Uma Abordagem Baseada em Gestão do Conhecimento para Gerência de Requisitos em Desenvolvimento Distribuído de Software

Rodrigo Espindola, Leandro Lopes, Rafael Prikladnicki, Jorge Luiz Nicolas Audy
Programa de Pós-Graduação em Ciência da Computação
Faculdade de Informática
Pontifícia Universidade Católica do Rio Grande do Sul
{respindola, lteixeira, rafael, audy}@inf.pucrs.br

Resumo. A distribuição das equipes de desenvolvimento tem provocado diversos desafios ao processo de software. Dentre os desafios, a engenharia de requisitos destaca-se, sofrendo impacto de fatores como distância, diferença de cultura e fuso-horário, bem como dos meios de comunicação disponíveis. Nesse contexto, o gerenciamento das informações relacionadas a requisitos torna-se crítico para garantir que as informações necessárias sobre um determinado domínio ou aplicação estão disponíveis para as equipes de desenvolvimento, bem como que estas informações sejam organizadas de forma a permitir futuro acesso por projetos de manutenção no mesmo escopo. Nesse sentido, este artigo propõe-se a analisar a engenharia de requisitos em projetos de desenvolvimento distribuído de software sob a ótica da gestão de conhecimento. A principal contribuição apresentada é a identificação das informações pertencentes às categorias de conhecimento envolvidas, bem como a forma com que devem ser disponibilizadas.

Palavras-chave: Engenharia de Requisitos, Manutenção de Software, Gestão de Conhecimento.

1. Introdução

Hoje em dia, é cada vez mais significativo o número de empresas que estão distribuindo seus processos de desenvolvimento de software ao redor do mundo. Desta forma, tem-se criado um cenário onde projetos de software são desenvolvidos com equipes distribuídas, caracterizando assim o desenvolvimento distribuído de software (DDS). Este ambiente criou uma nova classe de problemas a serem resolvidos pelos pesquisadores na área de desenvolvimento de software, centrada no DDS. Muitas vezes estes problemas ocorrem, pois não existe uma identificação correta das pessoas envolvidas nos projetos, da informação que deve percorrer os projetos e as organizações envolvidas em projetos distribuídos. Do ponto de vista de processo de desenvolvimento, a engenharia de requisitos é tida como um dos principais desafios para projetos desenvolvidos neste contexto. Em projetos distribuídos, percebe-se uma grande dificuldade das organizações em elicitar, analisar e gerenciar corretamente os requisitos. Desta forma, alguns estudos foram desenvolvidos nos últimos anos [1], [2], [3], [4], [5], [6]

buscando entender quais são as principais dificuldades presentes na engenharia de requisitos, em um cenário específico de desenvolvimento distribuído de software.
Este artigo tem por objetivo analisar a engenharia de requisitos em projetos de DDS sob a ótica da gestão de conhecimento. Desouza e Evaristo [7] realizaram um estudo sobre gestão de conhecimento em projetos distribuídos, onde procurou entender que tipo de conhecimento está envolvido em projetos desta natureza e onde este conhecimento deve estar para acesso das pessoas envolvidas. Desta forma, este estudo analisou a proposta de Evaristo buscando adaptá-la para o contexto de engenharia de requisitos. Propõe-se então uma forma de trabalhar com requisitos de software em projetos de DDS, aproveitando-se de um modelo que trabalha a gestão de conhecimento.
Como resultado, identifica-se quais tipos de conhecimento sobre requisitos são necessários ao longo de um projeto de software, quem deve ter conhecimento e como este conhecimento pode ser armazenado em um repositório para consultas futuras. Este artigo está organizado da seguinte forma: a seção 2 apresenta a base teórica, a seção 3 apresenta os trabalhos relacionados, a seção 4 apresenta uma descrição dos desafios na ER para Manutenção de Software em DDS, a seção 5 apresenta a classificação do conhecimento em gerência de requisitos com base na abordagem de Desouza e Evaristo [7], a seção 6 apresenta uma análise das implicações das abordagens de gestão do conhecimento para gerência de requisitos e, por fim, a seção 7 apresenta as conclusões deste artigo e os trabalhos futuros.

