Apoio de Gerência de Conhecimento na Engenharia de Requisitos

Julio Cesar Nardi[1], Ricardo de Almeida Falbo[2]

[1] Centro Federal de Educação Tecnológica do Espírito Santo (CEFET-ES) – UnED/Colatina. Av. Arino Gomes Leal, 1700, Bairro Santa Margarida–29.700-603– Colatina – ES – Brasil

[2] Departamento de Informática - Universidade Federal do Espírito Santo (UFES) Av. Fernando Ferrari s/n, Campus de Goiabeiras–29.060-900– Vitória – ES – Brasil

julionardi@yahoo.com.br, falbo@inf.ufes.br

Abstract. *This paper presents a Knowledge Management approach to support the Requirement Engineering (RE) process in ODE, an Ontology-based software Development Environment. Ontologies and analysis patterns are considered knowledge items, besides learned lessons, message packages and artifacts generated by ODE´s tools. Thus, developers and requirements engineers can reuse those items during the RE process.*

Resumo. *Este artigo apresenta uma abordagem de Gerência de Conhecimento para suportar a Engenharia de Requisitos (ER) em ODE, um ambiente de desenvolvimento de software baseado em ontologia. Ontologias e padrões de análise são considerados itens de conhecimento, juntamente com lições aprendidas, pacotes de mensagens e artefatos gerados pelas ferramentas de ODE. Assim, desenvolvedores e engenheiros de requisitos podem reutilizar estes itens durante o processo de ER.*

1. Introdução

A busca por um desenvolvimento de software que conduza a produtos de qualidade tem levado as organizações de software a aprimorarem seus processos. Neste contexto, uma área de crescente importância é a Engenharia de Requisitos (ER), a qual contempla um processo que é apontado como de extrema importância para o sucesso de um projeto [Hofmann et al. 2001]. A ER aborda aspectos relacionados ao desenvolvimento e à gerência dos requisitos, e envolve atividades relacionadas ao levantamento, análise, modelagem, negociação, documentação, verificação, validação e controle de mudanças em requisitos.

É consenso que se os requisitos capturados nas fases iniciais do desenvolvimento forem completos, claros, consistentes e rastreáveis, dentre outras características de qualidade de requisitos [Wiegers 2003], a busca por qualidade nas atividades subseqüentes do processo de software será favorecida.

Uma vez que a qualidade dos requisitos é determinante para o sucesso de um projeto, é importante apoiar a reutilização neste contexto, pois ela tem sido apontada como um fator chave para o aumento da qualidade e da produtividade no

desenvolvimento de software. Com a reutilização, o esforço de desenvolvimento do software tende a diminuir e a qualidade aumentar, pois os itens reutilizados já devem ter passado por verificações e validações em outros projetos [Gimenes et al. 2005].

No contexto da ER, a reutilização pode ser útil, sobretudo na reutilização de requisitos de sistemas similares e de modelos (com destaque para os modelos conceituais) [Falbo et al. 2007]. Entretanto, outros itens de conhecimento, tais como lições aprendidas e melhores práticas, úteis a quaisquer atividades do processo de software, podem ser igualmente relevantes. Assim, é importante gerenciar conhecimento relativo à ER.

A importância de aspectos relacionados à reutilização e à gerência do conhecimento é tal que diversos modelos e normas de qualidade de processo incorporam processos relativos a esses tópicos. A ISO/IEC 12207 em sua Emenda 1 [ISO 2002] passou a contemplar uma atividade de Gestão do Conhecimento como parte de seu Processo de Recursos Humanos. Além disso, ela possui dois processos dedicados à reutilização, o Processo de Gestão de Programa de Reúso e o Processo de Engenharia de Domínio. O modelo de Melhoria de Processo do Software Brasileiro (MPS.BR) [Softex 2006] definiu para 2007 a inclusão em seus guias dos processos de Gerência de Reuso e de Gerência de Conhecimento (GC) [Softex 2007]

No entanto, um problema recorrente na área de estudo da reutilização se mostra um obstáculo: a seleção de itens para reuso. Para ilustrar esse problema no contexto da ER, pode-se citar o trabalho descrito em [Falbo et al. 2007], no qual um processo de ER baseado em reutilização de ontologias e padrões de análise foi aplicado a três projetos tratando da alocação de recursos humanos a projetos em organizações de software. Nesse estudo, concluiu-se que a reutilização realmente trouxe benefícios no processo de ER, mas a seleção de modelos para reuso foi apontada como um fator decisivo para o sucesso.

Uma maneira de tratar esse problema consiste na aplicação de técnicas de gerência de conhecimento para apoiar a ER. Para tal, é importante integrar essas facilidades nas próprias ferramentas de apoio à ER. De fato, idealmente, as ferramentas de apoio à ER devem estar integradas a outras ferramentas de apoio ao processo de software, uma vez que requisitos são trabalhados em todas as fases do desenvolvimento. Uma forma de se obter essa integração é trabalhar com Ambientes de Desenvolvimento de Software (ADSs). Um ADS pode ser visto como um conjunto de ferramentas e facilidades integradas capazes de apoiar todas as atividades do processo de software, ou pelo menos porções significativas dele [Harrison et al. 2000].

A relevância do processo de ER na produção de software de qualidade, a possibilidade de reutilização de itens de conhecimento ao longo desse processo e a identificação da necessidade de se ter um ferramental integrado de apoio, tanto à execução das atividades da ER quanto à reutilização no âmbito dessas atividades, determinaram os objetivos deste trabalho no sentido de estender a infra-estrutura de Gerência de Conhecimento (GC) do ambiente ODE (*Ontology-based Software Development Environment*) [Falbo et al. 2003], para apoiar a ER. Por meio dos novos serviços disponibilizados por essa infra-estrutura, é possível reutilizar diversos itens de conhecimento produzidos pelas ferramentas de apoio à ER de ODE, tais como lições aprendidas, modelos de análise, padrões de análise, ontologias, requisitos etc.

