Apoio Automatizado à Definição de Processos de Software em Níveis

Gleidson Bertollo
Ricardo de Almeida Falbo

Departamento de Informática
Universidade Federal do Espírito Santo
Av. Fernando Ferrari, CEP 29060-900, Vitória – ES – Brasil
{gbertollo, falbo}@inf.ufes.br

Resumo

Um importante requisito para se obter qualidade em software consiste em definir e seguir um processo de software. Entretanto, definir um processo de software não é uma tarefa trivial. Processos devem ser definidos caso a caso considerando-se as especificidades da aplicação, a tecnologia a ser adotada na sua construção e o grupo de desenvolvimento. Para facilitar essa tarefa, organizações podem (e devem) definir processos padrão, que sirvam de ponto de partida para a definição de processos de projetos específicos. Deste modo, a definição de processos de software pode se dar em níveis: definir um processo padrão de desenvolvimento, especializar esse processo para paradigmas, tecnologias e domínios específicos e, por fim, instanciar esses processos para um projeto de software. Este trabalho apresenta uma ferramenta de apoio à definição de processos de software em níveis, que está integrada a um ambiente de desenvolvimento de software.

Palavras-chave: Processo de Software, Processo de Software Padrão, Ambiente de Desenvolvimento de Software.

Abstract

An important requirement for software quality is to define and follow a software development process. However, software process definition is a complex task. Processes should be defined for each case, considering the application characteristics, the development team and the technology to be applied. Although different projects require processes with specific features in order to regard its peculiarities, it is possible to establish a set of software process assets for use in software process definition. This collection of software process assets is called standard software process. This way, software process definition can be done in different levels of abstraction. First, a standard software process is defined for the organization. Based on the organizational software process, specialized standard processes can be defined considering certain paradigm, technology or domain application. Finally, project processes can be instantiated from standard processes. This paper presents a tool that supports this leveled approach to software process definition. This tool is integrated to a software engineering environment.

Keywords: Software Process, Standard Software Process, Software Engineering Environment

1 - Introdução

O desenvolvimento de produtos de software de qualidade, dentro do cronograma e considerando os custos planejados sempre foi um desafio para as organizações de desenvolvimento de software. Pesquisas indicam que a principal causa dos problemas no desenvolvimento de software é a falta de um processo de desenvolvimento de software claramente definido e efetivo. A qualidade de um produto de software depende fortemente da qualidade do processo de software utilizado em seu desenvolvimento [1]. Conhecer os

processos significa conhecer como os produtos são planejados e produzidos [2]. Assim, a definição de um processo de software se tornou um requisito básico para se chegar a produtos de software de qualidade.

Porém, a definição de um processo de software não é uma tarefa trivial. Processos devem ser definidos caso a caso, levando-se em consideração as especificidades do projeto em questão. Deve-se considerar a adequação às tecnologias envolvidas, ao tipo de software em questão, ao domínio de aplicação, ao grau de maturidade (ou capacitação) da equipe em engenharia de software, às características próprias da organização e às características do projeto e do grupo de desenvolvimento [3]. 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*) a ser utilizado na definição de processos de software de uma organização. Essas coleções de ativos de processo de software constituem os chamados processos padrão de desenvolvimento de software. Processos para projetos específicos podem, então, ser definidos a partir da instanciação do processo de software padrão da organização, levando em consideração suas características particulares. Esses processos instanciados são ditos processos de projeto [2].

De fato, o modelo de definição de processos baseado em processos padrão pode ser estendido para comportar vários níveis. Primeiro, deve-se definir um processo padrão da organização, contendo os ativos de processo (atividade, artefatos, recursos e procedimentos) que devem fazer parte de todos os processos de projeto da organização. Esse processo padrão pode ser especializado para agregar novos ativos de processo, considerando aspectos, tais como tecnologias de desenvolvimento, paradigmas ou domínios de aplicação. Assim, obtém-se processos mais completos, que consideram características da especialização desejada. Por fim, a partir de um processo padrão ou de um processo especializado, é possível instanciar um processo de projeto, que será o processo a ser utilizado em um projeto de software específico. Para definir esse processo devem ser consideradas as particularidades de cada projeto [3].

Apesar de facilitar a definição de processos de software, o modelo anteriormente apresentado é complexo e requer que essa tarefa seja realizada por engenheiros de software experientes, com elevado grau de conhecimento e atualização. De fato, para profissionais menos experientes, esta é uma atividade que requer alguma forma de ajuda e o apoio automatizado é uma maneira de oferecer essa ajuda [3].

Este trabalho apresenta uma ferramenta de apoio à definição de processos em níveis, integrada ao ambiente ODE (*Ontology based software Development Environment*) [4]. ODE é um Ambiente de Desenvolvimento de Software Centrado em Processo baseado em ontologias. Embora ODE seja baseado em diversas ontologias, a principal delas é uma ontologia de processo de software [5]. ODE se utiliza desta para estabelecer uma ligação explícita entre as ferramentas do ambiente e os processos definidos, bem como para organizar sua estrutura interna e oferecer apoio à definição de processos.

A seção 2 discute processos de software e sua automatização através de Ambientes de Desenvolvimento de Software (ADSs) Centrados em Processo. Essa seção apresenta, ainda, o ambiente ODE, um ADS Centrado em Processo, ao qual a ferramenta proposta está integrada. A seção 3 trata de um modelo para definição de processos de software em níveis. Na seção 4, discute-se como se dá a definição de processos em ODE. Finalmente, as seções 5 e 6 apresentam trabalhos correlatos e as conclusões deste trabalho.

2 – Processos e Ambientes de Desenvolvimento de Software

Um processo de software pode ser visto como o conjunto de atividades, métodos, práticas e transformações que guiam pessoas na produção de software. Um processo eficaz

deve, claramente, considerar as relações entre as atividades, os artefatos produzidos no desenvolvimento, as ferramentas e os procedimentos necessários e a habilidade, o treinamento e a motivação do pessoal envolvido [5].

Com a crescente preocupação com a qualidade de software, cada vez mais as organizações de desenvolvimento de software têm procurado seguir orientações de modelos e padrões de qualidade do processo, tais como ISO/IEC 12207 [6], CMM [2] e ISO 9001 [7]. Esses modelos e padrões provêem uma importante e bastante abrangente contribuição para a área de processos de software. Contudo, a observação dos vários aspectos descritos nesses modelos e normas e a própria complexidade inerente a grandes projetos de software tornam os processos de software bastante complexos.

Nesse contexto, torna-se essencial prover ferramentas para auxiliar o engenheiro de software em suas atividades. O suporte de ferramentas influencia diretamente o tempo de desenvolvimento do software, o seu custo e a qualidade do produto desenvolvido [8].

