Uma Ferramenta Baseada em Conhecimento para Apoiar a Definição de Processos de Software em Níveis

Fabiano Borges Ruy, Gleidson Bertollo, Ricardo de Almeida Falbo

Departamento de Informática – Universidade Federal do Espírito Santo
Av. Fernando Ferrari, CEP 29060-900, Vitória – ES - Brasil

{fruy, gbertollo, falbo}@inf.ufes.br

Abstract. *This paper presents ODE's software process definition tool, that supports process definition in various levels of abstraction, including standard, specialized and project processes. To provide a knowledge-based support to this task, it is used ODE's knowledge integration infrastructure, that handles knowledge bases using an inference engine.*

Resumo. *Este artigo apresenta a ferramenta de definição de processos de software do ambiente de desenvolvimento de software ODE. A ferramenta contempla a definição de processos em diversos níveis de abstração, incluindo processos padrão, especializado e de projeto. Para prover um suporte baseado em conhecimento a esta tarefa, é utilizada a infra-estrutura de integração de conhecimento de ODE, que manipula bases de conhecimento através do uso de uma máquina de inferência.*

1. Introdução

Conhecer os processos significa conhecer como os produtos são planejados e produzidos. Neste sentido, a definição de um processo de software é um requisito básico para a obtenção de produtos de software de qualidade. Quando são definidos, processos devem levar em consideração especificidades do projeto em questão. Embora diferentes projetos requeiram processos com características específicas para atender às suas particularidades, é possível estabelecer um conjunto de ativos de processo de software (*software process assets*) comum a todos os processos de software de uma organização, constituindo um Processo Padrão da organização, a partir do qual podem ser definidos processos específicos de projetos, levando em conta suas características particulares.

Este trabalho apresenta a ferramenta de definição de processos de software do ambiente ODE (*Ontology-based software Development Environment*) [1]. A ferramenta contempla a definição de processos em diversos níveis de abstração, incluindo processos padrão, especializado e de projeto. Para prover apoio baseado em conhecimento à definição de processos, é utilizada a camada de inferência da infra-estrutura de integração de conhecimento de ODE, que manipula bases de conhecimento através da realização de inferências.

Este artigo está organizado da seguinte forma: a seção 2 apresenta o ambiente ODE e a camada de inferência de sua infra-estrutura de integração de conhecimento; na seção 3, é apresentada a ferramenta de definição de processos e discute-se o apoio baseado em conhecimento por ela oferecido; finalmente, na seção 4, são apresentadas as considerações finais deste trabalho.

2. O Ambiente ODE

ODE é um Ambiente de Desenvolvimento de Software Centrado em Processo, que tem sua fundamentação baseada em ontologias. A premissa de ODE é a seguinte: se as ferramentas são construídas baseadas em ontologias, a sua integração pode ser facilitada, pois os conceitos envolvidos estão bem definidos na ontologia. Ontologias reduzem confusões terminológicas e conceituais, facilitando o entendimento compartilhado e a comunicação entre

as pessoas e entre as ferramentas do ambiente [1]. Uma ontologia que merece destaque, no contexto deste trabalho, é a ontologia de processo de software desenvolvida em [2].

Em seu corrente estágio, ODE possui diversas ferramentas integradas e sua abordagem de integração considera as seguintes dimensões:

- *Integração de Dados*: diz respeito aos meios como as ferramentas compartilham dados, incluindo serviços de repositório e de compartilhamento de dados;
- *Integração de Conhecimento*: com o aumento da complexidade dos processos de software, faz-se necessário considerar informações de natureza semântica e gerenciar o conhecimento obtido ao longo dos projetos. O conhecimento, assim como os dados, deve estar disponível no ambiente para ser compartilhado por diferentes ferramentas;
- *Integração de Apresentação*: trata de aspectos de padronização das interfaces com o usuário;
- *Integração de Controle*: refere-se à habilidade de uma ferramenta notificar ou iniciar uma ação em outra;
- *Integração de Processo*: diz respeito à ligação entre as ferramentas e o processo de software. Para integrar ferramentas, é preciso ter um foco forte no gerenciamento do processo de software. O processo deve definir que ferramentas um desenvolvedor pode utilizar e quando ele deverá ter acesso às mesmas, em função da atividade do processo que ele está desempenhando.