Este artigo está organizado da seguinte forma: a seção 2 aborda os temas Engenharia de Requisitos e Gerência de Conhecimento, procurando dar destaque a itens de conhecimento importantes para a ER, como ontologias e padrões de análise; a seção 3 apresenta o ambiente ODE, sua infra-estrutura de Gerência de Conhecimento, sua infra-estrutura de caracterização de itens de software e suas ferramentas de apoio à ER; a seção 4 trata do apoio da GC na ER no contexto de ODE, abordando a reutilização de itens de conhecimento úteis para a ER, tais como ontologias, modelos de análise, padrões de análise e requisitos; a seção 5 discute trabalhos relacionados; e finalmente, na seção 6, apresentam-se algumas considerações finais.

2. Engenharia de Requisitos e Gerência de Conhecimento

Segundo Zave (1997), a Engenharia de Requisitos (ER) se preocupa com as metas do mundo real que motivam o desenvolvimento de funções e a descoberta de restrições em sistemas de software. Também se preocupa com o relacionamento desses fatores com as especificações precisas do comportamento do software e com sua evolução. Assim, as funções e restrições descobertas, e que serão desenvolvidas e gerenciadas, são denominadas requisitos, os quais, em primeira instância, podem ser classificados como funcionais e não-funcionais.

No que diz respeito às normas e modelos de qualidade, o trato com requisitos é amplamente abordado, dada sua importância. O CMMI (*Capability Maturity Model Integration*) [Chrissis et al. 2003], em seus cinco níveis de maturidade, trata de requisitos, basicamente, no nível 2, no qual aborda a Gerência de Requisitos, e no nível 3, no qual trata do Desenvolvimento de Requisitos. Na Gerência de Requisitos objetiva-se garantir consistência dos requisitos e consistência entre os requisitos e os vários produtos gerados ao longo do processo de software. Já o Desenvolvimento de Requisitos visa a produzir e analisar os requisitos de clientes bem como os requisitos de produto.

O MPS.BR [Softex 2006], organizado em sete níveis de maturidade (de A a G), define no nível G o processo de Gerência de Requisitos e no nível D o processo de Desenvolvimento de Requisitos. Tais processos têm propósitos semelhantes aos homônimos do CMMI.

A ISO/IEC 12207, mais especificamente sua Emenda 1 [ISO 2002], destaca em seu Processo de Desenvolvimento três sub-processos: Levantamento de Requisitos, Análise dos Requisitos do Sistema e Análise dos Requisitos do Software. O propósito do Levantamento de Requisitos é obter, processar e acompanhar as necessidades e os requisitos do cliente ao longo da vida do produto e/ou serviço. A Análise dos Requisitos do Sistema tem como propósito transformar os requisitos dos envolvidos em um conjunto de requisitos técnicos desejados para o sistema, que guiarão o seu projeto (*design*). Por fim, a Análise dos Requisitos do Software visa a estabelecer os requisitos dos elementos de software do sistema.

Cada um destes processos apresentados pelos modelos e normas de qualidade e que estão contextualizados na ER concentra uma complexidade considerável, de modo que a busca por facilitadores para esse processo tem sido foco de pesquisas. Nesse contexto, a Gerência de Conhecimento se mostra uma opção interessante.

A Gerência de Conhecimento (GC) é uma área de pesquisa inerentemente interdisciplinar e envolve gerência de recursos humanos, cultura organizacional, métodos e ferramentas de tecnologia da informação [O´Leary et al. 2001]. Ela fornece as bases para a construção de repositórios de conhecimento organizacionais, bem como para o provimento de serviços de apoio à captura, disseminação, uso e evolução do conhecimento, de modo que o conhecimento gerado na organização seja armazenado de uma forma que facilite a sua recuperação e utilização.

A GC pode fornecer soluções para necessidades de vários negócios, tais como a diminuição de tempo e custos, aumento da qualidade e melhoria na tomada de decisão, uma vez que propicia a troca de experiências pessoais, fornecendo mais conhecimento aos membros da organização [Rus et al. 2002]. No contexto do desenvolvimento de software, a GC pode facilitar a aquisição de conhecimento sobre novas tecnologias, o acesso ao conhecimento de domínio, a troca de conhecimento acerca de práticas e políticas locais, a captura do conhecimento de quem sabe o quê e a colaboração e troca de conhecimento [Rus et al. 2002].

Contudo, há de se destacar que, para ser efetiva, a GC tem de estar integrada ao processo de negócio ao qual ela fornece apoio [Staab et al. 2001]. Considerando que o foco deste trabalho é o processo de ER, é importante integrar a GC a esse processo. Tendo isso em mente, dois aspectos têm de ser mais detalhadamente analisados: (i) a escolha de itens de conhecimento relevantes para a ER e (ii) a forma como os serviços de GC devem ser integrados à ER para que sejam facilmente utilizados.

No que se refere aos itens de conhecimento relevantes para a ER, é importante destacar dois grupos: itens gerais, úteis para quaisquer atividades do processo de software, tais como lições aprendidas e melhores práticas, e itens específicos para a ER, tais como requisitos, modelos etc. Nossa atenção neste artigo recai sobre o último grupo e, portanto, é importante discutir duas das áreas onde estão concentrados os principais esforços relativos à reutilização na ER: Padrões de Análise [Robertson et al. 1999] [Fowler 1997] e Engenharia de Domínio [Gimenes et al. 2005].

Padrões de software podem ser vistos como experiências reutilizáveis, uma vez que documentam problemas contextualizados, recorrentes e comuns, que desenvolvedores de software têm enfrentado e encontrado soluções bem sucedidas [Gimenes et al. 2005]. Um padrão de análise é um padrão de software que reúne um grupo de conceitos que representa uma construção comum na modelagem de negócio [Fowler 1997]. Padrões de análise são geralmente definidos a partir de modelos conceituais de aplicações, sendo descobertos e não inventados [Fowler 1997]. Ou seja, a partir da análise de diversos eventos de negócio do mesmo tipo, um padrão de análise pode ser derivado por meio da abstração de elementos comuns [Robertson et al. 1999].

A Engenharia de Domínio é o processo voltado para a produção de componentes reutilizáveis, englobando atividades de Análise, Projeto e Implementação de Domínio, as quais objetivam, respectivamente, representar requisitos comuns de uma família de aplicações por meio de modelos de domínio, disponibilizar modelos arquiteturais para aplicações a partir de um único modelo de domínio e disponibilizar implementações de componentes que representam funcionalidades básicas de aplicações relacionadas a um domínio [Gimenes et al. 2005].

