Uso de Agentes no Processo de Requisitos em Ambientes Distribuídos de Desenvolvimento

Miriam Sayão
PUC-RS, PUC-Rio
Brasil
miriam@inf.puc-rio.br

Julio Cesar S. P. Leite
PUC-Rio
Brasil
www.inf.puc-rio.br/~julio

Resumo: Dentre as atividades de Engenharia de Software desenvolvidas no decorrer do desenvolvimento de um sistema de software, as mais intensivas em comunicação são aquelas ocorridas na fase do Processo de Requisitos. Nesta fase interagem diferentes atores ou interessados[1]: usuário, cliente, desenvolvedor, engenheiro de requisitos, coordenador do projeto, inspetores de qualidade. Cada um desses atores desempenha um papel no processo, visando atingir seus próprios objetivos; todos colaboram visando atingir um objetivo comum, que é a obtenção, ao final do processo de desenvolvimento, de um produto de qualidade que atenda às necessidades dos clientes e usuários. Defendemos a idéia que o paradigma de agentes é adequado para apoiar a solução de problemas encontrados no desenvolvimento distribuído de software, no qual fatores como distância, dificuldades de comunicação e diferenças em fusos horários colocam dificuldades adicionais aos atores. Nossa argumentação é validada pela modelagem das atividades de verificação e validação de requisitos, com o uso da modelagem **i*** para mostrar a intencionalidade de diferentes atores nesse processo.

Keywords: gerenciamento de requisitos, desenvolvimento distribuído de software, agentes

1. Introdução

No processo de desenvolvimento do software, atividades relacionadas ao Processo de Requisitos envolvem a elicitação, modelagem, verificação e validação dos requisitos. Tais atividades, pela sua própria natureza, são mais intensivas em comunicação que as demais atividades em um processo de desenvolvimento de software. No contexto do processo de requisitos, tomemos como exemplo as atividades de verificação e validação (V&V): o processo de validação de requisitos envolve diferentes atores, como por exemplo engenheiro de requisitos, representantes de clientes e usuários e coordenador ou gerente do processo de desenvolvimento. Nas atividades de verificação em documentos de requisitos interagem diferentes atores: inspetores, secretário, coordenador, autor do documento. Cada um desses atores participa visando atingir seus próprios objetivos, e a colaboração entre

[1] denominamos interessados aos *stakeholders* do processo, aqueles que possuem algum interesse no sistema a ser desenvolvido

eles é necessária para atingir a meta maior associada a cada processo: 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?".

Diversos artigos relatam a exacerbação de dificuldades relacionadas às atividades de comunicação e coordenação no processo distribuído de desenvolvimento de software [Carmel99] [Paré99] [Bianchi02] [Damian03] [Herbsleb03]. Diferenças culturais implicam em dificuldades de compreensão da linguagem natural utilizada nos documentos de requisitos; diferenças em fusos horários podem dificultar a comunicação síncrona entre os interessados. Nesse cenário, praticamente inexistem os encontros informais entre desenvolvedores, que muitas vezes resultam em compartilhamento de conhecimento relacionado ao projeto ou mesmo em auxílio na resolução de problemas. Encontros virtuais ou troca de informações síncronas não substituem o relacionamento face a face, e a confiança entre os interessados e o espírito de grupo são afetados.

A orientação a agentes tem sido apresentada como adequada para tratar problemas complexos ou de natureza intrinsecamente distribuída [Jennings00] [Message01] ou ainda sistemas nos quais atores desempenham diferentes papéis e interagem visando atingir diferentes objetivos [Cysneiros03]. O uso de agentes tem sido visto também na solução de problemas relacionados ao processo de desenvolvimento de software [Dellen96] [Gaeta02] [Grundy05]. Propomos, portanto, a utilização do paradigma de agentes em apoio aos interessados em atividades relacionadas ao Processo de Requisitos, tratando especificamente da verificação e validação dos requisitos e incluindo aspectos de rastreabilidade, gerenciamento da evolução dos requisitos ao longo do processo de desenvolvimento distribuído e compartilhamento efetivo de informações relacionadas ao sistema em desenvolvimento.

Nosso principal objetivo é explorar positivamente as possibilidades trazidas pela distribuição do trabalho no contexto do processo de requisitos, visando à obtenção de um documento de requisitos que atenda às necessidades dos usuários (validação), aos requisitos de qualidade esperados (verificação) e respeitando a distribuição das atividades. Neste contexto, a distribuição é uma característica desejável: Filkenstein coloca que o futuro das ferramentas para gerenciamento de requisitos está na distribuição, considerando a tendência de globalização de atividades atualmente existente [Finkelstein00]. As restrições e dificuldades hoje encontradas têm levado muitas empresas a deslocar equipes para interagir diretamente com os clientes e usuários, evitando as dificuldades simplesmente pela eliminação da distribuição nas atividades do processo de requisitos [Zowghi02] [Damian03a] [Lopes05].