Como parte da infra-estrutura para apoiar a integração de conhecimento em ODE, foi desenvolvida uma camada de inferência. Seu objetivo é prover meios para que seja possível representar e manipular bases de conhecimento através de uma linguagem lógica, dotando o ambiente de capacidades de inferência, de modo que as ferramentas possam oferecer apoio baseado em conhecimento à realização de tarefas.

A idéia é expressar o conhecimento na forma de fatos e isolar as regras da aplicação, representando-os logicamente em bases de conhecimento. Separar regras da aplicação é uma maneira flexível de manter o conhecimento sem a completa reestruturação das aplicações. O resultado esperado é o desenvolvimento de sistemas mais robustos com uma base de conhecimento maleável que possa crescer e mudar na medida em que o negócio evolui [3]. Além disso, o uso de técnicas de Inteligência Artificial é capaz de aperfeiçoar a manipulação dessas regras, proporcionando ao sistema um caráter mais dinâmico e permitindo a realização de inferências.

Com a introdução de uma linguagem lógica em sistemas orientados a objetos (OO), como é o caso do ambiente ODE, é possível aproveitar os benefícios de um segundo paradigma. Diversas operações, convencionalmente feitas por meio de uma linguagem OO, produzem melhores resultados quando realizadas através de inferências lógicas. Situações como as que envolvem recursividade ou lidam com muitos relacionamentos entre objetos, exigindo casamento de padrões ou vários acessos às informações dos objetos, são mais adequadamente tratadas com uma abordagem lógica. Esses tipos de situações são bastante comuns, especialmente em operações envolvendo conhecimento.

A utilização da camada de inferência de ODE consiste basicamente de três passos, apresentados na figura 1. No primeiro passo, o Gerente de Conhecimento cria as estruturas das bases de conhecimento, que são compostas por predicadores, definidos por reflexão computacional a partir das classes e métodos do sistema, e por regras, que possuem conclusão e condições montadas com os predicadores definidos. A qualquer momento os predicadores e regras podem ser editados, reestruturando as bases de conhecimento.

Na instanciação de uma base de conhecimento, passo 2, os predicadores de sua estrutura são utilizados para, junto a dados extraídos do sistema, criar os fatos que se unem às regras para compor a base de conhecimento. Ao final da instanciação, é gerada a representação

lógica da mesma na forma de um arquivo Prolog, que é utilizado na realização de inferências.

No último passo, a realização de inferências, há interação com uma máquina Prolog incorporada à camada para que sejam realizadas as consultas lógicas. Existem duas formas disponíveis de realizar inferências usando a camada. A primeira, e mais simples, é via interface gráfica, onde o usuário pode montar as consultas desejadas e obter os resultados de inferências apenas visualmente. A segunda, que proporciona melhores resultados, requer implementação. O desenvolvedor deve montar a consulta via código, utilizando as funcionalidades da camada. Em ambos os casos, a camada de inferência transforma os objetos da consulta em dados lógicos e, então, realiza a inferência sobre o arquivo gerado na instanciação da base de conhecimento, utilizando a máquina Prolog. Finalmente, o resultado da inferência é transformado em objetos e retornado à aplicação.