Claramente a Análise de Domínio é a atividade diretamente ligada à reutilização na ER [Falbo et al. 2007]. No que concerne a itens de conhecimento a serem reutilizados, os seus produtos, ou seja, modelos de domínio, são os objetos de interesse. Atualmente, ontologias têm ganhado destaque como modelos de domínio.

Ontologias, na corrente discussão da Ciência da Computação, são recursos baseados em computador que representam semânticas compartilhadas sobre domínios [Spyns et al. 2002]. Essa definição é importante no contexto deste trabalho por três razões: (i) contextualiza ontologias na área da Ciência da Computação; (ii) considera ontologias como recursos baseados em computador; e (iii) destaca o fato de ontologias tratarem semânticas acordadas sobre domínios, o que reforça a idéia de utilização de ontologias para a captura de conhecimento compartilhado.

Utilizar ontologias e padrões de análise para a modelagem conceitual de sistemas abre espaço para uma visão mais abrangente do domínio tratado, levando a um melhor entendimento do mesmo. Isso pode diminuir o tempo gasto na especificação de requisitos, pois os analistas já têm uma fonte preliminar para aprender sobre o domínio, que estabelece uma terminologia comum aos especialistas do negócio [Cota et al. 2004].

Padrões de Análise, assim como ontologias, descrevem aspectos no nível de conhecimento. Assim, uma vez que eles provêem conhecimento sobre soluções bem sucedidas a problemas recorrentes no desenvolvimento de software [Devedzic 1999], eles favorecem a reutilização [Cota et al. 2004]. No entanto, ontologias capturam a estrutura conceitual intrínseca de um domínio, enquanto padrões de análise focalizam a estrutura de uma aplicação [Devedzic 1999].

Finalmente, em relação à forma como os serviços de GC devem ser integrados à ER, essa questão passa pela automatização do processo da ER. Para ser efetivamente utilizado, um sistema de GC deve estar integrado ao ambiente de trabalho existente. No caso do desenvolvimento de software, esse ambiente de trabalho pode ser um Ambiente de Desenvolvimento de Software (ADS) [Falbo et al. 2004] e, portanto, é importante que as facilidades de GC estejam disponíveis no próprio ambiente, mais especificamente nas ferramentas de apoio à ER, dado o foco deste trabalho.

Assim, visando a facilitar a institucionalização da reutilização no processo da ER, neste trabalho foram introduzidos serviços de GC nas ferramentas de apoio à ER do ADS ODE [Falbo et al. 2003], com destaque para o tratamento de ontologias e padrões de análise como itens de conhecimento.

3. O Ambiente ODE

O ambiente ODE (*Ontology-based Software Development Environment*) [Falbo et al. 2003] é um Ambiente de Desenvolvimento de Software (ADS) centrado em processo e baseado em ontologias. A premissa do Projeto ODE é a seguinte: se as ferramentas do ADS são construídas baseadas em ontologias, a integração das mesmas é facilitada, pois os conceitos envolvidos estão bem definidos e compartilhados. Tal ambiente vem sendo desenvolvido no Laboratório de Engenharia de Software da Universidade Federal do Espírito Santo (LabES-UFES) desde 1999.

Constituindo a base ontológica de ODE, há várias ontologias, dentre elas ontologias de Processo de Software, de Artefato de Software, de Gerência de

Configuração de Software, de Qualidade de Software, de Organizações de Software e de Requisitos de Software. Essas ontologias fornecem uma conceituação sólida para a construção de diversas ferramentas do ambiente, incluindo a sua infra-estrutura de Gerência de Conhecimento [Falbo et al. 2004] e a infra-estrutura de caracterização de itens de software [Carvalho et al. 2006]. Em especial, merece destaque no contexto deste trabalho a Ontologia de Requisitos de Software [Nardi et al. 2006], usada como base para a construção da ferramenta de apoio à Engenharia de Requisitos - ReqODE [Martins et al. 2006] - e como forma de estabelecer uma conceituação comum acerca dos elementos relacionados à ER, de modo que a integração entre as ferramentas de ODE e os serviços de gerência de conhecimento possa ser estabelecida mais facilmente.

Além de ReqODE, no apoio ao processo de ER, o ambiente ODE conta, ainda, com uma ferramenta de apoio à modelagem UML, denominada OODE. Por fim, há diversas outras ferramentas que, de alguma forma, interagem com as ferramentas de apoio à ER, dentre elas, ferramenta de definição do escopo de um projeto e decomposição do produto em módulos, ferramenta de definição de processos de software, ferramenta de estimativas baseadas em pontos de função e pontos de casos de uso, ferramenta de alocação de recursos, ferramenta de gerência de riscos etc. A seguir, são brevemente apresentadas as ferramentas de apoio à ER (ReqODE e OODE), a infra-estrutura de gerência de conhecimento de ODE e a infra-estrutura de caracterização de itens de conhecimento.

3.1. As Ferramentas de Apoio à ER de ODE

OODE é a ferramenta de modelagem UML de ODE. Como qualquer ferramenta de modelagem UML, OODE possui funcionalidades para a elaboração dos principais diagramas da UML, dentre eles diagramas de casos de uso, de classes e de estados, agrupados em um modelo de objetos. OODE permite, ainda, a criação de ontologias e padrões de análise, ambos usando como notação a UML.

ReqODE [Martins et al. 2006] é a principal ferramenta de apoio ao processo ER de ODE, permitindo tratar requisitos desde o levantamento, quando são capturados, até a gerência de requisitos. Nesta ferramenta são registradas diversas informações relacionadas a um requisito, dentre elas: um identificador único, uma sentença descrevendo sucintamente do que trata o requisito, uma descrição detalhada, tipo, prioridade, estado e razões de criação do requisito, requisitos que compõem um determinado requisito, requisitos que dependem dele e requisitos conflitantes, módulo do projeto ao qual o requisito foi alocado, interessados e responsáveis, casos de uso derivados do requisito e classes que o implementam e artefatos que foram gerados durante o desenvolvimento tendo como base o requisito, entre outros. Essas informações são usadas para prover diversos tipos de relatórios de rastreabilidade.

3.2. Infra-Estrutura de GC de ODE

A infra-estrutura de Gerência de Conhecimento (GC) de ODE [Falbo et al. 2004] conta com uma memória organizacional, estruturada em repositórios de conhecimento, e cinco tipos básicos de serviços.