2. Base Teórica

2.1 Engenharia de Requisitos

O processo de engenharia de requisitos é crítico para o sucesso no desenvolvimento de um software. Uma incorreta identificação dos requisitos de um software pode levar ao desenvolvimento de um produto que não atende aos objetivos para o qual foi planejado, sendo total ou parcialmente desperdiçado. Durante o processo de desenvolvimento, problemas nos requisitos podem gerar a necessidade de um novo ciclo de especificação, projeto, codificação e teste, afetando diretamente os custos e prazos envolvidos.
A engenharia de requisitos pode ser definida como a ciência e disciplina preocupada com a análise e documentação dos requisitos, incluindo análise das necessidades e análise e especificação dos requisitos. Além disso, a engenharia de requisitos fornece mecanismos apropriados para facilitar as atividades de análise, documentação e verificação [8].
Um processo de engenharia de requisitos é um conjunto estruturado de atividades que são seguidas para derivar, validar e manter um documento de requisitos de um sistema. Uma descrição completa de um processo inclui as atividades que devem ser conduzidas, a estrutura ou agenda destas atividades, as entradas e saídas de cada atividade e as ferramentas utilizadas para suportar a engenharia de requisitos [9].

2.2 Desenvolvimento Distribuído de Software (DDS)

Observou-se na última década um grande investimento na conversão de mercados nacionais em mercados globais, criando novas formas de competição e colaboração entre os países [10]. Entretanto, o mercado global de software vinha passando por diversas crises. Por um lado, um grande número de falhas em projetos. De outro, uma crescente demanda, atingida pela escassez de recursos capacitados. Nesse ambiente, muitas organizações encontraram no Desenvolvimento Distribuído de Software (DDS) uma alternativa. Atualmente um grande número de organizações desenvolve software com equipes geograficamente distribuídas [11], [12], [1].

Entre os fatores que têm contribuído para o crescimento do DDS, podemos destacar o custo mais baixo e disponibilidade de mão de obra; a evolução e maior acessibilidade a recursos de telecomunicação; a evolução das ferramentas de desenvolvimento; a necessidade de possuir recursos globais para utilizar a qualquer hora; as vantagens de estar perto do mercado local; a formação de equipes virtuais para explorar as oportunidades de mercado; e a pressão para o desenvolvimento *time-to-market*, utilizando as vantagens proporcionadas pelo fuso horário diferente, no desenvolvimento conhecido como *round-the-clock*, ou seja, o desenvolvimento quase que contínuo.

Neste sentido, o DDS tem atraído um grande número de pesquisas na área de Engenharia de Software [1], [2], [3], [4], [5], [6]. São freqüentes os esforços que os pesquisadores têm feito no intuito de entender os fatores que permitem organizações multinacionais ou virtuais a obterem sucesso trabalhando através das fronteiras físicas e culturais dos países. Em suma, existe uma série de problemas e desafios inerentes ao desenvolvimento de software. E o DDS, ao acrescentar fatores como dispersão geográfica e dispersão temporal acentuou alguns dos desafios existentes e acrescentou novos desafios ao processo de desenvolvimento.

2.3 Engenharia de Requisitos em DDS

Em ambientes de DDS, dificuldades como distância, comunicação e cultura causam um aprofundamento dos problemas inerentes ao processo de engenharia de requisitos, que adquire um caráter ainda mais crítico [13]. Como a distância envolvida pode compreender países distantes, comumente, a linguagem e a cultura são diferentes. Com isso, os problemas causados por ambigüidade e falta de clareza nos requisitos são intensificados. Sem o conhecimento presencial do ambiente onde o software será inserido, a compreensão das razões e expectativas do software por parte da equipe de desenvolvimento é reduzida [14].

A comunicação também apresenta novos desafios. Com a perda de contato face-a-face entre a equipe de desenvolvimento e os usuários, existe uma maior dificuldade de esclarecimento em caso de dúvidas. Além disso, com a utilização de meios de baixo contexto, como correio eletrônico, o contato informal entre os membros dos diversos grupos é limitado, reduzindo a confiança entre eles.

O gerenciamento de informações também influencia a engenharia de requisitos em ambientes DDS. A gestão do conhecimento atua de forma transversal às categorias identificadas, auxiliando, por exemplo, nos processos utilizados e na gerência de in-

formações culturais. O processo engenharia de requisitos, por exemplo, envolve documentos cujo conteúdo pode ter diversas origens.

2.4 Principais informações e artefatos envolvidos na engenharia de requisitos

A engenharia de requisitos, como fase inicial do processo de desenvolvimento de software, envolve diversas formas de conhecimento, influenciados por fatores humanos, sociais e organizacionais. O processo de engenharia de requisitos, além de garantir o desenvolvimento e aprovação de um artefato de requisitos (comumente chamado de SRS - *Software Requirements Specification* [15]), envolve informações de diversas origens e propósitos, necessitando gerenciar as necessidades e expectativas dos envolvidos de forma a produzir um software adequado.
Segundo Kotonya [16], as principais entradas e saídas do processo de engenharia de requisitos são:

- Informações sobre o sistema existente;
- As necessidades dos *stakeholders*;
- Padrões organizacionais;
- Regulamentos, como de segurança e saúde;
- Informações sobre o domínio da aplicação;
- Requisitos acordados pelos *stakeholders*;
- Especificação do sistema;
- Modelos do sistema.