A contribuição deste artigo é nossa argumentação e justificativa pela adequação do paradigma de agentes às atividades do processo de requisitos, num contexto de desenvolvimento distribuído. Nossa argumentação considera que o paradigma de agentes vem sendo apresentado como alternativa para a modelagem e implementação de sistemas complexos e distribuídos, dado que agentes autônomos podem atingir seus objetivos mesmo em ambientes dinâmicos, são capazes de interações em alto nível, e conseguem operar em estruturas organizacionais flexíveis [Jennings00].

Na seção 2 são apresentados os principais desafios para a Engenharia de Requisitos no contexto do Desenvolvimento Distribuído de Software, e suas implicações para atividades de validação e verificação no processo de requisitos. A seção 3 apresenta nossa proposta e nossa argumentação para uso de agentes no contexto do Processo de Requisitos, e a seção 4 apresenta os trabalhos relacionados. Conclusões estão presentes na seção 5 e finalmente, a seção 6 traz a bibliografia de referência.

2. Desenvolvimento Distribuído de Software (DDS): desafios para o Processo de Requisitos

Diversos relatos envolvendo *surveys* e experimentos identificaram aquilo que imaginamos sejam os maiores desafios enfrentados pelo conjunto de interessados nas tarefas e atividades associadas à etapa de requisitos. Carmel [Carmel99] considera que os impactos trazidos pela distribuição do desenvolvimento estão relacionados à distância, diferenças culturais e de fusos horários entre interessados. Essas dimensões são discutidas no contexto do Processo de Requisitos em [Zowghi02], para quem distância e fusos horários impactam diretamente na comunicação entre interessados, afetando as atividades de elicitação, negociação e priorização de requisitos. Dentre as diferenças culturais, aquelas relacionadas à normas de comportamento, linguagem e costumes podem levar à falta de confiança entre equipes e tendem a impactar negativamente atividades do processo de requisitos.

Em [Prikladnicki04] são relatados os principais resultados de um *survey* envolvendo duas organizações e vinte e dois desenvolvedores utilizando DDS; os participantes consideraram que o maior desafio no desenvolvimento distribuído em DDS é a Engenharia de Requisitos. A distância entre os participantes dificulta a convergência de idéias e impacta na estabilidade dos requisitos; dificuldades relacionadas à comunicação e falta de confiança entre equipes eram menores quando havia um padrão bem definido a ser seguido, tanto para o processo de desenvolvimento em si quanto para os artefatos gerados.

O *survey* relatado em [Damian03a] apresenta resultados de um estudo de caso envolvendo quatro sites situados em diferentes países, com a participação de vinte e quatro interessados. Dificuldades de comunicação

face-a-face tornam a interação entre grupos dependente das características das ferramentas disponíveis e interferem diretamente nas várias etapas do Processo de Requisitos. Sem um adequado compartilhamento da informação, a confiança entre equipes é atingida, os encontros virtuais necessários para atividades como priorização e negociação de requisitos tendem a não ser efetivos. A diversidade cultural, inclusive diferenças lingüísticas, afeta a compreensão comum dos requisitos e a convergência entre diferentes interesses. *Delays* de tempo devidos aos diferentes fusos horários impactam nas atividades de priorização e negociação, se não devidamente gerenciados. Uma grande contribuição desse trabalho é discutir e apresentar de forma esquemática as dimensões de comunicação, conhecimento, cultura e diferenças temporais no contexto do desenvolvimento distribuído e identificar claramente as atividades afetadas.

2.1. DDS: impactos nas atividades de Requisitos

Nossa proposta de trabalho está diretamente relacionada às dificuldades de comunicação entre interessados, e a aspectos do processo de requisitos que são diretamente afetados por esses problemas. Comunicação é fator crítico de sucesso em projetos distribuídos: já em 1996 Gorton&Matwani [Gorton96] apresentaram resultados de um experimento que aponta como sendo de 22% o tempo utilizado em comunicação entre os membros de uma equipe. A maior parte desse tempo -17%- foi utilizado em comunicação assíncrona, com uso de e-mail e de uma ferramenta de *groupware* desenvolvida especialmente para a empresa. Um estudo feito por Cherry&Robillard [Cherry04], identificou que atividades de *cognitive synchronization* são responsáveis por aproximadamente 29% do tempo de desenvolvimento de um software (*cognitive synchronization* são atividades de comunicação entre dois ou mais desenvolvedores, visando confirmar que eles compartilham o mesmo conhecimento ou a mesma representação do objeto em questão). Outros estudos relacionados em [Cherry04] apontam para atividades de comunicação consumindo de 15 a 41% do tempo total do desenvolvimento.