Os repositórios de conhecimento da memória organizacional de ODE contêm itens de conhecimento, que podem ser formais ou informais, como mostra a Figura 1.

Os itens de conhecimento formais são os diversos artefatos produzidos pelas ferramentas do ambiente. Já os informais incluem lições aprendidas e pacotes de mensagens [Falbo et al. 2004].

Os tipos de serviços oferecidos pela infra-estrutura são [Falbo et al. 2004]:

- captura e criação: artefatos produzidos pelas ferramentas do ambiente, quando colocados sob gerência de configuração, são disponibilizados também como itens de conhecimento da infra-estrutura de GC de ODE [Nunes et al. 2006]. Para tratar lições aprendidas, há um serviço específico que permite o registro, aprovação e disponibilização. Por fim, há um serviço de empacotamento de mensagens trocadas por membros da organização;
- recuperação e acesso: serviço geral de busca de itens de conhecimento, com iniciativa por parte dos usuários do ambiente;

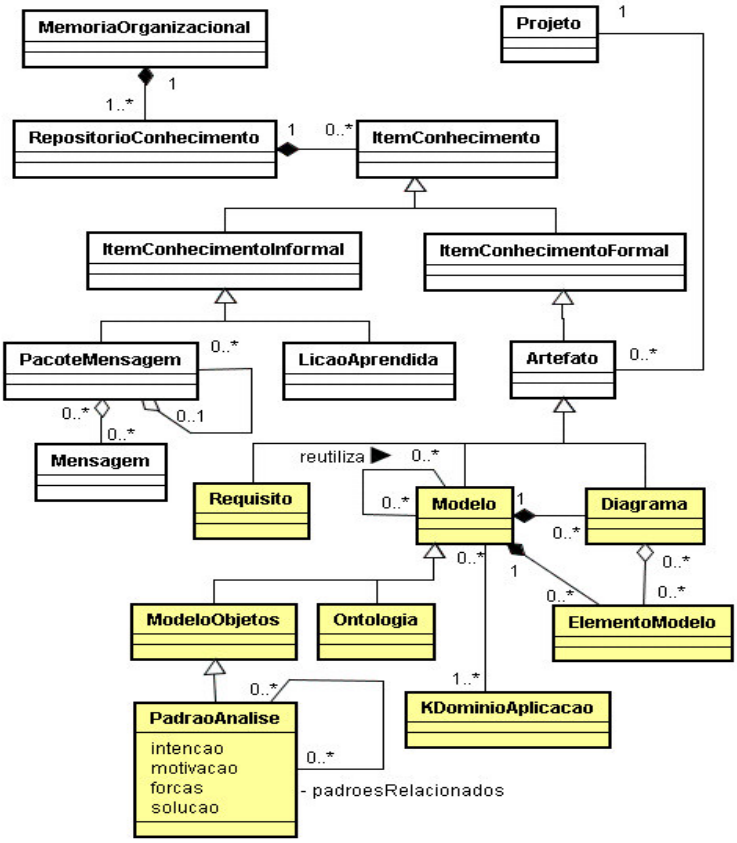

Figura 1. Diagrama de classes parcial da Infra-Estrutura de GC de ODE

- disseminação pró-ativa: serviço análogo ao anterior, mas iniciado pelo sistema, fornecendo, com base nas necessidades dos desenvolvedores, itens relevantes para a execução de uma determinada atividade. Uma vez que depende dessa atividade, esse serviço não é geral, precisando ser especializado pelas ferramentas que o implementam;
- feedback da relevância: durante o uso de um item de conhecimento, o desenvolvedor pode dar informações acerca da eficácia do mesmo em uma dada situação; e

- manutenção: com base no uso e no feedback provido pelos desenvolvedores, permite a exclusão de itens da memória organizacional.

3.3. Infra-Estrutura de Caracterização de Itens de Software em ODE

A fim de facilitar a recuperação de itens de software relevantes para uma dada situação, ODE conta com uma infra-estrutura de caracterização [Carvalho et al. 2006], que utiliza técnicas de Raciocínio Baseado em Casos para recuperação de itens similares.

Por meio desta infra-estrutura, itens de software como, por exemplo, projetos e módulos podem ser caracterizados. Assim, um item passa a ter uma *Caracterização* que contempla *Características* que determinarão um item. Tais características podem ser [Carvalho et al. 2006]:

- *CaracteristicaValorOrdenado*: característica cujos possíveis valores têm uma ordem intrínseca estabelecida entre eles. Ex: complexidade de projeto (alta, média ou baixa);
- *CaracteristicaValorNaoOrdenado*: característica cujos possíveis valores não têm uma ordem pré-estabelecida entre eles. Ex: forma de desenvolvimento (desenvolvimento interno, desenvolvimento terceirizado etc);
- *CaracteristicaConhecimento*: característica que têm como possíveis valores objetos que são instâncias das ontologias de ODE. Ex: paradigma (orientado a objetos, estruturado etc).[1]

No caso das características de valor não-ordenado e das características de conhecimento, por exemplo, utilizam-se, para se calcular a similaridade entre objetos caracterizados, tabelas de similaridade que relacionam valores às características. Desta forma, com estas tabelas, pode-se definir valores de similaridade entre, por exemplo, paradigmas de desenvolvimento, que são características de projeto. A partir do cruzamento destes valores chega-se a um número que sugere a similaridade entre projetos no que tange ao paradigma de desenvolvimento. Isto pode ser combinado com as similaridades de outras características para estabelecer uma similaridade global entre projetos [Carvalho et al. 2006].

É importante destacar também que, dependendo da atividade executada no processo de desenvolvimento, o cálculo da similaridade pode ser computado de maneira diferente. Por exemplo, considerando a atividade de Análise de Requisitos, pode-se estar interessado em recuperar modelos de requisitos e, para tal, o nível de experiência da equipe com o processo de software adotado tem menor impacto do que, por exemplo, o tipo de paradigma utilizado no processo, uma vez que diferentes paradigmas podem utilizar representações distintas para seus modelos. Desta forma, esta infra-estrutura foi utilizada a fim de recuperar itens de conhecimento relevantes, em especial, para que projetos similares pudessem ser identificados e, a partir deles, se pudesse selecionar, por exemplo, requisitos e modelos para reuso.