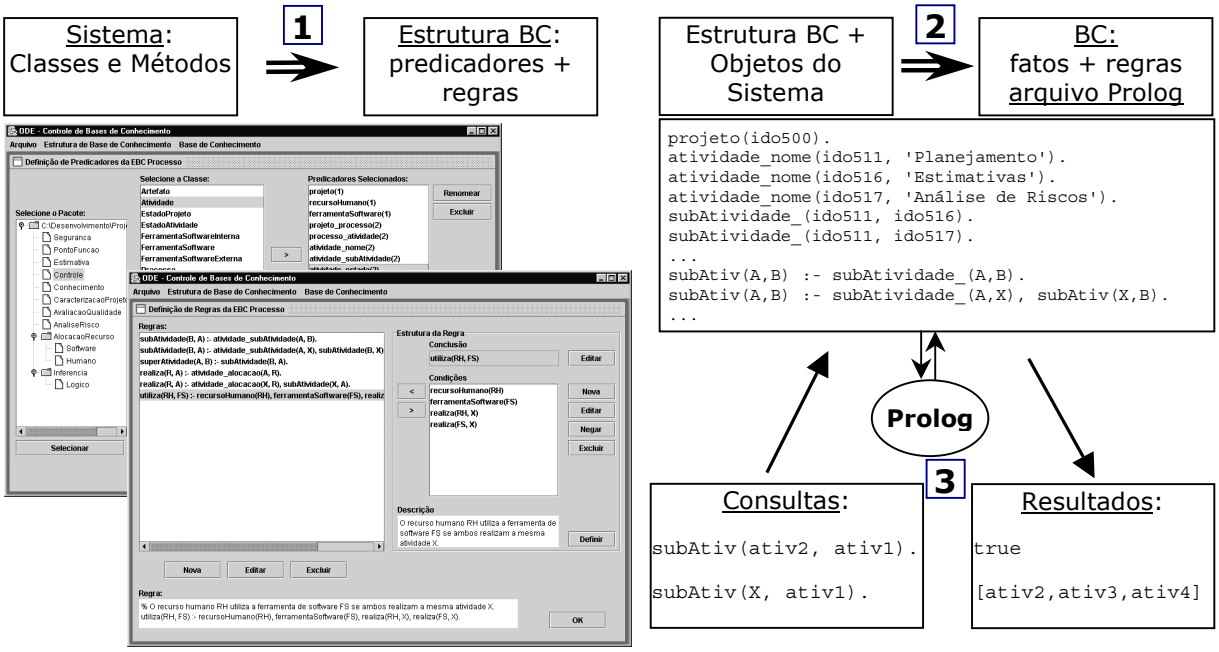

Figura 1 – Esquema de utilização da Camada de Inferência

A camada disponibiliza ao ambiente uma forma simples de criação, manutenção e validação de bases de conhecimento e uma abordagem de manipulação do conhecimento representado que oferece vantagens em relação à convencionalmente utilizada. Além disso, o sistema continua a operar com objetos, tratados, agora, segundo um outro paradigma.

3. Definição de Processos de Software em ODE

A abordagem utilizada para a definição de processos de software em ODE é uma adaptação do modelo proposto em [4], que propõem a definição de processos em três níveis de abstração: definição do processo padrão, especialização do processo padrão e instanciação para projetos específicos.

Um processo de software deve ser definido especificamente para cada projeto de software, no qual são consideradas suas particularidades. Contudo, pode-se estabelecer um processo padrão de desenvolvimento de software para uma organização, que conterá os ativos de processo que deverão fazer parte dos processos de qualquer projeto dessa organização. Dessa forma, é possível obter padronização e economia de tempo e esforço na definição de novos processos. A cultura organizacional e a prática da organização em engenharia de software influenciam diretamente na definição do seu processo de software padrão.

Um processo padrão pode, então, ser especializado para considerar tecnologias de desenvolvimento, paradigmas ou domínios de aplicação específicos. Durante a especialização, ativos de processo poderão ser adicionados ou modificados, de acordo com o contexto para o

qual se está realizando a especialização. Porém, os elementos básicos definidos no processo padrão deverão sempre estar presentes nos processos especializados, mantendo, assim, uma conformidade entre os dois processos.

Os processos padrão e especializados descrevem as atividades que devem pertencer aos processos de quaisquer projetos de uma organização, ou seja, descrevem o conhecimento a respeito dos processos da organização. A figura 2 apresenta as classes do pacote *Conhecimento* relacionadas à definição de processos. Deve-se observar a importância dos objetos deste pacote, que são utilizados para prover o apoio baseado em conhecimento à tarefa de definição de processos, sendo os principais objetos, neste contexto, manipulados pela camada de inferência. Além disso, este modelo foi desenvolvido tomando por base a ontologia de processos de software definida em [2].

