2
Processo de Requisitos, PLN e Agentes

O ciclo clássico de desenvolvimento de um produto de software tem início pelo Processo de Requisitos e é nesta fase que são determinados os requisitos que o software em construção deverá atender. Este processo gera um conjunto de artefatos que constituem uma *baseline* para o registro e acompanhamento da evolução dos requisitos ao longo do ciclo de desenvolvimento, possibilitando um efetivo gerenciamento de requisitos. A qualidade é fator crítico no documento de requisitos, dado que este será a base para as demais atividades ao longo do processo de desenvolvimento. Várias são as propostas para estruturação das atividades no processo de requisitos.

2.1.
Processo de Requisitos: estruturação das atividades

Segundo Leite [Leite94], este processo é composto pelas fases de elicitação, modelagem e análise de requisitos, representadas no diagrama SADT da Figura 2.

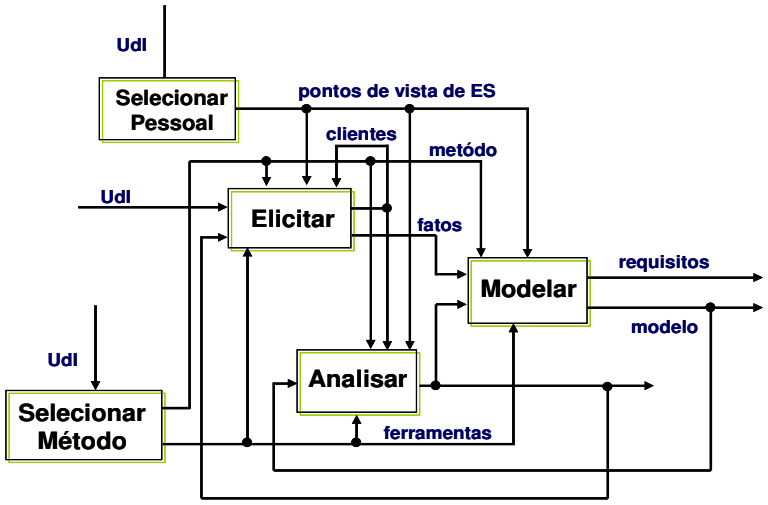

Figura 2 - Modelo SADT da Engenharia de Requisitos [Leite94]

Durante a fase de elicitação são utilizadas diferentes técnicas para a descoberta dos requisitos junto ao conjunto de interessados no sistema, com o objetivo de identificar necessidades e expectativas em relação ao sistema a ser desenvolvido. Estas necessidades e expectativas serão estruturadas e registradas de forma sistemática na fase de modelagem, seguindo um método previamente definido.

A última etapa deste processo é denominada de análise, que envolve as atividades de verificação e validação e tem por objetivo avaliar a qualidade da representação dos requisitos em relação a aspectos como consistência e completude (verificação) e identificar se os requisitos correspondem às expectativas dos clientes e usuários (validação). O contexto geral onde se insere o sistema a ser desenvolvido é denominado de Universo de Informações (UdI) e inclui todas as fontes de informação e pessoas relacionadas ao software (este contexto também é identificado como Universo do Discurso ou domínio da aplicação).

Durante a fase de elicitação a comunicação entre engenheiro de requisitos e clientes e usuários visa à identificação de funcionalidades a serem atendidas pelo sistema a ser desenvolvido. Neste processo, reconhecidamente um dos mais intensivos em comunicação no desenvolvimento de software, surgem termos próprios do domínio da aplicação. Tais termos devem ser registrados num léxico para a aplicação, possibilitando a todos os interessados o compartilhamento de uma mesma compreensão desse Universo de Informação. Várias são as técnicas utilizadas para a elicitação dos requisitos: entrevistas, reuniões, leitura de documentos, workshops e mesmo reuso de requisitos. O sistema a ser desenvolvido deverá ainda respeitar regras e padrões da organização para o qual ele está sendo construído, além de atender à legislação em vigor. Restrições derivadas do contexto de operação também devem ser consideradas e, portanto, nesta etapa não apenas os clientes e usuários devem ser ouvidos.

Na fase de modelagem os requisitos são registrados de acordo com um modelo tal como sentenças de requisitos, cenários, casos de uso, estórias do usuário. O uso da linguagem natural no registro dos requisitos facilita a comunicação entre os clientes, usuários e engenheiro de requisitos e possibilita, posteriormente, a validação desses mesmos requisitos por parte dos clientes e

usuários. O uso de modelos formais para o registro de requisitos facilita atividades de verificação, mas dificulta atividades de validação, pois clientes e usuários teriam que utilizar e compreender esses modelos formais; já o uso da linguagem natural não traz dificuldades adicionais ao processo, pois ela é a linguagem utilizada normalmente pelos interessados.

Na fase de análise os requisitos são verificados em relação ao modelo sendo utilizado e em relação ao atendimento das solicitações de clientes e usuários. Durante a verificação busca-se identificar no documento de requisitos discrepâncias, erros e omissões; este processo é realizado pelos profissionais de software. O documento é avaliado em relação ao modelo utilizado para registro dos requisitos, e uma avaliação sintática verifica se o documento de requisitos respeita a sintaxe do modelo utilizado. Também é verificado se o documento atende aos padrões estabelecidos pela organização, ou atende a modelos de qualidade. Verificações semânticas são realizadas, analisando-se o conjunto de requisitos e buscando encontrar inconsistências entre pares ou conjuntos de requisitos. A avaliação em relação à completude é parte deste processo. O objetivo desta fase é garantir a qualidade do documento que será utilizado posteriormente como guia para as demais atividades do processo de desenvolvimento do software.

Na validação o conjunto de requisitos é avaliado pelos clientes e usuários, e busca-se confirmar se suas necessidades e expectativas estão contempladas no documento analisado. Assim como na fase de elicitação, é importante que, na validação, participem usuários e clientes representando as diferentes visões dos envolvidos ou atingidos pelo sistema em desenvolvimento. Em caso de conflitos, negociações entre os interessados poderão solucionar o impasse ou levar a novas definições. Dependendo da técnica utilizada para o processo de desenvolvimento, a priorização dos requisitos junto com o cliente definirá quais os requisitos a serem trabalhados em cada incremento do software a ser liberado. Estas atividades resultam no comprometimento de clientes e usuários com os requisitos registrados no documento de requisitos.

Kotonya e Sommerville [Kotonya98, Sommerville04] estruturam as atividades do processo de requisitos nas fases de estudo de viabilidade, elicitação e análise, validação e gerenciamento de requisitos. O estudo de viabilidade utiliza regras do negócio, alguns requisitos preliminares e uma descrição de alto nível do

sistema a ser desenvolvido, apresentando a forma como o sistema deverá suportar ou apoiar processos do negócio. Este estudo resulta num parecer recomendando ou não a continuidade do projeto. Na fase de elicitação e análise de requisitos os interessados buscam obter maior conhecimento sobre o domínio da aplicação e identificar tanto os serviços que o sistema deverá prover como as restrições operacionais a respeitar.

É ainda na fase de análise que são identificados e negociados os eventuais conflitos entre requisitos, derivados das diferentes necessidades trazidas pelos interessados, e efetuada a priorização dos requisitos. As atividades relacionadas à representação dos requisitos em algum modelo ou linguagem padronizada são denominadas de especificação dos requisitos. A fase de validação busca mostrar que os requisitos correspondem ao sistema que o usuário deseja obter. Os requisitos são checados ainda em relação a aspectos como consistência, completude e testabilidade, e também quanto à viabilidade de sua implementação. A Figura 3, adaptada de [Sommerville04], apresenta esquematicamente estas fases.

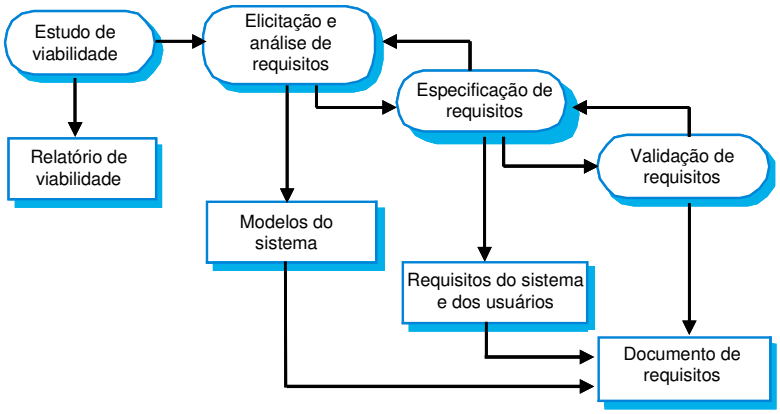

Figura 3 - Processo de Requisitos [Sommerville04]

Ao longo do Processo de Requisitos, e mesmo durante o processo de desenvolvimento do software, não é incomum que os requisitos já definidos sofram alterações devido a diferentes motivos como por exemplo mudanças no contexto onde o software está inserido, novas expectativas por parte dos clientes/usuários, negociação entre clientes e desenvolvedores, etc. Tanto a proposta de Leite quanto a de Kotonya e Sommerville consideram e acomodam a inevitabilidade dessas mudanças. A Figura 4, adaptada de [Sommerville04], mostra um modelo genérico para as etapas que envolvem elicitação, classificação e organização dos requisitos, priorização e negociação, e a documentação dos

requisitos. Como um modelo geral, ele pode ser instanciado a cada processo de desenvolvimento; a espiral mostra que estas atividades são cíclicas, e a cada execução do ciclo deve aumentar a compreensão dos interessados e a completude dos requisitos.