[1] Os possíveis valores de paradigma são conhecimentos armazenados em ODE que, por definição, são instâncias do conceito *Paradigma* da Ontologia de Processo.

4. Gerência de Conhecimento na Engenharia de Requisitos

Um dos interesses de se utilizar a Gerência de Conhecimento na Engenharia de Software em geral, e mais especificamente na Engenharia de Requisitos, é decorrente da constante expectativa de aumento da qualidade e produtividade, o que pode ser obtido por meio da reutilização. O desenvolvimento com reutilização considera que esforços e experiências que foram adquiridos em projetos anteriores podem ser uma rica fonte de informação para novos projetos. Neste sentido, buscou-se desenvolver uma abordagem na qual itens de conhecimento produzidos pelas ferramentas do ambiente, úteis para a ER, pudessem ser gerenciados visando à reutilização.

Antes do desenvolvimento deste trabalho, eram considerados itens de conhecimento em ODE lições aprendidas, pacotes de mensagens e artefatos em geral, sem um tratamento específico para os artefatos da ER. Dessa maneira, a infra-estrutura de GC de ODE foi estendida para dar um tratamento diferenciado aos artefatos relevantes para a ER. A Figura 1 mostra os principais elementos trabalhados, a saber: requisitos, diagramas e modelos, incluindo modelos de objetos, padrões de análise e ontologias.

Requisitos são capturados na ferramenta ReqODE e, assim como os demais artefatos de ODE, são disponibilizados pela infra-estrutura de GC quando submetidos à gerência de configuração. Graças à integração de ferramentas em ODE, é possível, a partir de ReqODE, utilizar a infra-estrutura de caracterização e cálculo de similaridade entre projetos de ODE. Assim, podem ser solicitados projetos similares ao projeto corrente e, a partir deles, requisitos podem ser reutilizados, como ilustra a Figura 2.

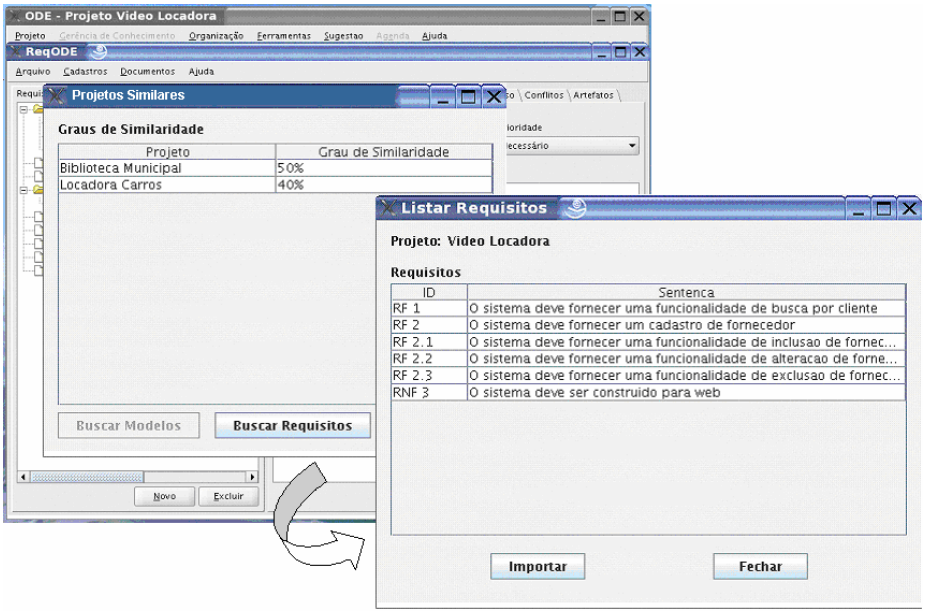

Figura 2. Reutilização de requisitos de projetos similares em ReqODE

Para modelos em geral, definiu-se um auto-relacionamento (associação *reutiliza* na Figura 1) para registrar as origens de um modelo, ou seja, a partir de quais outros modelos um determinado modelo foi criado. Assim é possível saber que modelos serviram de base para a elaboração de outros, facilitando, inclusive, a posterior

identificação de padrões de análise pela constatação de que determinada solução é recorrentemente aplicada.

Por meio de OODE, engenheiros de requisitos podem construir modelos de objetos e diagramas para projetos específicos. Durante a construção desses modelos é possível reutilizar modelos de objetos e diagramas de projetos similares, ontologias e padrões de análise, todos construídos nesta mesma ferramenta e que estão disponibilizados como itens de conhecimento formal na memória organizacional da infra-estrutura de GC do ambiente. Assim, o engenheiro de requisitos pode reutilizar modelos em diferentes níveis de abstração, desde ontologias, passando por padrões de análise e chegando a modelos de objetos. Vale ressaltar que, do mesmo modo que requisitos e demais artefatos de ODE, modelos de objetos e diagramas só são disponibilizados pela infra-estrutura de GC quando submetidos à gerência de configuração.

A reutilização de modelos de objetos e diagramas de projetos similares se dá de forma semelhante à reutilização de requisitos. Inicialmente, recuperam-se os projetos similares - por meio da infra-estrutura de caracterização - e, a partir da escolha de um desses projetos, são listados os modelos de objetos correspondentes. De posse desses modelos, o desenvolvedor pode, por exemplo, solicitar a importação de um deles para o projeto que está sendo trabalhado. Assim, todos os diagramas e elementos de modelo do modelo importado podem ser reutilizados, como ilustra a Figura 3.