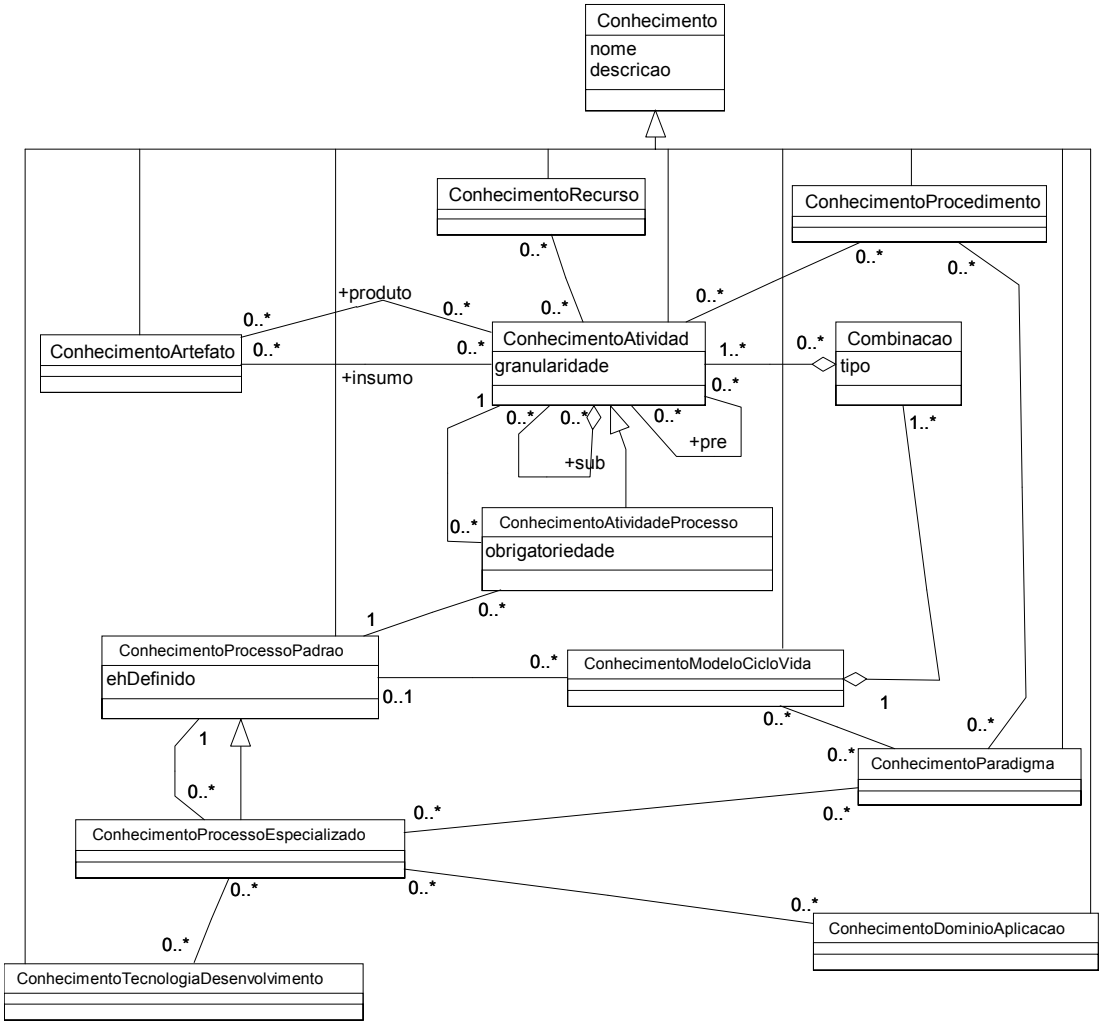

Figura 2 – O Pacote Conhecimento

Após encerrar a definição do processo padrão ou especializado, é preciso definir seus modelos de ciclo de vida (MCV), com base nas macro-atividades (atividades que não fazem parte de outras atividades) definidas. Deve-se manter a conformidade do modelo com a ordem de precedência das atividades definida no processo padrão ou especializado. A definição de processos padrão e especializados, e seus MCVs relacionados é de responsabilidade do Gerente de Conhecimento.