Muitas ferramentas foram desenvolvidas como forma de automatizar as diversas tarefas que compõem o processo de desenvolvimento de software. Essas ferramentas, conhecidas como ferramentas CASE (*Computer Aided Software Engineering*), significaram um grande avanço na área de Engenharia de Software, pois a partir delas possibilitou-se um aumento de produtividade e de qualidade, uma maior flexibilidade para mudanças e uma melhor documentação, facilitando, assim, o desenvolvimento e a manutenção dos sistemas construídos.

Contudo, embora as ferramentas CASE possam resolver um problema específico para o qual foram construídas, elas não se propõem a tratar o processo de desenvolvimento de software como um todo. Para serem realmente eficazes, é preciso que essas ferramentas trabalhem em conjunto, de forma integrada, provendo o compartilhamento de informações, de modo a apoiar o desenvolvedor ao longo de todo o processo de software.

A identificação da necessidade de apoio integrado ao longo de todo o processo de software representa a gênese dos Ambientes de Desenvolvimento de Software (ADSs) [8]. ADSs buscam combinar técnicas, métodos e ferramentas para apoiar o engenheiro de software na construção de produtos de software, abrangendo todas as atividades inerentes ao processo, tais como planejamento, gerência, desenvolvimento e controle da qualidade [5]. ADSs podem ser definidos como uma coleção de ferramentas integradas que auxiliam o trabalho do engenheiro de software durante as atividades que compõem o processo de desenvolvimento [8].

A integração torna-se, assim, um fator de fundamental importância. Porém, construir ferramentas que se comunicam e compartilham informações não é tarefa simples. Integração demanda representação consistente da informação, interfaces padronizadas, significados homogêneos para a comunicação entre engenheiros de software e ferramentas e uma efetiva abordagem que possibilite aos ADSs mudar entre várias plataformas [9].

A integração de ferramentas em ADSs pode ser tratada como uma questão de seis dimensões [10, 5]:

- *Integração de Apresentação*: suportada por serviços de interface com o usuário, tem o objetivo de criar uma uniformidade entre as interfaces do ambiente, aumentando a sua usabilidade.
- *Integração de Controle*: refere-se à habilidade de uma ferramenta notificar ou iniciar uma ação em outra, controlando os eventos ocorridos e compartilhando funcionalidades.
- *Integração de Processo*: diz respeito à ligação explícita 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.

- *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. Desta forma, o conhecimento, assim como os dados, deve estar disponível no ambiente para ser compartilhado por diferentes ferramentas.
- *Integração de Plataforma*: refere-se à independência da plataforma sobre a qual funcionará o ambiente e suas ferramentas.

Dada a sua grande importância, estudos sobre integração de processo em ADSs deram origem a uma nova classe de ambientes, os Ambientes de Desenvolvimento de Software Centrados em Processo [1]. Um ADS Centrado em Processo é aquele que explora uma definição explícita do processo de software e pode ser visto, de fato, como a automatização desse processo, incluindo mecanismos para: *(i)* guiar a seqüência de atividades definida; *(ii)* gerenciar os produtos que estão sendo desenvolvidos; *(iii)* executar ferramentas necessárias para a realização das atividades; *(iv)* permitir comunicação entre as pessoas; *(v)* colher dados de métricas automaticamente; (vi) reduzir erros humanos e *(vii)* prover controle do projeto à medida que este vai sendo executado [11].

São várias as experiências relatadas na literatura referentes à automatização da definição de processos de software, grande parte delas realizadas no contexto de ADSs Centrados em Processo, tais como [5, 12].

Este trabalho trata a definição de processos no contexto do ambiente ODE (*Ontology based software Development Environment*) [4]. ODE é um ADS Centrado em Processo baseado em ontologias. Uma ontologia define um vocabulário específico usado para descrever uma certa realidade, mais um conjunto de decisões explícitas fixando de forma rigorosa o significado pretendido para o vocabulário. Uma ontologia envolve, então, um vocabulário de representação que captura os conceitos e relações em algum domínio e um conjunto de axiomas, que restringem a sua interpretação [13].

A premissa do projeto de ODE é a seguinte: se as ferramentas de um ADS são construídas baseadas em ontologias, a integração dessas ferramentas pode ser facilitada, pois os conceitos envolvidos estão bem definidos na ontologia. Essa é uma das principais características que distingue ODE de outros ADSs, sua base ontológica. Especialmente em ADSs, ontologias reduzem confusões terminológicas e conceituais, facilitando o entendimento compartilhado e a comunicação entre pessoas com diferentes necessidades e pontos de vista. Além disso, a padronização de conceitos provida por uma ontologia permite que a comunicação entre as ferramentas que compõem o ambiente seja aprimorada [14]. Dentre as ontologias que compõem a base ontológica de ODE, uma delas merece destaque no contexto deste trabalho: a ontologia de processo de software [5].

A arquitetura de ODE reflete sua base ontológica. Ela possui dois níveis: o Nível Base e o Meta-Nível. O nível base define as classes que controlam os processos definidos no ambiente (o pacote *Controle*) e suas ferramentas. O meta-nível, ou pacote *Conhecimento*, define as classes que descrevem o conhecimento sobre os objetos do nível base. A figura 1 mostra a relação entre esses dois níveis.

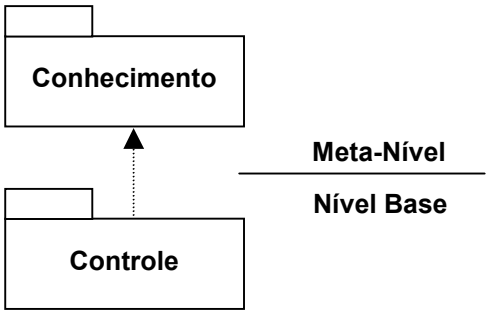

Figura 1 – Arquitetura em Níveis de ODE

As classes do meta-nível são derivadas diretamente de ontologias e seus objetos podem ser vistos como itens de instanciação de uma ontologia. Elas constituem o conhecimento do ambiente, podendo ser utilizado por todas as ferramentas que o compõem. Uma ontologia não tem o objetivo de descrever todo o conhecimento envolvido em um domínio, mas deve modelar o que é essencial para contextualizar esse domínio. Dessa forma, novas classes, associações, atributos e operações são criados para tratar decisões específicas na modelagem do nível base.

As classes do nível base não derivam diretamente de ontologias, mas, muitas vezes, são também baseadas nelas. Muitas classes desse nível possuem uma classe de conhecimento correspondente. Dessa forma, o meta-nível pode ser usado para descrever características do nível base.