A proposta de [Lopes05] para o Processo de Requisitos de certa forma corrobora o que foi apontado em [Cherry04]: nessa proposta, uma equipe co-localizada aos usuários elabora o Documento de Requisitos (SRS ou Software Requirements Specification) e o repassa à equipe que executará o desenvolvimento. Esta última equipe, após analisar os requisitos relacionados na especificação, identifica as possíveis fontes de problemas devidos por exemplo à ambigüidade ou falta de clareza na definição dos requisitos. Após a reescrita, o documento de requisitos é objeto de negociações entre equipe de desenvolvimento e equipe de requisitos, até que a equipe de desenvolvimento

tenha confiança no seu entendimento acerca do estabelecido no documento de requisitos.

Nossas observações junto a duas empresas que utilizam DDS e vários relatos mostram que é bastante comum a situação de uma equipe elaborando o documento de requisitos e posteriormente repassando esse documento para a equipe de desenvolvimento [Zowghi02] [Damian03a] [Prikladnicki03] [Audy04a]. Também encontramos a situação onde parte da equipe que efetuará o desenvolvimento se desloca até os clientes para as atividades do processo de requisitos, diminuindo desta forma os problemas decorrentes dos fatores culturais e lingüísticos para a equipe de desenvolvimento. Nestas situações, a distribuição não acontece nas atividades do processo de requisitos. Acreditamos que isto se deve em parte à inexistência de ferramentas para suportar adequadamente a distribuição dos interessados nas atividades do processo de requisitos e em parte para evitar os problemas de comunicação decorrentes da distância.

3. Agentes nas Atividades de Verificação e Validação de Requisitos no Desenvolvimento Distribuído de Software

O paradigma de orientação a agentes para desenvolvimento de software tem sido objeto de trabalhos desde o início da década de 80 [Faltings00]. A literatura aponta para diversas e às vezes conflitantes definições para o termo *agentes;* a propriedade de *autonomia* é apontada como fundamental para muitos autores [Wooldridge99a]. O termo *autonomia* é utilizado para indicar que agentes podem agir sem necessariamente necessitar da intervenção de usuários humanos ou outros sistemas. Aplicações utilizando agentes são encontradas em diferentes domínios, e recentemente no domínio da própria Engenharia de Software [Himmelspach03] [Dhavachelvan04] [Grundy05].

Wooldridge define agente como *um sistema de computação situado em algum ambiente e capaz de ações autônomas sobre este ambiente de forma a atingir os objetivos desejados* [Wooldridge99a]. O agente monitora o ambiente através de sensores, e pode disparar ações que irão atuar sobre esse mesmo ambiente; a forma como isso ocorre aparenta não-determinismo. Outra definição para agentes encontrada na literatura, e que será utilizada no contexto deste trabalho, estabelece que *agente é um elemento interativo, adaptativo e autônomo que possui um estado mental, e estas são as três características fundamentais que definem um agente* [Silva03]. Estado mental é definido como sendo o conjunto de crenças, metas, planos e ações. O conhecimento que o agente tem sobre si mesmo, sobre o ambiente e sobre outros agentes é denominado de crenças. Metas remetem aos objetivos que o agente deve atingir; planos definem as seqüências de ações a serem executadas pelo agente, visando atingir os objetivos propostos.

Jennings e Wooldridge [Jennings01] colocam que a tecnologia de desenvolvimento orientada a agentes ainda está num estágio inicial, mas que sua aceitação aponta para um futuro significativo na engenharia de software. Acreditamos que o uso de agentes é apropriado no contexto do Processo Distribuído de Requisitos, porque o uso de agentes em sistemas de software propicia:

