PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA

ELIANA BEATRIZ PEREIRA

UMA PROPOSTA PARA ADAPTAÇÃO DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE BASEADOS NO *RATIONAL UNIFIED PROCESS*

PORTO ALEGRE
2005

ELIANA BEATRIZ PEREIRA

UMA PROPOSTA PARA ADAPTAÇÃO DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE BASEADOS NO *RATIONAL UNIFIED PROCESS*

Dissertação apresentada como requisito para obtenção do grau de Mestre, pelo Programa de Pós-Graduação da Faculdade de Informática da Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Dr. Ricardo Melo Bastos

PORTO ALEGRE
2005

Dados Internacionais de Catalogação na Publicação (CIP)

P436p Pereira, Eliana Beatriz
Uma proposta para adaptação de processo de desenvolvimento de software baseados no Rational Unified Process / Eliana Beatriz Pereira. - Porto Alegre, 2005.
125 f. : il.

Dissertação (Mestrado) - Fac. de Informática, PUCRS, 2005.

1. Engenharia de Software. 2. Programas - Projetos. I. Título.

CDD 005.1

Ficha Catalográfica elaborada pelo Setor de Processamento Técnico da BC-PUCRS

PUCRS
Campus Central
Av. Ipiranga, 6681 - prédio 16 - CEP 90619-900
Porto Alegre - RS - Brasil
Fone: +55 (51) 3320-3544 - Fax: +55 (51) 3320-3548
Email: bceadm@pucrs.br
www.pucrs.br/biblioteca

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
Reconhecido pelo Parecer No. 930/98.C.N.E. Homologação Publicada no D.O.U. de 30/12/98.

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "***Uma Proposta Para Adaptação de Processos de Desenvolvimento de Software Baseados no Rational Unified Process***", apresentada por Eliana Beatriz Pereira, como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Sistemas de Informação, aprovada em 19/12/2005 pela Comissão Examinadora:

Prof. Dr. Ricardo Melo Bastos – Orientador — PPGCC/PUCRS

Prof. Dr. Marcelo Blois Ribeiro — PPGCC/PUCRS

Prof. Dr. Toacy Cavalcante de Oliveira — PPGCC/PUCRS

Prof. Dr. Arndt Von Staa — PUCRJ

Homologada em 02/04/2007, conforme Ata No. 008 pela Comissão Coordenadora.

Prof. Dr. Fernando Luís Dotti
Coordenador.

AV. IPIRANGA, 6681 - PRÉDIO 16/SALA 106
90619-900 - PORTO ALEGRE -RS -BRASIL
TELEFONE: 55 51 33203611 FAX 55 51 3320.3621
http: //www.inf.pucrs.br e-mail: ppgcc@inf.pucrs.br

“É melhor tentar e falhar que preocupar-se e ver a vida passar.
É melhor tentar, ainda que em vão, que sentar-se fazendo nada até o final.
Eu prefiro na chuva caminhar que em casa me esconder.
Prefiro ser feliz embora louco, que em conformidade viver...”
(Martin Luther King)

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus, minha fonte de inspiração e luz, por mais esta etapa cumprida em minha vida.

Aos meus pais, Arnaldo e Flávia pelo exemplo de vida, dedicação e amor. Obrigado por sempre me incentivarem a seguir em frente. Esta conquista, sem dúvida, também é de vocês!

À minhas irmãs, Márcia e Patrícia, pelo carinho e amor. Vocês contribuíram muito para esta vitória!

Aos meus tios, Paulo e Maria Tereza que me acolheram em sua casa durante estes anos tão importantes de minha vida. As palavras de carinho e incentivo de vocês foram muito importantes para amenizar minha ansiedade diante dos obstáculos. Vocês contribuíram muito para meu crescimento pessoal.

Ao meu orientador, Ricardo Bastos, por acreditar no meu trabalho. Obrigada pela compreensão nos momentos difíceis desta caminhada. Aprendi contigo mais que questões técnicas, recebi lições de dedicação e caráter que jamais esquecerei.

A Giovanni Balsini por sempre incentivar e acreditar o meu trabalho. Obrigado pelo seu amor e compreensão, tornando esta caminhada menos difícil.

À minha grande amiga Elisa Cerri e Cerri, que conheci durante o mestrado. Você dividiu comigo muitos momentos bons e ruins. Compartilhou experiências novas e foi uma ótima parceria para passar noites em claro, seja nas festas tomando umas cervas ou no CDPe realizando esta pesquisa.

Ao meu bolsista e amigo Rafael de Toledo Brito. Sua participação neste trabalho foi imprescíndivel. Obrigada pelo apoio e dedicação. Considero esta vitória sua também!