No caso do tratamento de processos em ODE, as classes do *Pacote Controle*, que correspondem ao nível base, como mostra a figura 1, são responsáveis pela definição e controle dos processos de software em ODE. Elas não derivam diretamente da ontologia de processo de software [5], mas são baseadas nelas. Algumas classes desse nível têm uma classe correspondente no *Pacote Conhecimento* (meta-nível), preservando as mesmas restrições contidas no modelo de conhecimento, este sim baseado na ontologia. Dessa maneira, o *Pacote Conhecimento* é usado para descrever características dos objetos do *Pacote Controle*.

3 - Modelo para Definição de Processos de Software em Níveis

Para se definir um processo de software, é necessário um nível de experiência e conhecimento bastante alto por parte do engenheiro de software, pois inúmeras variáveis devem ser consideradas, como, por exemplo, características da equipe de desenvolvimento, características específicas do projeto, nível de conhecimento da organização em engenharia de software, entre outras. Um processo de software deve ser definido especificamente para cada projeto de software.

Um elemento chave, conforme advogado pelas diversas normas e modelos de maturidade de processo, para apoiar a definição de processos para projetos e, de fato, para se atingir a qualidade do processo, consiste na definição de um processo padrão, estabelecendo um conjunto básico de ativos de processo de software (*software process assets*) que deve ser utilizado na definição de todos os processos de software de uma organização. Esse processo deve servir de base para os processos específicos, ditos processos de projeto, e deve passar por constantes melhorias, aproveitando as experiências adquiridas nos projetos.

Com um processo padrão definido para a organização, processos de projeto passam a ser, então, definidos como uma adaptação (ou instanciação) do processo de software padrão da organização, levando em consideração suas características particulares.

De fato, o modelo de definição de processos baseado em processos padrão pode ser estendido para comportar vários níveis. Rocha et al. [3] propõem um modelo para definição de processos de software que consiste em três etapas: definição do processo padrão, especialização do processo padrão e instanciação para projetos específicos. Como produtos de cada etapa, tem-se processos em diferentes níveis de abstração. A Figura 2 apresenta os diferentes níveis de definição de processos, assim como os produtos gerados após cada etapa e os fatores que influenciam a definição de processos nos diferentes níveis de abstração.

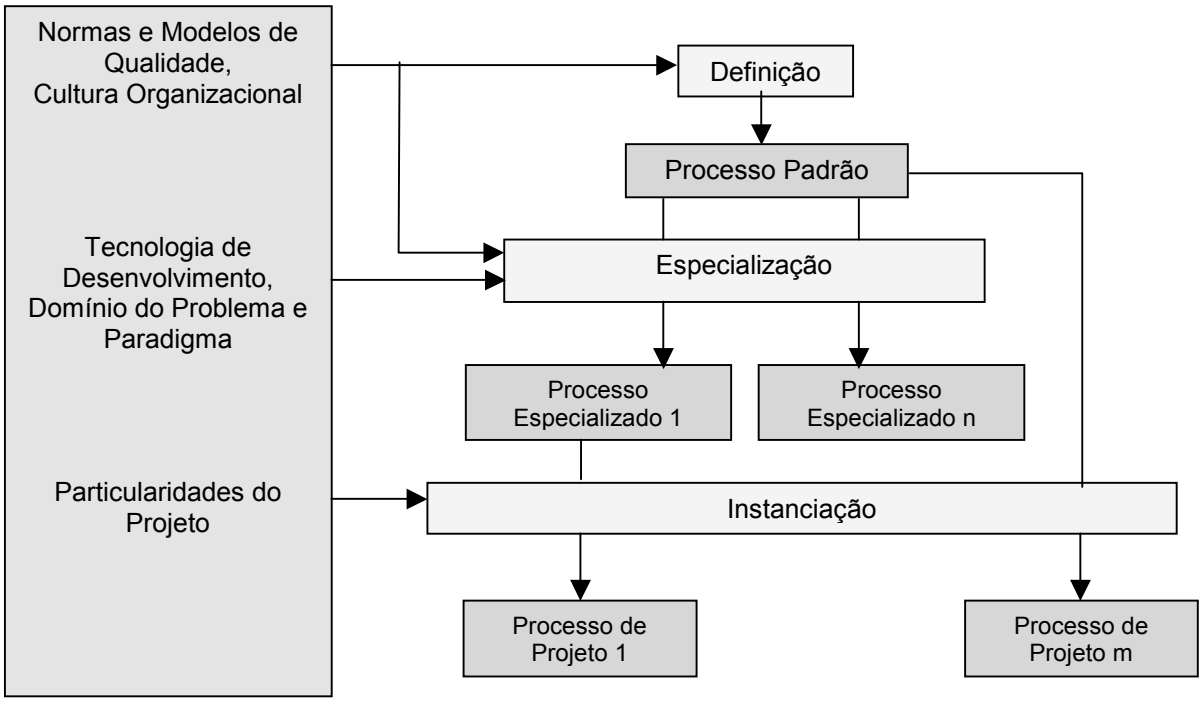

Figura 2 – Modelo para Definição de Processos em Níveis (adaptado de [3])

Primeiro se define o processo padrão da organização. Um processo de software padrão da organização é a definição operacional de um processo básico que servirá como ponto de partida para a definição de processos de software de uma organização. Dessa forma, é possível fazer uma economia de tempo e esforço na definição de novos processos. Esse processo contém os ativos de processo (atividade, artefatos, recursos e procedimentos) que deverão fazer parte dos processos de qualquer projeto da organização. Para uma organização estar apta a definir um processo padrão de desenvolvimento, ela já deve possuir um bom nível de maturidade. Normas e modelos de qualidade existentes, tais como a ISO/IEC 12207 [6] e o CMM [2], podem servir de base, auxiliando e direcionando essa definição. Porém, as particularidades e as características da organização também devem ser levadas em consideração. A cultura organizacional e a prática da organização em engenharia de software influenciam diretamente na definição do seu processo de software padrão.

A seguir, o processo padrão pode ser especializado para considerar tecnologias de desenvolvimento, paradigmas ou domínios de aplicação específicos. Diferentes tipos de software exigem diferentes práticas de desenvolvimento e, conseqüentemente, diferentes processos de desenvolvimento. 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. Vale ressaltar que, apesar da figura 2 mostrar apenas três níveis de definição

de processos, na realidade, são possíveis vários níveis de especialização. Por exemplo, pode-se definir um processo especializado para Desenvolvimento Web (especialização com base na tecnologia de desenvolvimento), que poderia ser novamente especializado, agora considerando Desenvolvimento Web Orientado a Objetos (especialização com base em um paradigma).