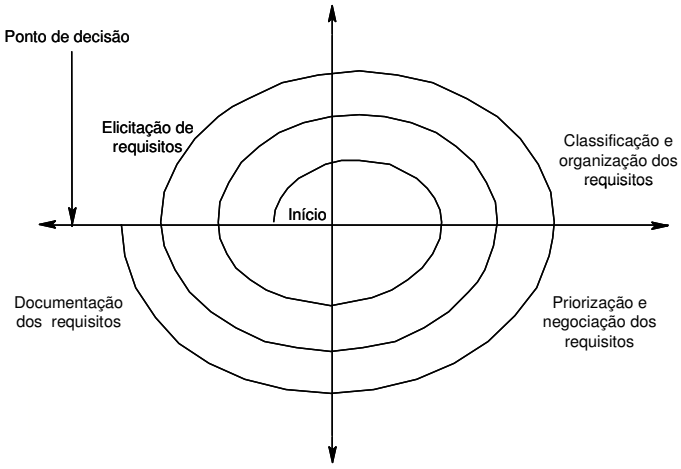

Figura 4 - Ciclo de elicitação e análise [Sommerville04]

Os vários artefatos gerados durante o processo de requisitos visam registrar as necessidades e expectativas dos interessados, facilitar a comunicação com clientes e usuários e também servir como base para o desenho e implementação do sistema. Um desses artefatos, denominado de *Documento de Requisitos*, é a base para o desenvolvimento do software; os requisitos nele registrados delimitam a abrangência do software, estabelecem funcionalidades requisitadas pelo conjunto de clientes e usuários, fornecem subsídios para o processo de verificação e validação do software construído. Não se questiona a importância do processo de requisitos para o sucesso de um projeto de software; sem um documento de requisitos de qualidade, as estimativas de custos ficam prejudicadas, o cronograma de execução passa a ser apenas uma estimativa, não há como afirmar que um projeto foi concluído (como saber se todos os requisitos foram implementados?), o processo de validação do software pelos usuários é dificultado.

Estas duas abordagens para o processo de requisitos diferem na estruturação das várias atividades, mas coincidem na essência das mesmas. O que Leite denomina de análise de requisitos, englobando verificação e validação, Kotonya e Sommerville chamam de validação. Adotaremos neste trabalho a estruturação de

Leite, dado que ela separa mais claramente as atividades a serem executadas nesta fase.

O conjunto de atores envolvido num processo de requisitos inclui representantes do cliente e dos usuários, gerente do projeto e engenheiros de requisitos, de software e de sistemas, entre outros. Cada um desses atores participa visando atingir seus próprios objetivos, e a colaboração entre eles é necessária pois existe uma meta maior associada a cada fase. Considerando especificamente as atividades de V&V: (i) na verificação é preciso determinar se um artefato de requisitos atende aos preceitos de qualidade estabelecidos pela organização; significa verificar se ele está internamente completo, consistente e correto, o que possibilita passar às etapas seguintes no processo de desenvolvimento e (ii) na validação é preciso assegurar que este mesmo artefato atende às necessidades dos clientes e usuários.

Repetindo o que já foi expresso no capítulo 1: na verificação, a meta envolve obter respostas à pergunta "Estamos construindo o produto corretamente?". Na validação, a pergunta a ser respondida é "Estamos construindo o produto desejado pelos clientes e usuários?" [Sommerville04]. Essas meta-questões serão respondidas pelos interessados, com utilização de métodos e técnicas apropriados a cada caso.

2.2. Processo de requisitos em ambientes distribuídos de desenvolvimento

Os desafios enfrentados no DDS têm sido avaliados segundo diferentes dimensões. Segundo [Paré99], times virtuais devem ser avaliados segundo as dimensões de tecnologia, contexto (ambiental e organizacional), processo e dinâmica do time (padrões de comunicação, cooperação, compartilhamento de informações) e estratégias para gerenciamento do desenvolvimento. Este *framework* [Paré99] foi utilizado como ponto de partida para um estudo em empresas utilizando DDS; os resultados desse estudo mostram que a distância dificulta o comprometimento dos times e alinhamento com propósitos assumidos. Atividades sociais ou mesmo encontros casuais também são minimizadas; tais atividades permitem o desenvolvimento do espírito de equipe e diminuem a possibilidade de ocorrência de conflitos. A distância também afeta atividades do

gerenciamento do projeto: a liderança utiliza a comunicação para manter a união da equipe e motivar para o andamento do projeto. Projetos bem sucedidos exigem planejamento detalhado, maior esforço e disciplina para seu acompanhamento, gerenciamento por objetivos e uso de métodos padronizados.

Considerando-se especificamente o Processo de Requisitos, [Damian03a] esquematiza conforme Figura 5 as dimensões derivadas da distribuição dos interessados, os desafios identificados e as atividades por eles afetadas. Nesse estudo, os problemas derivados da distribuição geográfica são vistos como associados à comunicação, conhecimento, cultura e diferenças temporais.

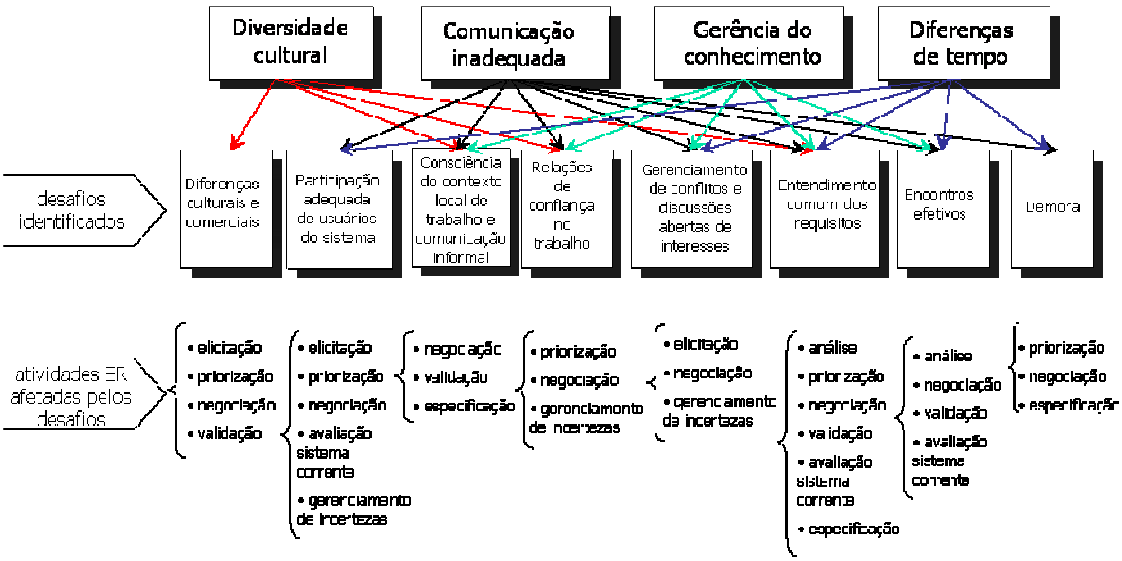

Figura 5 - Impactos do desenvolvimento distribuído no Processo de Requisitos [Damian03a]

Conforme pode ser verificado na Figura 5, atividades de verificação e validação são diretamente atingidas pelos seguintes fatores: diversidade cultural, comunicação inadequada, gerência do conhecimento e diferenças temporais. A literatura sobre o tema apresenta inúmeros trabalhos investigativos do impacto da distribuição das atividades no processo de desenvolvimento, e os resultados encontrados não são divergentes na sua essência [Carmel99] [Paré99] [Bianchi02] [Herbsleb03] [Prikladnicki04] [Cherry04].