a) **autonomia**: agentes podem monitorar o ambiente, detectando modificações significativas ou ocorrência de eventos e comunicá-las aos envolvidos. No contexto em pauta, agentes podem monitorar mudanças em artefatos de requisitos, comunicando automaticamente modificações aos interessados;
b) **flexibilidade**: agentes podem ser dinamicamente inseridos ou retirados do ambiente de execução, de forma transparente. O conjunto de interessados de um software evolui ao longo do processo de requisitos, e o conjunto de agentes pode refletir essa evolução;
c) **interatividade**: agentes podem se comunicar através de mensagens; atividades de comunicação entre interessados são essenciais às atividades do gerenciamento de requisitos e agentes podem realizar parte dessas comunicações;
d) **agregação do conhecimento distribuído**: agentes podem colaborar e cooperar nas atividades que exigirem agregação de conhecimento para tomada de decisão. O conhecimento relacionado ao sistema está distribuído entre interessados geograficamente dispersos, e agentes podem colaborar nessa agregação para a tomada de decisões;
e) **acesso controlado à informação**: colocar restrições ao acesso de agentes é tarefa facilmente realizável. No contexto de desenvolvimento de software, as restrições aos diferentes interessados com diferentes direitos de acesso às informações podem ser atendidas pelos agentes;
f) **racionalidade**: *crenças* e *metas* no modelo de agentes podem facilmente representar conhecimentos e objetivos dos participantes do processo de requisitos;
g) **continuidade temporal**: agentes podem estar ativos de forma contínua, monitorando os ambientes, detectando eventos e tomando decisões de forma independente, o que é essencial no caso do desenvolvimento *follow-the-sun*, quando os interessados geograficamente dispersos estarão ativos desempenhando diferentes tarefas durante as 24 horas do dia.

Nossa proposta visa repassar a agentes de software parte das tarefas e das comunicações que atualmente ocorrem entre agentes reais - os interessados*;* pretendemos abranger atividades relacionadas à verificação e à validação de requisitos. Visualizamos agentes de software como

representantes dos diferentes atores nesses processos, propiciando a efetiva distribuição do trabalho no contexto do processo de requisitos e diminuindo os problemas de comunicação entre os interessados reais.

No processo de **verificação**, uma atividade possível é a inspeção do documento de requisitos. O processo de inspeção caracteriza-se pela utilização de uma técnica de leitura aplicável a um artefato, buscando a localização de erros ou defeitos no mesmo, segundo um critério pré-estabelecido. A inspeção pode utilizar diferentes técnicas de leitura; a técnica escolhida deve identificar as informações a serem verificadas e apoiar a localização de defeitos nestas informações. Das técnicas existentes, identificamos a PBR (*Perspective Based Reading*) como sendo adequada aos nossos propósitos. PBR [Basili96] considera as diferentes perspectivas (visões) dos atores do processo e utiliza diferentes *checklists* para guiar a leitura e análise do artefato.

No processo de inspeção segundo a técnica PBR, os diferentes atores envolvidos no processo de desenvolvimento vêem os artefatos sob diferentes pontos de vista, e portanto são selecionados de acordo com a utilização que farão do artefato. Numa inspeção do documento de requisitos, podem ser selecionados, por exemplo: usuário final, projetista, programador, representante da equipe de manutenção, executor de testes e outros. O documento de requisitos será visto através de diferentes visões: o usuário final deseja ver ali refletido o conjunto de funcionalidades a ser atendido pelo software; o projetista irá utilizá-lo como base para a criação da estrutura do software, considerando as funcionalidades e restrições descritas, e o testador verá o documento de requisitos como fonte primeira para a geração dos testes, que deverão assegurar que o mesmo atende às necessidades (funcionais e não funcionais) registradas. Na nossa visão, as tarefas de inspeção podem ser parcialmente automatizadas por agentes. A cada ator corresponderia um agente que agiria em seu nome, executando ações em acordo com a perspectiva de seu papel e seguindo o estabelecido em um plano pré-definido. Parte das tarefas da inspeção seria automaticamente realizada, com tratamento da linguagem natural utilizada no SRS.

No processo de **validação**, engenheiro de requisitos e representantes do cliente e dos usuários avaliam o SRS e interagem com o objetivo de assegurar que os requisitos relacionados pelo engenheiro de requisitos no SRS correspondem ao esperado pelos usuários e cliente. Cada representante dos usuários trará a visão do grupo ao qual ele pertence: operacional, gerencial, decisório. Neste processo visualizamos agentes com o propósito de identificar se as intenções do ator a quem ele representa estão refletidas no documento sendo avaliado.

Visando ilustrar nossa proposta, apresentamos na figura 1 o modelo de dependência estratégica **i*** [Yu02] dos processos de Validação e Verificação de requisitos. Nesse modelo são mostradas as metas de cada

agente e as dependências entre agentes na consecução de seus objetivos ou metas. Os principais atores (cliente, desenvolvedor, engenheiro de requisitos, gerente do projeto, inspetor) possuem relações de dependência por recursos, por *goals* (metas) ou por *softgoals*, que representam a intencionalidade dos atores.

