4
Proposta de uma Arquitetura de Agentes

A abordagem de desenvolvimento centralizado de software tem sido amplamente utilizada por grandes e médias equipes de desenvolvimento. O desenvolvimento tradicional de software tem ocorrido em ambientes onde os artefatos utilizados ou gerados no desenvolvimento de software residem em um servidor local disponível aos interessados através de uma rede local. Entretanto, na economia globalizada atual, o desenvolvimento colaborativo de software envolvendo várias equipes distribuídas em locais diferentes está se tornando uma norma, ao invés de uma exceção [Wongthongthan06].

Freqüentemente, encontramos equipes de desenvolvimento distribuídas por diferentes cidades, regiões, e até mesmo em diversos continentes. Os modelos e metodologias atuais de desenvolvimento não contemplam as questões de projeto e desenvolvimento colaborativo de software [Wongthongthan06].

Um ambiente colaborativo de projeto é um ambiente automatizado que habilita pessoas (incluindo projetistas, engenheiros, administradores, clientes e usuários) a colaborar e interagir no desenvolvimento de novos projetos, considerando sua localização geográfica e os meios de interação [Wu06]. Wu declara também que projetos colaborativos de sistemas envolvem equipes multidisciplinares de projeto e várias ferramentas de engenharia de software; alguns dos principais problemas encontrados neste contexto incluem a dificuldade de troca de informações entre ferramentas devido aos diferentes padrões adotados, e a quase inexistência de mecanismos para auxiliar os usuários em suas tarefas colaborativas.

A arquitetura descrita neste capítulo tem por objetivo dar suporte ao processo de verificação e validação de requisitos em ambientes distribuídos de desenvolvimento. Este cenário requer que aplicações possam ser configuradas dinamicamente, e que arquiteturas flexíveis e habilitadas a evoluir sejam utilizadas para dar suporte à dinâmica destes sistemas, à constante evolução dos requisitos e às mudanças na equipe envolvida. A arquitetura proposta envolve o

uso de agentes de software.

O uso de agentes em ambientes distribuídos de desenvolvimento é relatado em [Li03], que propõe uma ferramenta para gerenciamento de dados utilizando uma abordagem baseada em agentes. Agentes de software também são utilizados para dar suporte ao gerenciamento de uma ontologia de engenharia de software destinada a apoiar uma equipe de desenvolvimento num ambiente geograficamente distribuído [Wongthongthan06]. Vários outros relatos apontam soluções para ambientes distribuídos utilizando agentes de software [Gaeta02] [Chang01], este último especificamente numa ferramenta de suporte ao processo de requisitos num ambiente distribuído.

Nos experimentos realizados identificamos que diferentes técnicas para tratamento de linguagem natural podem ser compostas e associadas a outras de forma a prover um novo tratamento para os dados. Desta forma, um conjunto inicial de tratamento da informação destinado a apoiar atividades de verificação e validação pode ser rapidamente incrementado, reutilizando algumas das técnicas já implementadas. Web Services provêem um paradigma apropriado para o desenvolvimento de sistemas com estas características porque são independentes de aplicações, plataformas e provedores.

4.1. Adequação de Web services e agentes

Tipicamente, em ambientes distribuídos de projeto, os recursos de informação são dinâmicos e muitas vezes heterogêneos. Além disso, estes ambientes de computação normalmente são dinâmicos, onde recursos de informação podem ser conectados e desconectados a qualquer momento. Agentes de software capturam e implementam serviços como funcionalidades e papéis. Da perspectiva de agentes, serviços Web são simplesmente entidades programáticas que podem ser chamadas para executar uma determinada atividade, tipicamente uma função unitária. Para que serviços Web possam trabalhar juntos com agentes, é necessário agregar propriedades comportamentais e de agenciamento, como colaboração e interação. É necessário dispor de uma arquitetura para que os agentes possam interagir, colaborar, compor e gerenciar serviços.

Diferentes de *sites* Web e aplicações *desktop*, serviços Web não são

projetados para interação direta com agentes humanos; eles operam no nível do código, são chamados e trocam informações com outros softwares através dos padrões estabelecidos para a Web. Um serviço Web pode ser usado quando o construtor de uma aplicação deseja expor alguma operação reativa, expressa como uma função, com ou sem parâmetros, que pode retornar ou não uma resposta. Na essência, um serviço Web funciona como uma invocação remota de um método, usando mensagens encapsuladas em XML sobre HTTP [Li03a].

Web Services são apropriados para incorporar regras de negócios em ambientes distribuídos [Li03a], enquanto agentes de software são apropriados em tais serviços para implementar atividades de interação e coordenação. Desta forma, propomos utilizar uma arquitetura orientada a serviços que utiliza agentes de software para compor a solução.

Pesquisadores da área de PLN tem disponibilizado ferramentas *open source* ou mesmo *free* para a comunidade em geral. Uma das motivações para a escolha da plataforma a ser utilizada na implementação da nossa proposta é relacionada, portanto, à habilidade de encapsular ferramentas já disponíveis na forma de serviços Web, possibilitando sua utilização quase que imediata.

4.2. Plataforma utilizada: MIDAS

Para demonstrar o funcionamento e viabilidade da solução, uma plataforma para a execução do modelo deve ser utilizada. Pesquisando as atuais plataformas para o desenvolvimento de SMAs para a Web, podem ser encontradas várias alternativas, tais como JADE [Bellifemine01], AgentScape [Overainder04], MIDAS [Haendchen05] [Haendchen07], dentre outras. Para os propósitos deste trabalho, alguns requisitos assumem um papel fundamental na escolha da plataforma. Primeiro, a plataforma deverá prover as funcionalidades básicas para a integração entre Web Services e agentes. Além destas funcionalidades, é necessário que existam mecanismos para dar suporte ao modelo de comunicação entre os agentes.