Finalmente, no último nível do modelo de definição de processos está a instanciação de um processo padrão ou de um processo especializado para um projeto específico. Essa instanciação consiste na adaptação de um processo para um projeto de software. Nessa adaptação, devem ser consideradas as particularidades do projeto e as características da equipe de desenvolvimento. Define-se, nesse momento, o modelo de ciclo de vida a ser utilizado durante o desenvolvimento e pode-se incorporar novas atividades ao processo, bem como alterar seus artefatos consumidos e produzidos, recursos utilizados e procedimentos adotados.

É importante ressaltar que estabelecer um processo bem definido para um projeto de software não garante a qualidade do produto final, porém a probabilidade de se obter, ao término de sua execução, o software planejado, atendendo os requisitos levantados, aumenta consideravelmente. Conhecer o processo significa conhecer como os produtos são planejados e produzidos. A partir da definição de processo, é possível definir-se medições e coletar dados de execução. Isto dá visibilidade aos gerentes sobre o andamento dos projetos. Uma vez definido o processo e estabelecido o treinamento, o processo fluirá por si só, sem depender de pessoas chave. Ou seja, quando uma pessoa deixar a equipe, outra poderá assumir seu lugar, guiando-se por este processo [2].

4 - Definição de Processos de Software em ODE

A definição de processos em ODE segue o modelo de definição de processos apresentado na seção anterior. Os processos padrão e especializado descrevem as atividades que devem pertencer aos processos de quaisquer projetos de uma organização, ou seja, esses processos descrevem o conhecimento a respeito dos processos dessa organização. Dessa forma, de acordo com a arquitetura de ODE, processos padrão e especializado são definidos no Meta-Nível e sua modelagem encontra-se no *Pacote Conhecimento*.

A definição do modelo de conhecimento sobre processos de software tem por objetivo apoiar a aquisição, organização, reuso e compartilhamento do conhecimento sobre processos de desenvolvimento de software. A figura 3 apresenta as classes do *Pacote Conhecimento* relacionadas com a definição de processos padrão e especializados. Essas classes são derivadas diretamente da ontologia de processo de software descrita em [5].

Tomando como referência essa ontologia, a definição de um processo envolve definir as atividades do processo, assim como os artefatos (insumos e produtos), recursos (humanos, hardware e software) e procedimentos (métodos, técnicas, normas e roteiros) necessários para sua execução.

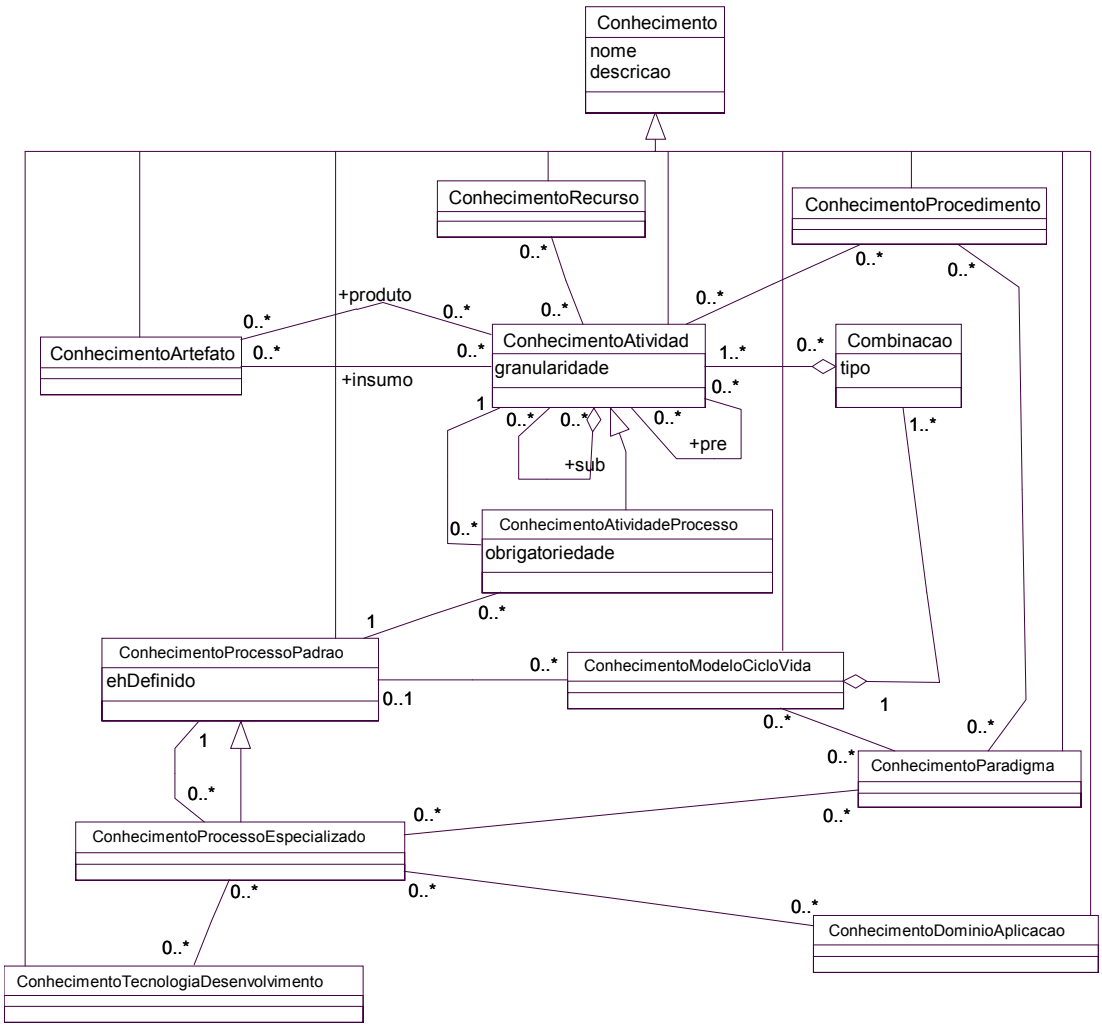

Figura 3 – O Pacote Conhecimento

A classe *ConhecimentoProcessoPadrao* representa os processos padrão definidos no ambiente ODE. Cada processo padrão pode ser especializado, incluindo características relacionadas com paradigmas, tecnologias de desenvolvimento e domínios de aplicação específicos. Dessa forma, a partir do processo padrão de desenvolvimento de uma organização, é possível definir, por exemplo, um processo especializado para o paradigma orientado a objetos ou, então, um processo específico para desenvolvimento Web. Os processos especializados são representados pela classe *ConhecimentoProcessoEspecializado* e as caracterísiticas de suas especializações representadas pelas associações com as classes *ConhecimentoParadigma*, *ConhecimentoTecnologiaDesenvolvimento* e *ConhecimentoDominioAplicacao*.