Concluindo, como resultado da literatura pesquisada e das informações iniciais coletadas junto a empresas, os principais desafios são:

a) **comunicação e linguagem**: a linguagem natural, normalmente utilizada para registrar os requisitos, é inerentemente ambígua, e além disso, as

diferenças culturais podem levar a uma compreensão equivocada. Conflitos entre requisitos colocados por diferentes interessados também são mais difíceis de serem negociados; a convergência de idéias é comprometida. Do nosso ponto de vista, sem dúvida o fato da língua utilizada para comunicação e documentação não ser a língua nativa de todos os envolvidos é o fator de maior impacto dentre as diferenças culturais. Dos relatos obtidos junto às organizações objeto da avaliação inicial do nosso estudo de caso, mesmo quando utilizada a mesma língua podem ocorrer ambigüidades - caso do português do Brasil e de Portugal;

b) **fusos horários**: se adequadamente explorados, os diferentes fusos horários das equipes podem levar a um bom aproveitamento do tempo (caso do desenvolvimento *follow-the-sun* ou *24-hour*). Em oposição, se a localização geográfica das equipes e as diferenças de horário não possibilitarem coincidência parcial de turnos de trabalho, encontros virtuais serão possíveis apenas se uma das partes se dispuser a modificar seu horário de trabalho, antecipando ou postergando horários de entrada ou de saída do trabalho, ou mesmo de intervalos. Atividades do Processo de Requisitos que exigem negociação ou comprometimento entre interessados são diretamente impactadas pela diferença de fusos horários;

c) **fatores culturais e contextuais e distribuição do trabalho**: fatores culturais e contextuais estão associados à confiança entre equipes [Paré99] [Audy04] [Damian03] e impactam decisões gerenciais relacionadas à distribuição do trabalho. Confiança é necessária para manutenção do espírito de equipe, para troca efetiva de informações entre integrantes de equipes distantes e para realizar a atribuição das tarefas. Aspectos associados à linguagem estão colocados no item a acima;

d) **gerenciamento do conhecimento**: o volume de informações a ser compartilhado durante o desenvolvimento de um sistema envolve documentação técnica, artefatos gerados nas várias etapas do processo, registros de comunicações formais e informais trocadas entre clientes, usuários e desenvolvedores, "dicionários de viagem", ... O compartilhamento efetivo dessas informações é um desafio: mesmo com a utilização de um repositório [Gorton96] [Damian03] existem decisões associadas: utilizar repositório central, ou distribuir as informações?

replicar a base de informações? que informações devem ser compartilhadas? como manter desenvolvedores atualizados em relação a alterações importantes?

e) **coordenação do projeto**: atividades de gerenciamento e coordenação do desenvolvimento são diretamente afetadas pela distribuição geográfica: a atenção do gerente será compartilhada entre as várias equipes, exigindo maior disciplina e atenção. O gerente precisa obter o comprometimento das várias equipes, motivá-las e manter o espírito de grupo mesmo quando parte das equipes está distante. Análises de impacto motivadas por alterações em requisitos poderão implicar em modificações no trabalho das equipes distantes. Para gerar novas estimativas de prazos, o gerente necessitará ter conhecimento da carga de trabalho da equipe distante. Eventos como este aumentam as necessidades de comunicação. Ainda, se o sistema a ser desenvolvido necessita de hardware e/ou software especial, o gerente deve se certificar que todas as equipes têm acesso aos recursos necessários, evitando que as atividades de desenvolvimento sofram interrupções.

2.3. Análise de Requisitos: importância das atividades de verificação e validação

É fato conhecido que o custo de descobrir e corrigir um defeito na fase de testes de software é de 5 a 100 vezes maior que o custo de descobrir e corrigir o problema ainda no Processo de Requisitos [Boehm76] [Bohem01] [Rosenberg98] [Blackburn01]. A correção de um defeito identificado na fase de testes pode implicar em re-trabalho nos artefatos gerados nas fases anteriores: requisitos, arquitetura, projeto e implementação. Problemas decorrentes de atividades desenvolvidas ainda no processo de requisitos são reportados tanto pela literatura acadêmica como por empresas que atuam no mercado de desenvolvimento.

Após avaliar um bom conjunto de organizações, Capers Jones [Jones96] afirma que o processo de requisitos é deficiente em mais de 75% das empresas; isto mostra que obter os requisitos corretos é também uma das mais difíceis atividades de um projeto de software. Um estudo envolvendo empresas européias de desenvolvimento de software registrou que mais de 50% delas identificou

problemas na área de especificação de requisitos [El-Emam00].

Defeitos em requisitos, isoladamente, é a falha com maior freqüência em projetos de software: erros em requisitos são responsáveis por uma taxa de 70 a 85% do custo do re-trabalho [Lefingwell97] [Bohem01].Estudos relativamente recentes indicam que aproximadamente 50% das falhas detectadas na fase de testes são causadas por defeitos em requisitos [Blackburn01], e que a correção dessas falhas implica em re-trabalho muitas vezes evitável. Detectar e corrigir problemas ainda na fase de requisitos é, portanto, uma medida fundamental para contribuir para a conclusão do software dentro dos custos e cronogramas previstos.

Em relação à importância da validação, lembramos que o Chaos Report 2004 do Standish Group [Standish04], envolvendo 9.236 projetos de software, mostra que mesmo entre os projetos que foram concluídos no prazo e dentro do orçamento previsto, apenas 66% atenderam a todas as características solicitadas pelos clientes. Isto reafirma a relevância do envolvimento de clientes e usuários no processo de validação dos requisitos, para que o conjunto de requisitos definidos realmente corresponda às expectativas.

No contexto do desenvolvimento distribuído, os participantes desse processo estarão geograficamente separados. A interação face-a-face ou mesmo mediada por ferramentas de comunicação nem sempre será viável, devido à distância e à diferença de fusos horários entre clientes, usuários e engenheiros de requisitos. Nesse caso as dificuldades inerentes aos processos de verificação e validação serão amplificadas, devido às dificuldades de comunicação e diferenças culturais já relatadas.

A automação parcial de atividades associadas aos processos de verificação e validação contribuirá para a descoberta de defeitos ainda no processo de requisitos, o que, acreditamos, melhorará a qualidade do documento de requisitos e diminuirá o re-trabalho decorrente da descoberta tardia de defeitos.

2.4. Verificação e Validação: uso de PLN e Agentes de Software

Técnicas de processamento da linguagem natural têm sido aplicadas a documentos de requisitos com diferentes objetivos: melhoria da qualidade dos

requisitos, agrupamento de requisitos relacionados, identificação de abstrações, identificação de entidades relevantes. Por outro lado, agentes tem sido utilizados em diversas áreas do conhecimento, e também no contexto de desenvolvimento distribuído de software. Apresentaremos alguns desses trabalhos de forma breve, dado que eles são relacionados ao trabalho desenvolvido nesta tese.

Um trabalho pioneiro na avaliação da qualidade de documentos de requisitos escritos em linguagem natural foi desenvolvido pelo Software Assurance Technology Center (SATC) do Goddard Space Flight Center (GSFC-NASA) [Hammer96] [Wilson97] [Rosenberg98]. Uma ferramenta, denominada de ARM (Automated Requirement Measurement), extrai um conjunto de métricas. A ferramenta obtém valores indicativos de linhas de texto (indicador do tamanho do documento), imperativos (frases indicando ação), continuação (frases que introduzem a especificação de requisitos de nível mais baixo), diretivas (referências para tabelas, figuras, notas), frases dúbias (contendo palavras ambíguas), incompletos (os denominados TBD´s - *to be defined*) e escolhas/opções (palavras que mostram ausência de definição). Os valores obtidos devem ser comparados em relação à média de outros documentos da organização, apontando desvios e pontos que devem receber avaliação e talvez medidas corretivas.

Gervasi e Nuseibeh [Gervasi02] defendem a utilização de métodos formais "leves" para a validação automática de requisitos escritos em linguagem natural. Um documento de requisitos com aproximadamente 250 páginas foi analisado com a utilização de diversas ferramentas, entre elas Circe e Cico, que dão suporte à extração de modelos dos requisitos e sua validação. O trabalho realizado coletou também um conjunto de métricas sobre o documento de requisitos, o sistema nele descrito e o próprio processo de escrita do documento.

Em outra linha de trabalho, Palmer e Liang [Palmer92] buscaram técnicas para agrupar requisitos, idealizando um algoritmo que foi denominado de Two Tiered Clustering (TTC). O objetivo do algoritmo é agrupar um conjunto de M requisitos, de forma que o número N de grupos seja tal que N<<M. Os requisitos são inicialmente agrupados considerando os verbos extraídos do documento, e um thesaurus de verbos cuja construção teve por base um trabalho anterior em requisitos de um sistema operacional. Esses grupos iniciais são então subdivididos em conjuntos similares, utilizando a medida do coseno (seção 2.5.3).

O gerenciamento de processos distribuídos de engenharia de software, apoiado por agentes, foi descrito em [Gaeta02]. Desenvolvido no contexto do projeto GENESIS (GEneralised eNvironment for procEsS management in cooperatIve Software engineering), visando apoiar o gerenciamento de projetos e a comunicação entre engenheiros de software, responsabiliza agentes pela manipulação de exceções, pela sincronização de processos entre os sites distribuídos e pela monitoração e coleta de informações relacionadas a processos.