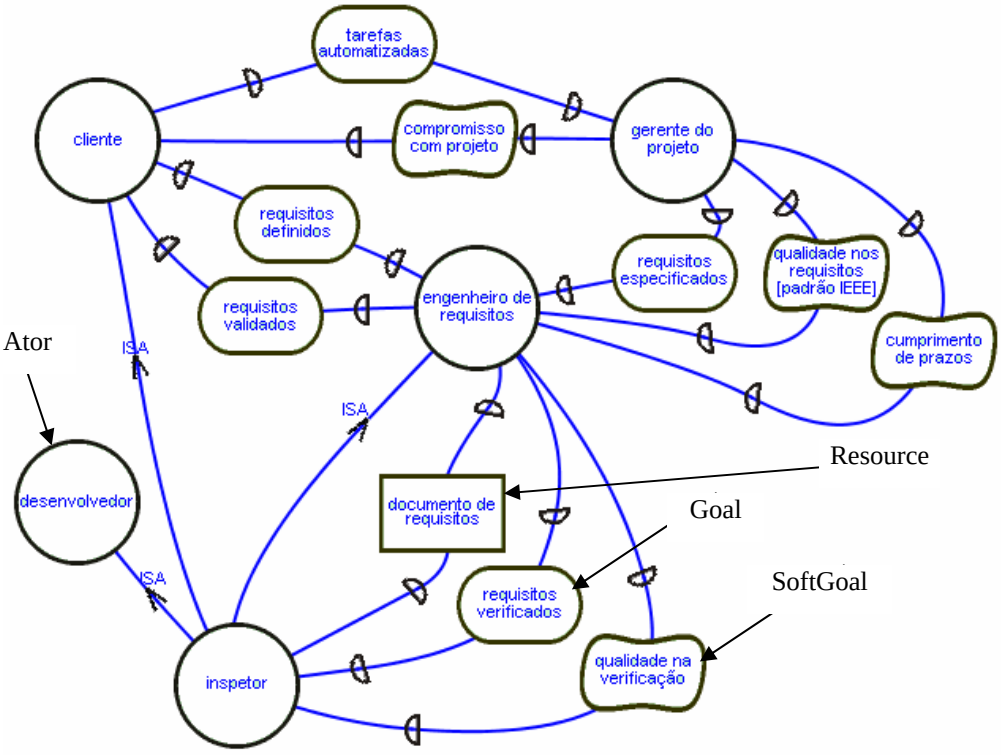

Fig 1 - Modelo de dependência estratégica para verificação e validação de requisitos

Além dos agentes já relacionados, outros poderão também ser utilizados, com propósitos de atender atividades não estritamente relacionadas à V&V. Como exemplo, um agente responsável pela manutenção das informações de rastreabilidade poderia atuar no processo de validação e associar requisitos aos representantes de cliente e usuários. Um outro agente poderia monitorar modificações no SRS e notificar de forma automática os interessados nessas alterações. A implementação desta proposta, na qual estamos trabalhando, envolve atendimento de características como:

- **agência**: diferentes papéis a serem desempenhados pelos agentes, na representação dos interessados reais;

- **serviços de comunicação**: para possibilitar a troca de informações entre usuários e agentes de software, e para notificação dos interessados na ocorrência de eventos;
- **serviços de gerenciamento**: para possibilitar aos agentes o monitoramento de modificações no ambiente e eventos significativos nesse contexto;
- **persistência de dados**: armazenamento de informações relacionadas a projetos e usuários e uma *baseline* para artefatos de requisitos [Leite95];
- **serviços de autenticação**: para validação dos usuários aptos a utilizar o sistema;
- **geração de relatórios**: para impressão de relatórios e de artefatos armazenados no sistema;
- **interface com usuários**: para possibilitar fácil interação entre usuários e sistema;
- **integração com outras ferramentas**: o uso de XML deverá possibilitar a troca de informações entre ferramentas.

4. Trabalhos relacionados

A área de pesquisa em Desenvolvimento Distribuído de Software tem recebido maior atenção da academia nos últimos dez anos, com um bom número de publicações, muitas das quais envolvendo experimentação e *surveys* [Gorton96] [Paré99] [Regnell01] [Damian01] [Damian03] [Herbsleb03] [Audy04] [Prikladnicki04]. Na extensa pesquisa que realizamos, os trabalhos diretamente relacionados a este abordam aspectos pontuais do Processo de Requisitos em DDS e não utilizam o paradigma de agentes.