A classe *ConhecimentoAtividade* descreve o conhecimento sobre atividades típicas de processos de software, bem como suas relações com outros ativos de processo, tais como os possíveis artefatos consumidos e produzidos por atividade deste tipo, recursos que podem ser utilizados e procedimentos que podem ser adotados na sua realização. Estas relações são representadas, respectivamente, pelas associações com as classes *ConhecimentoArtefato*, *ConhecimentoRecurso* e *ConhecimentoProcedimento*. Além disso, uma atividade pode ser decomposta em sub-atividades e é possível definir a precedência entre atividades, estabelecendo, assim, uma ordem de execução.

Para cada processo padrão ou especializado definido, devem ser estabelecidas as atividades que o compõem. As atividades de um processo padrão ou especializado são representadas pela classe *ConhecimentoAtividadeProcesso,* uma subclasse de *ConhecimentoAtividade*. Para cada uma dessas atividades, deve ser definida a sua obrigatoriedade, isto é, se esta atividade poderá ou não ser excluída de um processo definido com base no dado processo padrão ou especializado.

4.1 – Definição de Processos Padrão e Especializados em ODE

A figura 4 apresenta a definição de processos padrão ou especializados em ODE. 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 sub-atividades, pré-atividades, artefatos (insumos e produtos), recursos (humanos, hardware e software) e procedimentos (métodos, técnicas, roteiros e normas).

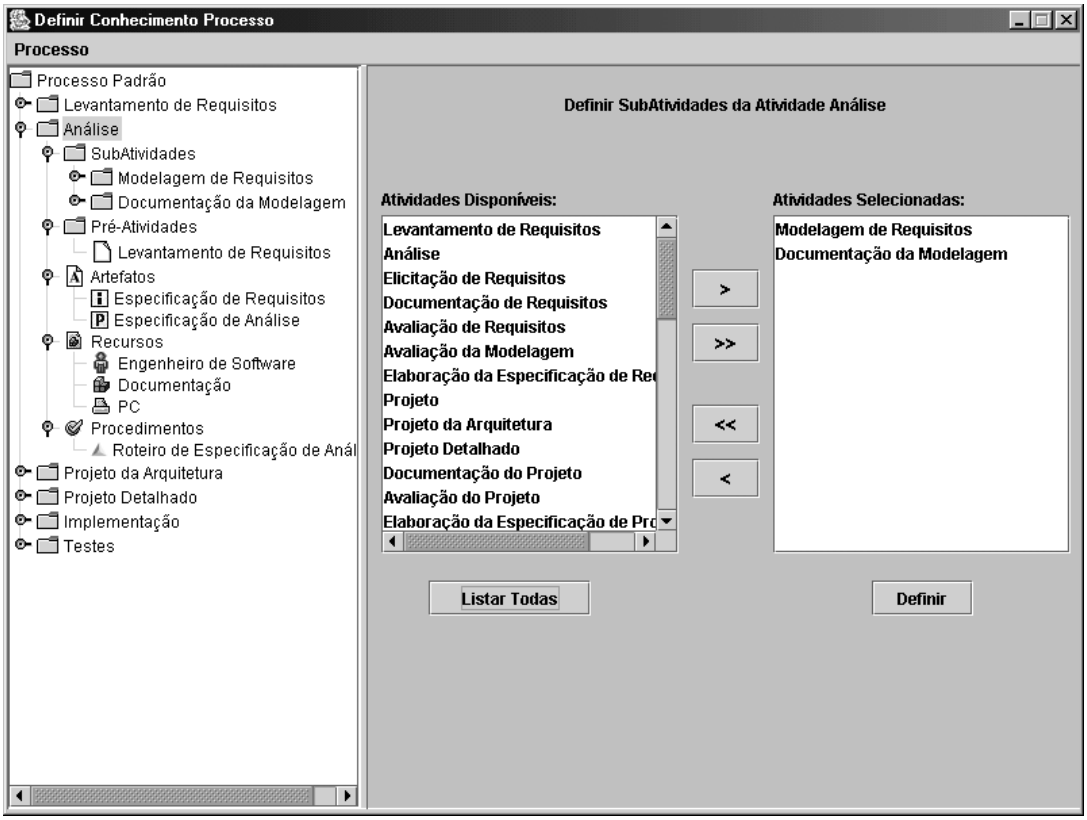

Figura 4 – Definição de processos padrão e especializado

Na parte direita da figura encontra-se um painel que possibilita a escolha dos ativos de processo para o processo padrão em definição. A lista mais à esquerda contém os ativos sugeridos pelo ambiente, com base em seu repositório de conhecimento. A lista mais à direita contém os elementos já definidos para o processo. O Gerente de Conhecimento, responsável por definir processos padrão ou especializados, pode preterir a sugestão do ambiente e pedir que todos os itens do repositório de conhecimento sejam listados.

Para se definir um processo especializado, deve-se escolher o processo padrão que se deseja especializar. Todos os elementos definidos no processo padrão são automaticamente incorporados ao novo processo. Dessa forma, o processo especializado sempre terá como base um conjunto de ativos de processo pré-definidos pelo processo padrão, podendo ser

adicionados, modificados ou excluídos elementos de acordo com as características da especialização. Porém, o processo especializado deve manter uma conformidade com o processo padrão, não sendo possível excluir atividades obrigatórias.

Após encerrar a definição do processo padrão ou especializado, é preciso definir os modelos de ciclo de vida adequados para o processo. Isso é necessário para permitir que os modelos de ciclo de vida estejam em conformidade com a precedência entre atividades definida no processo padrão ou especializado. Os modelos de ciclo de vida são representados pela classe *ConhecimentoModeloCicloVida* na figura 3.

Um modelo de ciclo de vida é definido com base nas macro-atividades (atividades que não fazem parte de outras atividades) do processo. As macro-atividades são agrupadas em combinações de atividades, que podem ser seqüenciais ou iterativas. As combinações iterativas podem ter mais de um ciclo de execução. Para exemplificar, suponha que um processo padrão possua as seguintes macro-atividades: Levantamento de Requisitos, Análise, Projeto da Arquitetura, Projeto Detalhado, Implementação e Testes. Pode-se definir um modelo de ciclo de vida incremental com duas combinações de atividades: a primeira seqüencial, incluindo as atividades de Levantamento de Requisitos, Análise e Projeto da Arquitetura; a segunda iterativa, consistindo de Projeto Detalhado, Implementação e Testes.