O *blackboard* tem sido utilizado para este propósito: a arquitetura baseada em *blackboards* é uma das mais utilizadas em sistemas multi-agentes cognitivos, e foi inicialmente criada para reconhecimento de voz no sistema HearsayII

[Ferber99]. Neste modelo de arquitetura, a comunicação entre agentes é possibilitada através do compartilhamento de informações armazenadas no *blackboard*. Isto significa que agentes podem ser inseridos dinamicamente no sistema, sem que sua identidade seja conhecida a priori pelos agentes já ativos no ambiente. Da mesma forma, um agente pode ser removido do ambiente sem que isto cause postergação indefinida em agentes que com ele trocavam informações de forma síncrona – outros agentes do ambiente podem assumir o seu papel, ou executar as atividades pelas quais o agente removido era responsável. Podemos dizer então que o uso de *blackboard* facilita atividades de interação e comunicação entre agentes, permitindo uma boa representação do ambiente de desenvolvimento de software, no qual humanos podem ser incluídos ou retirados de um determinado projeto de desenvolvimento de software.

Esta propriedade assume importante papel para os propósitos deste trabalho. Considerando as características abertas das aplicações a serem geradas, as facilidades obtidas pela utilização e possibilidade de reuso de *Web Services* e a necessidade de utilizar um modelo de comunicação onde os agentes possam se comunicar de forma anônima, a plataforma MIDAS foi escolhida para implementar a arquitetura proposta. Ela atende aos principais requisitos discutidos: propicia reuso em larga escala de serviços distribuídos e utiliza o *blackboard* como um meio de comunicação entre os agentes.

A arquitetura de MIDAS aplica os padrões definidos pela arquitetura de referência WSA, sendo baseada na coexistência de vários *containers*, cada um executando uma JVM (Java Virtual Machine). Cada máquina virtual provê um ambiente completo de execução, onde aplicações baseadas em agentes podem executar de forma concorrente no mesmo *host*. A Figura 29 mostra a arquitetura genérica do sistema, detalhando os elementos da arquitetura do servidor de front-end (FES) e dos containers de agentes (AC).

Na Figura 29 o *front-end* da plataforma Midas e o *container* de agentes e componentes já são apresentados na forma como estão instanciados para o nosso trabalho. Da plataforma Midas apresentaremos brevemente apenas características importantes no contexto deste trabalho, e detalharemos os aspectos dos agentes e componentes da aplicação.

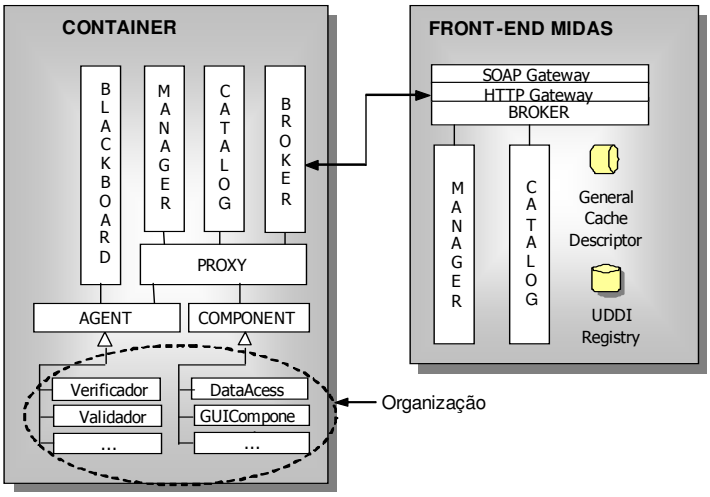

Figura 29 - Arquitetura da plataforma Midas

O *front-end* Midas possui agentes internos que provêem serviços de infra-estrutura à própria plataforma; na Figura 29 eles são representador por BROKER, MANAGER e CATALOG. Estes três agentes provêem serviços de comunicação, gerenciamento do ciclo de vida dos agentes e manutenção do catálogo de serviços para toda a plataforma. A comunicação ocorre através de duas portas de comunicação, via http ou via SOAP, esta última necessária para *web services.*

No container estão presentes agentes de infra-estrutura do container: BROKER, MANAGER, CATALOG e PROXY. Os três primeiros são instâncias dos correspondentes no *front-end* MIDAS, agora responsáveis pela comunicação, gerenciamento e catálogo de serviços do próprio container. O agente PROXY possibilita que os agentes de aplicação possam invocar serviços de forma transparente, sem necessitar conhecer a localização do serviço. O BLACKBOARD mantém informações do ambiente de execução, possibilitando que os agentes da aplicação monitorem as mudanças neste ambiente e tomem as ações apropriadas a cada caso.

Os agentes da aplicação são instanciados estendendo a classe abstrata AGENT. Esta classe abstrata contém os métodos comuns a todos os agentes da aplicação, como métodos para controle do ciclo de vida do agente e as interfaces para monitoração do *blackboard.* A classe abstrata COMPONENT representa componentes puramente reativos, e deve ser estendida para implementar, por exemplo, objetos de acesso a bancos de dados, interfaces de comunicação com usuários e objetos que encapsulam regras de negócio específicas do domínio da

aplicação. Agentes e componentes da aplicação compõem uma organização.