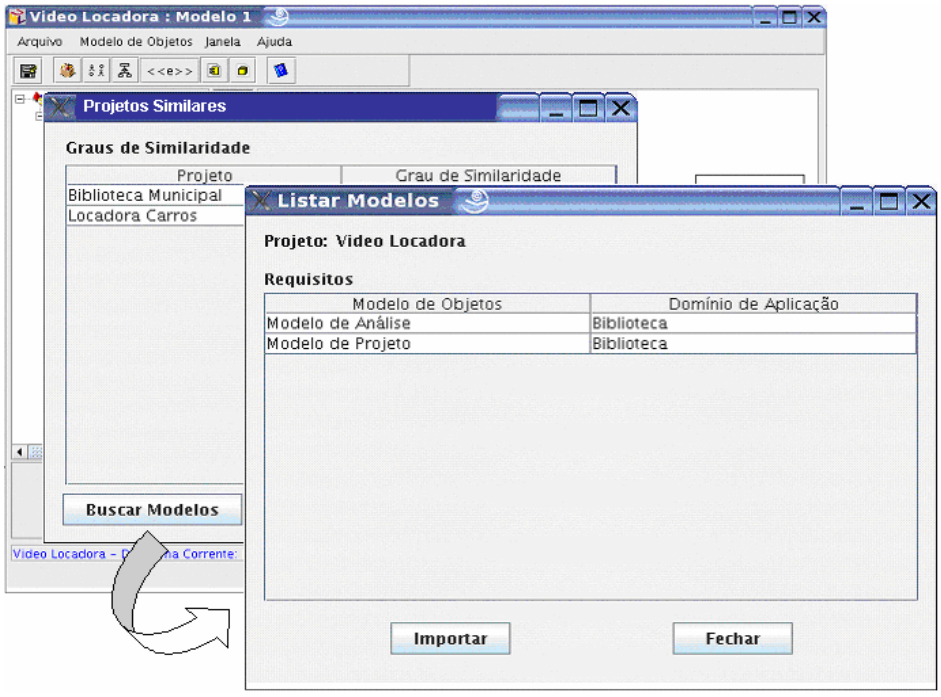

Figura 3. Reutilização de modelos de objetos de projetos similares em OODE

Uma vez que padrões de análise e ontologias descrevem aspectos em um nível mais alto de abstração, foi necessária uma abordagem de reutilização ligeiramente diferente. Primeiro, tais modelos não são produzidos no contexto de projetos. Assim, a ferramenta OODE passou a ser disponibilizada para engenheiros de conhecimento como parte das ferramentas de registro de conhecimento de ODE (serviço de captura e

criação), sendo que, neste contexto, apenas ontologias e padrões de análise podem ser elaborados. Segundo, para apoiar a identificação de similaridade neste caso, modelos passaram a ser caracterizados pelos domínios de aplicação relacionados, de modo a especializar a busca (serviço de recuperação e acesso). Assim, esses modelos podem ser recuperados, por exemplo, informando-se o domínio de aplicação e, a partir daí, ter toda sua estrutura importada para facilitar a construção de modelos de objetos, como ilustra a Figura 4.

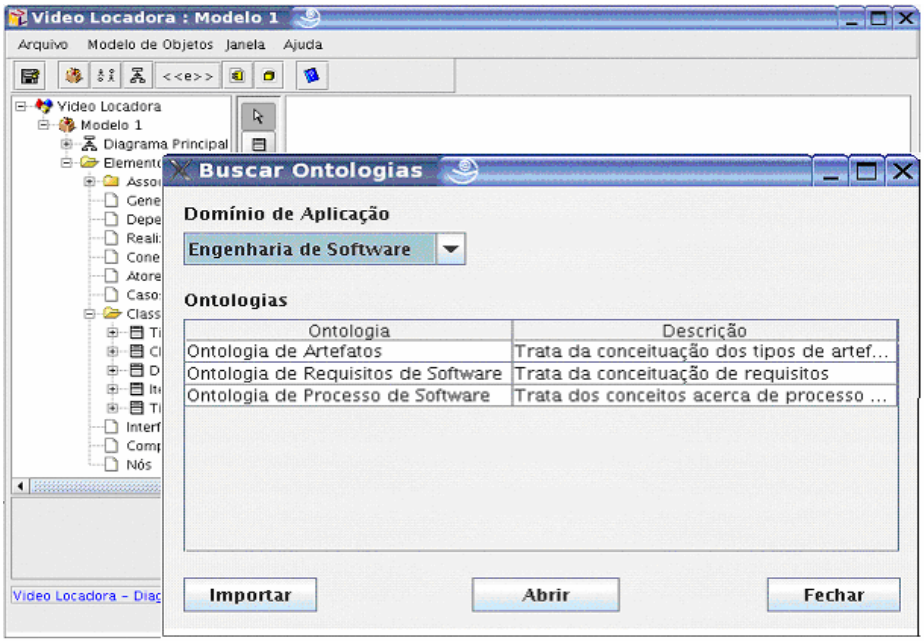

Figura 4. Reutilização de ontologias em OODE

Conforme citado anteriormente, para que seja possível reutilizar ontologias e padrões de análise, estes devem ter sido previamente criados e disponibilizados no ambiente. Este trabalho é desempenhado pelo engenheiro de conhecimento, que pode utilizar OODE para construir ontologias e padrões de análise, além de ter a incumbência de manter esses itens de conhecimento atualizados e consistentes. A Figura 5 mostra a ferramenta OODE sendo usada para a criação de um padrão de análise.

Ao engenheiro de conhecimento é também dada a possibilidade de reutilizar ontologias, padrões de análise e modelos de objetos. Assim, ele pode reutilizar ontologias e padrões de análise para construir novos modelos na forma de outras ontologias e outros padrões de análise. Além disso, o engenheiro de conhecimento tem a possibilidade de reutilizar modelos de objetos em busca de soluções de sucesso para problemas recorrentes e, a partir daí, construir novos padrões de análise.

Por fim, vale lembrar que continua sendo possível a todos os usuários de ODE, incluindo aí engenheiros de requisitos e de conhecimento, reutilizar lições aprendidas, pacotes de mensagens e outros artefatos diversos produzidos pelas ferramentas existentes.