É importante ressaltar que a ordem das atividades em uma combinação deve estar coerente com a ordem de precedência das macro-atividades definidas no processo padrão ou especializado. Suponha que, no exemplo anterior, o processo padrão defina que a atividade Implementação é uma pré-atividade para a atividade Teste. Na definição de um modelo de ciclo de vida para este processo, a atividade Teste não poderia aparecer em uma combinação antes da atividade Implementação, nem fazer parte de qualquer combinação de atividades anterior a essa.

A figura 5 apresenta a definição de modelos de ciclo de vida de um processo padrão ou especializado em ODE. Inicialmente, o Gerente de Conhecimento deve definir o número de combinações de atividades necessário para descrever o modelo de ciclo de vida. Para cada combinação, deve-se informar o seu tipo (seqüencial ou iterativa) e o número de atividades que a compõem. Finalmente, as atividades de cada combinação devem ser selecionadas dentre as macro-atividades do processo padrão ou especializado para o qual o modelo de ciclo de vida está sendo definido.

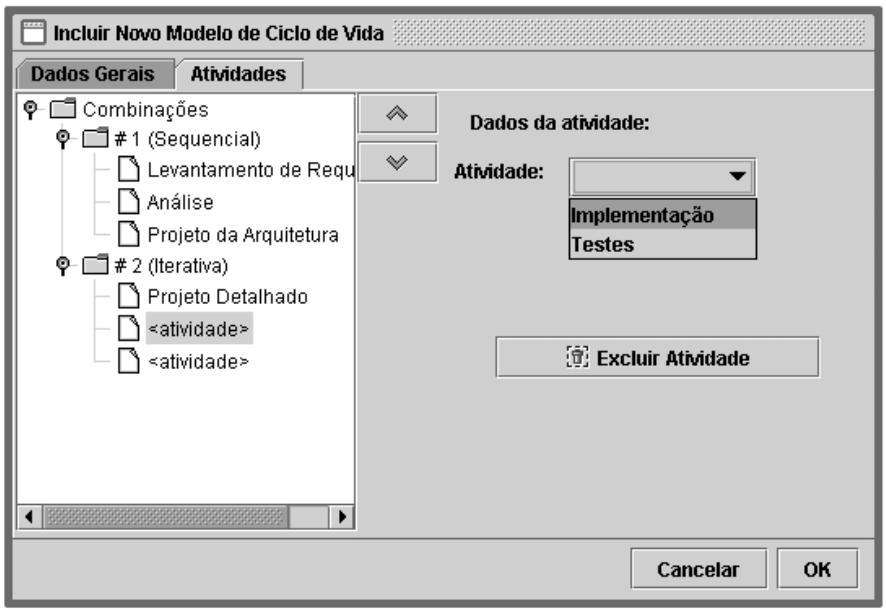

Figura 5 - Definição de Modelo de ciclo de vida

4.2 – Definição de Processos de Projeto em ODE

Definidos os modelos de ciclo de vida para um processo padrão ou especializado, estes podem ser utilizados na definição de processos de projeto. Entretanto, ODE oferece duas modalidades de apoio à definição de processos de projeto: definição de processos com base no repositório de conhecimento do ambiente e definição de processos a partir de processos padrão/especializados. De fato, ambas as modalidades apóiam-se nos repositórios de conhecimento de ODE. Contudo, a abordagem adotada é bastante diferente.

Uma vez que muitas organizações não têm processos padrão de software definidos, a primeira modalidade de definição de processo de projeto em ODE não requer um processo padrão definido. Neste caso, o Gerente de Projeto tem de fazer todo o trabalho, sem ter um modelo de processo como base. Ele deve escolher um modelo de ciclo de vida geral, isto é, um modelo que não foi definido para um processo padrão específico, e a partir dele definir novas atividade, subatividades, artefatos, recursos e procedimentos [4].

No segundo caso, um processo padrão/especializado é selecionado. A seguir, o Gerente de Projeto deve selecionar, dentre os modelos de ciclo de vida definidos para o processo selecionado, o modelo de ciclo de vida que deseja adotar no projeto em questão. Caso o modelo de ciclo de vida selecionado possua alguma combinação iterativa, deve ser informado o número de iterações. Neste momento, o ambiente gera um processo inicial de projeto contendo as atividades definidas pelo processo padrão, bem como os demais ativos de processo definidos. A partir daí, o Gerente de Projeto pode fazer as definições específicas do projeto, como mostra a figura 6.

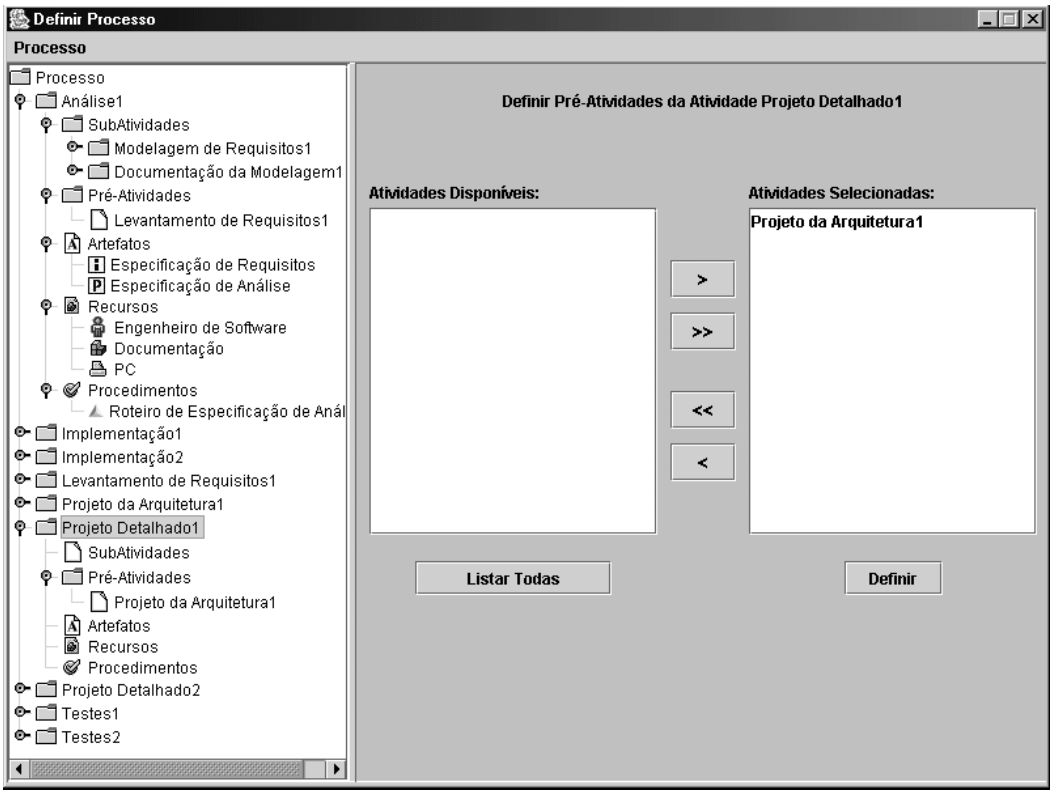

Figura 6 –Definição de processos de projeto

Deve-se ressaltar que, ao contrário da definição de processos padrão e especializados, que ocorre no meta-nível da arquitetura de ODE, a definição de processos de projeto ocorre no nível base, mais especificamente no pacote *Controle*, mostrado na figura 7.