Neste ambiente, componentes são entidades puramente reativas, e implementam aspectos característicos do domínio da aplicação como, por exemplo, interface com o usuário e mecanismos para acesso aos dados.

4.3. Aplicação: características, agentes e modelos de papéis

A ferramenta de suporte às técnicas propostas foi visualizada para atender às seguintes características:

- repositório de artefatos de requisitos e de informações sobre projetos
- servidor mantendo repositórios e provendo serviços de comunicação
- comunicação entre sites via protocolo http
- acesso remoto via *browser* e protocolo de comunicação *http*
- autenticação de usuários
- perfis de usuários, com diferentes direitos de acesso
- controle de modificações em requisitos
- serviços de PLN e outros disponibilizados via *web services*
- agentes como assistentes pessoais, como provedores de serviços, monitores de modificações.

4.3.1. Perfil de usuários

Definimos três tipos básicos de perfis para os participantes do processo de requisitos na plataforma: gerente de projeto, engenheiro de requisitos e representante de clientes e usuários. Ao gerente de projeto são delegados acesso a todas as funcionalidades incluindo manutenção de artefatos de requisitos, agendamento das atividades de V&V e definição dos participantes para os processos de V&V. O engenheiro de requisitos não tem acesso ao agendamento das atividades de V&V, mas pode efetuar manutenções nos artefatos do repositório, e o representante de clientes e usuários participa de atividades de V&V, mas tem acesso apenas para leitura aos artefatos do repositório.

4.3.2.
Uso do blackboard

O *blackboard* é uma metáfora ao meio ambiente onde os agentes desempenham seus papéis. Todos os agentes monitoram o ambiente, identificam a ocorrência de eventos neste ambiente e decidem se devem executar alguma ação em função do ocorrido. Os principais eventos de interesse dos agentes são chegada de mensagens, solicitações de serviços e modificações no estado do documento de requisitos.

A comunicação entre agentes se processa através deste mecanismo: o agente envia uma mensagem, no formato apropriado, que é colocada no *blackboard*. Os agentes monitoram este ambiente e detectam a modificação ocorrida. Cada um dos agentes então verifica se ele é o destinatário da mensagem e, em caso positivo, trata a mensagem recebida de acordo com o seu papel.

O *blackboard* é utilizado também para manter registro do estado do documento de requisitos. No estado *aberto* requisitos podem ser inseridos, excluídos ou modificados; nos estados de *pré-verificação* ou de *verificação* não são mais aceitas modificações. Após a conclusão da verificação o estado retorna para *aberto*. De maneira análoga, no estado de *pré-validação* e de *validação* não são aceitas modificações em requisitos.

Solicitações de serviços são feitas também através do *blackboard*: o agente que requisita o serviço coloca a requisição no *blackboard*, cada agente que monitora o ambiente detecta a chegada de uma solicitação e verifica se é ele que deve atender à solicitação. A solicitação de serviços tem um protocolo padrão, que registra o nome do serviço e parâmetros de entrada e saída.

4.3.3.
Repositório de artefatos e de informações sobre projetos

A organização deve manter um repositório de artefatos relacionados ao projeto, entre eles o documento de requisitos. Requisitos são armazenados individualmente e, eventualmente reunidos num único documento. Cada requisito possui um conjunto de atributos que o identifica e provê outras informações.

Para ter acesso aos requisitos e demais artefatos (utilizados ou gerados na fase de requisitos), os participantes do projeto são cadastrados. Uma importante

informação, além do perfil do participante, é o endereço eletrônico. Através deste endereço o participante recebe informações relativas a modificações em requisitos, ao agendamento de atividades de V&V, a resultados parciais de atividades de verificação em requisitos.

4.3.4. Agentes: modelo de papéis

A aplicação desenvolvida é um sistema que combina agentes e componentes. Agentes, nesta plataforma, possuem papéis bem definidos, descritos a seguir.

4.3.4.1. Agente Manager

O agente Manager é um assistente pessoal do gerente do projeto, e responsável por monitorar atividades do gerente do projeto, detectar o agendamento de atividades de V&V no documento de requisitos, notificar os interessados e o agente Verificador/Validador sobre o evento. A Figura 30 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Manager**	**Colaboradores**
Responsabilidades ▪ Identificar agendamento da verifição ou da validação: – monitorar atividades do gerente humano e identificar o agendamento de uma data para a verificação ou validação – modificar estado do documento de requisitos de aberto para pre-verificação ou pré-validação ▪ Enviar notificações: – notificar Comunicador da agenda definida para a verificação/validação de requisitos – notificador Verificador/Validador	▪ Comunicador ▪ Verificador ▪ Validador