A utilização de agentes de software para atividades de gerenciamento e controle da evolução de requisitos em ambientes distribuídos de desenvolvimento foi relatado em [Chang01]: agentes móveis foram utilizados para gerenciamento e controle de versões de documentos de requisitos, distribuídos pelos diversos sites de uma organização.

Na área de testes de software, agentes foram propostos como um novo paradigma para incrementar o desenvolvimento de software numa ampla gama de aplicações [Ponnurangam05]. O sistema é composto por agentes pessoais, que interagem com o usuário e executam tarefas específicas, e agentes de serviço, que executam em *background* e atendem ao usuário de forma indireta. Os agentes de serviço ou são especializados em algum tipo de teste de software, ou são responsáveis pela distribuição de recursos de forma eqüitativa e eficiente.

Técnicas de processamento da linguagem natural e uso de agentes de software para apoio a atividades do processo de desenvolvimento de software, portanto, já têm sido apresentados na literatura. Esta proposta utiliza algumas técnicas de tratamento da linguagem natural para apoiar atividades de V&V em requisitos, e agentes de software na ferramenta de suporte.

2.5. Alguns métodos e técnicas de PLN para apoio ao processo proposto

Na proposta apresentada nesta tese as estratégias para apoio às atividades de verificação e validação de artefatos de requisitos utilizam diversos recursos da área de Processamento da Linguagem Natural. Nas seções a seguir descreveremos brevemente os métodos, técnicas e ferramentas dessa área que subsidiaram a construção da nossa estratégia de trabalho para V&V.

2.5.1.
Part-Of-Speech Tagger (POS tagger)

Em processamento da linguagem natural *taggers* são sistemas que analisam um texto e inserem etiquetas morfológicas, gramaticais ou sintáticas a cada item lexical. Um *part-of-speech tagger* é um etiquetador morfossintático, que analisa o texto e identifica as categorias gramaticais como substantivos, adjetivos, pronomes e verbos, dentre outras. Para sinais de pontuação pode ser utilizado o próprio sinal, enquanto que para palavras estrangeiras e fórmulas utiliza-se um rótulo único.

O etiquetador basicamente insere uma etiqueta do conjunto utilizado (*tagset*) junto ao texto; nesta tarefa ele pode utilizar um léxico e um conjunto de procedimentos que apóiam o processo de definir a etiqueta a ser utilizada numa determinada palavra. Estes dois componentes – léxico e conjunto de procedimentos – fazem parte do modelo da língua utilizado para a tarefa de etiquetagem [Aires00]. Alguns etiquetadores exigem que o texto de entrada esteja num formato específico - por exemplo, o etiquetador pode exigir que o texto esteja *tokenizado*, com apenas uma palavra ou caracter de pontuação por linha. A Figura 6 mostra um esquema genérico para um etiquetador morfossintático, com a fase de *tokenização* já incorporada.

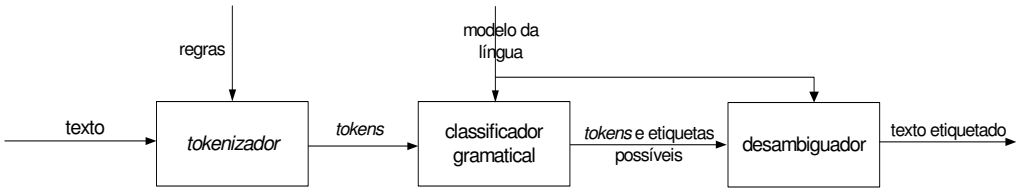

Figura 6 - Visão geral de um etiquetador morfossintático

Após o texto ser *tokenizado* a fase de classificação gramatical tem início. Para cada *token* o classificador busca no léxico as classes gramaticais possíveis. Se o *token* não é encontrado no léxico, então o etiquetador utiliza procedimentos específicos visando encontrar uma classificação do mesmo. Nos casos onde há ambigüidade, ou seja, o *token* pode receber mais de uma etiqueta, o desambiguador utiliza informações do contexto para realizar a desambiguação e definir a etiqueta mais provável para o *token* sendo analisado.

O modelo da língua utilizado pelo etiquetador pode ser baseado em regras,

em casos ou em árvores de decisão, e neste caso o etiquetador é denominado de simbólico ou lingüístico. O modelo pode utilizar representação baseada em Markov, em árvores de decisão probabilísticas ou distribuição estatística de palavras no texto, e neste caso o etiquetador é denominado de probabilístico ou estatístico. Neste trabalho foi utilizado o etiquetador QTAG, detalhado em [Mason97], e que é classificado como um etiquetador probabilístico.

O QTAG é denominado de etiquetador probabilístico porque utiliza a etiqueta mais provável para um termo específico, após identificar no modelo de linguagem todas as possíveis etiquetas (alguns termos podem ser classificados em várias categorias). Para casos onde ao termo sendo analisado podem ser atribuídas diversas categorias, a definição da etiqueta a ser utilizada considera o contexto no qual o termo está inserido. Isto significa que etiquetas de termos próximos ao termo em pauta também são avaliadas, e será utilizada a etiqueta correspondente à seqüência de maior freqüência.

O QTAG utiliza um dicionário que registra os termos da língua (o léxico) juntamente com as probabilidades associadas a cada possível etiqueta para o termo. Inicialmente o etiquetador consulta o dicionário, obtém informações sobre o termo sendo avaliado e as combina com informações do contexto, considerando as etiquetas dos dois termos anteriores. A etapa seguinte avalia ainda o termo em pauta em relação aos dois termos posteriores, e a etiqueta de maior probabilidade de ocorrência é escolhida.

Os recursos utilizados por este etiquetador são um dicionário de termos, com etiquetas e freqüências associadas, e uma matriz de seqüências de etiquetas e respectivas freqüências. Estes recursos podem ser abstraídos de um corpus anotado, o que possibilita que ele seja adaptado para trabalhar com diferentes linguagens; apesar de ter sido desenvolvido originalmente para a língua inglesa, já existem modelos para os idiomas romeno e sueco [Tufis98], além do português.

Para a língua inglesa, a taxa de precisão das etiquetas inseridas é da ordem de 96,3% [Mason97]. Para a língua portuguesa, este etiquetador foi treinado pelos pesquisadores T. Sardinha e R. Lima-Lopes, associados ao Lael/PUCSP, utilizando um corpus de textos jornalísticos composto por aproximadamente 500 mil palavras e etiquetado por lingüistas. Conforme dados experimentais, a precisão deste etiquetador para textos em língua portuguesa é da ordem de 93% [Sardinha04]. O conjunto de etiquetas para a língua portuguesa está relacionado

na Tabela 1, e a Tabela 2 apresenta o resultado do etiquetador para a frase "O modelo de língua utilizado pelo etiquetador pode ser baseado em regras.".

Tabela 1 - Etiquetas utilizadas pelo QTAG

Etiqueta	Significado
ADJ	Adjetivo
ADV	Advérbio
ARTD	Artigo definido
ARTI	Artigo indefinido
CJ	Conjunção
CPR	Contração de preposição e artigo, pronome ou advérbio
ESTR	Palavra estrangeira
IN	Interjeição
N	Substantivo
NUM	Numeral
PART	Particípio passado
PRN	Pronome
PRP	Preposição
PT	Pontuação
V	Verbo

Tabela 2 - Frase etiquetada pelo QTAG

```
<w pos="ARTD">O</w>
<w pos="N">modelo</w>
<w pos="CPR">da</w>
<w pos="N">língua</w>
<w pos="PART">utilizado</w>
<w pos="CPR">pelo</w>
<w pos="N">etiquetador</w>
<w pos="V">pode</w>
<w pos="V">ser</w>
<w pos="PART">baseado</w>
<w pos="PRP">em</w>
<w pos="N">regras</w>
<w pos="PT">.</w>
```

No contexto desta tese, este etiquetador foi utilizado como parte dos procedimentos necessários para a identificação de termos ou expressões representando atores em documentos gerados e/ou manipulados durante o processo de requisitos. A descrição completa dos procedimentos é apresentada no capítulo 3.

2.5.2. Representação de documentos: abordagem bag-of-words

Um dos problemas a ser resolvido quando visamos ao processamento automatizado de documentos escritos em linguagem natural é a escolha da estrutura a utilizar para a representação desses documentos; este problema já foi enfrentado pelas áreas de Recuperação de Informação (RI) e Mineração de Textos (MT). Em RI o problema consiste em recuperar documentos previamente armazenados num repositório, de forma a atender a uma determinada consulta realizada por um usuário. Em MT, o problema consiste em extrair conhecimento de um repositório, possivelmente utilizando algoritmos de aprendizado de máquina. Em ambos os casos impõe-se a necessidade de estruturar os documentos

analisados, visando possibilitar a sua manipulação.