Ainda são considerados, durante a gerência de requisitos os artefatos necessários às atividades de controle de alterações e avaliação do impacto das mudanças, incluindo solicitações de mudança, aprovações, avaliação do impacto, rastreabilidade dos requisitos, entre outros.
Dependendo do processo de desenvolvimento de software utilizado, estas informações podem ser documentadas de formas distintas. O RUP [17] (*Rational Unified Process*), por exemplo, utiliza como artefatos em seu *workflow* de gerenciamento de requisitos o documento visão (*Vision*), o modelo de casos de uso (*Use Case Model*) e a especificação suplementar (*Supplementary Specification*). Estes dois últimos documentos formam a especificação dos requisitos do software (*Software Requirements Specification* – SRS). De forma similar, outros processos de software tendem a seguir o padrão da IEEE [15], e utilizar um documento de especificação de requisitos (SRS). Além disso, para controle de alterações, o uso de um documento de requisição de mudanças (*Change Request*) é implementado por diversos processos.

3. Trabalhos Relacionados

3.1 Estudos de Damian e Zowghi sobre ER em DDS

Os estudos de Damian e Zowghi [13][14] têm como foco principal o entendimento e descrição do impacto exercido pela distância na negociação de requisitos de software. Em [14], é apresentada a relação entre cultura, conflitos e distância na negociação de

requisitos de forma distribuída globalmente. Para isso foi conduzida uma pesquisa baseada em estudos de caso, visando esclarecer o impacto causado pela distribuição dos *stakeholders* nas atividades de ER em ambientes de DDS.
Estes estudos sugerem que a distância tende a exacerbar os problemas fundamentais da ER, como a falta de comunicação entre *stakeholders*, bem como os fatores de natureza política, organizacional e social. Sugere também que pesquisas de campo devem ser realizadas para que se entenda o impacto dos problemas de comunicação e coordenação nas atividades de ER, identifique os desafios apresentados pelas organizações distribuídas, e apresente recomendações para superar os problemas associados com a distância.

3.2 Abordagem de Desouza e Evaristo para Gestão de Conhecimento

Desouza e Evaristo [7] apresentam uma abordagem hibrida para a gestão de conhecimento em projetos distribuídos. Através da análise do trabalho de Hansen e seus colaboradores [18], que dividem as abordagens de gestão de conhecimento em codificação e personalização, os autores traçam um paralelo das mesmas com dois modelos computacionais muito populares: cliente-servidor e ponto-a-ponto (P2P), respectivamente.
De acordo com os autores, na abordagem de codificação o conhecimento individual é condensado, contextualizado e centralmente disponibilizado em banco de dados. Esta abordagem parte da premissa que o conhecimento pode ser facilmente extraído e codificado, tornando-a muito similar ao modelo cliente-servidor. Já a personalização é exatamente o oposto, pois reconhece a dimensão tácita do conhecimento e assume que o conhecimento é compartilhado principalmente pelo contato direto entre os indivíduos. Esta abordagem tem um caráter distribuído que a torna muito semelhante ao modelo P2P.
As duas abordagens têm implicações na gestão de conhecimento em projetos distribuídos, levando os autores a analisar suas vantagens e desvantagens sobre três dimensões da gestão do conhecimento: compartilhamento, controle e estruturação do conhecimento. Com base na análise citada, Desouza e Evaristo propõem uma abordagem híbrida visando maximizar os benefícios dos modelos cliente-servidor e P2P, quando aplicados à gestão do conhecimento.
Neste trabalho a abordagem híbrida é aplicada à gestão de conhecimento em ER para DDS. Vários dos desafios encontrados neste cenário podem ser tratados através desta abordagem. Mas possivelmente, o processo de gerência de requisitos seja o beneficiado pela adoção de práticas de gestão do conhecimento, dado que é o processo afetado pelas dificuldades evidenciadas pela manutenção de software. O próximo capítulo apresenta uma análise destas dificuldades.

4. Desafios na ER para Manutenção de Software em DDS

O processo de ER exige a manipulação de uma grande quantidade de informações. Tipicamente, além dos documentos de requisitos propriamente ditos, é necessária também a manutenção de informações sobre controle de versão, relatórios de análise de impacto, estimativas de custos e todas as informações necessárias à gerência do

processo de submissão e apreciação das requisições de mudança de requisitos. Fazendo com que estes processos tenham uma grande capacidade de produzir e consumir conhecimento.