Figura 30 - Responsabilidades e colaborações do agente Manager

4.3.4.2. Agente Analisador Léxico (Lexical)

Este agente centraliza os serviços de tratamento léxico a ser efetuado sobre o documento de requisitos. Fornece serviços para vários agentes da plataforma,

atendendo requisições para conversão de documentos, extração do radical de palavras, obter contextos, identificar colocações, categorizar requisitos, identificar termos não dicionarizados. A Figura 31 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Analisador Léxico**	**Colaboradores**
Responsabilidades ▪ Converter documentos: – conversão de documentos de requisitos armazenados em formato doc, pdf ou html para formato texto puro ▪ Extrair requisitos: – analisar documento de requisitos, extraindo e armazenando em arquivos independentes cada um dos requisitos ▪ Fornecer stem de palavras: – a partir de uma palavra, retornar o radical ou *stem* da mesma ▪ Obter contextos (Concordanceador): – analisar o documento de requisitos e retornar as sub-sentenças onde constem o termo em análise ▪ Categorizar requisitos: – para cada item de mais alto nível da taxonomia, gerar e consolidar os padrões de segundo nível – identificar requisitos associados a cada um dos padrões ▪ Gerar matriz termo-documento: – gerar matrizes termo-documento, para uso em serviços que utilizem a abordagem *bag-of-words*	▪ Verificador ▪ Gerador de Visões ▪ Construtor do Léxico ▪ Estatístico

Figura 31 - Responsabilidades e colaborações do agente Analisador Léxico

4.3.4.3. Agente Construtor do Léxico (LexicalConstructor)

O agente Construtor do Léxico é responsável pela manutenção do léxico da aplicação, visando à atualização da base de conhecimentos para o domínio da organização. A Figura 32 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Construtor do Léxico**	**Colaboradores**
Responsabilidades ▪ Criar ou atualizar o léxico da aplicação: – Identificar termos não dicionarizados – identificar atores e recursos, via análise do documento de requisitos e extração de sintagmas nominais relevantes – em interação com o agente Léxico, buscar contextos para cada um dos terrmos não dicionarizados	▪ Analisador Léxico

Figura 32 - Responsabilidades e colaborações do agente Construtor do Léxico

4.3.4.4. Agente Gerador de Visões (Mapper)

O agente Gerador de Visões é responsável pelo enriquecimento de uma taxonomia básica fornecida pelos usuários, categorizar os requisitos tendo por referência esta mesma taxonomia e gerar visões textuais e gráficas dos agrupamentos obtidos. A Figura 33 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Gerador de Visões**	**Colaboradores**
Responsabilidades ▪ Gerar uma taxonomia: – utilizar termos fornecidos pelos usuários, que correspondem ao nível mais alto da taxonomia – em interação com o agente Léxico, identificar colocações e obter padrões léxicos que irão compor o segundo nível da taxonomia ▪ Categorizar requisitos: – em interação com o agente Léxico, identificar requisitos associados a cada um dos ramos da taxonomia ▪ Gerar visões: – gerar visão textual dos agrupamentos identificados – gerar mapa XTM para possibilitar visões gráficas ▪ Enviar notificações: – notificar agente verificador/validador e demais interessados, via agente Comunicador	▪ Analisador Léxico ▪ Comunicador ▪ Verificador ▪ Validador

Figura 33 - Responsabilidades e colaborações do agente Gerador de Visões

4.3.4.5. Agente Estatístico (Statistics)

O agente Estatístico é responsável pelo cálculo do mapa de freqüência e métricas como escore T e informação mútua, trabalhando sobre os contextos extraídos do documento de requisitos e relacionados a um determinado tema. A Figura 34 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Estatístico**	**Colaboradores**
Responsabilidades ▪ Calcular freqüências: – recebe um conjunto de sub-sentenças (concordances) relacionadas a um determinado tema e calcula frequencias do nodo (o termo principal) e dos dems termos presentes ▪ Calcular métricas para colocações: – calcula os valores de escore T e informação mutua para os termos (colocações), a partir dos valores de frequencia já obtidos. Apenas as colocações que possirem score T >=2 e informação mútua > 3 serão selecionadas	▪ Analisador Léxico ▪ Gerador de Visões

Figura 34 - Responsabilidades e colaborações do agente Estatístico

4.3.4.6. Agente Verificador

O agente verificador é responsável por atividades de verificação dos requisitos, como identificar requisitos em duplicidade e omissões em requisitos não funcionais. Para identificação de requisitos em duplicidade inicialmente é gerada a matriz termo-documento e são calculados os índices de similaridade de Dice, Jaccard e do coseno. Para omissões em requisitos não funcionais é utilizada uma taxonomia de RNF´s instanciada para a organização. A essa taxonomia são agregados termos em linguagem natural que são normalmente utilizados para fazer referência a esses requisitos não-funcionais. A Figura 35 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Verificador**	**Colaboradores**
Responsabilidades ▪ Identificar requisitos em duplicidade: – em interação com o Agente Léxico, gerar a matriz termo-documento para o conjunto de requisitos – executar análise estatística sobre pares de documentos, retornando medidas de similaridade (índices de similaridade de Dice, Jaccard e cos-seno) entre eles – gerar matriz de similaridades, e extrair pares de requisitos cujo índice seja maior que o limiar definido ▪ Identificar omissões em RNF´s: – utilizando uma taxonomia de requisitos não funcionais instanciada para a organização, identificar omissões e gerar relatório com análise dos requisitos emr elação a omissões e inconsistências detectadas ▪ Enviar mensagens: – notificar interessados, via agente Comunicador, dos resultados obtidos nas atividades de pré-verificação	▪ Analisador Léxico ▪ Comunicador

Figura 35 - Responsabilidades e colaborações do agente Verificador

4.3.4.7. Agente Validador

O agente Validador é responsável pela coleta e distribuição dos artefatos para a validação, envio de notificação aos envolvidos e consolidação parcial do relatório. A Figura 36 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Validador**	**Colaboradores**
Responsabilidades ▪ Organização da validação: – utilizar termos fornecidos pelos usuários, que correspondem ao nível mais alto da taxonomia – em interação com o agente Léxico, identificar colocações e obter padrões léxicos que irão compor o segundo nível da taxonomia ▪ Gerar visões: – gerar visão textual dos agrupamentos identificados – gerar mapa XTM para possibilitar visões gráficas ▪ Enviar notificações: – notificar interessados, via agente Comunicador	▪ Analisador Léxico ▪ Gerador de Visões ▪ Comunicador

