objetivos, este trabalho se apresenta como uma pesquisa exploratória, pois propõe um método alternativo que se soma aos já existentes, no esforço para encontrar informações que podem propiciar inovação a produtos de software, particularmente inovações incrementais ou radicais, associadas a opiniões de usuários deste produto, disponíveis na Web.

Do ponto de vista dos procedimentos técnicos, esta pesquisa envolveu um levantamento bibliográfico e um estudo de caso. A partir dos estudos realizados no capítulo de referencial teórico foi possível formular um método para a solução do problema de pesquisa investigado e, em seguida, implementar um estudo de caso para validar a teoria levantada na pesquisa.

A unidade caso ou os objetos de validação das pesquisas deste trabalho foram, respectivamente, o Windows Vista®[1] e o fórum de Discussão do Clube do Hardware[2]. Este fórum apresenta um conjunto de tópicos com comentários sobre o sistema operacional Windows Vista®, inseridos por usuários.

Portanto, tem-se, conforme Jung [2004], um estudo de caso apoiado por pesquisa operacional, ou seja, pelo uso de ferramentas matemáticas relacionadas à mineração de dados, obedecendo a procedimentos seqüenciais e iterativos de: coleta de páginas Web, pré-processamento, mineração de opinião de usuários sobre o produto e avaliação dos resultados, considerando o tema inovação de produto de software.

Estes procedimentos serão detalhados na seção seguinte deste capítulo de metodologia, tendo-se em conta todos os aspectos da mineração de conteúdo Web.

3.2 Procedimentos metodológicos

Inicialmente, fundamentado em referencial teórico, considerou-se que as principais áreas da Engenharia de Software que possuem importante relação com inovação de produto são qualidade, avaliação e manutenção de software. Particularmente no que diz respeito à inovação incremental e radical do produto de software envolvendo as necessidades e expectativas de usuários.

Na pesquisa bibliográfica realizada neste trabalho, foi observado que inovações em produtos de software podem surgir a partir de opiniões manifestadas por usuários. Trata-se de uma abordagem interessante e, ao mesmo tempo, relevante, pois usuários possuem experiência sobre o funcionamento destes produtos e conhecem quais características estão atendendo, ou não, aos requisitos durante a operação ou uso.

[1] A seleção deste produto de software para estudo de caso visa apenas ilustrar a aplicação prática dos estudos teóricos sobre o levantamento e avaliação de opinião de usuários de software na Web, não tendo qualquer relação com o seu mantenedor: `http://windows.microsoft.com/pt-br/windows-vista/products/home`

[2] Disponível em: `http://www.clubedohardware.com.br`

Assim, entende-se que inovações incrementais ou radicais, a partir de opiniões de usuários são importantes aspectos a serem tratados em pesquisas na área de Engenharia de Software. Em função disto, aqui serão descritos os principais procedimentos relacionados à pesquisa de mineração de opiniões Web associadas às referencias de base da Engenharia de software apresentadas, ou seja, qualidade, avaliação, manutenção e inovação do produto de software.

Portanto, do ponto de vista de procedimentos metodológicos, obedeceu-se a seguinte seqüência: i) Estudo teórico relacionado aos temas qualidade, avaliação e manutenção de software, inovação do produto de software e mineração de dados, especificamente mineração voltada para Web; ii) Proposta e especificação de um método para levantamento e avaliação de opinião de usuários de produtos de software em uso na Web visando inovação do produto a partir da abordagem de mineração de opiniões de usuários na Web; iii) Validação do método proposto considerando um estudo de caso em que:

- Inicialmente pressupõe-se que o mantenedor do produto de software Windows Vista® encontra-se diante do problema de encontrar informações que podem auxiliar na tomada de decisão gerencial no sentido de inovação deste produto;

- Diante dessa necessidade, uma alternativa que se apresenta é o método proposto neste trabalho de pesquisa: levantamento e avaliação de opiniões de usuários deste produto na Web, para que estas opiniões sejam utilizadas pelo mantenedor no contexto de inovação do Windows Vista®.

- Recuperação e formação de uma coleção de páginas Web do sítio Clube do Hardware, onde foi constatado de antemão que havia um conjunto de páginas de discussão com muitos comentários sobre o Windows Vista®;

- Pré-processamento da coleção de páginas formada utilizando técnicas de limpeza, indexação e transformação formando uma coleção de documentos a serem submetidos ao algoritmo de mineração;

- Definição do algoritmo de mineração *SVM-Ligth* para tarefa de mineração de opinião de usuários na coleção de documentos originadas do pré-processamento;

- Mineração de documentos da coleção visando selecionar aqueles que possuem opiniões de usuários que potencialmente possam propiciar inovação incremental ou radical ao Windows Vista®;

- Avaliação dos resultados alcançados na mineração de documentos com opiniões de usuários no contexto de inovação incremental ou radical do Windows Vista®;

- E, finalmente, discussão dos resultados alcançados no estudo de caso, relacionando mineração de opinião de usuários de software na Web e inovação deste produto;

e, iv) Construção do capítulo de considerações finais deste projeto de pesquisa, sendo apresentadas também as principais limitações e sugestões de trabalhos futuros.

Os procedimentos metodológicos deste trabalho de pesquisa foram realizados principalmente nos laboratórios dos departamentos de Ciência da Computação das Universidades de Lavras (UFLA) e de Minas Gerais (UFMG), e envolveram o uso dos seguintes recursos tecnológicos:

- Microcomputador equipado com processador Intel Core2Duo™ de 2.4Gz, 2gb de memória principal, 160gb de memória para armazenamento e sistemas operacionais Linux Ubuntu[3] e Windows Vista®, onde se executou todas as tarefas do presente trabalho de pesquisa;

- *Crawler Wget*[4]: software utilizado para coleta das páginas Web do Clube do Hardware;

- *PHP*[5]: linguagem de programação utilizada para automatização das tarefas de pré-processamento relacionadas a mineração de dados;

- *Netbeans*[6]: ambiente para desenvolvimento e execução de sistemas *PHP*;

- *SVM-Light*[7]: algoritmo selecionado para a tarefa de mineração de opinião;

- *Dia*[8]: ferramenta utilizada para criação do diagrama de atividades UML que ilustra o processo representado no método a ser proposto;

- *UML*[9]: linguagem utilizada para criação do diagrama representativo do método a ser proposto.

[3]Disponível em: `http://www.ubuntu.com/`
[4]Disponível em: `http://www.gnu.org/software/wget/`
[5]Disponível em: `http://www.php.net/`
[6]Disponível em: `http://netbeans.org/`
[7]Disponível em: `http://svmlight.joachims.org/`
[8]Disponível em: `http://projects.gnome.org/dia/`
[9]Disponível em: `http://www.uml.org/`

No capítulo a seguir é apresentada a proposta e especificação do método para levantamento e avaliação de opinião de usuários de software em uso na Web, visando inovação deste produto a partir da abordagem de mineração de opiniões de usuários na Web.

Capítulo 4

Método proposto

A partir dos estudos referenciais e investigações realizadas passou-se, então, ao delineamento do método para levantamento e avaliação de opinião de usuários do produto de software em uso na Web (fóruns, blogs, redes sociais, entre outros), visando selecionar opiniões que potencialmente venham a propiciar inovação incremental ou radical de um produto de software mediante uma decisão gerencial por parte do mantenedor.

O processo utilizado no método, ilustrado na Figura 4.1, foi representado utilizando-se o diagrama de atividades da UML, que é bastante utilizado para descrever métodos, processos, funções ou operações de software.

O método para levantamento e avaliação de opinião Web propõe três partições que representam os papéis de Mantenedor, Avaliador e Desenvolvedor. Estas partições são diferenciadas por raias verticais (linhas tracejadas) e agrupam atividades a serem realizadas pelos respectivos papeis. Na partição Mantenedor estão atividades relacionadas à organização que produz e ou mantém o produto de software.

Na partição Avaliador estão as atividades que dizem respeito aos profissionais envolvidos na gerência da execução do método. É interessante que estejam envolvidos com estas atividades principalmente especialistas que detém conhecimento sobre o software em uso a ser avaliado e profissionais que representem o mantenedor, para acompanhamento e auxílio nas decisões que eventualmente precisam ser tomadas ao longo da execução do método. Ademais, profissionais que atuam como desenvolvedores também podem atuar no papel de Avaliador.

Finalmente, na partição Desenvolvedor estão as atividades inerentes às técnicas de mineração de dados necessárias para levantar e avaliar opiniões de usuários do produto de software em uso na Web, que podem ser realizadas por desenvolvedores com conhecimentos técnicos adequados.

O início da execução do método apresentado na Figura 4.1 dá-se a partir da atividade Definição do problema na partição Mantenedor.

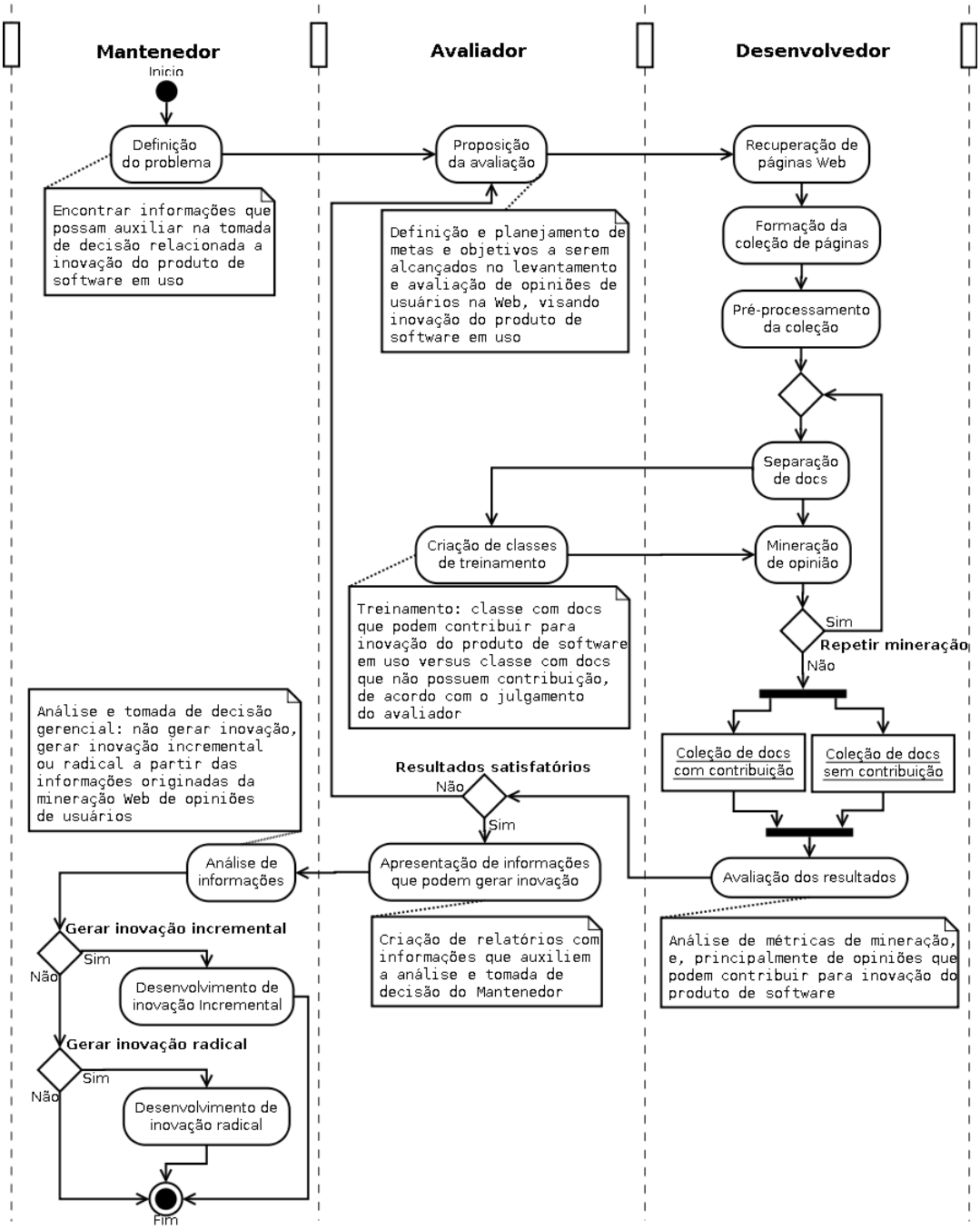

Figura 4.1. Método para levantamento e avaliação de opinião de usuários de software em uso na Web, no contexto de inovação

Definição do problema

Inicialmente, o mantenedor de software em uso enfrenta o problema de encontrar informações que possam auxiliar na tomada de decisão sobre inovação deste produto.

Retomando as idéias apresentadas no capítulo de referencial teórico, mantenedores de software vêem-se diante de questões como: qual a satisfação dos usuários em relação ao produto? Quais partes (funcionalidades) estão apresentando problemas quando esse sistema está em uso? O que aprimorar no produto? Quando lançar uma nova versão? Entre outras questões de mesmo tipo, que muitas vezes são respondidas com intuições e pressentimentos levando a tomada de decisões erradas que causam desperdício de recursos.

A partir da atividade Definição do problema no método para levantamento e avaliação de opinião Web há uma transição, representada por uma seta unidirecional, para a atividade Proposição da avaliação, que está na partição Avaliador.

Proposição da avaliação

Esta é uma atividade desempenhada pelo papel Avaliador, que representa um ou mais indivíduos que vão conduzir o processo de avaliação. Avaliador representa o papel de propor e conduzir o levantamento e avaliação de opiniões de usuários de software em uso na Web, com a finalidade de encontrar informações que auxiliem o mantenedor na tomada de decisão sobre inovação deste produto.

Nesta atividade algumas definições relacionadas às atividades da partição Desenvolvedor precisam ser observadas e planejadas. Inicialmente, é preciso definir quais os conhecimentos a serem alcançados ao final da execução das atividades relacionadas ao Desenvolvedor. Para isto é necessário identificar o problema ou características a serem investigadas no produto de software em uso, estudar o domínio deste problema ou características relevantes e utilizar o conhecimento gerado para traçar as metas e objetivos a serem alcançados.

É possível, por exemplo, conduzir a execução do método observando as características de qualidade (funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade, portabilidade, efetividade, produtividade, segurança, satisfação) relevantes para o produto de software em uso como um todo, ou, eventualmente, alguns de seus componentes ou características.

A segunda definição a ser observada na atividade Proposição da avaliação é a seleção da fonte (sítios de fóruns, blogs, redes sociais, entre outros) para coleta Web de forma que esta atividade seja a mais focada possível. Os profissionais no papel de Avaliador podem, por exemplo, pesquisar na Web, utilizando ferramentas de busca tais como Google[1], sítios onde comunidades de usuários colocam muitas informações (comentários, sugestões, sentimentos, opiniões) a respeito do produto de software sob avaliação. Ao contrário, coletar páginas Web não realizando estas observações pode tornar as atividades seguintes do método improdutivas sob o ponto de vista de mineração de dados.

A partir da atividade Proposição da avaliação há uma transição para a partição Desenvolvedor, especificamente para a atividade Recuperação de páginas Web.

Recuperação de páginas Web

Nessa atividade ocorre a coleta das páginas Web de onde serão extraídas as opiniões de usuários a respeito do produto de software. Esta coleta é focada em páginas que de antemão já se sabe que existem informações colocadas por usuários do produto sob investigação.