No contexto de manutenção de software, o conhecimento sobre requisitos é particularmente importante. Segundo Liu e seus colaboradores [19], compreender os requisitos de sistemas legados torna-se importante para:

- Compreender os *stakeholders*, os usuários e o contexto de negócio do sistema;
- Obter uma visão geral das funcionalidades e do ambiente do sistema;
- Compreender as intenções por trás do projeto original do sistema;
- Obter conhecimento sobre a interação do sistema com seus usuários no suporte às operações de negócio;

Este conhecimento é útil para melhorar o sistema legado, integrá-lo com o restante dos sistemas de informação ou realizar a reengenharia do sistema. Infelizmente, nem sempre é possível obter este conhecimento. De acordo com diversos autores [20][21][22][23][19], a ausência de documentação e a indisponibilidade dos *stakeholders* originais é um cenário muito comum na prática de manutenção de sistemas legados. Muitas vezes, o sistema em si, seu código-fonte e seus usuários são as únicas fontes de conhecimento sobre tais sistemas.

Um outro fator negativo para gerência de requisitos em projetos de manutenção é que a ER é, geralmente, discutida como uma fase inicial no desenvolvimento de software [23]. Entretanto, o conhecimento dos requisitos requer um esforço contínuo no refinamento progressivo das exigências contidas nas regras de negócio das organizações e em atendimento às necessidades dos *stakeholders*, durante todo o ciclo de vida do software. Assim, os requisitos devem ser gerenciados de forma a evoluírem em sincronia com o software. Práticas de ER que vinculem os requisitos ao escopo de projetos, ao invés do produto de software, podem levar a uma fragmentação das especificações de requisitos e dificultar a evolução e a transferência do conhecimento sobre a aplicação legada.

Além disto, em ambientes de DDS fatores como diferenças culturais e de idioma dificultam o processo de transferência de conhecimento sobre a aplicação entre os diversos *stakeholders* geograficamente distribuídos. De acordo com Damian [14] e Lopes [4], a gestão de conhecimento influencia a ER em ambientes de DDS. A ER em DDS envolve, por exemplo, documentos cujo conteúdo pode ter diversas origens. Captar, processar, armazenar e disponibilizar globalmente o conhecimento gerado e consumido pelos processos de manutenção e pela ER são questões que podem ser endereçadas pela gestão do conhecimento.

Cabe salientar também que os desafios aqui descritos não estão restritos ao escopo da manutenção de software, sendo muitas vezes causados por decisões tomadas e práticas adotadas durante o projeto de desenvolvimento de software. Mas é sob a perspectiva da manutenção de software que estes desafios são evidenciados. Uma parte significativa do conhecimento gerado durante o desenvolvimento de um novo produto de software, por exemplo, não é reaproveitada durante o projeto inicial. Portanto, a decisão de não preservar este conhecimento não acarreta nenhum prejuízo neste momento, podendo inclusive ser considerada como uma forma de economia de recursos e, con-

sequentemente, redução de custos. Mas a ausência deste mesmo conhecimento poderá acarretar dificuldades em futuros projetos de manutenção de software, principalmente se o projeto de manutenção for executado por outra equipe e em outra localidade.

5. Classificação do Conhecimento em Gerência de Requisitos

De acordo com Damm [24], o conhecimento gerado por projetos pode ser categorizado como conhecimento em projetos, conhecimento sobre projetos e conhecimento proveniente de projetos. O conhecimento em projetos compreende informações detalhadas que são geradas e utilizadas em cada projeto individual. Os envolvidos no projeto necessitam destas informações para saber quando, o que, quem, como, onde e por que algo é feito, visando promover uma coordenação eficiente e efetiva das atividades. O conhecimento sobre projetos compreende informações que a organização necessita para saber que projetos estão sendo conduzidos em qualquer momento. Estas informações contribuem para o planejamento e controle dos recursos utilizados nestes projetos. Já o conhecimento proveniente de projetos compreende as principais informações geradas com a execução do projeto. Este conhecimento é determinante do sucesso futuro do projeto e contribui para o aprendizado organizacional.