Figura 36 - Responsabilidades e colaborações do agente Validador

4.3.4.8. Agente Comunicador

O agente Comunicador do Léxico é responsável pelo envio de mensagens aos participantes do processo de desenvolvimento, notificando-os em relação a eventos de seu interesse, como definição de agendas para os processos de verificação/validação ou modificações ocorridas nos requisitos do sistema. A Figura 37 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Comunicador**	**Colaboradores**
Responsabilidades ▪ Envio de mensagens e notificações: – responsável pelas comunicações entre agentes e participantes do processo de requisitos	▪ Analisador Léxico ▪ Verificador ▪ Validador ▪ Construtor do léxico ▪ Observador

Figura 37 - Responsabilidades e colaborações do agente Comunicador

4.3.4.9. Agente Observador

O agente Observador é responsável pelo monitoramento de operações de escrita nos artefatos de requisitos e, em caso de alteração, é o responsável por notificar os interessados. A Figura 38 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Observador**	**Colaboradores**
Responsabilidades ▪ Identificar mudanças nos artefatos controlados: – Manter controle sobre documento de requisitos, identificando modificações ▪ Notificação de interessados: – Em colaboração com o agente Comunicador, notificar interessados sobre modificação ocorrida	▪ Comunicador

Figura 38 - Responsabilidades e colaborações do agente Observador

4.3.4.10. Agente Rastreador

O agente Rastreador é responsável pela construção de matrizes de rastreabilidade, e pela verificação destas matrizes com aquelas fornecidas pelo engenheiro de requisitos, apontando as divergências e apresentando a opção de tratar ou não cada uma das divergências apontadas. A Figura 39 ilustra as principais responsabilidades e colaborações deste agente.

Agente **Rastreador**	**Colaboradores** ▪ Comunicador
Responsabilidades ▪ Construção de matriz de rastreabilidade para dependencias entre requisitos: – Em colaboração com o agente Léxico, identificar dependências entre requisitos funcionais e não funcionais – Gerar matriz de rastreabilidade RF x RNF ▪ Notificação de interessados: – em colaboração com o agente Comunicador, notificar interessados após construção da matriz	

Figura 39 - Responsabilidades e colaborações do agente Rastreador

4.4. Uma visão da implementação

A visão de alto nível da plataforma de suporte ao sistema é apresentada na Figura 40: um servidor central que é ao mesmo tempo responsável por prover a comunicação entre os diferentes sites da organização e também mantém o repositório de informações sobre projetos. A comunicação entre as máquinas clientes, servidor local e servidor central é realizada via protocolo *http*,

possibilitando que qualquer máquina conectada à Internet possa acessar as informações armazenadas no servidor central com a utilização de um *browser*.

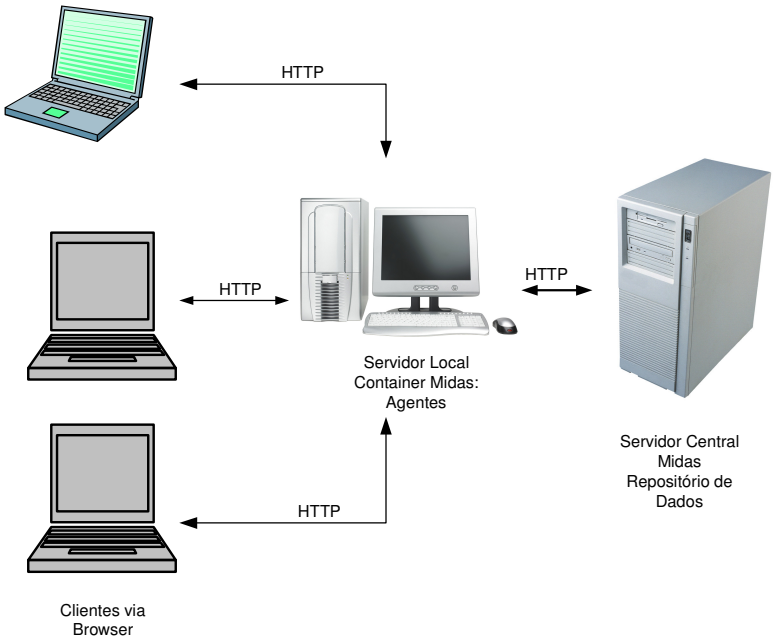

Figura 40 - Visão de alto nível da plataforma

A plataforma Midas provê uma interface, denominada Agent Server Manager, a qual possibilita o gerenciamento do ciclo de vida dos agentes. A Figura 41 apresenta a visão do sistema multi-agente criado, com os agentes referidos na seção anterior, através desta interface. É possível também a visualização dos serviços disponibilizados pelos agentes, como pode ser observado no detalhamento do agente Lexical.