Para focar ainda mais a atividade de coleta, Avaliador e Desenvolvedor podem trabalhar com o conceito conjunto de exemplos, estudado na seção 2.3 e apresentado na Figura 2.7. A proposta associada a esse conceito é trabalhar nas atividades relacionadas à mineração de dados com amostras de um conjunto de exemplos que é extraído de uma distribuição de dados do mundo real. Em razão disto, pode-se focar a atividade de coleta nestes conjuntos de exemplos que retratam bem a realidade; ao contrário, teria que ser coletada toda a distribuição do mundo real e prepará-la para a atividade de mineração; conforme mencionado isto pode tornar as atividades seguintes do método improdutivas sob o ponto de vista de mineração de dados.

A partir desta atividade de coleta há uma transição para a atividade Formação da coleção de páginas.

Formação da coleção de páginas

Esta atividade, também relacionada ao papel Desenvolvedor, representa a formação de uma coleção única de páginas recuperadas de uma ou mais fontes Web (páginas de fóruns, blogs, redes sociais, entre outros). A partir dela há uma transição para a atividade Pré-processamento da coleção formada.

[1]Disponível em: `http://www.google.com.br/`

Pré-processamento da coleção

Na atividade Pré-processamento ocorre a transformação da coleção de páginas formada em uma coleção de documentos com formato adequado à entrada do algoritmo de mineração. Em especial, nesta atividade, é preciso definir o que será considerado documento. Por exemplo, para páginas de fóruns de discussão, uma abordagem é transformar cada *post*[2] dos tópicos em um documento da nova coleção a ser formada. Em seguida, é necessário realizar técnicas de limpeza, indexação e, por fim, a transformação de cada documento em uma representação textual adequada à entrada do algoritmo de mineração de opinião.

Nesta atividade de pré-processamento é necessário adotar estratégias e técnicas que envolvem recursos computacionais, tais como: computadores, linguagens de programação, métodos e técnicas de processamento de texto, entre outras. Por isto é uma atividade a ser realizada por desenvolvedores com conhecimentos técnicos adequados.

A partir da atividade de pré-processamento há uma transição para uma junção (losango), que neste ponto do processo representa a saída de uma transição após a execução de uma das duas transições de chegada: uma originada da atividade de Pré-processamento, conforme mencionado, e outra da decisão "Repetir mineração", que será explicada mais adiante.

A transição de saída da junção chega a atividade Separação de *docs*.

Separação de *docs*

O estado do processo para levantamento e avaliação de opinião nesta atividade indica que a coleção de páginas recuperadas na Web foi transformada em documentos que estão prontos para treinamento e processamento no algoritmo de mineração.

O Desenvolvedor, em conjunto com o Avaliador, separa parte dos documentos da coleção para a atividade Criação de classes de treinamento e a outra parte restante é reservada para a atividade Mineração de opinião. Em especial, é importante observar que não ocorre um processamento paralelo ou simultâneo das atividades Criação de classes de treinamento e Mineração de opinião. Primeiro é necessário realizar a atividade Criação de classes de treinamento.

[2]Mensagem de usuários em páginas de fóruns de discussão.

Criação de classes de treinamento

Esta atividade refere-se à parte da coleção com representação textual selecionada para criação do classificador ou modelo de mineração de opinião. No método que está sendo proposto, a mineração de dados é fundamentada em técnicas de Aprendizagem de Máquina Supervisionada, que se refere à construção de classificadores através de processos indutivos, sendo que em um processo indutivo constrói-se automaticamente um classificador para uma classe C_i "observando" as características de um conjunto de documentos.

Estes documentos são previamente e manualmente classificados sob C_i por especialistas no domínio em investigação e compõem o conjunto de treinamento que é usado para treino do algoritmo de mineração. No processo indicado no método para levantamento e avaliação de opinião Web, Figura 4.1, esta atividade de treinamento é realizada pelo Avaliador, que são principalmente especialistas no domínio da avaliação, tais como profissionais que conhecem bem as características de qualidade de software e o produto sob avaliação.

Para possibilitar a avaliação dos resultados alcançados na mineração de opinião o Avaliador e o Desenvolvedor precisam definir um método para treinamento/mineração, tal como o *hold-out*. Neste método a coleção é separada, na atividade Separação de *docs*, em 2/3 para conjunto de treinamento e 1/3 para conjunto a ser minerado, aleatoriamente. Por fim, é preciso repetir este método três vezes realizando os procedimentos treinamento/mineração, sendo esta repetição representada pelo possível *loop* na decisão (losango) "Repetir mineração", após a atividade Mineração de opinião.

Após o treinamento do algoritmo de mineração ocorre a atividade Mineração de opinião, cuja entrada é a parte da coleção não separada para treinamento.

Mineração de opinião

Esta atividade é conduzida pelo Desenvolvedor e realizada automaticamente pelo algoritmo de mineração treinado pelo Avaliador. Especificamente neste estudo o algoritmo de mineração adotado foi o *SVM-Light* que é fundamentado no processo de classificação binária de texto. Conforme foi apresentado, este algoritmo tem sido bastante utilizado e apresenta bons resultados em atividades de classificação binária de texto, que podem ser mapeadas para problemas de mineração de opinião.

Como a mineração do método para levantamento e avaliação de opinião Web é fundamentada em classificação binária, o conjunto de treinamento criado manualmente pelo Avaliador é composto por duas partes - uma com documentos que podem contribuir para inovação do produto de software que está sendo avaliado, e outra com documentos que não apresentam esta contribuição. O algoritmo de mineração aprende o padrão de cada uma destas partes e torna-se apto a tomar uma decisão sobre a classificação para os demais documentos da coleção, ainda desconhecidos, submetidos à sua entrada.

As atividades de treinamento/mineração podem ser repetidas tantas vezes quanto necessário ou planejado inicialmente, de forma a produzir mais resultados ou analisar características desejadas. Por exemplo, após análise dos documentos encontrados no objeto Coleção de *docs* com contribuição, os Avaliadores podem decidir realizar outra classificação binária a partir destes documentos. A nova classificação seria, por exemplo, "*docs* com opiniões sobre usabilidade" *versus* "*docs* sem opiniões sobre usabilidade", a fim de tornar os resultados da mineração de opinião específicos apenas para *docs* que contêm opiniões sobre usabilidade e que potencialmente podem contribuir para inovação do produto de software neste sentido.

Esta estratégia de dividir o problema de mineração de opinião em vários problemas e depois mapear cada um destes para classificação binária ajuda a diminuir a dimensão dos resultados tornando-os mais específicos e conseqüentemente mais acessíveis à análise humana, e mais focados ao resultado de inovação desejado.

Ademais, a geração de um classificador ou modelo válido na atividade Criação de classes de treinamento pode exigir do Avaliador e Desenvolvedor modificações ou ajustes e novas execuções das atividades anteriores a esta. Este procedimento é considerado normal na área de mineração de dados, quando se busca um modelo de mineração que alcance resultados finais satisfatórios.

Avaliador e Desenvolvedor decidem por refazer, ou não, as atividades anteriores à mineração de opinião avaliando os resultados alcançados, que são representados no método da Figura 4.1 pelos objetos Coleção de *docs* com contribuição e Coleção de *docs* sem contribuição. Estes objetos recebem transições de uma separação (traço horizontal), que por sua vez, recebe uma transição da decisão "Repetir mineração".

Os objetos Coleção de *docs* com contribuição e Coleção de *docs* sem contribuição possuem transições para uma união, também representada por um traço horizontal, que expressa a unificação dos resultados para análise na atividade Avaliação dos resultados.

Avaliação dos resultados

A decisão por refazer ou não atividades anteriores para encontrar melhores resultados pode ser fundamentada principalmente avaliando os valores das métricas *Precision*, *Recall* e *Accuracy*, que fornecem uma visão sobre a eficiência da mineração de opinião.

Após análise dos resultados na atividade Avaliação dos resultados, se Avaliador e Desenvolvedor julgarem e decidirem (losango de decisão "Resultados satisfatórios") que a mineração de opinião foi eficiente e alcançou o planejamento feito inicialmente na atividade Proposição da avaliação, o processo do método para levantamento e avaliação de opinião Web alcança a atividade Apresentação de informações que podem gerar inovação. Ao contrário, o fluxo do processo é desviado para a atividade Proposição da avaliação, como forma de revisão das entradas utilizadas nas atividades já realizadas, em busca de resultados mais satisfatórios.

Apresentação de informações que podem gerar inovação

Nesta atividade o Avaliador pode confeccionar relatórios para apresentação ao Mantenedor que requisitou o levantamento e avaliação de opinião Web. Estes relatórios podem contemplar informações desde como foi conduzido todo o processo do método, o planejamento inicial com metas e objetivos a serem alcançados, até quais foram os resultados alcançados neste processo, principalmente quais foram os *docs* que possuem opiniões de usuários que podem contribuir para a inovação do produto sob investigação.

A partir da atividade Apresentação de informações que podem gerar inovação há uma transição para a atividade Análise de informações, que está na partição Mantenedor.

Análise de informações

Esta é uma atividade realizada pelo Mantenedor do produto de software sob avaliação para uma sub-sequente tomada de decisão gerencial. Após a atividade de Análise de informações há uma transição para a decisão "Gerar inovação incremental". A resposta positiva para esta decisão representa uma decisão tomada pelo Mantenedor de realizar a atividade Desenvolvimento de inovação incremental, por exemplo, uma atividade de manutenção no produto de software sob avaliação a partir das opiniões de usuários levantadas na Web.

Ao contrário, se a resposta para a decisão "Gerar inovação incremental" é negativa, há uma transição para a decisão "Gerar inovação radical". E, a partir desta decisão se o Mantenedor tomar uma decisão positiva há uma transição para a atividade Desenvolvimento de inovação radical.

A atividade Desenvolvimento de inovação radical consiste, principalmente, em aplicar os resultados recebidos e analisados no desenvolvimento de um novo produto de software. Esta decisão pode refletir uma estratégia de negócio por parte do Mantenedor, onde se implementa as idéias encontradas nas opiniões de usuários em um novo produto a ser lançado no mercado, pois estas idéias podem tornar o produto mais atraente ao consumidor/usuário/cliente.

A resposta negativa à decisão "Gerar inovação radical" reflete uma decisão por parte do Mantenedor de não gerar inovação - radical ou incremental - no produto de software avaliado a partir das informações recebidas, e finalmente chega-se ao fim do método.

As transições que saem das atividades que representam o desenvolvimento de inovação incremental ou radical também chegam ao fim do processo representado pelo método para levantamento e avaliação de opinião de usuários de software uso na Web, ilustrado na Figura 4.1.

É importante considerar que tanto a inovação incremental quanto radical a partir de opiniões de usuários levantadas na Web pelo método proposto pode gerar um produto de software de melhor qualidade, mediante uma decisão gerencial por parte do Mantenedor.

Ademais, o método para levantamento e avaliação de opinião de usuários de software em uso na Web, proposto neste capítulo, é um método que pode ser utilizado para avaliar e levantar opiniões disponíveis em páginas Web sobre qualquer produto de software, envolvendo qualquer tipo de analise relacionada às características ou funcionalidades deste produto, que no caso é alvo dos objetivos de inovação incremental ou radical a partir das opiniões de usuários.

No próximo capítulo é apresentado o estudo de caso realizado onde foi testado na prática o método proposto aqui neste capítulo. O software Windows Vista® foi o produto selecionado para ser submetido ao processo de levantamento e avaliação de opinião de usuários na Web e o fórum de discussão do Clube do Hardware foi o sítio selecionado para coleta de páginas Web, pois foi constatado de antemão uma quantidade significativa de páginas com comentários de usuários a respeito do Windows Vista®.

Capítulo 5

Estudo de caso do método proposto

Neste capítulo apresenta-se a experimentação e validação do método para levantamento e avaliação de opinião de usuários de software em uso na Web. Na seqüência estão descritas todas as práticas e testes relacionados às atividades deste método. Como principais componentes deste estudo de caso têm-se o fórum de discussão do sítio Clube do Hardware e o produto de software Windows Vista®, cujo mantenedor é a Microsoft®. Assim, considerando o processo do método apresentado na Figura 4.1, inicia-se a partir da atividade Definição do problema na partição Mantenedor.

Definição do problema
Considerou-se que a organização em estudo, no caso a Microsoft®, está diante do problema de encontrar informações que possam auxiliá-la a tomar decisão gerencial sobre inovação do produto de software Windows Vista®. Assim, uma alternativa possível para este problema enfrentado pela Microsoft® seria, portanto, a utilização do método para levantamento e avaliação de opinião de usuários do Windows Vista® em páginas Web, proposto pelo Avaliador[1] através da atividade Proposição da avaliação.

Proposição da avaliação
Nesta atividade o avaliador propõe ao mantenedor do Windows Vista® levantar e avaliar opiniões de usuários deste produto na Web, em busca de informações que possam auxiliar na decisão de propiciar inovação neste produto a partir de opiniões manifestadas por usuários, sendo este o objetivo deste trabalho.

[1]Para este estudo de caso o autor deste trabalho de pesquisa representou o papel de Avaliador e Desenvolvedor do método para levantamento e avaliação de opinião Web. Como avaliador ele tomou todas as decisões necessárias durante a execução deste estudo, inclusive aquelas que diziam respeito ao mantenedor, e executou todas as atividades técnicas relacionadas ao Desenvolvedor.

Diante da aceitação por parte do mantenedor, a primeira decisão do avaliador foi investigar (buscar) sítios Web com opiniões de usuários, não observando o nível ou conhecimento técnico destes, que pudessem contribuir de uma forma geral para inovação deste produto de software. Foram levantadas principalmente opiniões que envolvessem as características de qualidade de um produto de software, tais como: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade, portabilidade, efetividade, produtividade, segurança, satisfação.

O sítio Web selecionado foi o do Clube do Hardware[2]. Basicamente, este foi selecionado utilizando-se o Google[3] para pesquisar as palavras chave "Windows Vista", que dizem respeito ao produto de software em uso, do mesmo nome, da Microsoft®. Foram encontrados aproximadamente 221.000.000 resultados.

O Clube do Hardware encontrava-se entre os primeiros resultados. Este sítio foi fundado em 1996 e inicialmente era voltado para questões e discussões relacionadas a Hardware. No entanto, ao longo do tempo surgiram muitos tópicos de discussão com comentários de usuários do fórum sobre temas relacionados a software, dentre estes, sobre o Windows Vista®, fato constatado pelo avaliador. Mensalmente, segundo informações existentes no próprio sítio, aproximadamente nove milhões de pessoas buscam e inserem comentários, sugestões e dúvidas nos vários tópicos de discussão.

Definidos os objetivos a serem alcançados no estudo de caso, o produto de software a ser avaliado e a fonte de páginas Web onde será feita a mineração de opinião, a recuperação destas páginas é a próxima atividade do método, que acontece na partição Desenvolvedor.

Recuperação de páginas Web

Para executar esta atividade o desenvolvedor, que conforme mencionado também é o avaliador, observou especificamente a parte do sítio Clube do Hardware reservada às discussões sobre o Windows Vista® e constatou que havia 290 páginas[4]. Cada uma destas possuía 22 tópicos de discussão. Destes, 20 tópicos eram distintos e dois estavam em destaque e se repetiam em todas estas páginas.