O primeiro passo na aplicação da abordagem híbrida de gestão de conhecimento à ER em DDS consiste em aplicar esta categorização às informações relevantes para ER e, em particular, para a gerência de requisitos. Desta forma é possível elaborar uma lista de informações para cada categoria. Durante a execução de um projeto, são necessárias várias informações detalhadas sobre os requisitos e as requisições dos *stakeholders*. Dentre estas informações, as seguintes podem ser destacadas como conhecimento em projeto:

- Novos requisitos e seus respectivos estados, tais como “proposto”, “aprovado” ou “rejeitado”;
- Requisições de alteração de requisitos e seus respectivos estados;
- Relatórios de análise de impacto das alterações requisitadas;
- Estimativas de custo e esforço para novos requisitos ou requisições de alteração em requisitos já existentes;
- Informações sobre a priorização dos requisitos;
- Controle de versão e controle de alteração de todos os requisitos do software.

De uma perspectiva mais ampla, a organização necessita de várias informações sobre a alocação de suas necessidades em requisitos de software e o andamento dos respectivos projetos. Estas informações são úteis para garantir o alinhamento dos esforços de TI com as estratégias da organização, contribuindo assim para o planejamento e controle dos recursos de TI. Sob esta perspectiva podemos destacar as seguintes informações como conhecimento sobre projetos:

- Projetos em andamento e as necessidades organizacionais sendo tratadas pelos mesmos;
- Produtos de software sendo criados ou modificados por cada projeto;
- *Stakeholders* envolvidos em cada projeto;
- Localidades (sites) envolvidas em cada projeto;
- Indivíduo ou equipe responsável por requisitos em cada projeto;

- Métricas relacionadas aos requisitos.

Já o conhecimento proveniente de projetos inclui as informações geradas pelos processos de ER após a execução dos projetos. Muitas informações evoluem durante a execução destes processos e atingem um estado estável e definitivo somente ao término do projeto, quando os requisitos já estão implementados e o processo de gerência de alterações já está encerrado. Nesta categoria podemos destacar:

- Produtos de software criados ou modificados pelo projeto;
- *Features* implementadas;
- Versão final dos requisitos;
- Informações de rastreabilidade.

A partir desta categorização é possível analisar as implicações das abordagens cliente-servidor e P2P para cada categoria de conhecimento, bem como demonstrar as vantagens da abordagem híbrida.

6. Implicações das Abordagens de Gestão do Conhecimento para Gerência de Requisitos

As abordagens cliente-servidor e P2P apresentam vantagens e desvantagens de acordo com a categoria de conhecimento sendo considerada. Para avaliar qual abordagem oferece mais benefícios em cada categoria de conhecimento os autores da abordagem híbrida analisaram três dimensões da gestão de conhecimento: compartilhamento, controle e estruturação do conhecimento. Aqui, esta mesma análise é realizada para o contexto de ER em DDS.

6.1 Abordagem cliente-servidor

Na abordagem cliente-servidor, o conhecimento é codificado e centralmente disponibilizado, garantindo assim uma estruturação do conhecimento proveniente de múltiplos projetos. Os produtores do conhecimento precisam organizar e classificar o conhecimento de acordo com padrões previamente estabelecidos pela organização. Assim é possível oferecer recursos de busca baseada em filtros que permitem um acesso rápido ao conhecimento desejado. A Figura 1 ilustra esta abordagem.

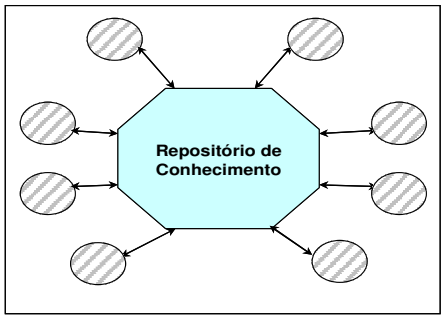

Figura 1. Abordagem cliente-servidor [7]

Do ponto de vista de ER em DDS, esta abordagem traz a vantagem de facilitar a transferência de conhecimento entre equipes geograficamente dispersas, pois garante que um mesmo padrão será utilizado para estruturar o conhecimento sobre requisitos de maneira uniforme por todas as equipes. Entretanto, esta abordagem pode dificultar o compartilhamento e o controle do conhecimento. Uma vez que o conhecimento precisa ser primeiro codificado de acordo com os padrões estabelecidos, os indivíduos tendem a retardar sua publicação até que tenham a confirmação e a estabilização do mesmo. Além disto, os indivíduos terão pouco controle sobre a visibilidade das informações a partir da sua publicação. Assim, uma parte do conhecimento sujeita a modificação mais freqüentes pode não ser disponibilizada com eficiência. Informações a respeito de requisições de alteração de requisitos e seu processo de avaliação, por exemplo, podem ser relegadas deixando-se que apenas as versões finais dos requisitos para publicação.
Desta forma, esta abordagem torna-se adequada para a gestão do conhecimento sobre projetos e proveniente de projetos, contribuindo para a preservação do conhecimento de longo prazo e para o aprendizado organizacional sobre as aplicações legadas e seus requisitos. Mas não é a opção mais adequada para o conhecimento em projetos, devido à natureza dinâmica deste conhecimento e às necessidades de controle e personalização existentes em cada projeto.