Aos amigos Túlio Baségio, Fabiana Rocha, Ana Paula Lemke, Fernanda Luna, Mariana Denes, Semadar Marques e Luciano Biasi pelo apoio, torcida e profunda amizade.

Aos professores Dr. Toacy Cavalcanti e Dr. Jorge Audy pelo acompanhamento do trabalho com sugestões e críticas, sempre procurando contribuir.

Ao Convênio Dell/PUCRS pelo apoio financeiro para realização deste trabalho.

RESUMO

O estabelecimento de um processo padrão nas organizações é um requisito essencial para aquelas que desejam tornarem-se competitivas em termos de qualidade e produtividade. Entretanto, é importante considerar que processos irão variar de uma organização para outra e, na maioria das vezes, até mesmo entre diferentes projetos de uma mesma organização. Isto indica claramente a necessidade prévia que as organizações de software têm em definir mecanismos para a adaptação de seus processos padrão, permitindo que as necessidades de seus projetos sejam devidamente atendidas. Entretanto, os desafios enfrentados pelas organizações na definição destes mecanismos são significativos, pois esta é uma tarefa não-trivial. Neste sentido, este trabalho tem como principal objetivo propor um meta-modelo para descrição do conjunto de elementos e relacionamentos necessários para adaptação de processos de desenvolvimento de software baseado no modelo de processo RUP. Ainda, um conjunto de assertivas estruturais para adaptação de processos é proposto a partir do meta-modelo, a fim de estabelecer procedimentos que devem ser realizados quando uma organização adapta seu processo padrão para um projeto específico. A contribuição principal das assertivas é auxiliar a adaptação de processos de software a partir do meta-modelo proposto, visando manter a conformidade com o processo padrão.

Palavra-Chave: Engenharia de Software, Processo de Desenvolvimento de Software, Processo de Adaptação, Processo Específico de Projeto.

ABSTRACT

The establishment of a standard process in organizations is an essential requirement for those want to become competitive in terms of quality and productivity. However, it is important to consider that processes change from one organization to another. Also, sometimes these change even among projects at the same organization. This indicates that software organizations need to define mechanisms for tailoring their standard processes, fulfilling the needs of their projects. However, the challenges faced by organizations during the definition of such mechanisms are not trivial. In this context, this work presents a metamodel describing a set of elements and relationships required to ensure tailoring of standard software development process based on the RUP process model. Also, a set of structural assertions for the process tailoring is proposed from a metamodel, in order to establish how the elements and relationships of the standard software process must be treated during tailoring. The main contribution of these assertions is helping software development organizations keeping the consistency and conformity of the tailored software process with the standard software process of the organization.

Key-words: Software Engineering, Standard Software Development Process, Process Tailoring, Project Specific Process.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CMMI	*Capability Maturity Model Integration*
GV-Model	*German Process Model*
IBM	*International Business Machines*
ISO	*International Organization for Standardization*
MAPS	Modelo Para Adaptação de Processos de Software
MOF	*Meta-Object Facility*
OMG	*Object Management Group*
OPEN	*Object-oriented Process, Environment and Notation*
PROCEPT	*Process Programming and Testing*
PROLOG	*Programming Logic*
RPW	*RUP Process Workbench*
RUP	*Rational Unified Process*
SPEM	*Process Engineering Metamodel Specification*
SW-CMM	*Software Capability Maturity Model*
UML	*Unified Model Language*
XP	*Extreme Programming*

SUMÁRIO

1. INTRODUÇÃO

Com a crescente demanda por tecnologia de informação, são crescentes também os esforços para garantir a qualidade dos produtos de software. Neste sentido, muitos estudos vêm sendo realizados e indicam cada vez mais que a qualidade do produto de software está fortemente relacionada com a qualidade do processo utilizado na sua construção (Fuggetta, 2000), (Welzel *et al*, 1995) e (Yoon *et al*, 2001).

Neste sentido, as organizações de software estão cada vez mais interessadas em utilizar processos de software bem definidos para o desenvolvimento de seus produtos. Atualmente, a implantação de um processo padrão de desenvolvimento de software nas organizações tem sido fortemente demandando pela indústria do setor e vários processos sugeridos por organizações e companhias internacionais (Yoon *et al*, 2001). Segundo Jalote (2002), algumas opiniões diferem na natureza e formalidade do processo de software, mas existe um consenso geral que o seu uso é extremamente importante para organizações que procuram aumento de produtividade e melhoria na qualidade de seus produtos.

Entretanto, a criação de um processo padrão de desenvolvimento de software não garante melhoria na qualidade, pois na maioria das vezes a utilização do mesmo processo para todos os tipos de projeto não é possível, isto porque, diferentes projetos apresentam necessidades diversas e específicas.