A Figura 5.1 apresenta o estado de uma página de tópicos considerada pelo avaliador no momento da pesquisa, sendo o primeiro tópico: "Destaque: [Resolvido] Tradução do Windows vista de um jeito mais simples .....".

[2]Disponível em: `http://www.clubedohardware.com.br`

[3]Disponível em: `http://www.google.com.br`

[4]Acessado em 02/11/2010 - `http://forum.clubedohardware.com.br/windows-vista/f128`

Ao observar os tópicos em geral o avaliador constatou que eles possuem uma ou mais páginas com *posts*[5]. No primeiro tópico da Figura 5.1, por exemplo, foi possível observar a presença de várias páginas que totalizavam 72 *posts* de respostas. Foi possível observar também que este tópico e seus *posts* foram visitados 48.393 vezes, dentre outras informações.

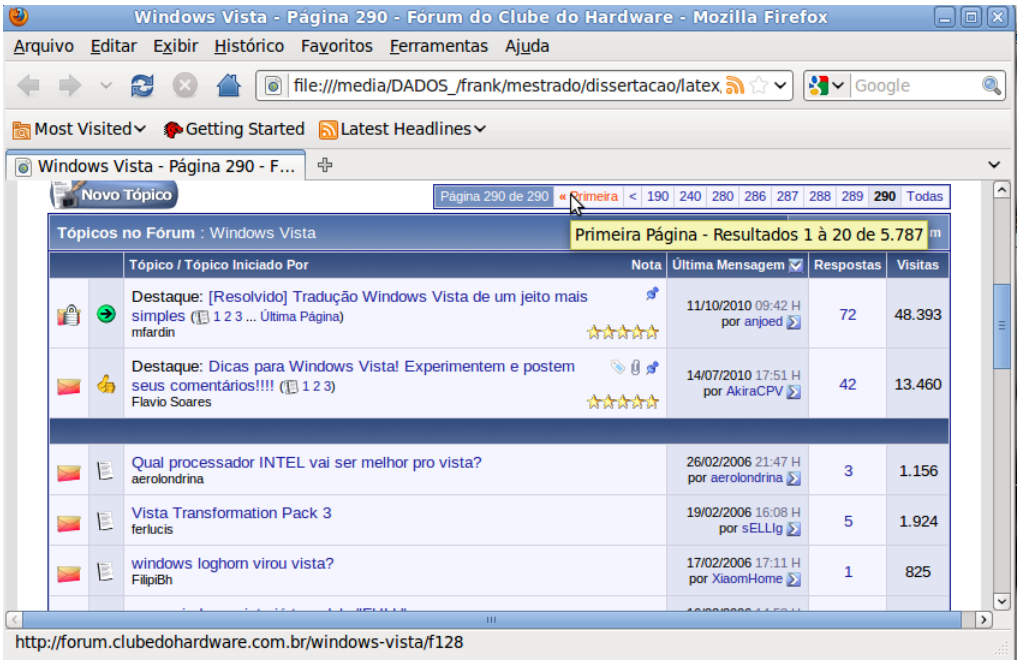

Figura 5.1. Página com tópicos de discussão sobre o Windows Vista®

Após observações e análises técnicas relativas a presente atividade o avaliador iniciou a atividade de coleta. O Web *crawler Wget* foi selecionado para realizar esta atividade, pois este *crawler* pode fazer uma coleta de todo o conteúdo do sítio ou pode coletar apenas determinadas partes, sendo esta última mais interessante para este trabalho.

O primeiro desafio encontrado ao iniciar a coleta focada estava relacionado ao armazenamento das páginas de cada tópico no sítio Clube do Hardware. Estas se encontravam em um diretório comum, onde páginas de tópicos sobre todos os temas de discussão do sítio estavam armazenados.

[5]Cada mensagem dentro de uma página de tópico é um *post* inserido por um usuário do fórum.

A Figura 5.2 apresenta esta situação visualizada via navegador Web. A página acessada através do endereço *http://forum.clubedohardware.com.br/windows-vista/f128* é uma daquelas que possuem ligações para as páginas de tópicos sobre o Windows Vista®. Observa-se nesta figura que o endereço apontado no rodapé do navegador, que se refere ao tópico apontado pela seta e destacado, mostra que a página deste tópico encontra-se no diretório *http://forum.clubedohardware.com.br/resolvido-windows-vista/873506*, não estando abaixo do diretório *http://forum.clubedohardware.com.br/windows-vista/f128*.

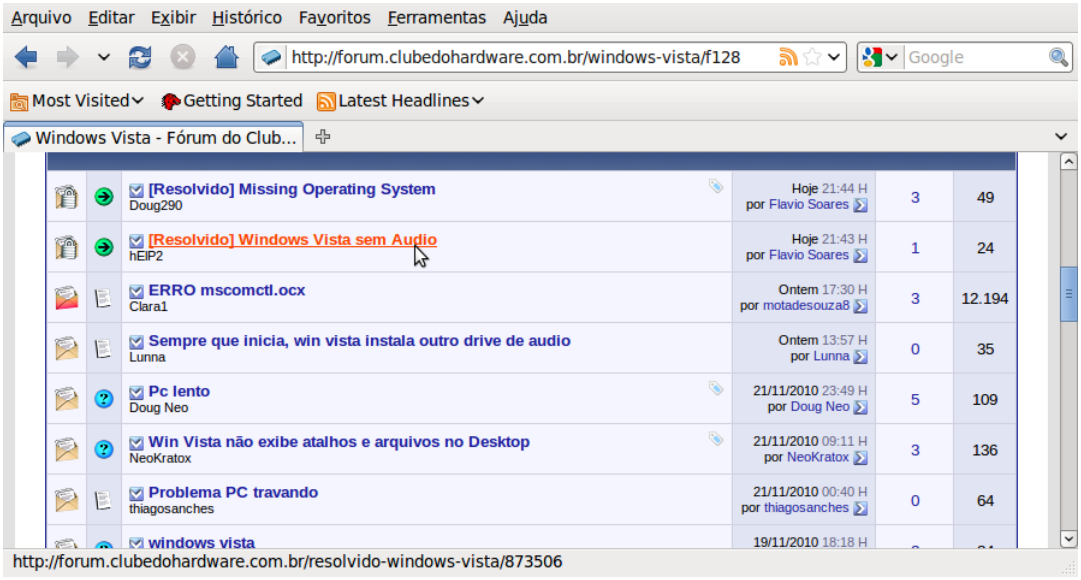

Figura 5.2. Ligações para páginas de tópicos de discussão sobre Windows Vista®

Em função deste problema, a coleta de páginas de tópicos sobre o Windows Vista® no sítio do Clube do Hardware foi dividida em duas etapas. Primeiro o avaliador coletou, com o comando *Wget*[6] abaixo, todas as páginas que estão dentro do diretório "windows-vista/f128", no endereço: *http://forum.clubedohardware.com.br/windows-vista/f128*.

> "wget –wait=10 –limit-rate=20K –reject='css, js, zip, jpg, gif, png, php' -r -np -x -k -nc -U Mozilla http://forum.clubedohardware.com.br/windows-vista/f128/"

[6]A definição e explicação dos parâmetros *Wget* utilizados neste trabalho de pesquisa podem ser observados no anexo A.

Estas páginas coletadas possuem apenas as ligações para as páginas de tópicos onde verdadeiramente estão os *posts* de usuários do Windows Vista®, que estão dentro do endereço *http://forum.clubedohardware.com.br/* juntamente com as centenas de páginas de todos os tópicos do sítio Clube do Hardware.

Em seguida, extraiu-se das páginas coletadas as ligações para as páginas de tópicos. O avaliador observou que estas ligações tinham o seguinte padrão: ***http://forum.clubedohardware.com.br/<texto>/<números>***

O avaliador observou que "<texto>" diz respeito ao título do tópico, tal como se demonstra no seguinte exemplo, onde o título do tópico é "erro-windows-vista": ***http://forum.clubedohardware.com.br/erro-windows-vista/602110***

Através de um *script PHP*[7] baseado em expressões regulares extraíram-se estas ligações. Estas ligações, então, foram armazenadas no arquivo texto "urls.txt".

A Figura 5.3 apresenta a estrutura do arquivo "urls.txt". Este arquivo totalizou 5121 ligações para páginas de tópicos de discussão sobre o Windows Vista®.

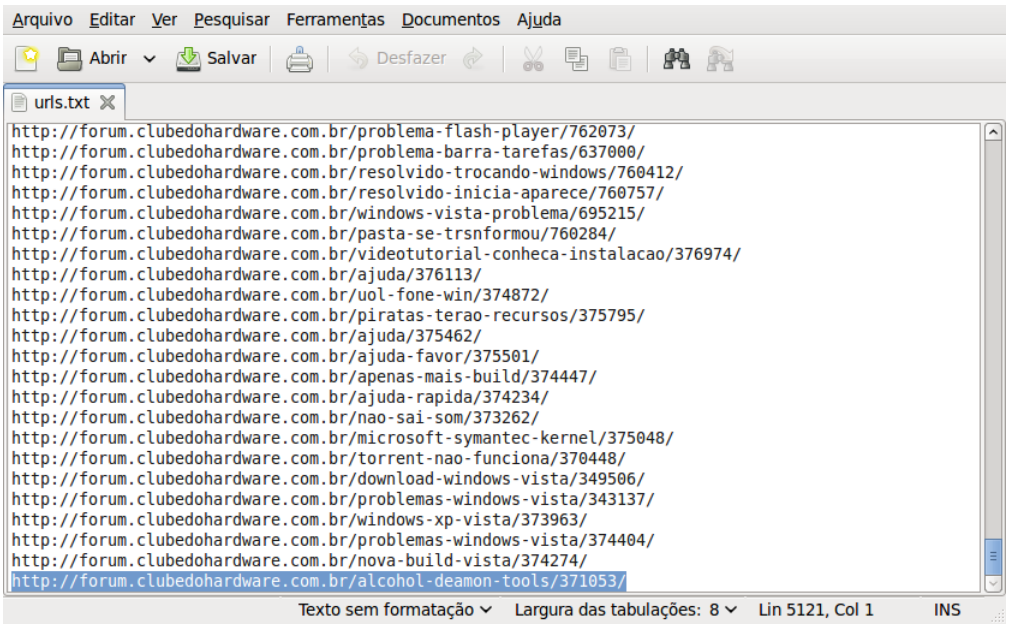

Figura 5.3. Arquivo urls.txt

Ao observar e analisar os títulos dos tópicos presentes nas ligações gravadas no arquivo "urls.txt" o avaliador decidiu coletar apenas uma parte das páginas apontadas por estas ligações, devido à grande abundância destas. Para este estudo de caso uma amostra destas páginas de tópicos será suficiente.

[7] Disponível em: `http://www.php.net`

Como o avaliador definiu inicialmente que o foco deste levantamento e avaliação é encontrar opiniões de usuários que possam contribuir de uma forma geral para inovação do Windows Vista®, decidiu-se investigar no arquivo "urls.txt" quais tópicos poderiam ser recuperados para esta finalidade.

Novamente, após observações e análises decidiu-se coletar páginas Web do Clube do Hardware cujos tópicos iniciam com a letra "P" de "Problema". Constatou-se que o arquivo "urls.txt" possuía 705 tópicos que iniciam com esta letra. Deste total, 435 tem a palavra ou termo "Problema".

A hipótese do avaliador é que neste conjunto de exemplos pode-se encontrar *posts* com opiniões que dizem respeito a problemas do Windows Vista®, que podem ser utilizados pelo mantenedor no auxílio à inovação deste produto. Para validar esta hipótese, o avaliador conduziu o processo de levantamento e avaliação de opinião em uma coleção de exemplos onde a maior parte dos tópicos possui a palavra "Problema".

Um segundo comando *Wget*, apresentado a seguir, iniciou a coleta de cada tópico armazenado no arquivo "urlsP.txt", que foi criado a partir do arquivo "urls.txt". Naquele arquivo há apenas endereços Web para os tópicos que começam com a letra "P".

"wget –wait=10 –limit-rate=20K –reject='css, js, zip, jpg, gif, png, php' –input-file='/home/frank/coleta/urlsP.txt' -c -r -np -x -k -nc -U Mozilla"

Este comando *Wget* fazia a leitura de cada linha do arquivo "urlsP.txt" e, em seguida, iniciava o download das páginas do tópico armazenando-as em um diretório no computador local. Em alguns casos havia mais de uma página para o mesmo tópico. Ao final, aqueles 705 tópicos totalizaram 824 páginas Web recuperadas no sítio Clube do Hardware, conforme mostra a Figura 5.4.

A abordagem anteriormente selecionada pelo avaliador de explorar os tópicos com a palavra "Problema" poderia perfeitamente ser outra. Por exemplo, poderia ser uma abordagem direcionada para tópicos onde há a palavra "Ajuda". Neste caso, o objetivo seria recuperar páginas Web e submetê-las ao método para levantamento e avaliação em busca de opiniões de usuários que podem propiciar inovação à ajuda *on-line* do Windows Vista®. Ou, ainda, utilizar todos os *posts* disponíveis visando outras abordagens. Este último caso, em especial, seria interessante havendo uma equipe de avaliadores/especialistas e uma estrutura computacional de dimensões adequadas, visto o grande volume de *posts* existentes.

Seguindo o processo do método para levantamento e avaliação de opinião Web, Figura 4.1, tem-se, a partir da atividade Recuperação de páginas Web uma transição para a atividade Formação da coleção de páginas Web.

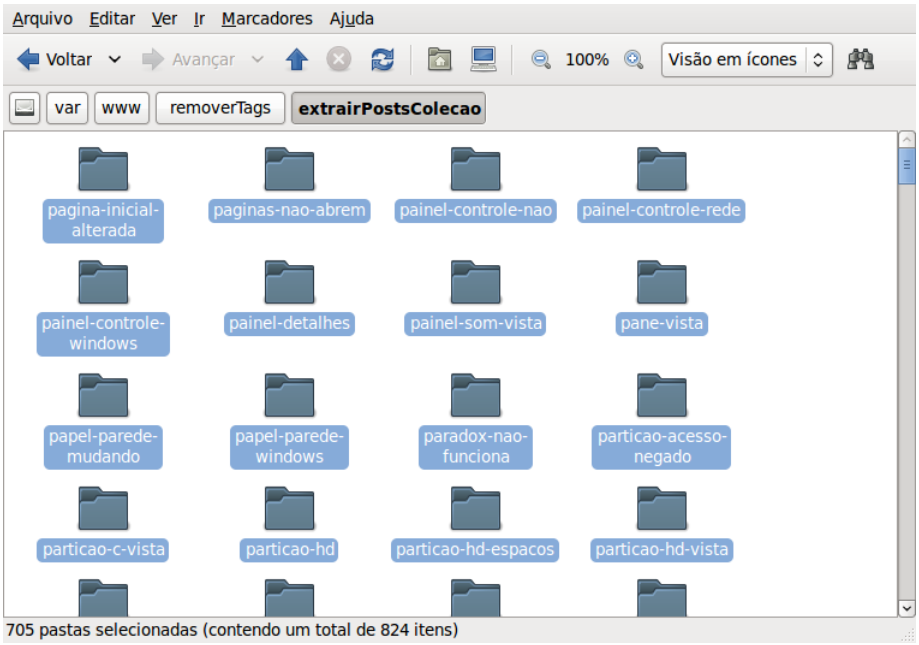

Figura 5.4. Coleção de pastas de tópicos que iniciam com a letra "P"

Formação da coleção de páginas

A execução do estudo de caso alcançou, então, a atividade Formação da coleção de páginas com abordagem para levantamento e avaliação de opinião bem definida e coleção de páginas Web formada, pois para este estudo de caso a coleta Web aconteceu somente em uma fonte, o sítio do Clube do Hardware.