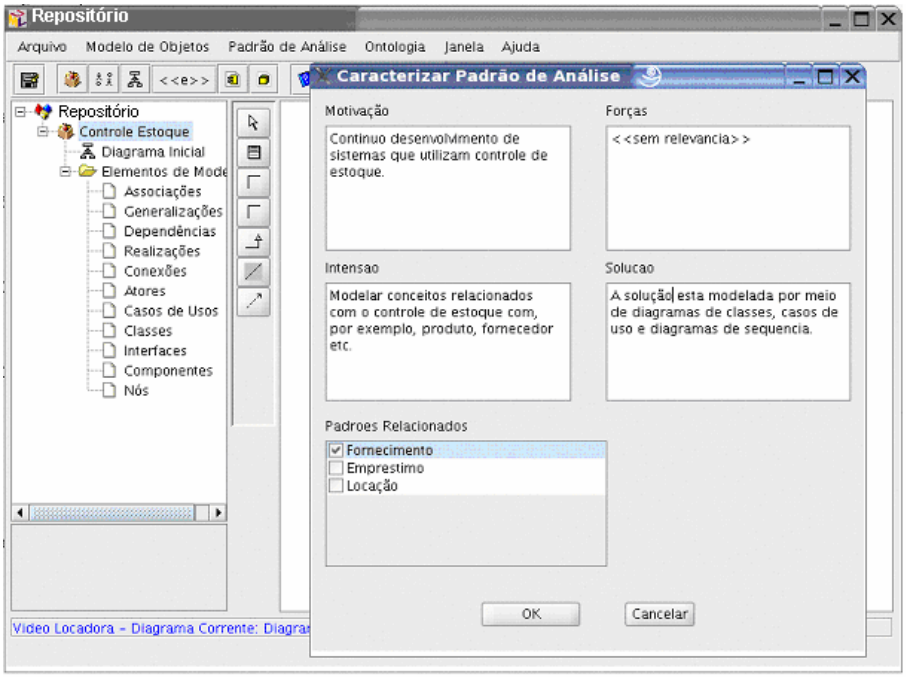

Figura 5. Elaborando um padrão de análise em OODE

5. Trabalhos Correlatos

Diversos trabalhos exploram o uso de Gerência de Conhecimento (GC) para apoiar atividades da Engenharia de Software, tal como [Rus et al. 2002]. Em menor proporção, são encontrados trabalhos que integram GC em Ambientes de Desenvolvimento de Software (ADSs), tais como [Santos et al. 2004] e [Holz 2003]. O primeiro explora o conceito de ADSs Orientados à Organização para introduzir gerência do conhecimento organizacional na Estação TABA. O segundo trata da GC no ambiente MILOS. Contudo, em ambos os casos, não foram encontradas referências explícitas ao apoio à Engenharia de Requisitos (ER).

Togneri et al. (2004) discutem diversos requisitos de um sistema de GC para apoiar a ER e apresentam CRETA, uma ferramenta de apoio à ER cooperativa. Em CRETA há serviços baseados em GC de apoio à ER, mas focados basicamente na definição de um processo de ER e no conhecimento a respeito de padrões de documentação e listas de verificação, em contraste com o apoio oferecido em ODE, que está centrado no reúso de requisitos e modelos.

Gomes et al. (2004) apresentam uma abordagem de Raciocínio Baseado em Casos para Gerência de Conhecimento no contexto da fase de projeto, usando a ferramenta REBUILDER. Os casos, armazenados em uma biblioteca, são representados por meio de diagramas de classes UML. Distintamente deste trabalho, a abordagem de REBUILDER foca a gerência do conhecimento no âmbito da atividade de projeto. Para se reutilizar diagramas, os mesmos são recuperados por analogia (de modo a completar um determinado diagrama de classes com base em outros) e por composição (gerando um novo diagrama de classes por meio da integração de casos existentes na biblioteca). A fim de tornar a identificação de casos mais eficiente, REBUILDER utiliza *WordNet* [Miller et al. 1990] como ontologia de senso comum.

6. Considerações Finais

Este artigo apresentou uma extensão da infra-estrutura de Gerência de Conhecimento (GC) de ODE para apoiar a condução das atividades do processo de Engenharia de Requisitos (ER), por meio da reutilização de itens de conhecimento relevantes, tais como requisitos, modelos de objetos, ontologias e padrões de análise. Em [Falbo et al. 2007], um estudo conduzido apontou a importância da reutilização desses itens, sobretudo ontologias, para se obter ganhos de qualidade e produtividade. Contudo, igualmente apontou o problema da seleção de itens para reuso como crucial para o sucesso de uma abordagem de reutilização na ER. Assim sendo, ainda que estudos aprofundados não tenham sido feitos a fim de medir com precisão os reais ganhos da abordagem de GC proposta, acredita-se que ela possa trazer benefícios reais. Neste sentido, acredita-se que esta abordagem pode favorecer políticas de reuso e Gerência de Conhecimento de forma que estas possam facilitar a execução de processos como os de Gerência e Desenvolvimento de Requisitos. Um exemplo disto pode ser a reutilização de requisitos de projetos similares o que facilita a identificação de requisitos e, portanto, o desenvolvimento de requisitos. Como um trabalho futuro, entretanto, pretende-se fazer estudos similares aos feitos em [Falbo et al. 2007], usando agora o apoio de GC oferecido pelas ferramentas de apoio à ER de ODE.

Outro aspecto importante a considerar é que apenas mecanismos de busca, como os implementados nas ferramentas OODE e ReqODE, podem ser insuficientes para resolver o problema. Conforme apontado por vários trabalhos, dentre eles [Falbo et al. 2004] e [Holz 2003], em organizações de software, desenvolvedores geralmente não sabem que existem itens de conhecimento relevantes para o seu trabalho, estão ocupados demais para procurar por itens ou não sabem como fazer a busca adequadamente. Assim, a disseminação pró-ativa passa a ser fundamental. Pretende-se, pois, explorar o uso da tecnologia de agentes no âmbito dos serviços de disseminação dessas ferramentas de forma a fornecer pró-ativamente requisitos, ontologias, padrões de análise e modelos de objetos úteis para os engenheiros de requisitos. A idéia é estender a abordagem já utilizada por outras ferramentas do ambiente para a disseminação pró-ativa de conhecimento, definida em [Falbo et al. 2005].

Sobre tudo o que fora citado, há que se ressaltar que apenas a utilização de um ferramental que apóie o reuso e a Gerência de Conhecimento não garante que uma organização alcançará maturidade nestes aspectos. Para tanto, deve-se definir uma política organizacional que favoreça a reutilização e promova, sobretudo, a construção de itens para reuso.

Agradecimentos

Este trabalho foi realizado com o apoio do CNPq e da CAPES, entidades do Governo Brasileiro dedicadas ao desenvolvimento científico e tecnológico, da FAPES, Fundação de Apoio à Ciência e Tecnologia do Espírito Santo, e das empresas VixTeam e Projeta, parceiras que têm financiado o projeto e dado feedback de sua aplicação a casos reais.