Uma das possíveis abordagens para a estruturação de documentos é denominada de *bag-of-words*. Nessa abordagem, cada documento é representado como um vetor dos termos que ocorrem no documento: os vetores são derivados de matrizes termo-documento, cuja estrutura é similar à apresentada na Tabela 3. Em matrizes termo-documento, colunas representam termos e linhas representam documentos. Cada um dos *n* documentos é representado por um vetor de tamanho *t*, onde *t* corresponde ao número de termos; o vetor pode representar todos os termos existentes no conjunto de documentos, ou ainda todos os termos relacionados num dicionário. Os valores nas células registram o peso que um determinado termo representa para cada um dos documentos que faz parte do repositório ou do conjunto avaliado.

Tabela 3 - Estrutura genérica para uma matriz termo-documento

	$termo_1$	$termo_2$	$termo_3$	$termo_4$	.....	$termo_t$
doc_1	$peso_{11}$	$peso_{12}$	$peso_{13}$	$peso_{14}$	.....	$peso_{1t}$
doc_2	$peso_{21}$	$peso_{22}$	$peso_{23}$	$peso_{24}$	.....	$peso_{2t}$
.....	.....	.....	.....	.....	.....	.....
doc_n	$peso_{n1}$	$peso_{n2}$	$peso_{n3}$	$peso_{n4}$	.....	$peso_{nt}$

O peso de um termo indica a relevância desse termo para o documento em questão, e seu valor é nulo para termos que não estejam presentes no documento. Existem diversas abordagens para definição do peso de cada termo num documento, a freqüência é uma medida presente em várias fórmulas para o cálculo do peso.

Entre as abordagens mais freqüentemente utilizadas estão *boolean*, *tf (*abreviação para *term frequency*) ou *tfidf (*abreviação para *term frequency inverse document frequency*).

Na medida *boolean*, o valor do peso é zero se o termo não está presente no documento, ou 1, caso contrário. Esta é uma medida extremamente simples de ser computada, e quando utilizada em operações de RI causa o retorno de todos os documentos da coleção que contenham o termo objeto da consulta. O uso da medida *boolean* implica em considerar que todos os documentos que contenham o termo são igualmente relevantes.

Na medida *tf* o valor do peso é a própria freqüência do termo no

documento. Em operações de RI isto torna viável classificar os documentos obtidos em resposta a uma dada consulta, colocando no topo da lista aqueles que se acredita melhor respondam à consulta realizada.

A medida *tfidf* , apresentada na equação (1), é calculada ponderando a freqüência do termo por um fator que minimiza o peso de termos presentes em grande parte dos documentos, pois tais termos não são considerados discriminantes. Termos menos freqüentes ou raros terão um fator de ponderação maior. Esta medida é bastante utilizada em processos de MT.

$$tfidf(t_j, d_i) = freq(t_j, d_i) \times \log \frac{n}{d(t_j)} \quad (1)$$

Nesta fórmula, $freq(t_j, d_i)$ representa a freqüência do termo j no documento i, n representa a quantidade de documentos da coleção e $d(t_j)$ representa a quantidade de documentos que contém o termo t_j. Termos que estejam presentes em todos os documentos terão peso nulo, pois log 1 vale zero; o uso da função *log* evita que um termo que apareça em apenas um documento tenha peso duas vezes maior que outro termo que esteja presente em dois documentos da coleção.

Um dos problemas enfrentados quando se utiliza a abordagem *bag-of-words* para estruturação de documentos, considerando todos os termos presentes no conjunto de documentos, é a grande dimensão da matriz termo-documento. Cada documento é representado por um vetor de termos, e todos os termos presentes no conjunto de documentos devem ser representados nesses vetores. A matriz gerada é possivelmente uma matriz esparsa, pois mesmo termos que apareçam em um único documento deverão estar representados. A redução da dimensionalidade da matriz de termos pode ser obtida com o uso dos *stems* dos termos (vide seção 2.4.4.) ou seleção dos termos a representar.

Uma maneira de encontrar termos pouco representativos combina as leis de Zipf [Zipf49] e os cortes de Luhn [Luhn58]. A teoria de Zipf afirma que o produto da freqüência de um termo e sua classificação (*rank*) é aproximadamente constante. Obtém-se a freqüência dos termos de um conjunto de documentos, gera-se a classificação desses termos em ordem descendente de freqüência (*rank*) e gera-se um gráfico de classificação versus freqüência, como pode ser visualizado na Figura 7.

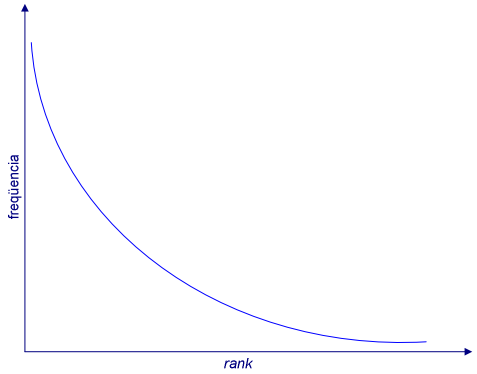

Figura 7 - Curva segundo a lei de Zipf

Esta teoria foi utilizada posteriormente por Luhn [Luhn58] que ponderou que os termos relevantes para a indexação de documentos ficam concentrados numa região delimitada por dois pontos de corte. Termos acima do limite superior e termos abaixo do limite inferior são considerados pouco discriminantes, por serem muito freqüentes ou por serem muito raros. Uma das formas de implementar o conjunto de termos que pertencem ao limiar superior é agrupar artigos, pronomes, advérbios, preposições e conjunções num conjunto denominado de *stoplist*. A Figura 8 mostra limites inferior e superior para uma curva de Zipf, segundo a abordagem de Luhn.

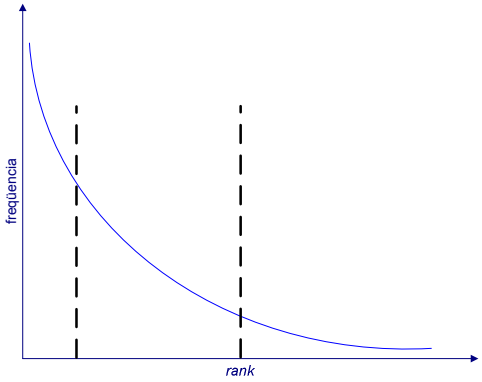

Figura 8 - Curva de Zipf com os cortes propostos por Luhn

A região que engloba os termos com maior freqüência inclui termos como artigos (o, a), preposições (de, da), conjunções (e, ou), que estão presentes em quase todos os documentos e não são significativos para representação dos documentos. A região central inclui termos como substantivos, adjetivos, verbos e que podem ser utilizados para a representação de documentos. A terceira região inclui termos com baixa freqüência de ocorrência, muitas vezes apenas uma ocorrência, e que são considerados “ruídos”.

A delimitação dessas regiões não é tarefa trivial, e à decisão sobre os limites está associado um certo grau de arbitrariedade. Para a terceira região, [Meadow00] sugere que seja utilizado o limiar de 1 ou 2 para a freqüência, [Daile96] sugere que o limiar utilizado seja 4. Para a primeira região, é comum o uso de um conjunto de *stopwords*[2] que agrega os termos da língua que são artigos, pronomes, preposições, conjunções. [Meadow00] observa que o conjunto de *stopwords* é sensível ao contexto: por exemplo, retirar o termo A num contexto de saúde pode excluir indevidamente referências à vitamina A.

No contexto desta tese a representação de documentos utilizou a abordagem *bag-of-words* para apoiar a identificação de termos relevantes em documentos gerados ou manipulados no processo de requisitos, e também para apoiar a identificação de similaridades entre requisitos.

2.5.3. Similaridade entre documentos

Um das formas de se identificar documentos similares utiliza a representação espaço-vetorial. No modelo espaço-vetorial [Salton88] são utilizadas matrizes termo-documento, já referidas na seção anterior. Cada elemento ou termo do vetor é considerado uma coordenada num espaço vetorial euclidiano t-dimensional, e a posição do documento em cada dimensão é dada pelo peso associado. Documentos localizados numa mesma região desse espaço possuem conteúdos similares, e uma forma de determinar essa similaridade está relacionada ao ângulo entre documentos representados nesse espaço vetorial. Uma representação do espaço vetorial assim definido pode ser visualizada na Figura 9.