A atividade sub-sequente é o pré-processamento da coleção de páginas formada.

Pré-processamento da coleção

O primeiro passo desta atividade que envolve técnicas de mineração foi transformar a coleção de páginas Web formada na atividade anterior em uma coleção de documentos. Cada página Web daquela coleção foi processada e cada *post* destas páginas transformou-se em um documento da nova coleção.

O avaliador estudou os códigos *HTML* das páginas Web do Clube do Hardware e percebeu que cada *post* localizava-se entre *tags* específicas. Diante desta constatação, foi possível a construção de um *script PHP* que extraiu cada *post* das páginas e transformou-o em um documento. Simultaneamente a esta transformação removeu-se as citações[8].

[8] *posts* que são repetidos dentro de *posts* com a finalidade de relembrar alguma questão levantada anteriormente.

A Figura 5.5 apresenta parte do *script* que extraiu os *posts* e transformou-os em documentos. Em especial, observa-se que as *tags* que estão dentro da variável "$buscarInicio" e "$buscarFim" são aquelas que delimitam os *posts* dentro do código *HTML* das páginas do Clube do Hardware.

O resultado desta fase inicial da atividade de pré-processamento foi a transformação das páginas recuperadas no Clube do Hardware em uma coleção com 3233 documentos contendo apenas texto. Cada um destes documentos recebeu o nome do tópico, seguido da palavra "Post" e um número inteiro seqüencial, conforme se observa na Figura 5.6.

Fundamentando-se no conceito de conjunto de exemplos o avaliador selecionou uma amostra exemplo de 1000 documentos desta coleção de 3233, para a continuidade da atividade de pré-processamento e sub-sequente execução de atividades de mineração. Para evitar que a amostra exemplo se concentrasse apenas em partes da coleção original, em detrimento de outras, o avaliador utilizou uma função aleatória que selecionou os documentos de maneira uniforme.

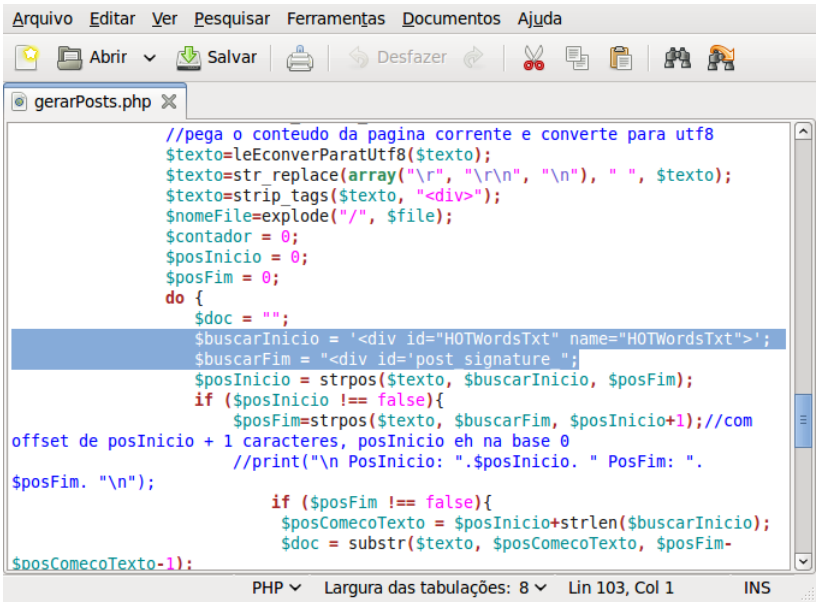

Figura 5.5. *Script PHP* utilizado para transformação de *posts* em documentos

A distribuição uniforme discreta[9] é uma distribuição de probabilidade que assume um número finito de valores com a mesma probabilidade. A adoção desta distribuição permitiu a seleção de uma coleção exemplo, a partir da coleção original de 3233 documentos, de maneira aleatória, uniforme e distribuída, apresentada na Figura 5.7.

[9]Disponível em: `http://en.wikipedia.org/wiki/Discrete_uniform_distribution`

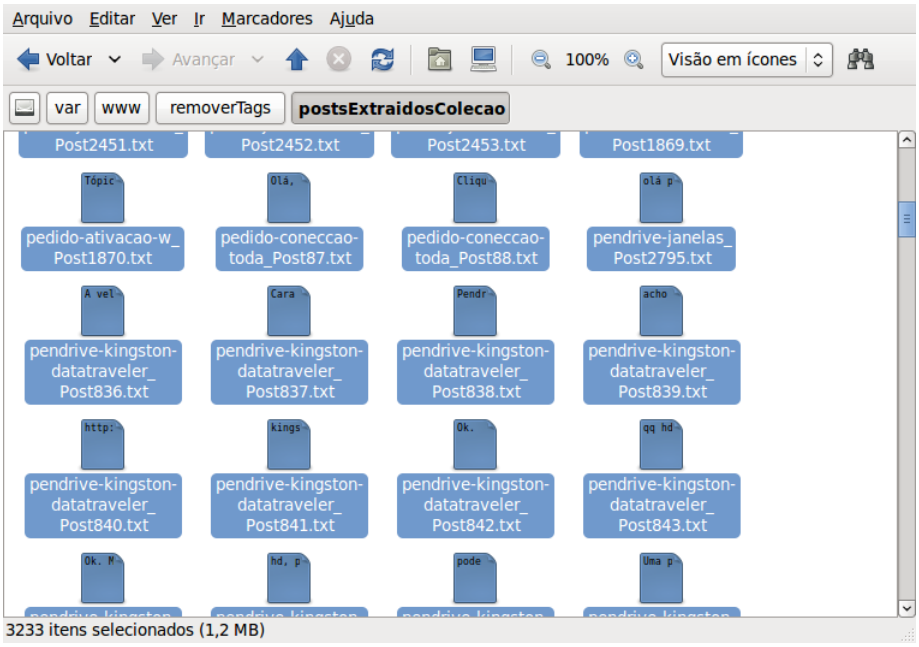

Figura 5.6. Coleção de documentos originados de *Posts* de páginas Web

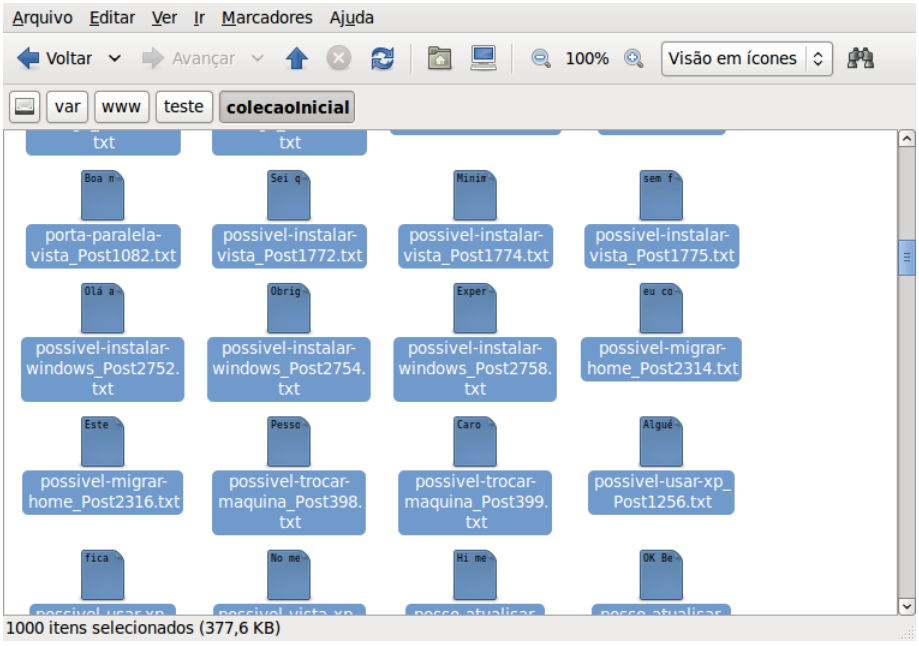

Figura 5.7. Coleção exemplo contendo 1000 documentos originados de *Posts* de páginas Web

A partir desta coleção exemplo, o avaliador iniciou a fase de criação da representação textual de cada documento para entrada no algoritmo de mineração. Foi aplicada uma limpeza removendo pontuações, números, acentos e algumas expressões indesejadas, tais como endereços Web que estavam presentes no conteúdo textual dos

documentos. Converteram-se as palavras ou termos para minúsculo e por fim colocou-se cada documento da coleção em uma única linha do arquivo "colecao.txt", Figura 5.8.

Com a aplicação destas técnicas de limpeza, nove documentos foram eliminados da coleção exemplo, que então totalizou 991 documentos, Figura 5.8. Estes documentos foram eliminados devido à pequena quantidade de palavras no conteúdo textual. Por exemplo, o documento de nome "problema-falha-logon_Post451.txt" possuía como conteúdo textual apenas o termo "upppppppppppppp". Este foi excluído durante a limpeza, o documento ficou vazio e não foi incluído no arquivo "colecao.txt".

Na seqüência destas últimas fases da atividade de pré-processamento, o avaliador criou o vocabulário de palavras ou termos distintos no arquivo "termos.txt", totalizando 6705 termos. Calculou-se o *tf* - freqüência do termo no documento - e o *df* - relevância do termo relacionada a documento/coleção - e armazenou-os no arquivo "tf.txt" e "df.txt", respectivamente. Isso possibilitou o cálculo do *tf-idf* de cada termo dos documentos que já estavam no arquivo "colecao.txt" e a gravação no arquivo "tfidf.txt".

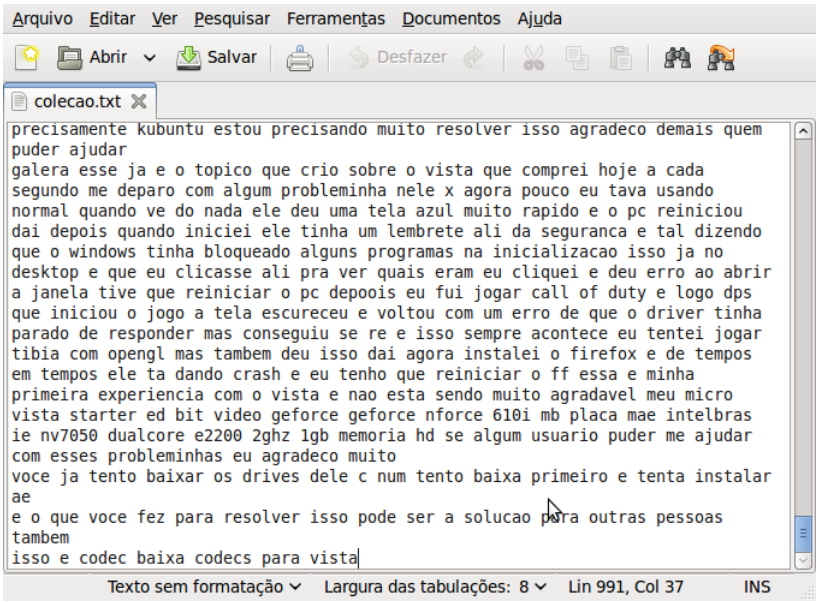

Figura 5.8. Coleção exemplo em um único arquivo onde cada documento está em uma linha

A Figura 5.9 apresenta os arquivos resultantes da atividade de pré-processamento. O arquivo "NomeDocsColecao.txt" armazena os nomes e a ordem dos 991 documentos que foram inseridos em "colecao.txt" e em "tfidf.txt". O arquivo "NomeDocsExit.txt" armazena o nome dos nove documentos que foram removidos nas fases de limpeza e criação do arquivo "colecao.txt".

Na seqüência (Figura 5.10) tem-se o arquivo "words" que é requerido pelo algoritmo que foi selecionado pelo avaliador para a atividade de mineração de opinião. Este arquivo contém os termos do vocabulário, um por linha e na seqüência em que estes aparecem no arquivo "tfidf.txt". O *tf-idf* dos termos são as medidas das características do conteúdo textual dos documentos da coleção que estão sendo exploradas na atividade de mineração.

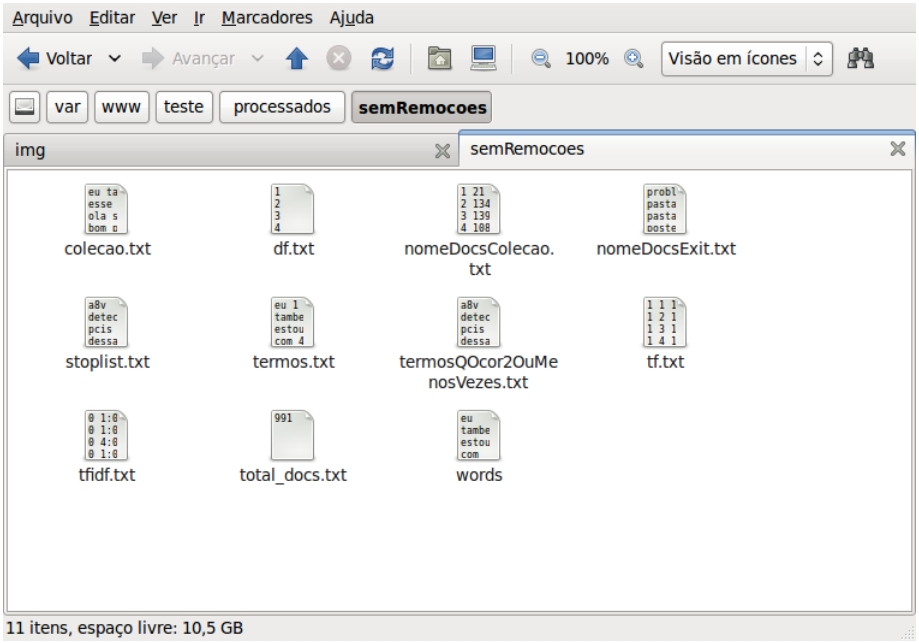

Figura 5.9. Arquivos resultantes da atividade de pré-processamento

Enfim, chegou-se a uma coleção pré-processada e já no formato *tf-idf*, pronta para a atividade seguinte do método proposto. No entanto, o avaliador decidiu pré-processar a coleção exemplo utilizando três outras abordagens, além da apresentada até aqui, onde não se removeu termos dos documentos. As três outras abordagens foram:

- Removeram-se termos considerados *stop-words* da língua portuguesa, inglesa e espanhola, pois o avaliador observou *posts*, quando transformados em documentos, que estavam nas línguas espanhola e inglesa;
- Removeram-se termos que ocorrem em dois ou menos documentos;
- E as duas abordagens anteriores simultaneamente: removeram-se termos considerados *stop-words* e termos que ocorrem em dois ou menos documentos.

O objetivo de ter quatro abordagens de pré-processamento é gerar mais resultados na atividade de mineração de opinião permitindo uma análise final comparativa.