Definidos os MCVs de um processo padrão ou especializado, estes podem ser utilizados na definição de processos de projeto. O Gerente de Projeto escolhe o processo que deseja instanciar, bem como o MCV que deseja adotar no projeto em questão. Neste momento, o ambiente gera um processo de projeto inicial contendo as atividades definidas pelo processo

padrão ou especializado e os demais ativos de processo definidos. Em seguida, o Gerente de Projeto pode incorporar novas atividades ao processo, bem como definir outros ativos de processo, de acordo com as características do projeto.

A janela para definição de processos padrão, especializado ou de projeto é similar e está apresentada na figura 3. A árvore localizada na parte esquerda da janela contém os ativos de processo já definidos para o processo em definição, isto é, as atividades do processo, suas subatividades, pré-atividades, artefatos, recursos e procedimentos. Na parte direita, encontra-se um painel que possibilita a escolha dos ativos de processo para o item selecionado na árvore. A lista mais à esquerda contém os ativos sugeridos a partir da realização de inferência nas bases de conhecimento definidas no ambiente. A lista mais à direita contém os elementos já definidos para o processo.

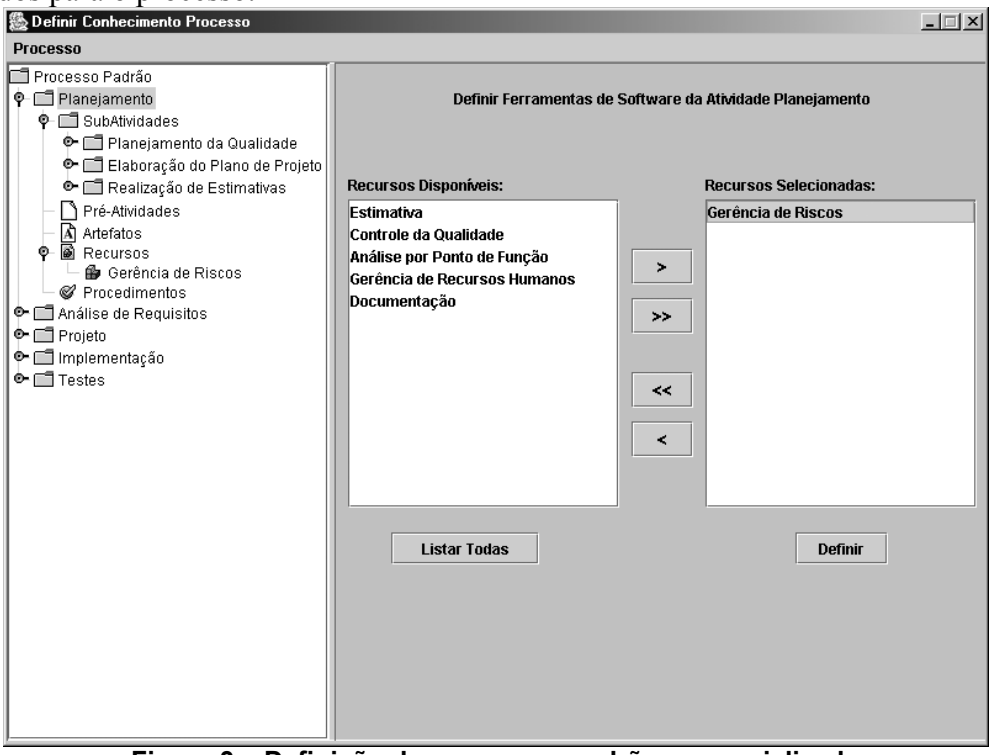

Figura 3 – Definição de processos padrão e especializado

As bases de conhecimento, previamente definidas pelo Gerente de Conhecimento com base no pacote *Conhecimento*, são instanciadas na inicialização da ferramenta e utilizadas para que sejam realizadas inferências no momento em que são feitas as sugestões. Dessa forma, é oferecido apoio baseado em conhecimento a essa tarefa.