A priorização de requisitos com uso distribuído da técnica Quality Function Deployment (QFD) para definição dos requisitos de dois diferentes sistemas de software é apresentada em [Hrones93]. A comunicação entre os interessados, localizados em sites geograficamente distantes da empresa Digital Corp., foi realizada com o uso de equipamentos de teleconferência. Regnell et al [Regnell01] apresentam um estudo de caso com priorização distribuída, cujo objetivo principal foi avaliar diferentes segmentos do mercado para definir os requisitos a serem considerados num sistema a ser desenvolvido. A negociação de requisitos em ambientes distribuídos foi abordada em [Damian01] [Damian03]. Nestes artigos Damian et al investigam o uso de um facilitador para a condução de processos de negociação de requisitos envolvendo interessados geograficamente separados.

Outro trabalho relacionado está descrito em [Lanubile03], onde é apresentada a ferramenta ÍBIS, para apoio à verificação do documento de especificação de requisitos de software com uso de inspeção segundo a

técnica PBR. A ferramenta, construída para uso em ambientes distribuídos de desenvolvimento, armazena o documento a ser inspecionado, possibilita o cadastro e a seleção dos inspetores, fornece *checklists* e formulários aos inspetores e registra os problemas detectados por cada um deles. Os relatórios são depois analisados e os problemas são agrupados; a consolidação é feita posteriormente pelo coordenador do processo. Não é feita referência à automação parcial de tarefas de inspeção, e a apresentação da arquitetura da ferramenta não menciona uso de agentes de software.

O trabalho mais próximo desta proposta está descrito por Gaeta et al [Gaeta02], onde o sistema desenvolvido para apoio ao gerenciamento de processos distribuídos de engenharia de software utiliza agentes de software. O trabalho, desenvolvido no contexto do projeto GENESIS (GEneralised eNvironment for procEsS management in cooperatIve Software engineering), utiliza técnicas de workflow e de gerenciamento de documentos para gerenciamento de projetos e comunicação entre engenheiros de software. O uso de agentes foi escolhido por propiciar uma abordagem menos invasiva para atividades de coordenação e controle entre sites. Neste trabalho os agentes são responsáveis por mecanismos de manipulação de exceções, pela sincronização de processos entre os sites distribuídos e pela monitoração e coleta de informações relacionadas a processos.

Em contraponto, nossa proposta visa colocar agentes de software como reais representantes dos interessados, tanto para aspectos relacionados à comunicação como para outros associados à tomada de decisões e execução automática de tarefas. Estamos iniciando o trabalho na construção de um *framework* que possibilite a agregação de serviços via *plug-ins*, atingindo aspectos relacionados não só a um efetivo compartilhamento de informações entre interessados geograficamente dispersos, mas também melhoria da qualidade do documento de requisitos e o gerenciamento da evolução dos requisitos.

5. Conclusões

O paradigma de agentes tem sido indicado para apoiar a solução de problemas de natureza intrinsecamente distribuída ou também sistemas nos quais atores desempenham diferentes papéis e interagem visando atingir diferentes objetivos. Este é o caso do Processo de Desenvolvimento de Software, onde atores como usuários, clientes, engenheiro de requisitos, projetistas, engenheiros de software e desenvolvedores atuam visando atingir, cada um deles, os seus próprios objetivos, e colaboram visando atingir um objetivo comum, que é o desenvolvimento de um software de qualidade que atenda às necessidades dos clientes e usuários. Nesse contexto se enquadra o Processo de Requisitos, fase na qual são gerados os requisitos que guiarão todo o processo de desenvolvimento. Ferramentas baseadas em agentes

também assistem na solução de problemas relacionados ao processo de desenvolvimento de software, e publicações recentes tem mostrado essa abordagem.

Nossa proposta identifica o Processo de Requisitos como área onde soluções baseadas em agentes são desejáveis, e justificamos nossa premissa apresentando características de agentes e sua adequação a atividades do Processo de Requisitos. Apresentamos o modelo de dependência estratégica relacionado às etapas de Verificação e Validação de Requisitos, onde são visualizados os principais atores, suas metas e as dependências entre eles.