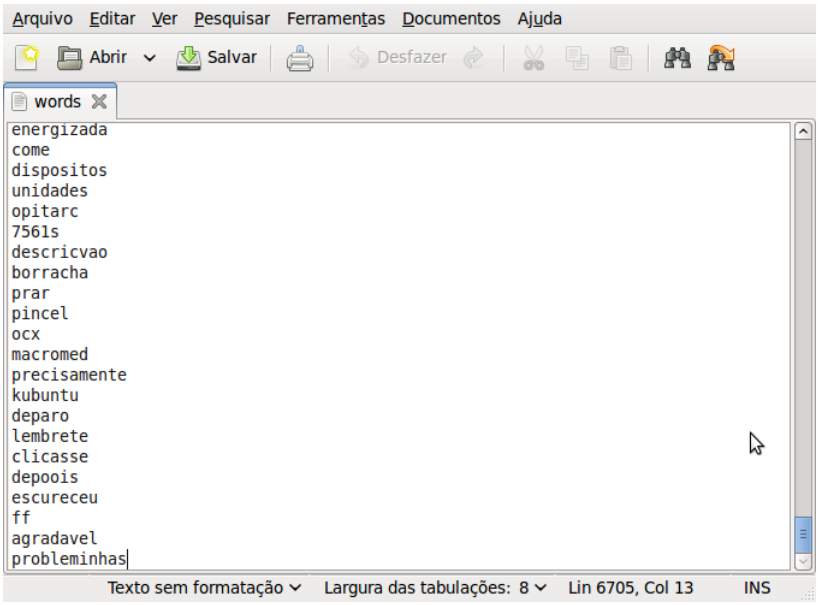

Figura 5.10. Formato do arquivo words requerido pelo *SVM-Light*

A Tabela 5.1 apresenta as quatro abordagens de pré-processamento adotadas sobre a coleção exemplo que foram testadas na atividade de mineração de opinião. É possível observar variações nos tamanhos do vocabulário de termos distintos e na quantidade final de documentos gravados no arquivo "coleta.txt". Cada uma dessas abordagens vai contribuir de maneira diferente na atividade de mineração de opinião.

Tabela 5.1. Abordagens de pré-processamento diferentes para a coleção exemplo

Id	Abordagem	Tam. vocabulário	Coleta.txt
1	Sem remoção de palavras ou termos	6705	991 *docs*
2	Remoção de *stop words*	6413	990 *docs*
3	Remoção de palavras que ocorrem em 2 ou menos documentos da coleção	2048	991 *docs*
4	Abordagens 2 e 3 juntas	1821	989 *docs*

A atividade de pré-processamento gerou quatro coleções com representação textual adequada para execução das atividades Separação de *docs*, Criação de classes de treinamento e Mineração de opinião. Na seqüência estas atividades são apresentadas em um mesmo tópico deste estudo de caso, pois são atividades fortemente relacionadas no método para levantamento e avaliação de opinião de usuários, onde estão envolvidas por um *loop* que pode ser iniciado a partir da decisão "Repetir mineração". Ademais, o autor deste trabalho observou que esta estratégia facilitaria a explicação de todas as ações e acontecimentos relacionados a estas atividades.

Separação de *docs*, Criação de classes de treinamento e Mineração de opinião Para realizar as atividades de mineração de opinião nas abordagens resultantes da atividade de pré-processamento, que estão com representação textual adequada, o avaliador utilizou a estratégia de treinamento *hold-out*. Cada uma das quatro coleções resultantes das quatro abordagens foi separada ou dividida nas proporções 2/3 para conjunto de treinamento e 1/3 para conjunto de teste (conjunto a ser minerado), aleatoriamente.

Este processo de divisão ou separação foi feito três vezes para cada uma das quatro coleções apresentadas na Tabela 5.1, gerando três conjuntos treinamento/teste por coleção, conforme apresenta a Figura 5.11.

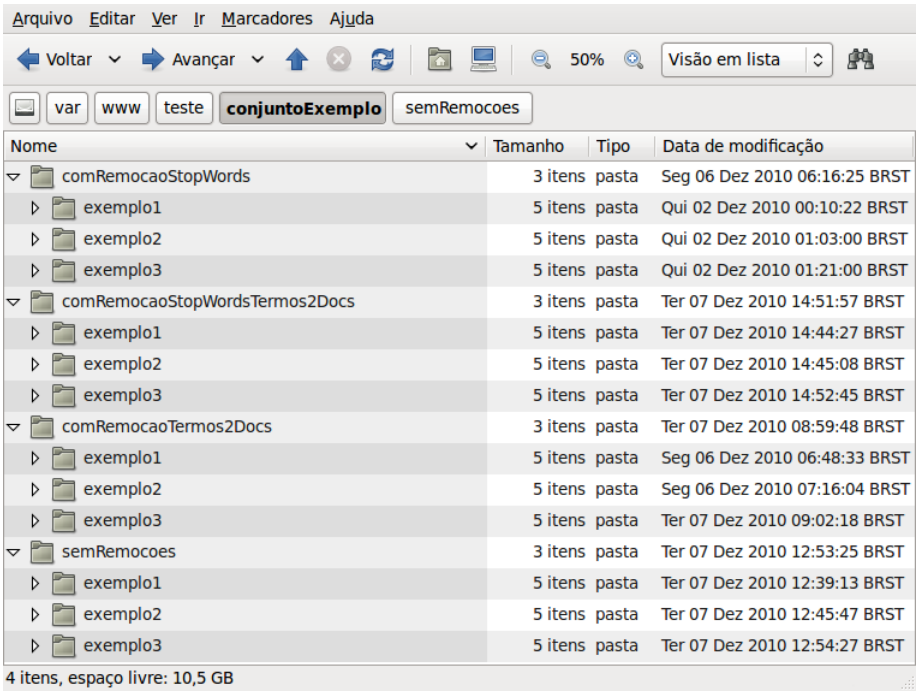

Figura 5.11. Conjuntos exemplos (treinamento/teste) gerados aleatoriamente para as quatro abordagens de pré-processamento

Dentro de cada um destes conjuntos exemplos, foram criados os conjuntos de treinamento e teste aleatoriamente, na proporção de 2/3 e 1/3, respectivamente. E o avaliador, cumprindo o papel de especialista no domínio da avaliação de opinião, realizou a classificação manual destes dois conjuntos, criando uma classe de documentos que podem contribuir para inovação do Windows Vista®, e outra onde há documentos que não apresentam este tipo de contribuição.

O conjunto de treinamento foi aquele utilizado para treinar o algoritmo de mineração de opinião, que é fundamentado em classificação binária. E o conjunto de teste foi aquele a ser minerado pelo algoritmo treinado. Este conjunto também foi classificado manualmente para que no momento da classificação o algoritmo faça a análise de efetividade, comparando sua classificação com a classificação do avaliador/especialista, apresentando os valores para as métricas *Precision*, *Recall* e *Accuracy* em sua saída padrão.

Para realizar a classificação manual dos conjuntos treinamento/teste, o avaliador adotou o critério apresentando na Tabela 5.2, que foi fundamentado nos conhecimentos adquiridos no Capítulo 2 - Referencial teórico, em especial baseado nas características de qualidade da norma ISO/IEC 9126-1 [2001], bem como seu conhecimento de usuário do Windows Vista®.

Este critério foi criado à medida que cada documento era analisado pelo avaliador e o mesmo decidia sua classificação. Trata-se de um padrão ou modelo de julgamento do avaliador para diferenciar documentos que podem contribuir para inovação do Windows Vista®, daqueles que não podem.

Tabela 5.2. Padrão ou modelo de classificação manual criado e adotado pelo avaliador

Classes	Padrão ou modelo gerado pelo avaliador para classificação manual
Docs com contribuição	- aparece a palavra "Windows Vista" ou variações; - contém opinião (positiva ou negativa) ou dúvida; - aparecem características de qualidade: usabilidade, segurança, customização, compatibilidade, instalação, portabilidade e desempenho.
Docs sem contribuição	- não é explícito se o comentário é relativo ao Windows Vista®; - não contêm crítica (opinião) ou dúvida; - suporte: mostra como fazer sem crítica ou dúvida; - comentários sem nenhuma contribuição (ex.: Obrigada, vou tentar ...).

A Figura 5.12 apresenta a estrutura de pastas criadas para a abordagem de pré-processamento "com remoção de *stop-words*". A coleção que está dentro das pastas selecionadas totaliza 991 documentos, conforme indica a mensagem no rodapé desta figura. É possível constatar também nesta figura o número de documentos presentes nos conjuntos de treinamento e teste, separadamente, que juntos totalizam os 991 documentos, total da coleção sob análise. É possível ainda verificar as pastas de nome "comContribuição" e "semContribuição", onde estão os documentos selecionados manualmente pelo avaliador.

Neste passo, o avaliador realizou a classificação manual de todos os três conjuntos exemplos - exemplo1, exemplo2 e exemplo3 - de cada uma das coleções originadas da etapa de pré-processamento. Em seguida, com as classes definidas, o avaliador extraiu do arquivo "tfidf.txt" os documentos para treinamento e teste de mineração de cada exemplo.

Na Figura 5.12, observa-se que há dois arquivos: "treinamento.dat" e "teste.dat". Neles encontra-se a representação textual em forma de *tf-idf* dos documentos que estão armazenados nos diretórios treinamento e teste. Esta representação foi retirada do arquivo "tfidf.txt", onde inicialmente armazenou-se a representação de toda a coleção.

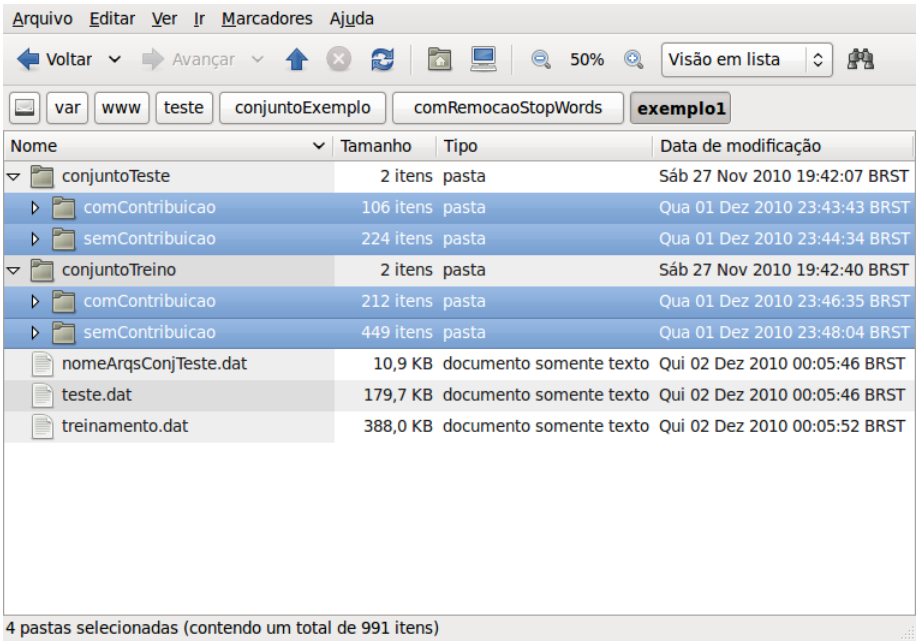

Figura 5.12. Conjuntos treinamento/teste da abordagem "com remoção de *stop-words*"

Para cada documento inserido dentro dos arquivos "treinamento.dat" e "teste.dat", o valor "1" foi inserido para indicar que o documento possuía contribuição, e "-1" para aqueles que não possuem contribuição para inovação do Windows Vista®.

A Figura 5.13 apresenta a estrutura de um arquivo "treinamento.dat" em *tf-idf*. Observa-se a presença de documentos marcados para as duas classes: "comContribuição" e "semContribuição".

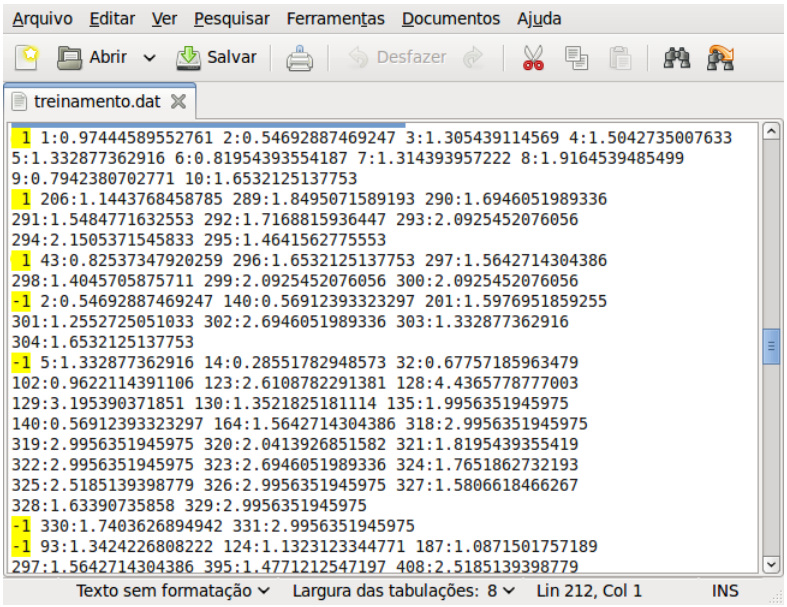

Figura 5.13. Exemplo de arquivo treinamento.dat

A estrutura dos arquivos "treinamento.dat" e "teste.dat" permitiu o treinamento e teste de mineração utilizando o *SVM-Light*, algoritmo selecionado pelo avaliador para a atividade de mineração de opinião.

O arquivo "teste.dat" é aquele que foi classificado com o modelo ou classificador gerado a partir do arquivo "treinamento.dat". Como o "teste.dat" também foi classificado manualmente pelo avaliador, que nessa pesquisa desempenha o papel de especialista, o *SVM-Light* exibe em sua saída as métricas de *Precision*, *Recall* e *Accuracy*, permitindo a avaliação dos resultados alcançados na atividade de mineração.

Isto é possível porque o *SVM-Light* faz a classificação automática de cada linha do arquivo "teste.dat" e compara com a classificação manual feita pelo avaliador, que também consta na linha. As métricas apresentadas na saída desse algoritmo são resultados da comparação entre classificação automática e classificação humana.

A Figura 5.14 apresenta os arquivos "treinamento.dat", "teste.dat" e "words" utilizados em uma atividade de mineração. E as Figuras 5.15 e 5.16 demonstram um exemplo de treinamento e a classificação, respectivamente, utilizando os "treinamento.dat" e "teste.dat" do exemplo1 da abordagem de pré-processamento "Sem remoção de palavras ou termos".