Para demonstrar o uso de inferências na ferramenta, toma-se o caso da definição de subatividades de uma determinada atividade. O ambiente sugere, em um primeiro momento, as subatividades associadas a uma atividade, conforme definido no modelo de objetos do pacote *Conhecimento*. Porém, a ferramenta fornece também uma forma mais elaborada dessa sugestão, considerando a regra de subatividades mostrada na figura 4. Assim, são sugeridas, também, as subatividades das subatividades, de acordo com a transitividade imposta pela regra.

```
% Regra de Subatividade (A é subatividade de B)
subAtividade(A, B) :- subAtividade_(A, B).
subAtividade(A, B) :- subAtividade_(A, X), subAtividade(X, B).

% Regra de Recurso (Atividade A utiliza o Recurso R)
utiliza(A, R) :- subAtividade(X, A), utiliza(X, R).
```

Figura 4 – Regras da Base de Conhecimento utilizada na Definição de Processos

De forma semelhante, a sugestão de recursos para uma atividade é mais efetiva se considerar a regra de recursos, exibida na figura 4. O uso dessa regra permite que uma atividade aloque também recursos que estejam mapeados para suas subatividades. Como pode ser visto na figura 3, é possível utilizar uma ferramenta de gerência de riscos na atividade de Planejamento, sem que necessariamente, neste processo, a atividade Análise de Riscos, à qual a ferramenta está associada, tenha sido definida como uma subatividade de Planejamento. Isso decorre da inferência realizada através do trecho de código abaixo e do fato do modelo de Conhecimento definir que Análise de Riscos é uma potencial subatividade de Planejamento.

```
//Obtém a sugestão de recursos
Variavel var = new Variavel("R");
Object[] termos = new Object[] {atividade.obterConhecimento(), var};
List lstRecursos = baseConhecimento.inferir("utiliza", termos);
...
```

Vale ressaltar que qualquer ferramenta do ambiente pode utilizar a camada de inferência, bastando para isso definir suas bases de conhecimento. As funcionalidades oferecidas pela camada podem ser empregadas em diversas situações, tais como indicação de sugestões, utilização de dados históricos, configuração do ambiente através do perfil do usuário e agentes inteligentes.

4. Considerações Finais

A ferramenta apresentada provê apoio automatizado à definição de processos de software. Mais do que isso, ela proporciona um suporte baseado em conhecimento a uma tarefa relativamente complexa.

Existe uma forte ligação entre a ferramenta e a ontologia de processos na qual o ambiente se fundamenta. O uso de inferências possibilitou à ferramenta operar com maior fidelidade em relação à ontologia, respeitando suas restrições e axiomas. As situações demonstradas fazem parte de um conjunto de operações que aproximam a ferramenta de seu modelo conceitual. Realizar esse tipo de operação, muitas vezes, é extremamente custoso se feito da maneira convencional. Porém, o uso de deduções lógicas possibilitou que as operações fossem implementadas de maneira mais fácil e obtendo melhor desempenho em sua execução.

A abordagem de definição de processos de software em níveis abre espaço para a melhoria contínua de processos de software, baseada na captura de experiências e informações de projetos que utilizam o processo padrão como base para a definição do seu processo. Nesse contexto, o uso de inferências pode apoiar a busca e disseminação de dados históricos, obtidos de projetos anteriores, fornecendo subsídios para um contínuo aprimoramento dos processos.

Agradecimentos

Os autores agradecem ao CNPq e à CAPES pelo apoio financeiro a este trabalho.

Referências Bibliográficas

[1] Bertollo, G., Ruy, F.B., Mian, P.G., Pezzin, J., Schwambach, M., Natali, A.C.C., Falbo, R.A. *ODE – Um Ambiente de Desenvolvimento de Software Baseado em Ontologias*. Anais do XVI SBES, Caderno de Ferramentas, Gramado, Outubro 2002.

[2] Falbo, R.A. *Integração de Conhecimento em um Ambiente de Desenvolvimento de Software*. Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro, Dezembro 1998.

[3] Rasmus, D.W. *Ruling Classes: The heart of knowledge-based systems*. Object Magazine, July 1995.

[4] Rocha, A. R. C., Maldonado, J. C., Weber, K. C., *Qualidade de Software: Teoria e Prática*. São Paulo: Prentice Hall, 2001.