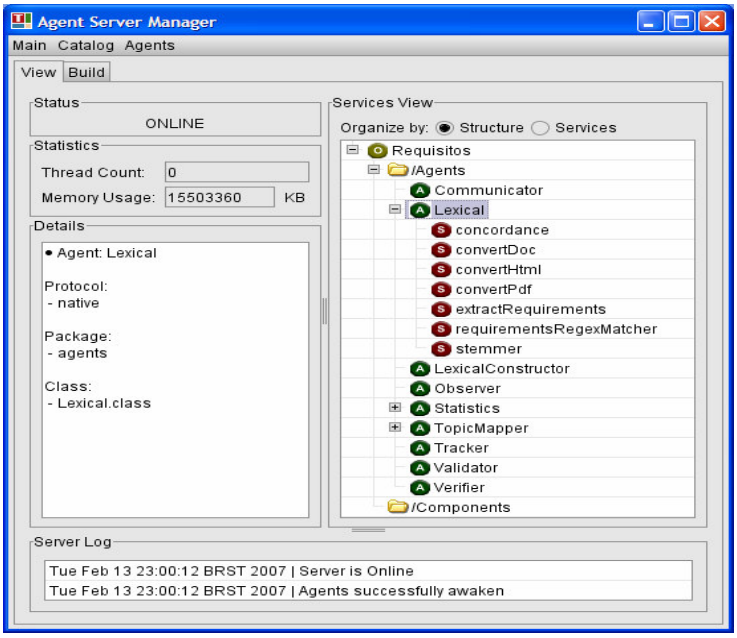

Figura 41 - Visão do sistema multi-agente

A arquitetura da aplicação multi-agente e é apresentada a seguir, com descrição e características principais.

4.4.1. Estrutura da aplicação multi-agente

A aplicação está estruturada em 3 subsistemas, a saber: *core*, *web* e *midas*. Para a implementação foi utilizada a linguagem Java, mas os diferentes softwares incorporados como serviços dos agentes estão desenvolvidos nas linguagens Perl, C e AWK. Descrevemos a seguir cada um desses subsistemas.

O subsistema *core* agrega componentes responsáveis pelos agentes e seus serviços e pelo acesso e manutenção do repositório de informações sobre projetos. É composto pelos pacotes *requisitos.agents* (agentes que compõem a aplicação), *requisitos.business.entities* (entidades da aplicação), *requisitos.dao*, *requisitos.dao.hibernate* e *requisitos.dao.jdbc* (responsáveis pelas funcionalidades de acesso aos repositórios de dados) e *requisitos.util* (alguns serviços implementados na linguagem Java, como envio de mensagens, cálculo de similaridade entre requisitos e stemização).

O sub-sistema *web* agrega componentes responsáveis pelas funções de interface com os usuários do sistema, incluindo validação de acesso. É composto pelos pacotes *requisitos.web.adapter* (processos da aplicação como mudanças em

requisitos, geração de mapas visuais e processos de verificação e validação), *requisitos.web.controller* (responsável pelas comunicações entre usuário/midas/repositório), *requisitos.web.servlets* e *requisitos.web.user* (responsáveis pelas operações de login/logout na aplicação) e *requisitos.web.util* (montagem de strings *http* para comunicação).

O subsistema *midas* é constituído pela própria plataforma de agentes que fornece a infra-estrutura de comunicação, possibilita o gerenciamento do ciclo de vida dos agentes e a manutenção do catálogo de serviços.

4.5. Interface com o usuário: processos abordados

A aplicação oferece interfaces para três processos: manutenção de requisitos, geração de mapas e processo de verificação e validação.

Manutenção de documentos de requisitos: nos estudos de caso que fizemos utilizamos documentos de requisitos cedidos por uma organização que utiliza DDS; tais documentos eram versões ainda não verificadas e validadas, mas em princípio deveriam ser completas no sentido de abarcar todas as funcionalidades e atender as restrições colocadas para o sistema. Para que efetivamente a plataforma possa ser utilizada em ambientes distribuídos, é necessário que os requisitos possam ser submetidos por engenheiros de requisitos localizados em ambientes geograficamente distantes. A Figura 42 apresenta a interface para a manutenção do documento de requisitos: requisitos podem ser incluídos, modificados ou excluídos, via Web.

Geração de visões (mapas): no processo de V&V os participantes podem desejar obter uma particular visão dos requisitos, buscando avaliar o conjunto de requisitos associados a um determinado tema. Para esse tipo de solicitação a plataforma provê uma interface onde o usuário seleciona o projeto, informa o tema e solicita as associações relevantes. A plataforma identifica as associações e as avalia através das medidas de informação mútua e score T; o usuário pode então solicitar a geração do mapa visual correspondente. Isto pode ser visualizado na Figura 43.