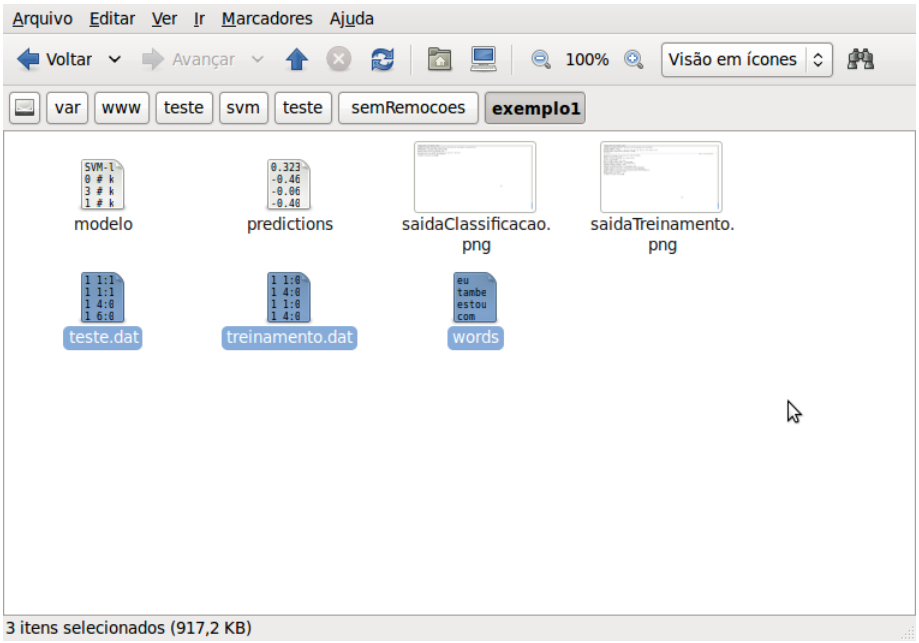

Figura 5.14. Arquivos treinamento.dat, teste.dat e words prontos para mineração de opinião

Ainda, observando a Figura 5.16, constata-se a presença do arquivo de nome "modelo" e um de nome "predictions" na primeira linha. O primeiro é o modelo ou classificador que foi gerado após o treinamento apresentado na Figura 5.15 e utilizado na classificação apresentada na Figura 5.16. E o segundo é o resultado da classificação realizada pelo *SVM-Light* sobre o arquivo "teste.dat", apresentado na Figura 5.16.

```
Arquivo  Editar  Ver  Terminal  Ajuda
frank@fmn:/var/www/teste/svm$ ./svm_learn teste/treinamento.dat teste/modelo
Scanning examples...done
Reading examples into memory...100..200..300..400..500..600..OK. (661 examples read)
Setting default regularization parameter C=0.0062
Optimizing.....................................................................................................
.....................................................................................done. (317 iterations)
Optimization finished (37 misclassified, maxdiff=0.00089).
Runtime in cpu-seconds: 0.05
Number of SV: 457 (including 147 at upper bound)
L1 loss: loss=92.60934
Norm of weight vector: |w|=1.05792
Norm of longest example vector: |x|=291.06465
Estimated VCdim of classifier: VCdim<=94816.94841
Computing XiAlpha-estimates...done
Runtime for XiAlpha-estimates in cpu-seconds: 0.00
XiAlpha-estimate of the error: error<=69.14% (rho=1.00,depth=0)
XiAlpha-estimate of the recall: recall=>15.57% (rho=1.00,depth=0)
XiAlpha-estimate of the precision: precision=>10.61% (rho=1.00,depth=0)
Number of kernel evaluations: 25361
Writing model file...done
frank@fmn:/var/www/teste/svm$
```

Figura 5.15. Treinamento do *SVM-light*

```
Arquivo  Editar  Ver  Terminal  Ajuda
frank@fmn:/var/www/teste/svm$ ./svm_classify teste/teste.dat teste/modelo teste/predictions
Reading model...OK. (457 support vectors read)
Classifying test examples..100..200..300..done
Runtime (without IO) in cpu-seconds: 0.00
Accuracy on test set: 79.09% (261 correct, 69 incorrect, 330 total)
Precision/recall on test set: 76.81%/50.00%
frank@fmn:/var/www/teste/svm$
```

Figura 5.16. Classificação utilizando *SVM-Light*

Por fim, o avaliador realizou o treinamento e o teste de mineração - atividades relacionadas as partições Avaliador e Desenvolvedor no método da Figura 4.1 - para cada um dos três exemplos das abordagens apresentadas na Tabela 5.1 de coleções resultantes da atividade de pré-processamento, gerando os valores para as métricas *Precision*, *Recall* e *Accuracy*.

Estes resultados serão apresentados e discutidos no contexto de inovação do Windows Vista®, observando qual das abordagens resultantes da atividade de pré-processamento alcançou melhores resultados na busca por documentos com contribuição para inovação deste produto de software.

Seguindo o processo do método para levantamento e avaliação de opinião de usuários de software em uso na Web, a próxima atividade deste estudo de caso é a Avaliação dos resultados alcançados nas atividades resultantes da mineração de opinião, que são representados, principalmente, pelos objetos: Coleção de *docs* com contribuição e Coleção de *docs* sem contribuição.

Avaliação dos resultados

Para avaliar os resultados alcançados no levantamento e avaliação de opiniões de usuários do Windows Vista® na Web, o avaliador/desenvolvedor observou a métrica *Accuracy*. Nesta situação específica a *Accuracy* é a medida para o número de documentos que foram classificados (minerados) corretamente para a categoria: *docs* com opiniões que podem contribuir para inovação do Windows Vista® *versus docs* que não possuem estas opiniões.

Cada uma das quatro coleções resultantes das variações da atividade de pré-processamento da Tabela 5.1 foi explorada na atividade de mineração separadamente. Com isto, o avaliador gerou quatro tabelas de resultados, correspondente a estas abordagens, onde pode ser visualizada a *Accuracy* e outras métricas. A abordagem de *Id* de número 1 na Tabela 5.1 é aquela onde não se removeu palavras ou termos dos documentos da coleção.

O resultado da atividade de mineração de opinião para esta abordagem é apresentado na Tabela 5.3. A primeira coluna desta tabela identifica o conjunto exemplo, a segunda o número de documentos que o *SVM-Light* classificou corretamente em comparação com a classificação humana, a terceira coluna diz respeito ao número de classificações erradas e as três últimas apresentam as métricas *Precision*, *Recall* e *Accuracy* do modelo ou classificador gerado no treinamento.

Nesta abordagem, o conjunto treinamento tem 661 documentos e o conjunto teste tem 330.

Tabela 5.3. Resultado da abordagem "Sem remoção de palavras"

Conj. Exemplo	Acertos	Erros	*Precision* (%)	*Recall* (%)	*Accuracy* (%)
Exemplo1	261 *docs*	69 *docs*	76.81	50.0	79.09
Exemplo2	270 *docs*	60 *docs*	82.61	54.29	81.82
Exemplo3	262 *docs*	68 *docs*	73.42	55.24	79.39

Na abordagem onde se removeu *stop-words* (*Id* de número 2 da Tabela 5.1), o conjunto treinamento e teste têm 660 e 330 documentos, respectivamente. A Tabela 5.4 apresenta os resultados alcançados nesta abordagem.

Tabela 5.4. Resultado da abordagem "Remoção de *stop-words*"

Conj. Exemplo	Acertos	Erros	*Precision* (%)	*Recall* (%)	*Accuracy* (%)
Exemplo1	264 *docs*	66 *docs*	82.26	48.11	80.0
Exemplo2	265 *docs*	65 *docs*	80.3	50.48	80.3
Exemplo3	263 *docs*	67 *docs*	76.39	52.38	79.64

Na abordagem onde se removeu termos que ocorreram em dois ou menos documentos (*Id* de número 3 da Tabela 5.1), o conjunto treinamento e teste constituiu-se de 661 e 330 documentos, respectivamente. A Tabela 5.5 apresenta os resultados alcançados nesta abordagem.

Tabela 5.5. Resultado da abordagem "Remoção de termos que ocorrem em 2 ou menos documentos da coleção"

Conj. Exemplo	Acertos	Erros	*Precision* (%)	*Recall* (%)	*Accuracy* (%)
Exemplo1	263 *docs*	67 *docs*	78.26	50.94	79.7
Exemplo2	269 *docs*	61 *docs*	81.43	54.29	81.52
Exemplo3	264 *docs*	66 *docs*	74.07	57.14	80.0

Por fim, os resultados obtidos na atividade de mineração quando se adotou as abordagens "Remoção de *stop-words*" e "Remoção de termos que ocorrem em dois ou menos documentos da coleção", simultaneamente, são apresentados na Tabela 5.6. Nesta abordagem o conjunto treinamento e teste constituíram-se de 661 e 329 documentos, respectivamente.

Tabela 5.6. Resultado das abordagens "Remoção de stop-words" e "Remoção de termos que ocorrem em 2 ou menos documentos da coleção"

Conj. Exemplo	Acertos	Erros	*Precision* (%)	*Recall* (%)	*Accuracy* (%)
Exemplo1	268 *docs*	61 *docs*	82.61	53.77	81.46
Exemplo2	271 *docs*	**58 *docs***	83.10	56.19	**82.37**
Exemplo3	265 *docs*	64 *docs*	76.62	56.19	80.55

A partir da observação e análise destes resultados, o avaliador constatou que o levantamento e avaliação de opinião de usuários sobre o Windows Vista® na Web alcançou os objetivos pretendidos. Foram alcançados valores de *Accuracy* que se encontram no mesmo intervalo alcançado, por exemplo, no trabalho de Wang et al. [2008], que é relacionado a este trabalho de pesquisa.

Ao contrário, se os resultados de *Accuracy* não fossem satisfatórios, o avaliador poderia retornar a qualquer uma das atividades do método para levantamento e avaliação de opinião Web, Figura 4.1, e fazer intervenções que poderiam melhorar os valores das métricas apresentadas nestas tabelas de resultados. Nas atividades que dizem respeito à mineração de dados Web poderia, por exemplo, adotar novas abordagens de coleta, de pré-processamento ou mesmo novos algoritmos para mineração, na busca por melhores resultados.

Depois de avaliado e constatado que método para levantamento e avaliação de opiniões alcançou os resultados esperados caminha-se para a apresentação de informações que podem gerar inovação ao mantenedor do Windows Vista®. Esta atividade e as atividades sub-sequentes serão apresentadas dentro da seção que discute os resultados alcançados neste estudo de caso, a seguir.

5.1 Resultados e discussão

Ao observar as Tabelas 5.3, 5.4, 5.5 e 5.6, referente ao estudo prático onde foram levantados e avaliados opiniões de usuários do Windows Vista® na Web, constata-se que o melhor resultado foi alcançado com o "Exemplo2" da Tabela 5.6, onde observa-se o maior valor de *Accuracy* dentre os demais resultados. Contudo, pode-se ainda observar que todos os resultados apresentados nas Tabelas 5.3 a 5.6 resultaram em valores de *Accuracy* que em geral são comparáveis com aqueles apresentados na literatura em trabalhos relacionados [Pang et al., 2002; Pang & Lee, 2008; Wang et al., 2008].

A Tabela 5.6 contém os resultados da abordagem de pré-processamento onde se removeu *stop-words* e termos que ocorrem em dois ou menos documentos da coleção (*Id* número 4 da Tabela 5.1). Dos 329 documentos classificados pelo *SVM-Light* no "Exemplo2", o modelo ou classificador gerado no treinamento errou 58 vezes, ou seja, a classificação automática do *SVM-Light* foi diferente da classificação humana feita pelo avaliador em 58 documentos do conjunto teste minerado.

A Figura 5.17 apresenta graficamente como foi a distribuição deste melhor resultado de mineração de opinião. Observando o eixo y, as marcações que estão acima do valor zero são os documentos classificados como aqueles que podem contribuir para inovação no Windows Vista®. Aquelas marcações que estão abaixo do valor zero correspondem aos documentos que não podem contribuir nesse sentido.

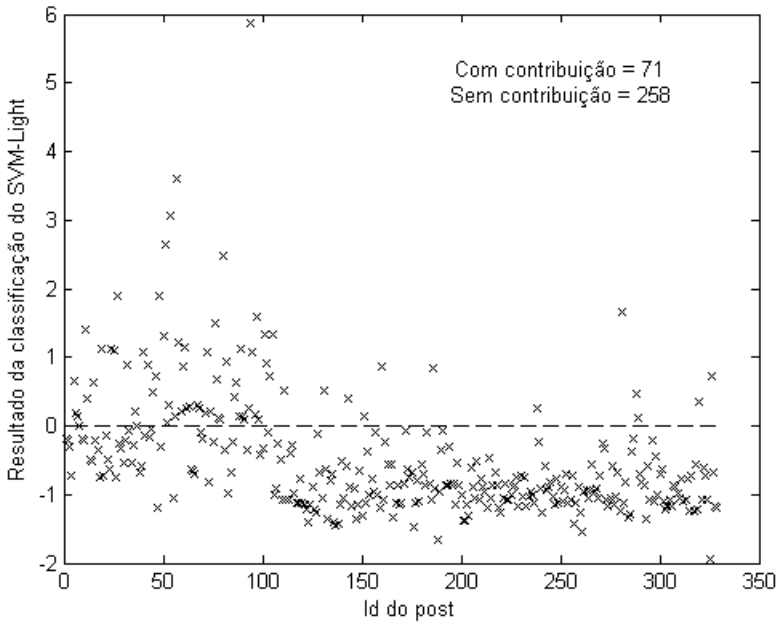

Figura 5.17. Distribuição espacial do melhor resultado da atividade Mineração de opinião

Este resultado, apresentado na Tabela 5.6 e Figura 5.17, refletiu numa *Accuracy* de 82.37%, maior valor dentre os demais testes de mineração e um resultado que está dentro do intervalo de acertos alcançados por trabalhos que utilizaram o algoritmo *SVM-Light* em tarefas relacionadas [Martins Jr., 2003].

Se o avaliador desejasse melhorar este resultado uma alternativa diferente daquelas relacionadas às atividades de mineração citadas anteriormente seria a análise dos 58 documentos onde a classificação automática foi diferente da classificação humana. A partir do conhecimento adquirido nesta análise, o avaliador poderia compreender porque a classificação foi diferente e, a partir disto, aprimorar o critério adotado para classificação manual, apresentando na Tabela 5.2.

Este critério foi o modelo de classificação manual adotado pelo avaliador para decidir a qual classe pertencia cada documento do conjunto de treinamento, quando este foi criado. É importante ressaltar que este critério tem uma forte relação com as dez características de qualidade de software estudadas no referencial teórico e também do conhecimento do software sob avaliação, ambas as situações oferecem um excelente suporte para avaliação do produto de software.

Então, a melhoria daqueles critérios apresentados na Tabela 5.2 pode refletir positivamente na *Accuracy* do modelo ou classificador gerado. Diante disto, localizar os documentos que foram classificados de forma distinta (*SVM-Light versus* Avaliador) é uma tarefa importante na busca por melhores resultados.

Para saber quais são esses documentos é necessário comparar as linhas dos arquivos "teste.dat" e "predictions". Onde o campo *<target>*, que tem os valores negativos ou positivos, estiverem com sinais opostos deve-se exibir e analisar os documentos, pois estes são aqueles que foram classificados de forma diferente.

Uma estratégia parecida com esta pode ser usada para localizar quais foram os documentos classificados como sendo aqueles que podem contribuir para inovação do Windows Vista®. Trata-se de localizar no arquivo "predictions" as linhas que estão com valor positivo no campo *<target>* e verificar se suas correspondentes também estão com valor positivo no arquivo "teste.dat".

Estas linhas correspondem àqueles documentos que foram classificados corretamente, ou seja, onde a classificação manual e automática coincidiu. E documentos correspondentes a estas linhas são aqueles que podem contribuir para inovação do Windows Vista®.