[2] *Stopwords* são aqueles termos da língua geral ou elementos tais como preposições, artigos, conjunções e outros termos que não apresentam relevância ou valor terminológico [Teline03]; o conjunto desses termos é denominado de *stoplist*

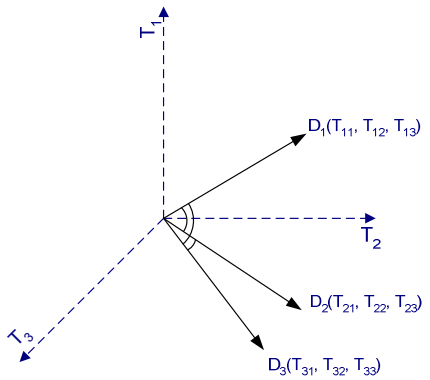

Figura 9 - Espaço vetorial para representação de documentos [Salton83]

No exemplo da Figura 9, o espaço vetorial é definido pelos eixos T_1, T_2 e T_3, que correspondem aos três termos utilizados na representação dos documentos D_1, D_2 e D_3. Os valores utilizados para a representação de cada documento, nesse espaço, são determinados pelos pesos associados aos termos T_1, T_2 e T_3 (e representados, na figura, por T_{11}, T_{12}, T_{13}, ...). Pode-se observar que o ângulo formado pelos vetores que representam os documentos D_2 e D_3 é menor que os ângulos formados por D_1 e D_3 ou D_1 e D_2, indicando um maior grau de similaridade entre eles. Métricas de similaridade baseadas nesta representação são a distância euclidiana e o coseno do ângulo formado pelos documentos.

Tomando como exemplo dois documentos quaisquer, que chamaremos de *x* e *y*, e adotando a representação espaço-vetorial para ambos, considerando um conjunto de *t* termos, o cálculo do coseno do ângulo formado pelo par de documentos é dado pela fórmula apresentada em (2a), e o cálculo da distancia euclidiana é dado pela fórmula em (2b).

$$\cos(x, y) = \frac{\sum_{i=1}^{t} x_i \, y_i}{\sqrt{\sum_{i=1}^{t} x_i^2} \sqrt{\sum_{i=1}^{t} y_i^2}} \quad (2a) \qquad \delta = \sqrt{\sum_{i=1}^{t} (x_i - y_i)^2} \quad (2b)$$

No cálculo do coseno, o numerador representa um produto vetorial, correspondendo à soma dos produtos dos pesos associados aos termos nos vetores representando *x* e *y*. Cada um dos componentes do denominador representa uma função sobre um único vetor, e seu produto é um fator de normalização para a medida. Se dois documentos são exatamente iguais, então seus vetores serão superpostos no espaço vetorial, o ângulo entre eles será nulo e o valor do coseno será 1. Como o denominador utiliza sempre valores positivos, teremos que valores

obtidos com aplicação desta fórmula resultarão sempre no intervalo [0,1]; quanto mais próximo de 1 o valor obtido, maior a similaridade entre os documentos.

O cálculo da distância euclidiana mostra quão próximos ou distantes estão dois documentos representados no espaço de documentos. Se aplicadas a vetores normalizados, os resultados dessas duas medidas serão idênticos [Maning99].

Outras formas para identificação de similaridade freqüentemente encontradas na literatura [Salton83] [Kowalski97] [Manning99] [Meadow00] são os coeficientes de Dice e de Jaccard, utilizando a mesma representação espaço-vetorial para os documentos. As fórmulas desses coeficientes estão apresentadas respectivamente nas equações (3) e (4).

$$Dice(x, y) = \frac{2\sum_{i=1}^{t} x_i \, y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2} \quad (3)$$

$$Jaccard(x, y) = \frac{\sum_{i=1}^{t} x_i \, y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2 - \sum_{i=1}^{t} x_i \, y_i} \quad (4)$$

No coeficiente de Dice, o produto vetorial das representações de x e y é dividido pela média dos pesos $\left(\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2\right)/2$ associados aos termos nos dois conjuntos.

No coeficiente de Jaccard, o produto vetorial das representações de x e y é dividido pela soma dos pesos que correspondem aos termos que não são comuns aos dois documentos. Para documentos iguais, estes dois coeficientes (Dice e Jaccard) também retornarão o valor 1.

As três medidas de similaridade apresentadas consideram, no numerador, os termos co-ocorrentes, pois se um determinado termo não ocorre em um dos documentos, seu peso será zero e o produto resultará também em zero. Desta forma, apenas termos que ocorrem em ambos os documentos serão considerados para efeito de cálculo do numerador dos coeficientes de similaridade.

No contexto desta tese, estas três medidas de similaridade foram utilizadas na identificação de requisitos similares, apoiando a verificação da existência de duplicidade em requisitos.

2.5.4. Stemização

O processo de stemização (do inglês *stemming*) visa à obtenção do radical de uma palavra, seja ela uma forma verbal flexionada, um substantivo, um adjetivo ou de outra classe de palavra. O processo de stemização é utilizado quando se deseja agrupar palavras com diferentes grafias, ou mesmo diferentes categorias gramaticais, mas relacionados a um mesmo conceito. Exemplo: as palavras guardar e guarda possuem o mesmo radical (guard), embora pertençam a classes diferentes de palavras: enquanto guardar é um verbo no infinitivo, guarda pode ser um substantivo ou outra forma flexionada do verbo guardar. Em processos onde a semântica deve se sobrepor à morfologia, o processo de stemização possibilita esse agrupamento.

A raiz (ou *stem*) resultante do processo de stemização não é necessariamente idêntica ao radical lingüístico, mas *servirá como uma denotação mínima não ambígua do termo. Stemming consiste em reduzir todas as palavras ao mesmo stem, por meio da retirada dos sufixos, permanecendo apenas um radical* [Chaves03].

Na área de recuperação de informação milhares de documentos são analisados, seus termos são extraídos e contabilizados, gerando matrizes do tipo termo-documento (vide seção 2.4.3). Os termos extraídos são utilizados como indexadores desses documentos, muitas vezes gerando o problema de manipular uma enorme escala de índices e gerar matrizes muito esparsas. Utilizar os radicais, e não os próprios termos, é uma forma de diminuir o número de índices, agilizar o processo de recuperação de informações e otimizar o espaço ocupado pelos índices, além de agrupar termos que apresentam similaridade morfológica e proximidade conceitual.

Um dos mais conhecidos algoritmos para o processo de obtenção do radical de uma palavra é o algoritmo de Porter [Porter80]. Este algoritmo está estruturado em cinco etapas, e cada uma delas gera uma transformação no termo sendo avaliado. Este algoritmo foi originalmente proposto para a língua inglesa, e foi adaptado para a língua portuguesa pelo próprio autor. Na fase de experimentação utilizamos diferentes implementações deste algoritmo, entre elas a implementação descrita em [Caldas01] e uma outra disponibilizada pelo software PreTexT

[Matsubara03]. Apesar de, na grande maioria das vezes essas duas implementações retornarem o radical que possibilita o agrupamento de palavras semanticamente relacionadas, em vários casos importantes para o nosso trabalho isso não aconteceu. A Tabela 4 apresenta alguns dos resultados obtidos do processo de extração do radical de palavras com uso dessas implementações, considerando um mesmo termo no singular e plural, ou vários termos remetendo a um mesmo conceito.

Tabela 4 - Resultados obtidos na aplicação de dois diferentes stemmers

Termo	Radical esperado	Radical obtido com [Caldas01]	Radical obtido com [Matsubara03]
hotel	hotel	hotel	hotel
hotéis	hotel	hoté	hot
avião	avi	aviã	avia
aviões	avi	aviõ	avio
aviação	avi	aviaçã	AviaCa
confirma	confirm	confirm	confirm
confirmação	confirm	confirmaçã	confirmaCa

Nosso primeiro experimento utilizou um documento de requisitos para a área do turismo, e pela tabela pode-se observar que nenhuma das implementações teve sucesso no agrupamento de palavras acentuadas ou com caracteres como o cedilha. Partimos então para a busca de um outro *stemmer*, e localizamos um algoritmo criado especificamente para a língua portuguesa. Esse algoritmo foi proposto por Orengo e Huick e está descrito em [Orengo01]. Este último algoritmo é um pouco mais sofisticado que o de Porter, pois considera as classes morfológicas das palavras sendo avaliadas. Ao termo em análise são aplicados oito procedimentos seqüenciais, a saber: redução do plural, redução do feminino, redução de advérbio, redução do aumentativo e do diminutivo, redução de formas nominais, redução das terminações verbais, redução da vogal temática e finalmente remoção dos acentos.

Na redução do plural busca-se retirar o sufixo indicativo de plural, normalmente a letra **s**. Exemplo de exceção a esta regra são as palavras que terminam em **s** e não representam termos no plural (exemplo: cais) ou palavras que terminam em **ns** e cujo singular termina em **m** (exemplo: itens). Na redução para o feminino substantivos e adjetivos terminados em **a** e são modificados para corresponder ao gênero masculino da palavra, trocando o **a** final por **o**. Encontramos como exceção neste caso palavras como portuguesa/português,

chinesa/chinês. Redução de advérbio retira o sufixo **mente**, a não ser que a palavra seja uma das exceções registradas, como a palavra experimente.