Referências

Carvalho, V.A., Arantes, L.O., Falbo, R.A. (2006) "EstimaODE: Apoio a Estimativas de Tamanho e Esforço no Ambiente de Desenvolvimento de Software ODE", Anais do V Simpósio Brasileiro de Qualidade de Software (SBQS´2006), Vila Velha, Brasil.

Chrissis, M. B., Kontad M., Shrum S, S. (2003) "CMMI: Guidelines for Process Integration and Product Improvement", Addison Wesley.

Cota, R.I., Menezes, C.S., Falbo, R.A. (2004) "Modelagem Organizacional Utilizando Ontologias e Padrões de Análise". VII Workshop Iberoamericano de Ingeniería de Requisitos y Desarrollo de Ambientes de Software, Arequipa, Perú, pp. 56-67.

Devedzic, V. (1999) "Ontologies: Borrowing from Software Patterns". ACM intelligence Magazine, Fall, pp. 14-24.

Falbo, R.A., Martins, A.F., Segrini, B.M., Baiôco, G., Dal Moro, R., Nardi, J.C. (2007) "Um Processo de Engenharia de Requisitos Baseado em Reutilização e Padrões de Análise", VI Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento (JIISIC'07), Lima, Peru (aceito para apresentação e publicação).

Falbo, R.A., Pezzin, J., Schwambach, M.M. (2005) "A Multi-Agent System for Knowledge Delivery in a Software Engineering Environment", 17th International Conference on Software Engineering and Knowledge Engineering, Taipei, China, pp. 253 – 258.

Falbo, R.A., Arantes, D.O., Natali, A.C.C. (2004) "Integrating Knowledge Management and Groupware in a Software Development Environment". Proceedings of the 5th International Conference on Practical Aspects of Knowledge Management, Karagiannis, D., Reimer, U. (Eds.): LNAI 3336, Springer-Verlag Berlin Heidelberg, Austria, pp. 94-105.

Falbo, R.A.; Natali, A.C.C.; Mian, P.G.; Bertollo, G.; Ruy, F.B. (2003) "ODE: Ontology-based software Development Environment", IX Congreso Argentino de Ciencias de la Computación, p. 1124-1135, La Plata, Argentina.

Fowler, M. (1997) *Analysis Patterns: Reusable Object Models*. Addison-Wesley Professional Computing Series.

Gimenes, I.M.S., Huzita, E.H.M. (2005) "Desenvolvimento Baseado em Componentes: Conceitos e Técnicas". Editora Ciência Moderna.

Gomes, P. Pereira, F. C. Paiva, P. Seco, N. Carreiro, P. Ferreira, J. L. Beno, C. (2004) "Rebuilder: A CBR Approach to Knowledge Management in Software Design." 2004. pp.31-42.

Harrison, W., Ossher, H., Tarr, P. (2000) "Software Engineering Tools and Environments: A Roadmap". Proceedings of The Future of Software Engineering (ICSE'2000). Limerick, Ireland, pp. 263 – 277.

Hofmann, H.F., Lehner, F. (2001) "Requirements Engineering as a Success Factor in Software Projects", IEEE Software, July/August.

Holz, H. (2003) "Process-Based Knowledge Management Support for Software Engineering", Doctoral Dissertation, University of Kaiserslautern, Online-Press.

ISO 12207 (2002) "Information Technology - Software life cycle processes, Amendment 1".

Martins, A. F., Nardi, J. C., Falbo, R. A. (2006) "ReqODE: Uma Ferramenta de Apoio à Engenharia de Requisitos Integrada ao Ambiente ODE". Sessão de Ferramentas do XX Simpósio Brasileiro de Engenharia de Software (SBES´2006), Florianópolis-BR.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., Miller, K. J. (1990) "Introduction to WordNet: an on-line lexical database". International Journal of Lexicography 3. 235-244.

Nardi, J. C., Falbo, R.A. (2006) "Uma Ontologia de Requisitos de Software". IX Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, La Plata, Argentina.

Nunes, V.B., Falbo, R.A. (2006) "Uma Ferramenta de Gerência de Configuração Integrada a um Ambiente de Desenvolvimento de Software", Anais do V Simpósio Brasileiro de Qualidade de Software (SBQS´2006), Vila Velha, Brasil.

O´Leary, D.E.; Studer, R. (2001) "Knowledge Management: An Interdisciplinary Approach" IEEE Intelligent Systems, January/February, vol. 16, No. 1.

Robertson, S., Robertson, J. (1999), "Mastering the Requirements Process", Addison Wesley, ACM Press.

Rus, I.; Lindvall, M. (2002) Knowledge Management in Software Engineering. IEEE Software 19(3) May/Jun. pp 26-38.

Santos, G., Villela, K., Schnaider, L., Rocha, A.R., Travassos, G.H. (2004) "Building Ontology-based Tools for a Software Development Environment", Advances in Learning Software Organizations, Melnik G. and Holz, H. (Eds.): LNCS 3096, pp. 19-30.

Softex (2007) "Comunicado Softex MPS 29/2006", http://www.softex.br/mpsbr/_comunicados/comunicado.asp?id=563.

Softex, S. (2006) "MPS.BR (Melhoria de Processo do Software Brasileiro): Guia Geral". Versão 1.1. Maio.

Spyns, P., Meersman, R., Jarrar, M. (2002) "Data Modelling Versus Ontology Engeneering". SIGMOD Rec., Vol. 31, No. 4, pp. 12-17.

Staab, S., Studer, R., Schurr, H. P., Sure, Y. (2001) "Knowledge Processes and Ontologies". IEEE Intelligent Systems, January/February, Vol. 16, No. 1.

Togneri D.F., Falbo, R.A., Menezes, C.S. , Wernesback, B.S., Almeida, D.Q., Côrtes, M.F. (2004) "Gerência de Conhecimento na Engenharia de Requisitos", VII Workshop Iberoamericano de Ingeniería de Requisitos y Desarrollo de Ambientes de Software (IDEAS'2004), Arequipa, Perú, pp. 257-262.

Zave, P. (1997) Classification of research efforts in requirements engineering**.** ACM Computing Surveys Journal, vol. 29, n. 4, pp. 315-321.

Wiegers, K.E. (2003) "Software Requirements", 2nd edition., Redmond, Washington: Microsoft Press.