Etapas futuras da presente pesquisa estarão concentradas em três vertentes. A primeira vertente é a construção de um suporte de software para que nossas idéias possam ser implementadas. Esse suporte proverá meios para que possamos simular com agentes algumas das atividades de V&V apontadas acima. Estamos avaliando alguns *frameworks* para construção de sistemas multi-agentes na intenção de identificar aquele que mais se adequa aos nossos propósitos, e nossa avaliação preliminar aponta para o uso do JADE. Em particular, estaremos trabalhando com os aspectos de inspeção com a técnica PBR e sua caracterização distribuída. Resultados anteriores obtidos na automação de inspeções em cenários com uso de técnicas de processamento da linguagem natural indicam que essa linha de trabalho deve ser aprofundada [Sayão03]. A segunda vertente é a aplicação de agentes de V&V em estudos de casos baseados em situações reais de desenvolvimento distribuído. A terceira vertente é o uso e o acompanhamento do sistema multi-agente de inspeção em um projeto real. Essas etapas futuras contribuirão para confirmar nossa hipótese da utilidade de sistemas multi-agentes na construção de requisitos em ambientes distribuídos.

Agradecimentos

aos profs Carlos J. de Lucena e Ricardo Choren, pelas revisões e sugestões.

6. Bibliografia

[Audy04] Audy, Jorge; Evaristo, Roberto; Watson-Manheim, Mary. "Distributed Analysis: the Last Frontier?" In: 37th Hawaii International Conference on System Sciences, 2004. Proceedings. pp. 1-9.

[Audy04a] Audy, Jorge L. N. & Lopes, Leandro. "Towards a reference model for requirement engineering in distributed software development". In: CAiSE - 16th International Conference on Advanced Information Systems Engineering, Riga, Letonia. 2004. Proceedings.

[Basili96] Basili, V. R.; Green S.; Laitenberger, O.; Lanubile, F.; Shull, F.; Sorumgard, S.; Zelkowitz, M. V. "The Empirical Investigation of Perspective-Based Reading", Empirical Software Engineering Journal, vol. I, 1996. pp. 133-164.

[Bianchi02] Bianchi, A.; Caivano, D.; Lanubile, F.; Rago, F. & Visaggio, G. "An Empirical Study of Distributed Software Maintenance". In: International Conference on Software Maintenance (ICSM02). Proceedings.

[Carmel99] Carmel, E. "Global Software Teams". Prentice-Hall, 1999.

[Cherry04] Cherry, S. & Robillard, P. "Communication Problems in Global Software Development: Spotlight on a New Field of Investigation". In: Third International Workshop on Global Software, May 24, 2004, Edinburgh, Scotland. Proceedings.

[Cysneiros03] Cysneiros, Luiz M. & Yu, Eric. "Requirements Engineering for Large Scale Multi-Agent Systems". In: 2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems, Portland, Oregon, USA, 2003. Proceedings. pp. 36-57.

[Damian01] Damian, D.; Eberlein, A.; Woodward, B.; Shaw, M. & Gaines, B. "An empirical study of facilitation of computer-mediated distributed requirements negotiations". In: Fifth IEEE International Symposium on Requirements Engineering (RE '01), August 27-31, 2001. Toronto, Canadá. Proceedings. pp. 128-135.

[Damian03] Damian, D.; Eberlein, A.; Shaw, M. & Gaines, B. "An exploratory study of facilitation in distributed requirements engineering". Requirements Engineering Journal, 8(1), 2003, pags 23-41.

[Damian03a] Damian, D. & Zowghi, D. "RE challenges in multi-site software development organizations". Requirements Engineering Journal, 8(3),2003, pp. 149-160.

[Dellen96] Delle, B. & Maurer, F. "Integrating Planning and Execution in Software Development Process". In: Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE'96). Proceedings, pp. 170-176.

[Dhavachelvan04] Dhavachelvan, P.; Uma, G.V. "Reliability Enhancement in Software Testing- An Agent-Based Approach for Complex Systems". Lecture Notes in Computer Science, vol. 3356,jan 2004. pp. 282 - 291.

[Faltings00] Faltings, B. Intelligente Agents: Software Technology for the new Millennium. In: Informatik 1/2000, pp. 2-5. Disponível em <http://www.svifsi.ch/revue/pages/issues/n001/no001.html>. Acesso 11.12.2003.

[Finkelstein00] Finkelstein, Anthony & Emmerich, Wolfganf. "The future of Requirements Management Tools". In *Information Systems in Public Administration and Law*, Quirchmayr, G., Wagner R. & Wimmer M. (eds), Oesterreichische Computer Gesellschaft, 2000.

[Gaeta02] Gaeta, M. & Ritrovato, P. "Generalised Environment for Process Management in Cooperative Software Engineering". In: 26th Annual International Computer Software and Applications Conference (COMPSAC'02). Proceedings.

[Gorton96] Gorton, I. & Motwani, S. "Issues in Co-operative Software Engineering Using Globally Distributed Teams". In: Information and Software Technology Journal ,vol 38(10), 1996. pp. 647-655.