A Figura 5.18 apresenta um destes documentos encontrado automaticamente no estudo de caso realizado neste capítulo. E a Figura 5.19 apresenta um documento que foi classificado como não contendo opiniões que podem contribuir para inovação do Windows Vista®.

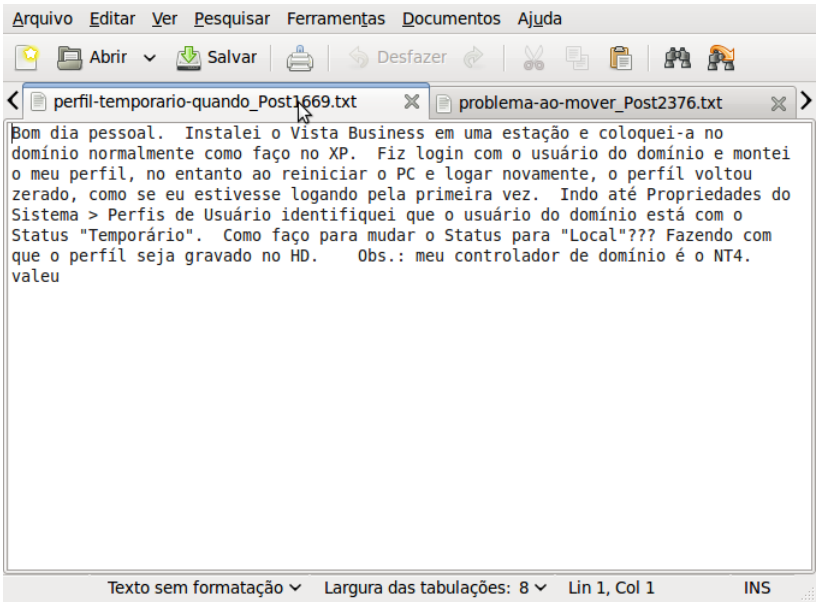

Figura 5.18. Documento que possui opiniões que podem contribuir para inovação do Windows Vista®

Na visão do avaliador, autor deste trabalho de pesquisa, o documento apresentado na Figura 5.18 contém opiniões sobre Windows Vista®, pois o usuário comenta sobre um problema que está enfrentando ao utilizar este sistema em um contexto particular.

Ainda na visão do avaliador, estas opiniões relatadas em forma de problema são ou representam oportunidades para o mantenedor do Windows Vista®. Uma investigação do problema apontado pelo usuário deste sistema poderá gerar conhecimento e este poderá ser utilizado para melhorar a qualidade da versão atual do Windows Vista® - inovação incremental - ou pode ser utilizado na criação de um novo Windows - inovação radical.

Por outro lado, no documento apresentado na Figura 5.19 não há opiniões que podem contribuir para a inovação do Windows Vista®. Trata-se de um suporte a alguma questão relacionada a este sistema levantada por outro usuário do fórum do Clube do Hardware.

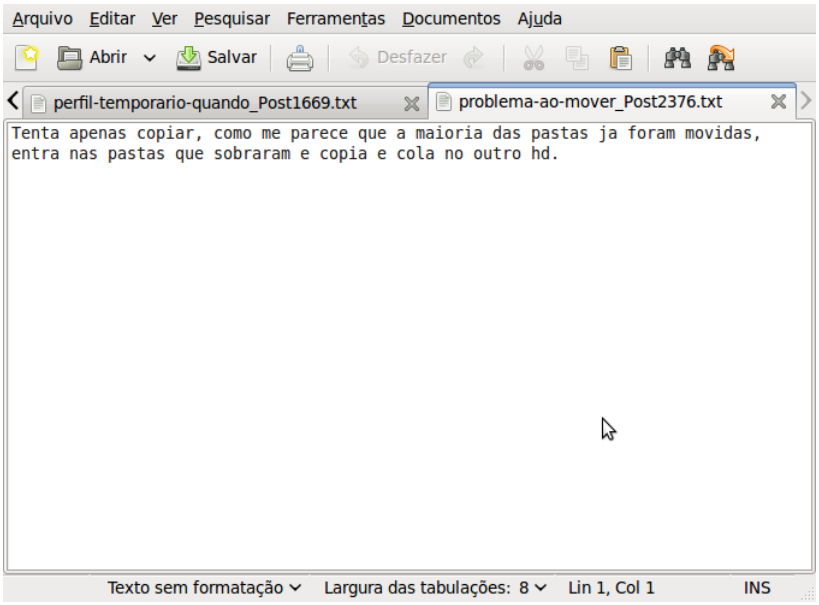

Figura 5.19. Documento que NÃO possui opiniões que podem contribuir para inovação do Windows Vista®

Ademais, a Tabela 5.7, exibida na próxima página, apresenta trechos de documentos que também foram classificados na abordagem que alcançou *Accuracy* de 82.37% como documentos que podem contribuir para inovação do Windows Vista® de uma maneira geral, observando as dez características de qualidade de software e o conhecimento que o avaliador possui como usuário deste produto.

Uma observação relevante relacionada ao método para levantamento e avaliação de opinião de usuários de software na Web diz respeito à necessidade de análise e geração de conhecimento a partir dos problemas encontrados durante a execução de cada atividade. Este comportamento por parte do avaliador garante a obtenção de melhores resultados em cada atividade e no final um melhor resultado geral, que especificamente neste trabalho foi avaliado pela métrica *Accuracy*.

Isto foi constatado no estudo de caso implementado neste capítulo, cujos resultados alcançados estão sendo analisados nesta seção. Neste estudo, o avaliador variou a atividade de pré-processamento criando quatro abordagens diferentes, gerando quatro coleções pré-processadas distintas (Tabela 5.1).

O avaliador observou ao pré-processar a coleção com a abordagem "sem remoção de palavras" que o tamanho do vocabulário estava muito grande. Ainda assim, prosseguiu e realizou as atividades de treinamento e mineração de opinião, que resultou nos valores apresentados na Tabela 5.3, onde a *Accuracy* máxima foi 81,82%.

Tabela 5.7. *Docs* que podem contribuir para inovação do Windows Vista®

Doc	Trecho com contribuição	Breve comentário do avaliador
problema-wista-ultimate_Post444.txt	"Ea galela, eu to com uma problema aqui no meu pc ..... Hoje pela manha estava rodando windows xp pro sp3 e então resolvi troca o meu xp pelo windows vista ultimate mais ai q ta o problema na hora de iniciar o windows aparece uma mensagem dando a opção de 2 windows a versão mais antiga e a versão do vista ..... Em relação a formatação do hd esta tudo correto porque antes de instalar o vista eu apaguei a partição do xp e recriei nova partição pro vista... Agora eu pergunto tem alguem que saber o que esta avendo com o meu windows vista? Se tiver poste .... Estou a espera"	Relato de problema sobre instalação do Windows Vista®. Como o usuário parece ter feito corretamente o processo de instalação é relevante que o mantenedor investigue em mais detalhes este problema para constatar se é uma falha do sistema ou não.
problemas-rede-wireless_Post223.txt	"..so q tem o problema, toda a vez q eu desligo o micro e volto a ligar o vista tem q reinstalar a placa de novo, já tentei de tudo ao meu alcanse e nada, se alguem tiver ideia, postem ai..."	O usuário relata problemas de compatibilidade. Diante disto o mantenedor pode estudar este problema e constatar se há falha do sistema do driver da placa ou do Windows Vista®.
problema-ao-instalar_Post765.txt	"Olá pessoal! Estou querendo formatar meu pc da assinatura novamente, e tenho o Vista home premium X 64 original, só que nunca consegui instalar ele com os 4 GB das corsair sempre ,tenho que por 1 GB no slot para não dar a maldita tela azul ,com 1 gb ele instala e depois boto os 4 gbs coms os sps e fixs e fica perfeito, Mas o que pode ser isso, configuração da bios em relação as memória,quando vou instalar o VIsta?? ou incompatibilidade de mobo ou memos???? o que pode ser? Toda vez que quero formatar o pc tenho que pedir 1 gb emprestado do meu primo e trocar as mem até conseguir instalar, é um saco, Ja li em vários lugares , mas ninguem disse algo concreto, nem o pessoal da microsof,ai é fogo Can someone help me? Abraço!"	Problema relacionado à instalação que pode ser investigado pelo mantenedor na tentativa de descobrir se há falha no Windows Vista®.

Não satisfeito e desejando investigar mais o avaliador resolveu então adotar técnicas de redução de dimensionalidade na atividade de pré-processamento. A análise e a decisão de adotar outras abordagens de pré-processamento viabilizaram alcançar uma *Accuracy* máxima de 82.37% e, conseqüentemente, um acerto maior no que diz respeito à quantidade de documentos com opiniões que podem contribuir para inovação do software sob avaliação.

A seguir, em relação às quatro abordagens adotadas para a atividade de mineração de opinião (Tabela 5.1), serão exibidos resultados que mostram detalhes interessantes e inerentes ao estudo de caso implementado. O gráfico da Figura 5.20 apresenta a pequena variação na quantidade de documentos de cada coleção para as variações de pré-processamento. No gráfico da Figura 5.21 observa-se a grande variação na quantidade de palavras ou termos do vocabulário de cada uma destas coleções.

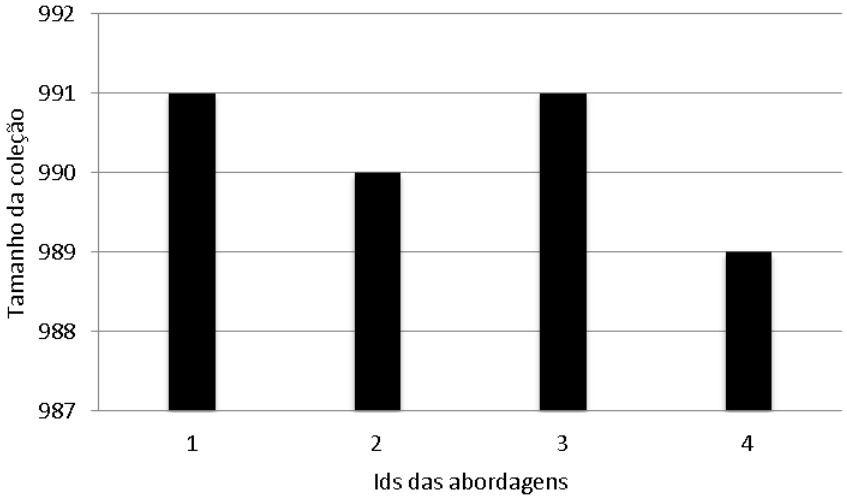

Figura 5.20. Alterações na quantidade de documentos da coleção para as variações de pré-processamento

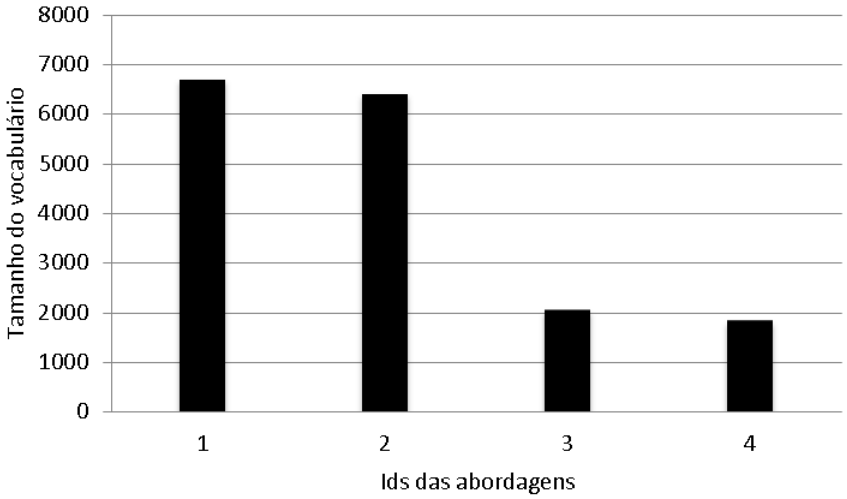

Figura 5.21. Alterações na quantidade de palavras ou termos dos vocabulários das coleções resultantes das variações de pré-processamento

A abordagem onde se reduziu a maior quantidade de palavras ou termos, *Id* número 4 da Figura 5.21, foi aquela que apresentou o melhor resultado para a métrica *Accuracy*. Isto pode ser confirmado observando o gráfico da Figura 5.22, onde os resultados das quatro abordagens de pré-processamento são apresentadas no mesmo espaço.

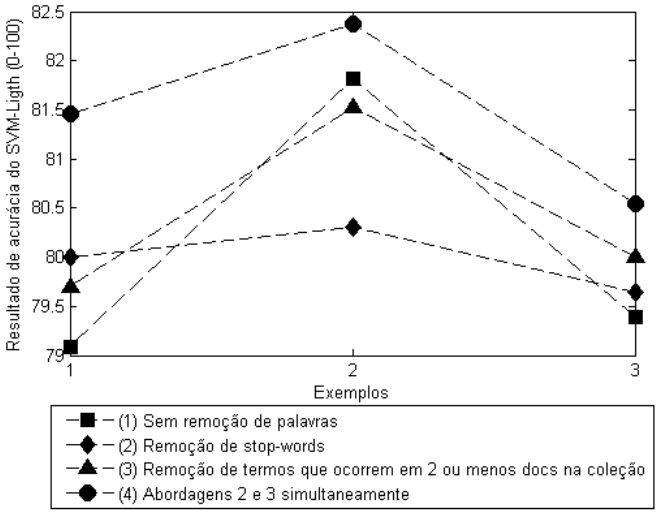

Figura 5.22. Resultados obtidos na atividade Mineração de opinião com as variações de pré-processamento

Na Figura 5.22 observa-se também o menor valor de *Accuracy* alcançado, que aconteceu no teste de mineração do exemplo 1 da abordagem "Sem remoção de palavras".

Analisando a relação entre os resultados apresentados nas Figuras 5.20 e 5.21 observa-se que o crescimento do valor de *Accuracy* é inversamente proporcional ao crescimento do número de termos do vocabulário. O exemplo 1 da abordagem "Sem remoção de palavras", onde não se removeu termos ou palavras dos documentos da coleção, foi aquele que alcançou o menor valor de *Accuracy*. Ao contrário da abordagem de Id número 4, onde se removeram *stop-words* e termos que ocorriam em dois ou menos documentos da coleção.

Então, através do estudo de caso realizado, é possível constatar que o número de termos ou palavras presentes no documento pode afetar positivamente ou negativamente o resultado alcançado no levantamento e avaliação de opinião na Web. Um número maior ou menor de opiniões de usuários do produto de software avaliado pode ser alcançado quando o número de termos ou palavras dos documentos da coleção é variado.

Em vista do que foi apresentado nesta seção, as observações e análises feitas levam a entender que o método para levantamento e avaliação de opinião de usuários de software em uso na Web, proposto e em seguida implementado através de um estudo de caso, foi validado pelos resultados alcançados neste estudo.

É importante observar que, durante a execução de cada atividade do método proposto, e particularmente observando os resultados na atividade Avaliação de resultados, o Avaliador pode variar as estratégias a serem adotadas, gerando novas abordagens, assim como foi feito na atividade de pré-processamento do estudo de caso. As novas abordagens geradas podem levar a resultados finais melhores naquilo que diz respeito ao levantamento e avaliação de opinião de usuários de software na Web.