6.2 Abordagem P2P

Por outro lado, na abordagem P2P o conhecimento é personalizado. Os indivíduos podem escolher seus próprios meios de categorização e codificação do conhecimento e o compartilhamento deste conhecimento se dá pelo contato direto entre os indivíduos. Neste caso, o papel da tecnologia é oferecer os meios necessários para facilitar a comunicação direta entre os indivíduos. Uma vez que os indivíduos são mais propensos a armazenar nos seus próprios repositórios as informações nas quais ainda estão trabalhando, esta abordagem pode reduzir os atrasos entre a criação do conhecimento e seu armazenamento no repositório. Ao contrario da centralização, onde o conhecimento é separado de seu criador, na abordagem P2P cada indivíduo possui controle sobre o conhecimento e sobre a visibilidade do mesmo. A Figura 2 ilustra esta abordagem.

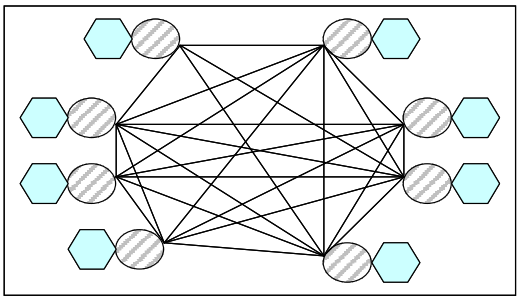

Figura 2. Abordagem P2P [7]

Este tipo de abordagem trás para a ER em DDS a vantagem de facilitar o compartilhamento e o controle do conhecimento de gerência de requisitos. Informações sobre requisições de alteração e sobre o acompanhamento do seu processo de avaliação, por exemplo, são muito dinâmicas, necessitando de constante atualização. Mas não há uma necessidade de transferência imediata deste conhecimento para outras equipes geograficamente dispersas, permitindo que este conhecimento possa ser armazenado localmente e disponibilizado sob demanda apenas para as equipes interessadas. Entretanto, esta abordagem pode dificultar a estruturação do conhecimento. Cada projeto e equipe podem utilizar seus próprios métodos para codificar e categorizar o conhecimento, dificultando assim a transferência deste conhecimento entre equipes geograficamente dispersas. Se cada equipe utilizar seus próprios métodos para armazenar as especificações de requisitos, por exemplo, com o passar do tempo esta documentação se tornará desestruturada.
Assim, esta abordagem torna-se adequada para a gestão do conhecimento em projeto. Proporciona a flexibilidade necessária para que cada equipe e projeto armazene o conhecimento dinâmico sobre a gerência de requisitos utilizando seus próprios processos e padrões. Mas não é a opção mais adequada para gestão do conhecimento sobre projetos e proveniente de projetos, pois pode levar a desestruturação deste tipo de informação.

6.3 Abordagem híbrida

A abordagem híbrida apresenta características que a torna adequada ao contexto da ER em DDS. Para preservar os benefícios de cada abordagem e evitar suas limitações esta abordagem possui dois componentes. O primeiro é um repositório centralizado contendo o conhecimento sobre e proveniente de projetos. Este componente também serve como um índice para o segundo componente: conhecimento em projeto armazenado nos repositórios das equipes geograficamente distribuídas. A Figura 3 ilustra esta abordagem.

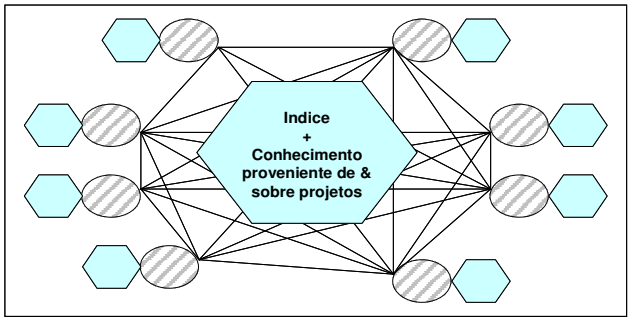

Figura 3. Abordagem Híbrida [7]