Redução do aumentativo e diminutivo retira sufixos, como inho ou inha (casinha, carrinho), indicativos de diminutivo, e aqueles indicativos de aumentativo, como ão e íssimo (buracão e felicíssimo). A redução de formas nominais atua sobre substantivos e adjetivos, identificando aqueles cuja terminação está relacionada numa lista com 61 sufixos; se sufixos são removidos nesta etapa, as próximas duas não serão executadas. Na redução das terminações verbais busca-se obter o radical do verbo, já que a língua portuguesa utiliza mais de 50 diferentes formas verbais para os verbos regulares. Na remoção de vogal temática busca-se obter o radical de palavras não tratadas anteriormente, como por exemplo a palavra menino, que será transformada no radical **menin**. O último procedimento, remoção dos acentos, retira acentos que poderiam diferenciar palavras com um mesmo radical, como por exemplo psicólogo e psicologia.

Cada procedimento possui um conjunto de regras, e apenas uma regra em cada procedimento é aplicada ao termo sendo avaliado. O procedimento visa remover o sufixo mais longo possível; no caso de redução do plural da palavra aluguéis será tratado o sufixo **éis**, ao invés de apenas **s**. O algoritmo utiliza um total de 199 regras, estruturadas num mesmo padrão. Esse padrão estabelece o sufixo a ser removido, o tamanho mínimo para o radical após a remoção, um sufixo a ser adicionado em substituição àquele sendo retirado e uma lista de palavras que são consideradas exceções à regra. Um exemplo de regra pode ser visualizado a seguir [Orengo01]:

“inho”, 3, “ ”, { “caminho”, “golfinho”, “padrinho”, “sobrinho”, “vizinho” }

Essa regra estabelece que palavras terminadas em *inho* (normalmente um diminutivo) terão este sufixo retirado, desde que o radical restante tenha tamanho mínimo 3 e que a palavra não seja uma das exceções relacionadas: caminho, golfinho, padrinho, sobrinho e vizinho.

Em [Orengo01] são relatados diversos experimentos comparando os algoritmos de Porter e de Orengo e Huick. Os resultados obtidos com a aplicação dos dois algoritmos a um mesmo vocabulário são resumidos a seguir. Considerando um conjunto inicial de 32.000 palavras, o *stemmer* de Orengo e

Huick apresentou redução de vocabulário de 51%, contra 44% obtida com o algoritmo de Porter. Um extrato de 1.000 palavras desse conjunto inicial foi então utilizado para verificação da correção dos *stems* obtidos, e o algoritmo de Orengo&Huick apresentou resultado correto em 96% dos casos, contra 71% obtidos pelo algoritmo de Porter. Um terceiro experimento foi realizado, desta vez considerando como entrada conjuntos de palavras morfológica e semanticamente relacionadas. Um novo conjunto de 1.000 palavras foi dividido em 170 grupos de palavras relacionadas, sendo efetuadas medidas utilizando índices de *overstemming* e *understemming*. Com o algoritmo de Orengo e Huick foram obtidos valores de 0,034 e 0,000985 para *understemming* e *overstemming*, e com o algoritmo de Porter foram obtidos os valores de 0,215 e 0,000211 para essas mesmas medidas. Novamente foram obtidos resultados melhores para o algoritmo de Orengo e Huick.

Após alguns experimentos realizados sobre o documento de requisitos já referido, e considerando as particularidades do nosso trabalho, optamos pelo uso do algoritmo proposto por Orengo e Huyck no contexto desta tese.

2.5.5. Concordanceador

Um concordanceador é um programa que avalia um documento e recupera os contextos nos quais uma determinada palavra ou expressão de busca está presente. Na Figura 10 observa-se os contextos para a palavra **concordanceador** considerando-se os dois primeiros parágrafos desta seção e uma janela de 5 palavras para cada lado da palavra **concordanceador**. A palavra ou expressão de busca geralmente é centralizada, e a concordância apresenta um número determinado de palavras à esquerda e à direita da palavra de busca. Numa leitura vertical podem-se observar padrões gramaticais e lexicais; uma leitura horizontal permite observar colocações e diferentes sentidos.

Um	**concordanceador**	é um programa que avalia
os contextos para a palavra	**concordanceador**	considerando-se os dois primeiros parágrafos
No ensino de línguas o	**concordanceador**	é utilizado para que o

Figura 10 - Contextos para a palavra concordanceador

As concordâncias ou contextos são exibidos e podem ser manipulados para

uso com diferentes finalidades, por exemplo para identificar colocações (palavras que aparecem freqüentemente próximas num texto), para identificar qualificadores para um substantivo, para apoio à tradução automática de textos, para identificação de expressões idiomáticas. No ensino de línguas, o concordanceador é utilizado para que o aprendiz possa identificar em que contextos uma determinada palavra costuma ser utilizada.

A forma mais comum de uso de concordanceador é denominada de KWIC (Key Words In Context), e é utilizada para a geração de índices remissivos em livros e outros documentos.

Um dos mais famosos usos de concordâncias está relacionado aos Manuscritos do Mar Morto. Esses manuscritos, descobertos na década de 40, foram colocados sob os cuidados de um grupo internacional de pesquisadores, que por muito tempo os manteve sob sigilo; até o início da década de 90 apenas um terço desses manuscritos havia sido liberada para outros estudiosos ou mesmo para o público em geral. Em 1991 um estudante, Martin Abegg, reconstruiu quase que integralmente o texto dos manuscritos tendo por base aproximadamente cinqüenta mil cartões com extratos dos manuscritos e um código de referências das palavras e sua posição nos manuscritos (indicadores de contexto ou **kwic**). Confrontado por esse trabalho, o grupo de pesquisadores então liberou os microfilmes dos manuscritos, e uma edição fac-símile dos manuscritos originais foi posteriormente publicada (veja em http://www.pennandteller.com/sincity/penn-n-teller/pcc/deadsea.html ou em http://www.byaaronhoward.com/index.php?action=details&record=7).

Dado um tema (palavra ou expressão), o concordanceador é utilizado para que sejam extraídos os contextos (*concordances*) para avaliar a existência de termos que co-ocorrem com o tema. No contexto desta tese, o concordanceador foi utilizado como parte do processo para a obtenção de colocações de temas relevantes de documentos de requisitos, visando à criação de uma taxonomia para a classificação dos requisitos.

2.5.6. Análise de conteúdo

Nas ciências sociais e humanas, duas linhas de trabalho para análise de

textos são análise do discurso e análise de conteúdo. Análise do discurso é uma metodologia qualitativa, interpretativa e construcionista para análise de fenômenos sociais [Bardin77] [Hardy04], e incorpora um conjunto de técnicas para conduzir uma investigação qualitativa e estruturada de textos. O objetivo da análise do discurso é descobrir a maneira como a realidade social existente é produzida; ela envolve o estudo sistemático de textos para encontrar evidências de seu significado e de como este significado se traduz numa realidade social. Isto inclui identificar características da linguagem utilizada pelos atores, as categorias utilizadas na estruturação ou organização de seu universo e ainda as metáforas ou analogias utilizadas na descrição dessas categorias [Lowe04].

Análise de conteúdo diverge da análise do discurso no sentido de ser fortemente baseada em métodos quantitativos. Historicamente, a análise de conteúdo evoluiu da avaliação de textos tendo por base a simples análise de freqüência de determinados termos para métodos e técnicas mais sofisticados, baseados em conceitos e em relações semânticas entre eles. O texto é analisado não apenas pelos conceitos explicitados, mas também por aqueles implícitos de forma proposital ou não pelo autor do mesmo. Enquanto a identificação de termos explícitos é relativamente simples de realizar, a identificação de termos implícitos pode estar sujeita à interpretação do avaliador. Uma forma de diminuir a subjetividade de diferentes pontos de vista envolve a utilização de dicionários especializados.

A análise de conteúdo, na prática, envolve o desenvolvimento de categorias analíticas que permitem verificar em que medida tais categorias estão ou não presentes no texto ou conjunto de textos analisado. Análise de conteúdo pode ser caracterizada como sendo objetiva, sistemática e quantitativa. Ela é objetiva na definição precisa das categorias de forma que diferentes investigadores possam aplicá-las e obter resultados equivalentes; é sistemática na utilização de regras claras e bem definidas para a inclusão ou exclusão de categorias, e quantitativa no sentido de gerar resultados tratáveis por técnicas estatísticas [Hardy04]. Análise do discurso e análise de conteúdo de alguma forma avaliam também o contexto onde se insere o texto ou documento, mesmo que isso seja apenas presente na escolha dos termos utilizados para registro das categorias escolhidas. Métodos de análise de conteúdo são baseados em dicionários e classificados como modelos de variáveis latentes.

Em tais modelos, variáveis não observáveis diretamente (que chamaremos de *x*) originam efeitos observáveis (que chamaremos de *y*). O processo de inferência num modelo de variáveis latentes implica na avaliação da presença de *x* quando um particular *y* é observado. Podemos associar a variável *x* a um conceito que se deseja avaliar se está ou não presente no documento e *y* às características observáveis como termos e suas freqüências. Análise de conteúdo especifica o mapeamento de *x* para *y* através da construção de um dicionário de termos e expressões; o dicionário define como um determinado conceito (*x*) é expresso através de termos e expressões (*y*).