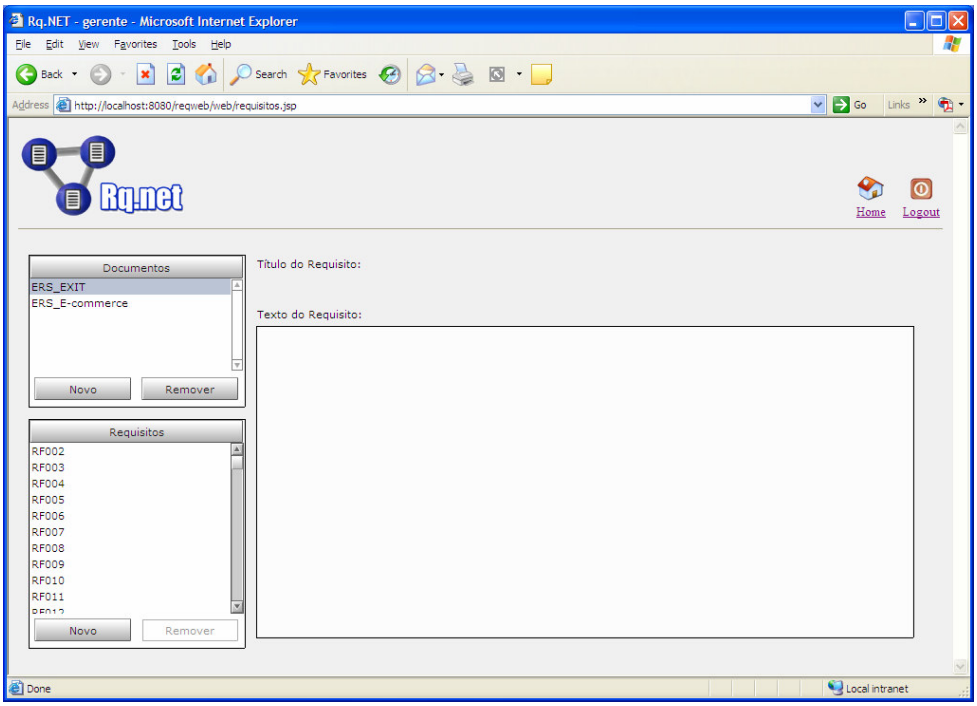

Figura 42 - Interface para manutenção de documentos de requisitos

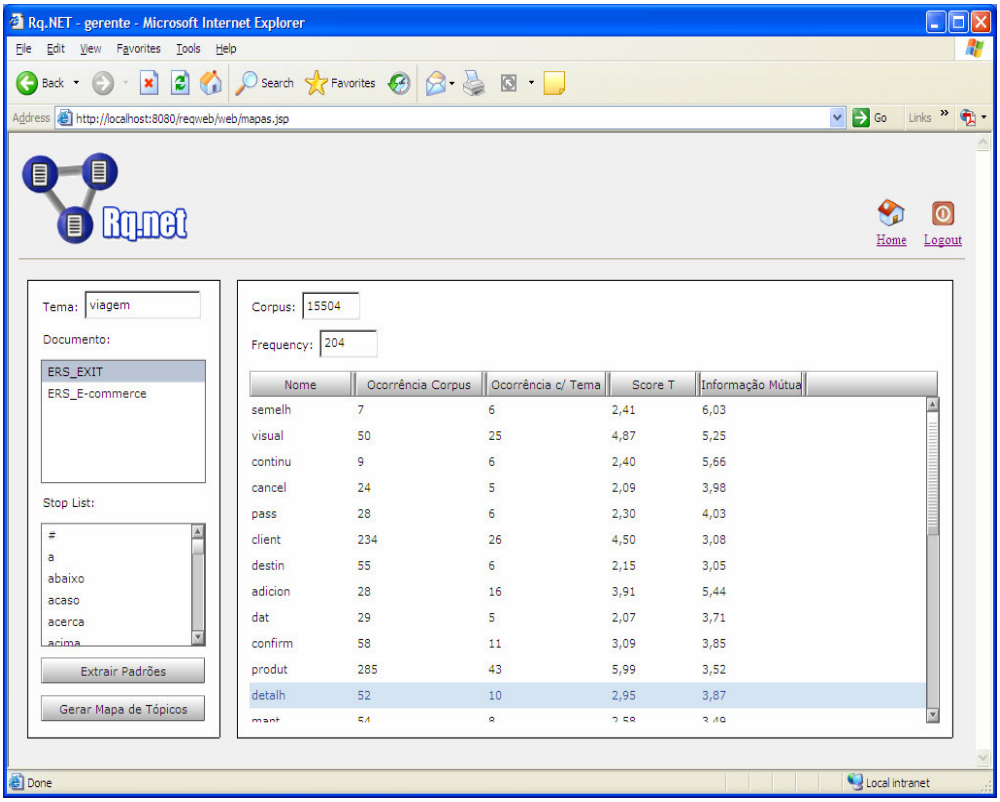

Figura 43 - Interface para identificação de requisitos associados a um tema

Processo de V&V: é responsabilidade do gerente do projeto escolher uma data para o processo de verificação do documento de requisitos. Nesse momento o seu agente pessoal, em colaboração com os demais agentes da plataforma, se encarrega de enviar mensagem aos participantes informando da data definida para a verificação. O estado do documento de requisitos é alterado, e não são mais aceitas inserções ou atualizações no documento. São realizadas algumas verificações, como a busca pela duplicidade em requisitos, e os resultados são enviados aos participantes também por mensagem eletrônica. O diagrama de atividades para este processo é mostrado na Figura 44.