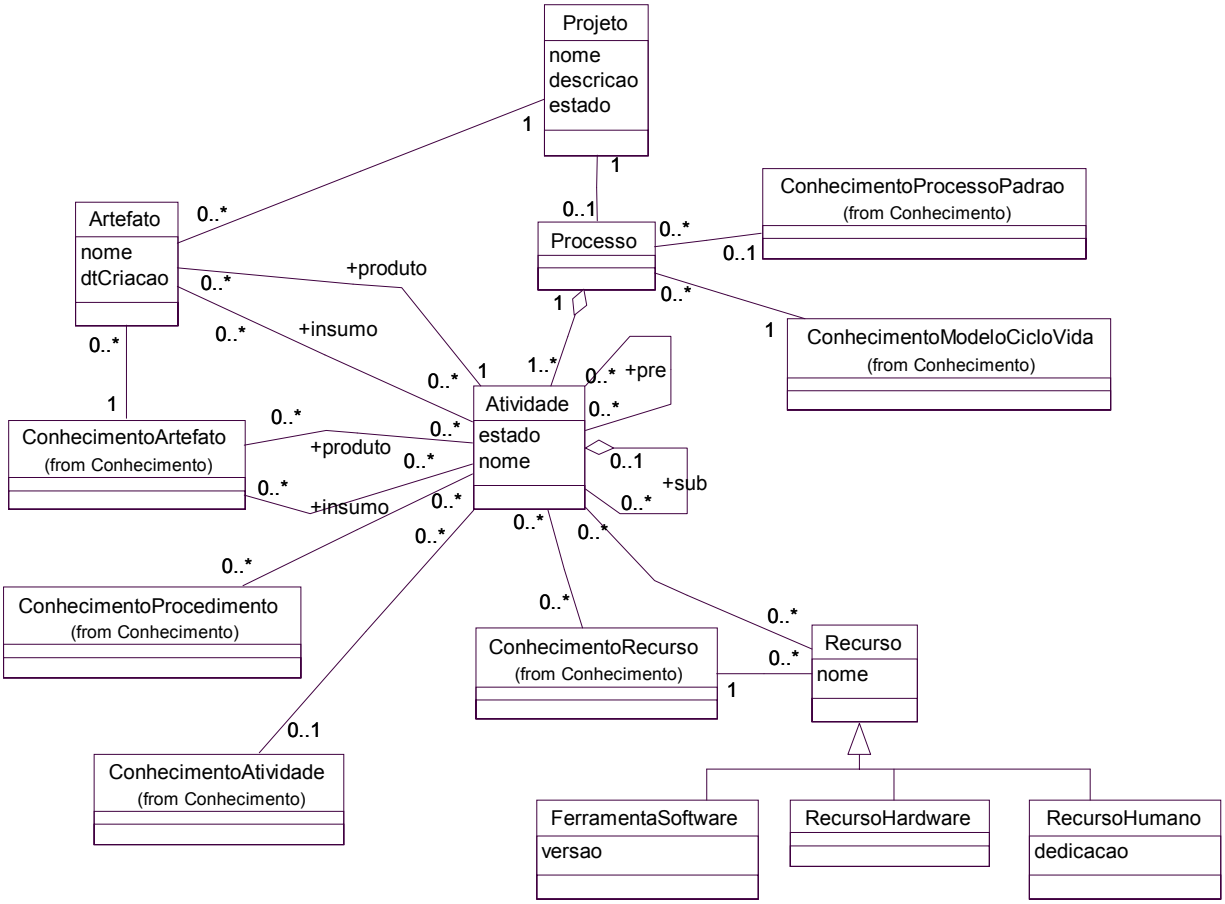

Figura 7 – O Pacote Controle

A classe *Processo* representa um processo de projeto, definido para um projeto de software específico (representado pela classe *Projeto*). Um processo de projeto pode ser definido tendo por base um processo padrão ou especializado, ou, então, ser definido a partir do repositório de conhecimento do ambiente, conforme anteriormente mencionado. Por isso, um *Processo* pode ter, ou não, uma associação com *ConhecimentoProcessoPadrao*.

O primeiro passo para a definição de um processo de projeto é a escolha do seu modelo de ciclo de vida. Para cada atividade do modelo de ciclo de vida escolhido, é criada uma ou mais instâncias da classe *Atividade*, dependendo do número de iterações definido para as suas combinações iterativas.

Quando se define um processo de projeto a partir de um processo padrão ou especializado, todos os seus ativos são replicados para o novo processo, de acordo com o processo padrão/especializado e o modelo de ciclo de vida escolhidos, não sendo possível excluir as atividades definidas como obrigatórias no processo padrão ou especializado.

A classe *Atividade* possui uma associação com a classe *ConhecimentoAtividade*, que representa o conhecimento referente à instância de uma atividade. Esse conhecimento será utilizado para se definir as subatividades, pré-atividades, insumos, produtos, recursos e procedimentos dessa atividade do processo. Porém, pode ser criada uma atividade específica para o processo em definição e que não possui um conhecimento prévio no ambiente. Essa é uma oportunidade para se trabalhar a melhoria de processos em ODE.

Durante a definição de um processo de projeto, é alocado um determinado tipo de recurso para uma atividade, assim como estabelecido um tipo de insumo e produto para essa atividade (representados, respectivamente, pelas classes *ConhecimentoRecurso* e

ConhecimentoArtefato). Por exemplo, o Gerente de Projeto define que precisará de um Analista para executar a atividade de Análise de Requisitos. "Analista" é um tipo de recurso humano (e, portanto, uma instância de *ConhecimentoRecurso*). Somente em um segundo momento, durante a alocação de recursos, é que se aloca uma pessoa (instância da classe *Recurso*) para a atividade. Essa pessoa deve exercer a função de Analista. Os procedimentos necessários para a execução das atividades são representados pela a associação com a classe *ConhecimentoProcedimento*.