A maneira de fornecer flexibilidade aos projetos sem deixar de padronizar acontece a partir da adaptação do processo padrão de desenvolvimento de software. Adaptar um processo padrão consiste em adicionar, excluir ou modificar alguns de seus elementos e relacionamentos, sendo o processo resultante mais apropriado para o alcance das metas do projeto (Yoon *et al*, 2001) e (Jalote, 2002). Assim, durante um processo de adaptação, a organização deve permitir um conjunto de derivações a partir de seu processo padrão, tornando possível seu ajuste a necessidades específicas de projetos ou negócios em particular.

Contudo, é importante considerar que permitir essas derivações a partir do processo padrão sem controle efetivamente implica que nenhum padrão existe (Jalote, 2002). Por isso, as organizações devem estar preparadas para fornecer algumas regras a ser seguidas durante o processo de adaptação. Elas devem garantir que a conformidade do processo padrão de desenvolvimento de software seja mantida, o que exige um esforço considerável por parte das

organizações. Esse esforço diz respeito principalmente a conhecer todos os elementos que constituem o processo padrão e os principais relacionamentos existentes entre eles, já que em um procedimento de adaptação tais elementos e relacionamentos vão sofrer alterações.

Partindo desse contexto, este trabalho envolve a definição de um meta-modelo para descrição do conjunto de elementos e relacionamentos necessários para adaptação de processos de desenvolvimento de software. O objetivo é tornar possível a identificação das relações de dependências entre os elementos do processo, permitindo a definição de algumas assertivas a serem aplicadas na adaptação do processo de modo a garantir a consistência do processo resultante.

Para a definição do meta-modelo, inicialmente uma pesquisa na literatura foi realizada visando a identificação de elementos e relacionamentos que constituem um processo padrão de desenvolvimento de software, bem como estes devem ser tratados durante o processo de adaptação. A partir disso, como forma de generalizar os resultados encontrados, realizou-se um estudo nos meta-modelos de processo RUP e SPEM, sendo proposta uma extensão ao meta-modelo RUP com a inclusão de novos elementos e relacionamentos para suporte ao processo de adaptação.

A motivação para escolha do RUP frente ao meta-modelo SPEM foi devido a sua relevância como processo de engenharia de software e sua ampla utilização no mercado. Além disso, outro fator determinante para sua seleção foi o fato de possuir uma extensa definição de processo associada, o que no contexto desta pesquisa é fundamentalmente importante para uso e avaliação do meta-modelo proposto.

Com base nesse meta-modelo, um conjunto de assertivas estruturais para adaptação de processos foi desenvolvido estabelecendo procedimentos que devem ser realizados quando uma organização adapta seu processo padrão para um projeto específico. A contribuição principal das assertivas é auxiliar a adaptação de processos de software a partir do meta-modelo proposto, visando manter conformidade com o processo padrão. Por fim, um protótipo de software também implementado no contexto desta pesquisa é apresentado.

1.1. Objetivos

O estudo realizado neste trabalho tem como objetivo geral desenvolver um meta-modelo e assertivas estruturais para a apoio a processos de adaptação. Como objetivos específicos têm-se:

- Aprofundar o estudo teórico sobre processos de adaptação;
- Identificar quais são os elementos e relacionamentos de um processo padrão de desenvolvimento de software utilizando como base o processo RUP;
- Analisar os relacionamentos existentes entre os elementos que constituem o processo RUP para identificação de fatores envolvidos no processo de adaptação e, a partir disso, identificar quais relacionamentos devem ser considerados durante um processo de adaptação;
- Propor a partir dos elementos e relacionamentos identificados uma extensão ao meta-modelo RUP para suporte a adaptação de processos;
- Definir um conjunto de assertivas estruturais para adaptação de processos de desenvolvimento de software a partir do meta-modelo proposto;
- Especificar uma ferramenta de software e desenvolver um protótipo que possibilite o uso do meta-modelo e assertivas estruturais propostos;
- Usar o meta-modelo e assertivas estruturais propostos a partir do protótipo desenvolvido e avaliar os resultados.

1.2. Etapas da Pesquisa

O trabalho apresentado neste documento foi realizado por meio de várias etapas, as quais estão representadas pela Figura 1 e são descritas a seguir:

Etapa 1: nessa etapa realizou-se um estudo sobre o processo RUP no qual se estabeleceram informações sobre os elementos que o constituem e seus possíveis relacionamentos, relevantes para a construção do processo de adaptação. Ainda nessa etapa, realizou-se levantamento bibliográfico e estudo do referencial teórico que permitiu aprofundar os conhecimentos sobre processos de adaptação.