O uso de um repositório centralizado para o conhecimento sobre e proveniente de projetos propicia a padronização e estruturação do conhecimento de longo prazo sobre os requisitos das aplicações. Os padrões estabelecidos para a publicação da espe-

cificação de requisitos no repositório centralizado podem garantir que os requisitos sejam documentados de forma homogênea. Como exemplo de possíveis padrões pode-se citar o idioma exigido pela organização para os documentos publicados, a técnica de especificação de requisitos utilizada (casos de uso, linguagem natural, etc.), sistema métrico utilizado e glossário unificado de termos. Além disto, o uso de um repositório centralizado pode viabilizar a especificação dos requisitos com foco no produto de software, evitando que os requisitos sejam documentados para o escopo de cada projeto individual e que informações específicas de um determinado contexto possam prejudicar a interpretação sobre a especificação do produto de software em projetos futuros ou por outras equipes.

Um índice centralizado para o conhecimento em projeto, que por sua vez permanece armazenado nos repositórios distribuídos, oferece um mecanismo eficiente de coordenação e comunicação entre as equipes distribuídas. Por exemplo, uma equipe responsável pela execução de um projeto de manutenção em um determinado produto de software pode localizar a especificação de requisitos que descreve o funcionamento do mesmo em um determinado momento. Mas pode também encontrar um índice para os demais projetos realizados sobre o mesmo produto de software, permitindo interagir diretamente com as equipes responsáveis pelos mesmos e obter informações mais detalhadas de seus repositórios sobre o conhecimento em projeto.

O conhecimento em projeto armazenado nos repositórios distribuídos, segundo componente da abordagem híbrida, proporciona um mecanismo eficiente e efetivo de captura do conhecimento em projeto. Uma vez que cada projeto é único e cada equipe é diferente, esta abordagem permite flexibilidade na adoção de processos de gerência de requisitos e padrões que sejam adequados à realidade de cada caso. Por exemplo, uma equipe em uma determinada localidade pode utilizar um sistema de automação de *workflow* para o acompanhamento das requisições de alteração de requisitos, enquanto outra pode simplesmente utilizar trocas de e-mails entre os *stakeholders* para a mesma finalidade. Também é possível evitar, com esta abordagem, a perda de informações que são relevantes apenas para um determinado projeto, mas que não agregam valor ao aprendizado organizacional ou não se enquadram nos padrões globais da organização. A priorização dos requisitos, por exemplo, é uma informação relevante durante a execução de um projeto, mas perde seu sentido ao termino do mesmo. Portanto, não há necessidade de sobrecarregar o repositório centralizado com informações que podem gerar resultados irrelevantes durante operações de busca por conhecimento.

7. Conclusão

O software é cada vez mais indispensável para a sociedade moderna [11], onde a globalização é uma característica fundamental. Assim como o processo de desenvolvimento de software tem se tornado cada vez mais complexo, a distribuição das equipes no tempo e no espaço tem tornado os projetos distribuídos cada vez mais comuns. Neste contexto, a Engenharia de Requisitos surge como um dos principais desafios no DDS. Muitas são as dificuldades existentes na ER em projetos de DDS. Entre elas, este artigo abordou principalmente a gerência de informações dispersas em múltiplas localidades.

Desta forma, este artigo procurou analisar, do ponto de vista da gestão de conhecimento, que contribuições esta área pode trazer para a engenharia de requisitos, de forma a minimizar os desafios existentes. Para isso, foi proposta uma abordagem de gestão de conhecimento para a engenharia de requisitos derivada da abordagem de [7], que é uma abordagem genérica para projetos distribuídos.

7.1 Trabalhos futuros

Acredita-se que exista uma grande oportunidade para continuar este estudo no sentido de avaliar em projetos reais como esta abordagem se aplicaria. E entende-se que a abordagem de gestão de conhecimento pode contribuir no sentido de melhorar a comunicação no projeto e ajudar os *stakeholders* e a organização a identificar que tipo de conhecimento de requisitos é necessário em que momento e a utilidade desta informação.
Com este propósito, pretende-se realizar um estudo de caso em uma organização que mantém um ambiente DDS. Atualmente, este estudo de caso está em fase de planejamento. A organização e os projetos já foram identificados e as respectivas equipes estão sendo treinadas para a adoção da abordagem descrita neste artigo. Com os resultados deste estudo de caso a abordagem será refinada e um modelo para gestão do conhecimento em gerência de requisitos será elaborado.

References

[1] Prikladnicki, Rafael; Audy, Jorge Luis Nicolas. MuNDDoS: Um Modelo de Referência para Desenvolvimento Distribuído de Software. In: 18 SBES - SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE, 2004, Brasília. 2004.