A janela de definição de processos de projeto, mostrada na figura 6, é similar à janela de definição de processos padrão/especializados, apresentada na figura 4, e segue o mesmo princípio, exibindo os elementos do processo já definidos na árvore no lado esquerdo da janela e possibilitando a escolha dos elementos no painel para seleção, com base no repositório de conhecimento do ambiente.

5 – Trabalhos Correlatos

Há vários trabalhos relatados na literatura que tratam do apoio automatizado à definição de processos de software, alguns deles realizados no contexto de ambientes de desenvolvimento de software centrados em processo.

Borges et al. [15] desenvolveram ProKnowHow, uma ferramenta de apoio à instanciação de processos de software com gerência de conhecimento. ProKnowHow permite a definição do processo padrão da organização e apóia a instanciação de processos de projeto a partir dele. Além disso, faz parte do repositório de conhecimento de ProKnowHow lições aprendidas sobre a instanciação e uso do processo padrão, que podem ser utilizadas para permitir o compartilhamento de experiências sobre instanciação de processos na organização. ProKnowHow, no entanto, não apóia a definição de processos padrão ou especializados. A ferramenta de ODE também oferece apoio à definição de processos com base em um repositório de conhecimento, entretanto, apenas instâncias da ontologia fazem parte deste repositório e ainda não estão disponíveis serviços de gerência de conhecimento mais elaborados.

No contexto da Estação TABA, há duas ferramentas de apoio à definição de processos de software: Assist-Pro [5] e Def-Pro [12]. Assist-Pro é um assistente inteligente para apoiar a definição de processos de projeto, cuja base de conhecimento é definida com base na ontologia de processo de software definida em [5] e também utilizada neste trabalho. Assist-Pro, no entanto, não considera a definição de processos padrão e especializados.

Machado [12] desenvolveu Def-Pro, uma ferramenta de apoio à definição de processos de software padrão para uma organização, com base na norma ISO/IEC 12207 [6], modelos de maturidade (por exemplo, CMM [2]), características da organização e no tipo de ADS para o qual está sendo definido o processo. Def-Pro, no entanto, não considera a definição de modelos de ciclo de vida adequados para processos padrão, o que pode vir a ser um problema na instanciação de processos de projeto.

6 - Conclusões e Trabalhos Futuros

Este trabalho teve como foco um importante segmento da área de Qualidade de Software, no qual vem sendo desenvolvidos diversos estudos: Definição de Processos de Desenvolvimento de Software. Essa definição é essencial para a construção de software, pois, a partir de um processo de desenvolvimento bem definido, é mais fácil desenvolver produtos de software de qualidade, que atendam aos requisitos especificados.

Hoje, existem diversas normas e modelos de qualidade que exigem das organizações

desenvolvedoras de software a definição de processos padrão de desenvolvimento. Esses processos devem conter atividades de garantam a qualidade do produto construído, tais como atividades de verificação, validação, revisão e análise crítica.

Na abordagem utilizada, optou-se por um modelo de definição de processos em níveis, pois a definição de processos de projeto não precisa "partir do zero". Ao contrário, pode partir de um processo padrão, cujos elementos básicos já estão definidos.

A automatização da definição de processo abre espaço para uma abordagem de melhoria contínua de processos de software, com base na captura de experiências e informações de projetos que utilizam o processo padrão como base para a definição do seu processo. Assim sendo, a perspectiva de aperfeiçoamento deste trabalho vai na direção da melhoria de processos com base em gerência de conhecimento. Oportunidades de melhoria, relatos de sucesso, melhores práticas e lições aprendidas devem ser capturados, analisados e, se pertinentes, seus resultados incorporados aos processos da organização.

Finalmente, encontra-se em estudo a implantação desta ferramenta em uma organização de desenvolvimento de software que se encontra em processo de certificação segundo a ISO 9001 [7]. Espera-se que, uma vez definido o processo padrão dessa organização, processos especializados possam ser definidos e que, a partir do conjunto de processos da organização (padrão e especializados), a ferramenta possa ser usada para apoiar a definição de processos de projeto.

Agradecimentos

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

Referências Bibliográficas

[1] Fuggetta, A. *Software Process: A Roadmap*, In: Proc. of The Future of Software Engineering, ICSE'2000, Limerick, Ireland, 2000.

[2] Fiorini, S. T., Von Staa, A., Baptista, R. M. *Engenharia de Software com CMM.* Brasport, 1998.

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

[4] 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 Simpósio Brasileiro de Engenharia de Software, Caderno de Ferramentas, Gramado, Outubro, 2002.

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

[6] NBR ISO/IEC 12207: 1995. *Tecnologia da Informação – processos de ciclo de vida de software*, 1995.

[7] ISO 9001:2000. *Quality management systems.* 2000.

[8] Harrison, W.; Ossher, H.; Tarr, P. *Software Engineering Tools and Environments: A Roadmap*, In: Proc. of The Future of Software Engineering, ICSE'2000, Limerick, Ireland, 2000.

[9] Pressman, R.S., *Software Engineering: A Practitioner's Approach*, 5th Edition, New York: McGraw-Hill, 2001.

[10] Pfleeger, S.L., *Software Engineering: Theory and Practice*, 2nd Edition, New Jersey: Prentice Hall, 2001.

[11] Christie, A.M., *Software process automation: the technology and its adoption*. Pittsburgh, 1995.

[12] Machado, L.F.D.C., Santos, G., Oliveira, K.M., Rocha, A.R.C., *Def-Pro: Uma ferramenta para apoiar a definição do processo padrão*. Anais da XI Conferência Internacional de Tecnologia de Software – XI CITS, Curitiba, 2000.

[13] Guarino, N., *Formal Ontology and Information Systems.* Proceedings of the First Int. Conference on Formal Ontology in Information Systems, Trento, Italy, June 1998.

[14] Falbo, R.A.; Guizzardi, G.; Natali, A.C.C.; Bertollo, G.; Ruy, F.B.; Mian, P.G. *Towards Semantic Software Engineering Environments*. In: Proc. of the 14th International Conference on Software Engineering and Knowledge Engineering, SEKE'02, Ischia, Italy, 2002.

[15] Borges, L.S., Falbo, R. A., *Uma Ferramenta de Apoio à Instanciação de Processos de Software com Gerência de Conhecimento*, Anais do I Simpósio Brasileiro de Qualidade de Software, Gramado, Brasil. Outubro 2002.