Análise do discurso e análise de conteúdo convergem na utilização de um conjunto de conceitos, chamados de categorias pela análise do discurso. Esse conjunto de conceitos é estruturado num dicionário, e a cada conceito é associado um conjunto de termos ou expressões a ele relacionados, em alguma medida. O processo de análise de conteúdo é baseado nesse conjunto de categorias e está estruturado em cinco etapas: preparação das informações, divisão do texto em unidades de análise, categorização das unidades, descrição e interpretação dos resultados [Moraes99]. A análise de conteúdo propriamente dita tem início apenas depois que os objetivos da análise estão claros, e as categorias (ou conceitos) estão definidas.

Preparação dos dados envolve definição da amostra a ser utilizada para o processo de análise, considerando os objetivos propostos; também significa aplicar as transformações necessárias que permitirão o trabalho com o texto. Unitarização implica em definir a unidade de análise, dividir o texto em unidades e identificar cada unidade - uma unidade pode ser uma palavra, uma expressão idiomática, um termo, uma frase ou um documento. Uma unidade deve ter um significado, deve ser completa em si mesma. A categorização pode utilizar critérios sintáticos ou semânticos, e, neste último caso, as categorias são denominadas de temáticas. A etapa de descrição, numa análise quantitativa, compreende a organização das categorias, freqüências e percentuais computados em tabelas e visualizados em gráficos ou outras formas de representação. A última etapa é a de interpretação de resultados, onde se buscam inferências que permitam generalizações a partir dos resultados obtidos na amostra ou documento analisado. A interpretação busca a compreensão sobre os conteúdos explícitos no texto, e também sobre aqueles não explicitados, latentes, não verbalizados pelos autores.

O processo de criação do conjunto de categorias a ser utilizado é central ao processo de análise de conteúdo, e pode tanto utilizar categorias extraídas do próprio texto ou derivar essas categorias de um referencial teórico relacionado à área de estudo. As categorias devem ser válidas em relação aos objetivos da análise, em relação à natureza do material avaliado e às questões que se busca responder por meio do trabalho de análise [Moraes99]. O conjunto de categorias deve ser exaustivo, no sentido de possibilitar que toda unidade de análise de conteúdo significativo seja classificada numa das categorias presentes no dicionário. Isto envolve escolha criteriosa das categorias e dos conjuntos de termos semanticamente relacionados a cada uma delas.

No contexto desta tese, a análise de conteúdo foi utilizada para a identificação de omissões em requisitos não funcionais, conforme apresentado no capítulo 3.

2.5.7. Agrupamento ou clusterização de documentos

Atividades de recuperação da informação e de mineração de dados estão muitas vezes associadas ao trabalho com um grande volume de documentos, ou corpora de informações textuais. Um dos desafios envolve agrupar documentos com características semelhantes, ou relacionados a uma mesma área. Nesses contextos, clusterização é definida como o processo de agrupar documentos similares tendo por base as palavras ou conceitos presentes nos documentos [Wives04]. A literatura pesquisada aponta muitas técnicas para o agrupamento de documentos [Kowalski97] [Manning99] [Meadow00] [Witten00].

Na área de recuperação de informação o agrupamento de documentos possibilita o armazenamento de documentos similares em áreas próximas da base de dados, agilizando o processo de recuperar todo um grupo quando uma consulta solicita documentos relacionados a um determinado tema. Para a descoberta do conhecimento na mineração de textos os grupos agilizam a identificação de associações entre palavras, facilitando o processo de criação de dicionários e de tesauros [Wives04].

Para o processo de clusterização não existe informação conhecida, a priori, sobre o número de grupos que resultará do processo de agrupamento, e os

métodos classificam documentos em grupos de acordo com um critério pré-determinado. Estes critérios são baseados em medidas de semelhança ou de diferença entre documentos, medidas estas obtidas a partir de representações estruturadas dos documentos. Uma representação freqüente é a matriz termo-documento, descrita na seção 2.5.2, e para o processo de agrupamento, utiliza-se uma seleção dos termos, que são denominados de atributos.

Na área de mineração de textos o uso de algoritmos de particionamento iterativo tem sido bastante freqüente [Wives04]. Dentre estes, um dos mais utilizados é o *k-means*, ou k-médias. Este algoritmo é um método iterativo para particionamento de conjuntos de dados onde o número *k* de grupos é indicado pelo usuário. Um conjunto inicial de *k* objetos é obtido pelo algoritmo de forma aleatória ou de acordo com alguma heurística; esses *k* objetos correspondem aos elementos centrais (ou centróides[3]) dos *k* agrupamentos. Em seguida é analisada a distância ou similaridade de cada documento aos centróides; utiliza-se como medida de similaridade a distância euclidiana. Cada documento é alocado ao agrupamento de centróide com a menor distância (ou maior similaridade) e o centróide desse agrupamento é então recalculado. O processo é repetido até que os centróides não mudem mais de posição.

Como o passo inicial deste algoritmo é aleatório, diferentes agrupamentos podem resultar de diferentes execuções do *k-means*, devido à escolha aleatória dos *k* centróides iniciais. É possível também que os agrupamentos resultantes da aplicação deste algoritmo não resultem em agrupamentos razoáveis [Witten00]. O problema também pode decorrer da escolha de um valor *k* não adequado ao conjunto de documentos que se deseja agrupar. O primeiro problema a resolver então está relacionado à definição do número *k* de agrupamentos que se deseja obter.

Uma boa estimativa para *k* pode ser obtida através de um outro algoritmo de particionamento iterativo, denominado de EM (Expectation-Maximization) [Manning99]. Este algoritmo, da mesma forma que o *k-means*, é iterativo e pode ser executado sem que seja informado o número de agrupamentos. Neste caso, o algoritmo divide os documentos em dois grupos e calcula um conjunto de valores que inclui média e desvio padrão para cada atributo, além do coeficiente

[3] centróide é a média do conjunto de características representativas de cada agrupamento

LogLikelihood[4], que é um coeficiente que mede a similaridade global. O número de grupos é incrementado e os demais procedimentos são repetidos até se obter um coeficiente ótimo para o coeficiente LogLikelihood. Nesse momento não ocorrem mais iterações, e o número ideal de grupos é aquele que otimizou a verosimilhança dos grupos encontrados.

No contexto desta tese técnicas de clusterização foram experimentadas, tendo como entrada documentos de requisitos. Utilizamos para os experimentos a implementação disponibilizada pela ferramenta WEKA [Witten00].

2.5.8. Recuperação de informações: medidas *recall* e *precision*

Na recuperação de informações as medidas recall e precision são utilizadas para avaliar a qualidade da estratégia utilizada. Conforme [Manning99], as medidas de *recall* e *precision* são definidas como:

$$Recall = \frac{tp}{tp + fn} \quad (5) \qquad Precision = \frac{tp}{tp + fp} \quad (6)$$

A utilização do diagrama apresentado na Figura 11 facilita a compreensão do significado dessas medidas. Nas fórmulas (5) e (6), *fp* indica a quantidade de falsos positivos, *fn* indica a quantidade de falsos negativos, *tp* indica a quantidade de positivos corretos (*true positives*) e *tn* indica a quantidade de negativos corretos (*true negatives*). Portanto, *recall* é a proporção dos objetos corretos retornados em relação ao total de objetos buscados (alvo), e *precision* é a proporção de itens corretos no conjunto de objetos recuperados. A Tabela 5 complementa essa definição.

Tabela 5 - Medidas utilizadas para cálculo de recall e precision

Objetos	*alvo (buscados)*	*não alvo*
selecionados	tp	fp
não selecionados	fn	tn

[4] o coeficiente Likelihood é também denominado de coeficiente de verosimilhança

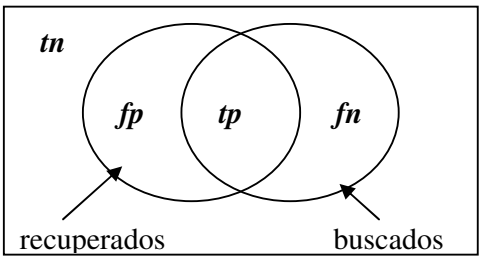

Figura 11 - Diagrama ilustrativo das medidas *precision* e *recall* [Manning99]

2.6. Processo de Requisitos, PLN e Agentes

Este capítulo condensou os principais conceitos do Processo de Requisitos (seções 2.1, 2.2 e 2.3), visando apresentar ao leitor os fundamentos necessários para uma boa compreensão do trabalho desenvolvido nesta tese. Também estão registrados os métodos e técnicas do Processamento da Linguagem Natural (seção 2.5) utilizados para atingir os objetivos propostos, dado que trabalhamos com requisitos expressos em linguagem natural. Como a ferramenta de suporte à abordagem proposta utiliza agentes de software, apresentamos também os conceitos básicos da área (seção 2.4), possibilitando ao leitor acesso à base conceitual necessária ao entendimento deste trabalho.