[2] Evaristo, Roberto; Audy, Jorge Luis Nicolas; Prikladnicki, Rafael; Avritchir, Jairo. Wholly Owned Offshore Subsidiaries for IT Development: A Program of Research. In: Proc. of the 38th Hawaii International Conference on System Sciences - HICSS. Hawaii, USA, 2005.

[3] Evaristo, Roberto; Scudder, Richard. Geographically distributed project teams: a dimensional analysis. In: HICSS, 2000, Havaí. Proceedings… EUA, p. 1-15, 2000.

[4] Lopes, Leandro; Majdenbaum, Azriel; Audy, Jorge Luis Nicolas. Uma proposta para processo de requisitos em ambientes de desenvolvimento distribuido de software. In: WER'03 - Workshop em Engenharia de Requisitos. Piracicaba, SP. RJ: PUCRJ, 2003.

[5] Lopes, Leandro; Audy, Jorge Luis Nicolas. Em busca de um modelo de referencia para engenharia de requisitos em ambiente de desenvovimento distribuido de software. In: WER'03 - Workshop em Engenharia de Requisitos. Piracicaba, SP. RJ: PUCRJ, 2003.

[6] Espindola, Rodrigo dos Santos; Majdenbaum, Azriel; Audy, Jorge Luis Nicolas. Uma Análise Crítica dos Desafios para Engenharia de Requisitos em Manutenção de Software. In: VII Workshop on Requirements Engineering, 2004, Tandil, Argentina, 2004.

[7] Desouza, K. C.; Evaristo, J.R. "Managing Knowledge in Distributed Projects". Comunications of the ACM. Abril 2004, Vol. 47, No.4. pp 87-91.

[8] Thayer, Richard; Dorfman, Merlin. "System and Software Requirements Engineering". IEEE Computer Society Press Tutorial. 2000. 718p

[9] Sommerville, Ian; Sawyer, Peter. “Requirements Engineering – a good practice guide”. New York: John Wiley & Sons Ltd, 1997, 391p.

[10] Carmel, E.. Global Software Teams – Collaborating Across Borders and Time Zones. Prentice Hall. 1999. 269p.

[11] Karolak, D.. Global Software Development – Managing Virtual Teams and Environments. IEEE Computer Society. Los Alamitos, EUA. 1998. 159p.

[12] Kiel, L. Experiences in Distributed Development: A Case Study, In. Workshop on Global Software Development at ICSE, Oregon, EUA. Proceedings… 2003.

[13] Zowghi, D., “Does Global Software Development need a different requirements engineering process?”, Proceedings of International Workshop on Global Software Development, 2002.

[14] Damian, D., Zowghi, D., “An insight into the interplay between culture, conflict and distance in globally distributed requirements negotiations”, Proceedings of the 36th Hawaii International Conference on Systems Sciences (HICSS’03), IEEE, 2002.

[15] IEEE, 1998. IEEE Recommended Practice for Software Requirements Specification, Std 830-1998. IEEE Computer Society. New York, USA

[16] Kotonya, G.; Sommerville, I. “Requirements Engineering: process and techniques”. New York: John Wiley & Sons Ltd, 1998, 282p.

[17] Kruchten, P., The Rational Unified Process: An Introdution. 2nd Edition. Addison Weasley. 2001.

[18] Hansen, M.T., Nohira, N., Tierney, T. What’s your strategy for managing knowledge? Harvard Business Review 77, 2 (Mar./Apr. 1999), pp 106-116.

[19] Liu, Kecheng; Alderson, Albert; Qureshi, Zubair. “Requirements Recovery from Legacy Systems by Analysing and Modelling Behaviour”. Proceedings of the International Conference on Software Maintenance, 1999, IEEE Computer Society, Los Alamitos, pp3-12.

[20] Ebner, G., Kaindl, H., “Tracing All Around in Reengineering”, IEEE Software, May 2002, pp.70-76.

[21] Pressman, R. S. Software Engineering: a practitioner’s approach. McGraw Hill, New York, 5th ed., 2001.

[22] White, Stephanie M. “Capturing Requirements for Legacy Systems”. Proceedings of the International Symposium and Workshop on Systems Engineering of Computer Based Systems, 1995. pp 251-256.

[23] Zanlorenci, E. P., Burnett, R. C. “Abordagem de Engenharia de Requisitos em Software Legado”. Workshop em Engenharia de Requisitos, Piracicaba-SP, Brasil, 2003, pp 270-284.

[24] Damm, D., Schindler, M., Security issues of a knowledge médium for distributed projects work. Intern. J. of Project Management 20. 2002. pp 37-44.