Esta busca por melhores resultados reflete-se em encontrar um maior número de documentos com opiniões que podem ser apresentados ao mantenedor do software avaliado para que este tome uma decisão gerencial no sentido de propiciar atividades de inovação incremental ou radical a esse produto, ou, decidir que o produto permaneça como se encontra, não propiciando inovação. Pode-se observar que estes são os acontecimentos finais possíveis ilustrados no processo do método para levantamento e avaliação de opinião de usuários de software em uso na Web, proposto no capítulo 4 e avaliado/validado neste capítulo.

No próximo capítulo são apresentadas as considerações finais deste trabalho de pesquisa, sendo apresentadas também as principais limitações e sugestões de trabalhos futuros.

Capítulo 6

Considerações finais

O presente trabalho encontra-se no escopo de pesquisas da área de Engenharia de Software onde investigar métodos que auxiliem a inovação do produto de software é um problema desafiador e aberto a pesquisas. Para contribuir neste sentido, este trabalho sinalizou como objetivo geral propor um método para levantamento e avaliação de opinião de usuários de software em uso na Web e discutir as potencialidades destas opiniões no apoio aos processos de inovação incremental ou radical deste produto.

Este método foi então proposto, fundamentando-se nos conhecimentos de Engenharia, principalmente qualidade, avaliação e manutenção de software, bem como inovação do produto de software a partir de opinião de usuários, associados a conhecimentos de mineração Web. E na seqüência este método foi validado através de estudo de caso onde se obteve resultados satisfatórios, tais como a indicação de opiniões que podem levar a inovação do produto de software avaliado.

A possibilidade de melhorar os resultados alcançados durante a execução do método para levantamento e avaliação de opinião de usuários na Web foi discutida no estudo de caso e é real, pois em cada uma das atividades tem-se a oportunidade de analisar o conhecimento do contexto e tomar decisões que afetem positivamente um resultado local, que podem influenciar também positivamente o resultado final.

Na busca por melhores resultados durante a execução do método proposto pode-se, por exemplo, traçar diferentes metas e objetivos a serem alcançados no levantamento e avaliação de opinião, tais como explorar características específicas de atenção visando inovação em usabilidade, ou eficiência, ou portabilidade, entre outras relacionadas às características de qualidade de software. Definir novos locais e estratégias para coleta de páginas Web. Estudar e adotar novas técnicas de pré-processamento. Testar novos algoritmos de mineração na atividade Mineração de opinião, tais como aqueles estudados por Gonçalves [2009]: *Decision Trees*, *Nearest Neighbors*, dentre outros.

No que diz respeito às pesquisas em Ciência da Computação e Engenharia de Software o método proposto e validado soma-se aos métodos já existentes focados em melhoria de qualidade do produto de software para contribuir com a inovação deste produto. Em especial, este método é uma abordagem que aproxima a Engenharia de Software de novas tecnologias que a farão chegar mais próximo das necessidades e expectativas de usuários do produto de software.

Neste sentido, tanto a sociedade consumidora que anseia e necessita cada vez mais de software de qualidade, quanto às organizações que precisam produzir software de qualidade para atender à estas necessidades e expectativas, podem se beneficiar com o resultado deste trabalho de pesquisa, pois este apresentou e validou um método para levantamento e avaliação de opinião de usuários na Web, que visa contribuir para inovação - incremental ou radical - do produto de software em uso melhorando sua qualidade em uso.

Entretanto, cabe ressaltar as principais limitações deste trabalho de pesquisa, que estão relacionadas a seguir e podem ser observadas em trabalhos futuros:

- No estudo de caso realizado apenas um avaliador (o autor do presente trabalho) foi responsável por decidir, conduzir e executar as atividades técnicas do método proposto, enquanto o recomendável seria a participação de uma equipe de avaliadores que envolvessem profissionais responsáveis e conhecedores do produto de software sob avaliação e analista responsável pela execução das técnicas de Mineração Web;

- Em função da necessidade se determinar um escopo relacionado ao tempo para execução deste trabalho de pesquisa, o estudo de caso explorou apenas uma fonte de páginas Web, sendo esta fonte parte de um fórum de discussão sobre o produto avaliado. O adequado seria, por exemplo, realizar uma coleta de páginas em várias fontes. Uma maior variedade de páginas com comentários de usuários a respeito do produto sob investigação e uma equipe composta por profissionais adequados poderia melhorar a qualidade do estudo de caso realizado.

- Na abordagem de mineração de páginas Web adotada cada *post* das páginas de discussão foi transformado em um documento da coleção. Outras abordagens podem ser observadas e testadas, como, por exemplo, a identificação e transformação de *posts* relacionados em um único documento ou a transformação de todos os *posts* de um tópico em um único documento.

Ainda no que diz respeito a trabalhos futuros, o autor desta pesquisa especula sobre a possibilidade de utilizar o método para levantamento e avaliação de opiniões na Web para identificar sentimentos de usuários de software em uso. A identificação de sentimentos negativos ou positivos de usuários deste produto pode ser uma fonte inicial de informação para estudos. O passo seguinte do estudo seria, por exemplo, a descoberta do motivo que levou o usuário a expressar determinado sentimento a respeito do produto de software avaliado.

Além disto, este autor especula também sobre a adaptação do método proposto para auxiliar potenciais compradores a decidirem por adquirir (comprar) ou não produtos de software em uso, a partir de opiniões de usuários deste produto levantadas e avaliadas na Web.

Parece ser interessante apoiar o potencial comprador nesse sentido. Afinal, ele está investindo recursos financeiros, que muitas vezes envolvem valores significativos, na aquisição de um novo produto. As opiniões de usuários que possuem experiências de uso do produto pretendido podem ajudar este comprador a acertar na decisão de aquisição.

Referências Bibliográficas

Abran, A. & Moore, J. W. (2004). *Guide to the Software Engineering Body of Knowledge - SWEBOK*. IEEE Computer Society.

Aranha, C. N. (2007). *Uma Abordagem de Pré-Processamento Automático para Mineração de Textos em Português: Sob o Enfoque da Inteligência Computacional*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro.

Baeza-Yates, R. A. & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Benbrahim, H. & Bramer, M. (2009). Artificial intelligence. chapter Text and hypertext categorization, pp. 11--38. Springer-Verlag, Berlin, Heidelberg.

Bodendorf, F. & Kaiser, C. (2010). Mining customer opinions on the internet - a case study in the automotive industry. *International Workshop on Knowledge Discovery and Data Mining*, 0:24--27.

Carrilho Jr., J. R. (2007). Desenvolvimento de uma metodologia para mineração de textos. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro.

Chen, B.; Ma, L. & Hu, J. (2010). An improved multi-label classification method based on svm with delicate decision boundary. *International Journal of Innovative Computing, Information and Control*, 6(4):1605--1614.

Chesbrough, H. (2003). *Open Innovation: the new imperative for creating and profiting from technology*. Harvard Business School Press.

Coelho, O. P. (2006). A arquitetura e o desenvolvimento de produtos de software. Technical report, Microsoft Corporation.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273--297.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, 3:1289--1305.

Freire, A. (2002). *Inovação - Novos Produtos, Serviços e Negócios para Portugal*. Verbo.

Gamon, M. (2004). Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics*, p. 841, Morristown, NJ, USA. Association for Computational Linguistics.

Gamon, M.; Aue, A.; Corston-Oliver, S. & Ringger, E. K. (2005). Pulse: Mining customer opinions from free text. In *IDA*, pp. 121--132.

Gil, A. C. (1991). *Como elaborar projetos de pesquisa*. Atlas.

Godfrey, M. & German, D. (2008). The past, present, and future of software evolution. In *Frontiers of Software Maintenance, 2008. FoSM 2008.*, pp. 129--138.

Godfrey, M. W.; Hassan, A. E.; Herbsleb, J.; Murphy, G. C.; Robillard, M.; Devanbu, P.; Mockus, A.; Perry, D. E. & Notkin, D. (2009). Future of mining software archives: A roundtable. *IEEE Software*, 26:67--70.

Gonçalves, M. A. (2009). Text classification. Capítulo de livro a ser lançado fornecido pelo Prof. Marcos durante o curso de recuperação de informação, ministrado no segundo semestre de 2009 do curso de mestrado em ciência da computação da Universidade Federal de Minas Gerais.

Gonçalves, T. & Quaresma, P. (2003). A preliminary approach to the multilabel classification problem of portuguese juridical documents. In *EPIA*, pp. 435--444.

Grützmann, A.; Zambalde, A. L.; Esmin, A. & Santos, L. M. (2010). Framework para a geração de inovação de produtos a partir de conhecimento extraído da web. *Abstract and Proceedings 7th CONTECSI*, pp. 1468--1482.

Guerra, A. C. & Colombo, R. M. T. (2008). *Qualidade de Produto de Software*. Ministério da Ciência e Tecnologia - MCT.

Hassan, A. E. (2008). The road ahead for mining software repositories.

Hassan, A. E. & Xie, T. (2010a). Mining software engineering data. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2*, ICSE '10, pp. 503--504, New York, NY, USA. ACM.

Hassan, A. E. & Xie, T. (2010b). Software intelligence: the future of mining software engineering data. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, FoSER '10, pp. 161--166, New York, NY, USA. ACM.

Higgins, A. (2007). Software innovation as maintenance. In McMaster, T.; Wastell, D.; Ferneley, E. & DeGross, J., editores, *Organizational Dynamics of Technology-Based Innovation: Diversifying the Research Agenda*, volume 235 of *IFIP International Federation for Information Processing*, pp. 475--479. Springer Boston. 10.1007/978-0-387-72804-9_34.

ISO/IEC 12207 (2008). Information technology - systems and software engineering - software life cycle processes. Technical report, International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 14598-1 (1999). Information technology software product evaluation - part 1 general overview. Technical report, International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 14764 (2006). Software engineering - software life cycle processes - maintenance. Technical report, ISO/IEEE.

ISO/IEC 9126-1 (2001). Information technology - software product quality - part 1 quality model. Technical report, International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 9126-2 (2003). Information technology - software product quality - part 2 external metrics. Technical report, International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 9126-3 (2003). Information technology - software product quality - part 3 internal metrics. Technical report, International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 9126-4 (2004). Information technology - software product quality - part 4 quality in use metrics. Technical report, International Organization for Standardization, Geneva, Switzerland.

Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. In Nédellec, C. & Rouveirol, C., editores, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pp. 137--142, Heidelberg et al. Springer.

Jung, C. F. (2004). *Metodologia para Pesquisa & Desenvolvimento*. AXCEL BOOKS.

Kagdi, H.; Collard, M. L. & Maletic, J. I. (2007). A survey and taxonomy of approaches for mining software repositories in the context of software evolution. *J. Softw. Maint. Evol.*, 19:77--131.

Koscianski, A. & Soares, M. S. (2007). *Qualidade de Software - Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software*. Novatec.

Lan, M.; Tan, C. L.; Su, J. & Lu, Y. (2009). Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:721--735.

Lientz, B. P. & Swanson, E. B. (1981). Problems in application software maintenance. *Commun. ACM*, 24(11):763--769.

Linden, G. S. (2008). Combinação de classificadores na categorização de textos. Master's thesis, Pontifícia Universidade Católica do Rio Grande do Sul.

Lippoldt, D. & Stryszowski, P. (2009). *Innovation in the Software Sector*. OECD 2009.

Lo, Y. W. & Potdar, V. (2009). A review of opinion mining and sentiment classification framework in social networks. In *Digital Ecosystems and Technologies, 2009. DEST '09. 3rd IEEE International Conference on*, pp. 396--401.

Maldonado, J. C.; da Rocha, A. R. C. & Weber, K. C. (2001). *Qualidade de Software: Teoria e Prática*. Prentice Hall.

Martins Jr., J. (2003). Classificação de páginas na internet. Master's thesis, USP – São Carlos.

Naur, P. & Randell, B. (1968). Software engineering: Report of a conference sponsored by the nato science committee. Technical report, Brussels, Scientific Affairs Division, NATO.

OECD (1997). *Manual de Oslo*, terceira edição edição.

Pang, B. & Lee, L. (2008). Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1--135.

Pang, B.; Lee, L. & Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pp. 79--86, Morristown, NJ, USA. Association for Computational Linguistics.

Pfleeger, S. L. & Atlee, J. M. (2006). *Software Engineering - Theory and Pratice*. Prentice Hall.

Pressman, R. S. (2006). *Engenharia de Software*. McGraw-Hill.

Rezende, S. O. (2005). *Sistemas Inteligentes: Fundamentos e Aplicações*. Manole, primeira edição edição.

Rose, J. (2010). Software innovation: Eight work-style heuristics for creative system developers. Disponível na Web.

Santos, C. R. & Brasil, V. S. (2010). Envolvimento do consumidor em processos de desenvolvimento de produtos: Um estudo qualitativo junto a empresas de bens de consumo. *Revista de Administração de Empresas*, 50(3):300--311.

Santos, L. M.; Esmin, A. A. A.; Zambalde, A. L. & Nobre, F. M. (2010). Twitter, análise de sentimento e desenvolvimento de produtos: Quanto os usuários estão expressando suas opiniões? *Revista de Ciências e Tecnologias de Informação e Comunicação do CETAC.MEDIA*, (13):1--12.

Schumpeter, J. A. (1942). *Capitalism, Socialism and Democracy*. Harper & Brothers.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1--47.

Shandilya, S. K. & Jain, S. (2009). Automatic opinion extraction from web documents. *Computer and Automation Engineering, International Conference on*, 0:351--355.

Soares, F. A. (2008). Mineração de textos na coleta inteligente de dados na web. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro.

Tapscott, D. & Williams, A. D. (2007). *Wikinomics: como a colaboração em massa pode mudar seu negócio*. Nova Fronteira.

Tigre, P. B. (2006). *Gestão de Inovação: a Economia da Tecnologia no Brasil*. Campus.

Von Hippel, E. (1986). Lead users: a source of novel product concepts. *Manage. Sci.*, 32(7):791--805.

Wang, X.; Zhang, L.; Xie, T.; Anvik, J. & Sun, J. (2008). An approach to detecting duplicate bug reports using natural language and execution information. In *ICSE '08: Proceedings of the 30th international conference on Software engineering*, pp. 461--470, New York, NY, USA. ACM.

Xie, T. (2010). Bibliografias. Bibliografias sobre Mineração de Dados de Engenharia de Software.

Xie, T.; Thummalapenta, S.; Lo, D. & Liu, C. (2009). Data mining for software engineering. *Computer*, 42(8):55--62.

Yang, Y. & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 412--420, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Anexo A

Principais parâmetros para configuração do Web Crawler Wget:

- -E, –adjust-extension: salva os documentos HTML/CSS com as extensões apropriadas.
- -r, –recursive: especifica como download recursivo.
- -np, –no-parent: não subir ao diretório-pai.
- -L, –relative: segue apenas links relativos.
- -x, –force-directories: força a criação de diretórios.
- -F, –force-html: trata o arquivo de entrada como HTML.
- -k, –convert-links: faz os links no HTML ou CSS baixado apontarem para os arquivos locais.
- -A, –accept=LISTA: lista separada por vírgulas das extensões aceitas.
- -R, –reject=LISTA: lista separada por vírgulas das extensões rejeitadas.
- -I, –include-directories=LISTA: lista dos diretórios permitidos.
- -nc, –no-clobber: ignora os downloads que sobrescreveriam arquivos existentes.

Fonte: opção de ajuda do *Wget*