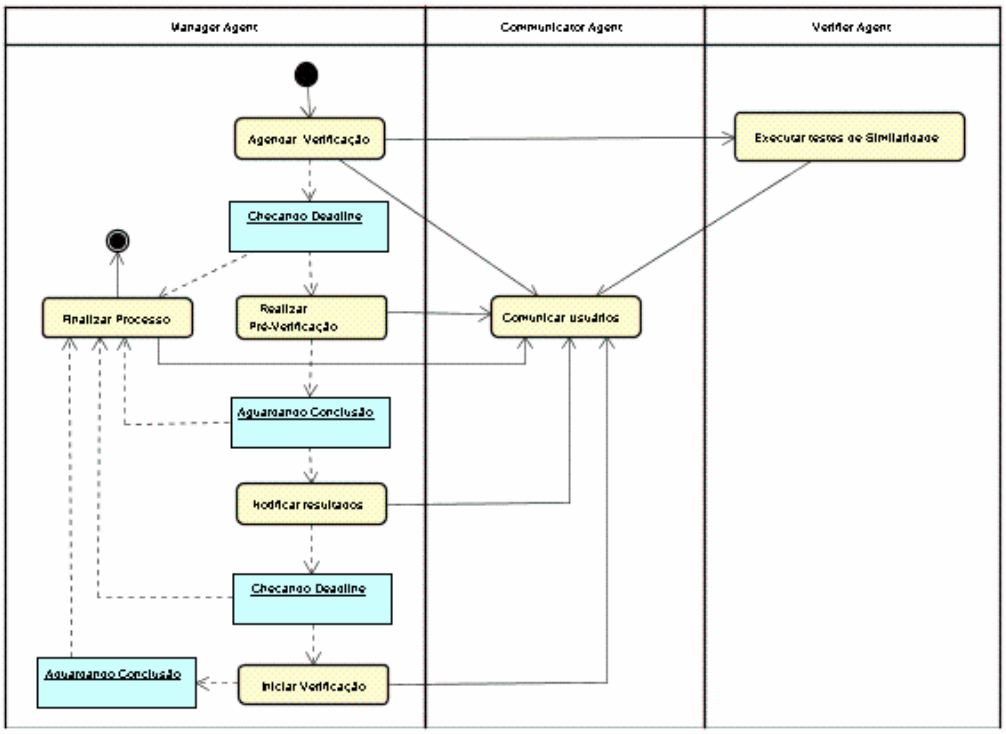

Figura 44 - Diagrama de atividades para o processo de Verificação

O diagrama de atividades para o processo de validação é equivalente ao diagrama das atividades de verificação, à exceção da verificação de similaridades, que não tem correspondente na validação.

4.6. Algumas considerações sobre a aplicação

A implementação da plataforma multi-agente para a aplicação apresenta as principais características e propriedades: uso de agentes pessoais, encapsulamento e reuso de aplicações completas como serviços de agentes (*web services*),

notificação dos interessados em eventos específicos no processo de V&V. Nem todos os serviços dos agentes estão disponíveis ainda através de interface com o usuário, pois a aplicação ainda está em fase de construção. O código fonte dos agentes aqui apresentados está disponível no Anexo D deste volume.

Os serviços e funcionalidades implementados por softwares disponibilizados pelas comunidades de PLN e incorporados e/ou utilizados na plataforma são:

1. software: Yoshikoder converter
    - finalidade: conversão de documentos dos formatos *doc*, *html* e *pdf* para txt
    - serviços ConvertDoc, ConvertHtml e ConvertPdf
    - agente: Analisador Léxico
    - obs: foram necessárias modificações para correção de bugs
2. software: Q-TAG
    - finalidade: POS Tagger: colocação de etiquetas gramaticais aos tokens
    - serviço: tagger
    - agente: Construtor do Léxico
    - obs: fornecido apenas o executável e arquivo de treinamento
3. software: qtoken
    - finalidade: divisão do texto em tokens
    - serviço: tokenizer
    - agente: Construtor do Léxico
    - obs: fornecido apenas o executável
4. software: pretex
    - finalidade: geração de matrizes termo-documento
    - serviço:
    - agente: Verificador
    - obs: implementado em Perl, o *stemmer* foi substituído por uma implementação baseada em [Orengo01]
5. software: yoshikoder
    - finalidade: geração de relatórios para análise do discurso

- serviço:
- agente: Verificador
- obs: modificado para permitir utilização de expressões (e não apenas palavras simples) e colocação dos menus em português. Ainda não incorporado à plataforma.

6. software: TMNAV
   - finalidade: exibição de mapas de tópicos
   - serviço:
   - agente: Verificador e Validador
   - obs: ainda não incorporado à plataforma.
7. software: analisador
   - finalidade: identificação de termos não dicionarizados
   - serviço:
   - agente: Construtor do Léxico
   - obs: implementado em C e AWK pelo pesquisador Akeo Tanabe, do LES/DI/PUC-Rio

A Tabela 10 relaciona agentes, Web services associados, software utilizado e estado atual da implementação, permitindo ao leitor uma visão geral dos resultados obtidos até o momento na ferramenta de suporte à estratégia utilizada.

Tabela 10 - Estado atual da implementação da ferramenta de suporte

AGENTE	SERVIÇO WEB	SOFTWARE	ESTADO
Comunicador			implementado
Analisador Léxico	conversão de formatos	Yoshikoder Converter	modificado e encapsulado
	Concordanceador		implementado
	individualizar requisitos		implementado
	identificador de padrões		implementado
	stemizador		implementado e encapsulado
Construtor do Léxico	tagger	QTAG	encapsulado*
	tokenizer	qtoken	encapsulado*
		analisador	ainda não encapsulado
Manager			implementado

AGENTE	SERVIÇO WEB	SOFTWARE	ESTADO
Observador			implementado
Estatístico	frequencyMap		implementado e encapsulado
	metrics		implementado e encapsulado
Gerador de Visões	build		implementado e encapsulado
	setAssociations		implementado e encapsulado
		TMNav	ainda não encapsulado
Rastreador			não implementado
Validador			não implementado
Verificador	TermDocument	pretext	encapsulado
	calculateSimilarity		implementado e encapsulado
	contectAnalysis	Yoshikoder	modificado ainda não encapsulado