[Grundy05] Grundy, J.; Ding, G. & Hosking, J. "Deployed software component testing using dynamic validation agents". The Journal of Systems and Software 74 (2005). pp 5-14.

[Herbsleb03] Herbsleb, J. & Mockus, A. "An empirical study of speed and communication in globally distributed software development". IEEE Transactions on Software Engineering, vol 29(6), jun 2003. pp. 481-494.

[Himmelspach03] Himmelspach, J.; Rohl, M. & Uhrmacher, A.M. "Simulation for testing software agents - an exploration based on James". In: Simulation Conference, 2003. Proceedings. pp. 799-807.

[Hrones93] Hrones, John Jr.; Jedrey, Benjamin; Zaaf, Driss. "Defining Global Requirements with Distributed QFD". In: Digital Technical Journal, vol 5(4), 1993. pp. 36-46.

[Jennings00] Jennings, N.R.. "On agent-based software engineering". In: Artificial Intelligence, vol. 117 (2000). pp. 277–296.

[Lanubile03] Lanubile, F.; Mallardo, T. "Preliminary Evaluation of Tool-based Support for Distributed Inspection". In: 26th Annual International Computer Software and Applications Conference (COMPSAC'02). Proceedings.

[Leite95] Leite, J.C.S.P. and Oliveira, A.P. "A Client Oriented Requirements Baseline". In: Second IEEE International Symposium on Requirements Engineering (RE'95), 1995. Proceedings. pp.108-115.

[Lopes05] Lopes, L.; Prikladnicki, R.; Audy, J. & Majdenbaum, A. "Requirements Specification in Distributed Software Development - A Process Proposal". In: 38th Hawaii International Conference on System Sciences - HICSS. Hawaii, USA, 2005. Proceedings.

[Message01] MESSAGE: Methodology for Engineering Systems of Software Agents - Final guidelines for the relevant problems areas where agent technology is appropriate. EURESCOM Participants in Project P907-GI (2001).

[Paré99] Paré, G. & Dubé, L. "Virtual Teams: An Exploratory Study of Key Challenges and Strategies". In: 20th International Conference on Information Systems, dez 1999, Charlotte, North Carolina, United States. Proceedings. pp. 479-483.

[Prikladnicki03] Prikladnicki, R., Audy, J.L.N., & Evaristo, R. "Global software development in practice: lessons learned". Software Process: Improvement and Practice, 8(4), 2003. pp. 267-281.

[Prikladnicki04] Prikladnicki, R. & Audy, J. "MuNDDoS - Um Modelo de Referência para Desenvolvimento Distribuído de Software". In: XVIII Simpósio Brasileiro de Engenharia de Software - 2004 - Brasília, DF, Brasil. Anais. pp. 289-304.

[Prikladnicki04a] Prikladnicki, R.; Audy, J. & Evaristo, R. "An empirical study on Global Software Development: Offshore Insourcing of IT Projects". In: Third International Workshop on Global Software, May 24, 2004, Edinburgh, Scotland.

[Regnell01] Regnell,B.; Höst, M.; Natt,J.; Beremark, P. & Hjelm, T. "An industrial case study on distributed prioritization in market-driven requirements engineering for packaged software". Requirements Engineering Journal 6(1), 2001. pp. 51-62.

[Sayão03] Sayão, M. "INSPETOR - Inspeção Automatizada de Cenários". Trabalho Individual de Programação, Departamento de Informática, PUC-Rio, 2003.

[Silva03] Silva, V.; Garcia, Alessandro; Brandão, A.; Chavez, C.; Lucena, C. & Alencar, P. "Taming Agents and Objects in Software Engineering". In: Software Engineering for Large-Scale Multi-Agent System, Garcia, A., Lucena, C., Zamboneli, F., Omicini, A, and Castro, J., Eds., LNCS, Springer-Verlag, 2003.

[Wooldridge99a] Wooldridge, M. Intelligent Agents. In: Weiss, Gerhard (Ed.). *Multiagent Systems - A Modern Approach*. MIT Press, 1999. pp. 27-77.

[Yu02] Yu, E., "Agent-Oriented Modelling: Software Versus the World". In: Agent-Oriented Software Engineering AOSE-2001 Workshop Proceedings, Montreal, Canada - May 29th 2001. LNCS 2222.

[Zowghi02] Zowghi, Didar. "Does Global Software Development need a different Requirements Engineering Process?" In: International Workshop on Global Software Development - ICSE 2002, Orlando, Florida, USA, 2002. Proceedings. pp. 53-55.