Etapa 2: aqui analisaram-se os resultados da etapa 1 para a identificação dos elementos de um processo padrão de desenvolvimento de software e fatores envolvidos na construção de processos de adaptação. Baseado nisso, uma extensão ao meta-modelo RUP foi proposta com a inclusão de novos elementos para suporte ao processo de adaptação. Ainda nessa fase, um conjunto de assertivas estruturais para adaptação de processos baseado no meta-modelo proposto foi desenvolvido.

Etapa 3: nessa etapa uma ferramenta para uso do meta-modelo e assertivas estruturais propostos foi especificada e um protótipo desenvolvido.

Etapa 4: a fase final constitui o uso do meta-modelo e assertivas estruturais propostos por intermédio do protótipo desenvolvido. Nessa fase, o RUP foi utilizado como processo padrão a ser adaptado. Algumas de suas disciplinas foram cadastradas no protótipo e simulações foram realizadas a fim de avaliar as assertivas estruturais para inclusão e exclusão de atividades.

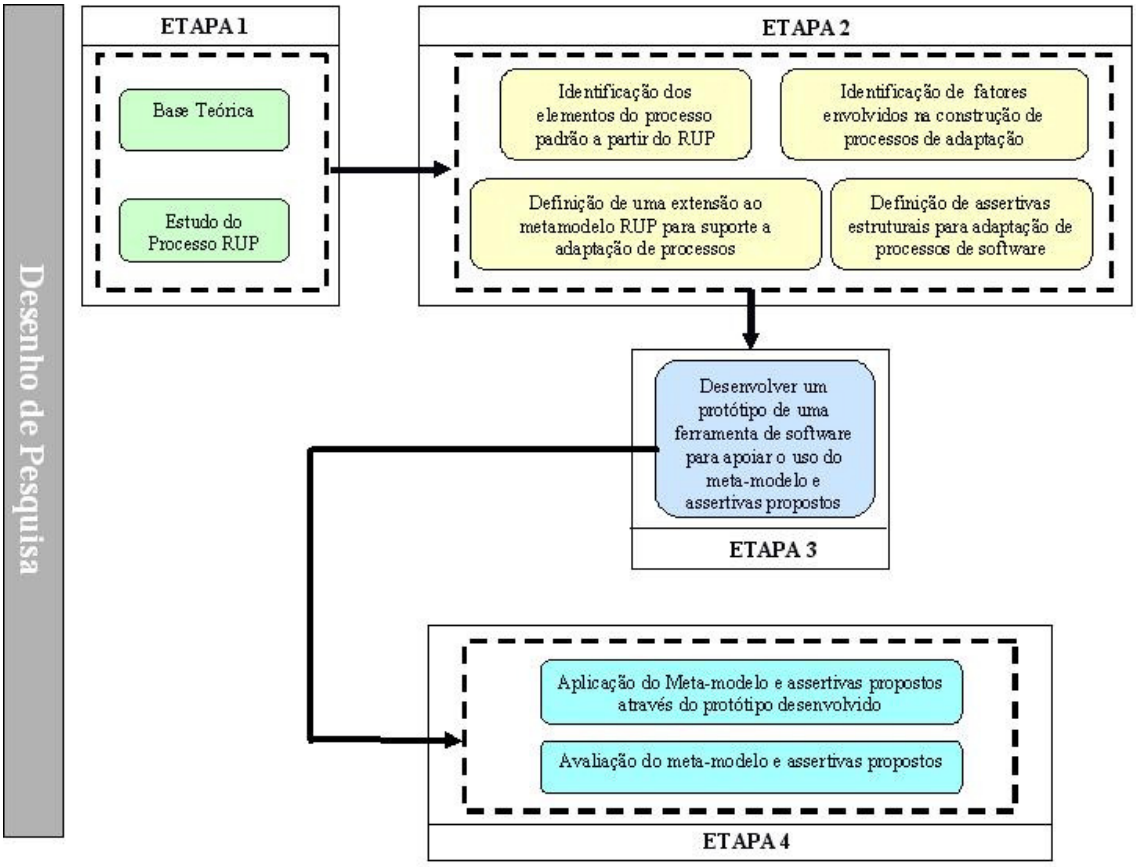

Figura 1 - Desenho de pesquisa proposto

1.3. Estrutura do Trabalho

O restante desse documento está estruturado da seguinte forma: o capítulo 2 fornece uma visão geral sobre processos de desenvolvimento de software e adaptação de processos. Ainda, descreve resumidamente como é tratada a adaptação de processos em algumas normas e modelos de maturidade.

No capítulo 3 apresentam-se o processo de desenvolvimento RUP e o meta-modelo para a definição de processos SPEM. A descrição é realizada em termos de quais elementos e relacionamentos os constituem, bem como qual a relação desses com o processo de adaptação.

Ainda, nesse capítulo é realizada uma análise de conformidade do processo RUP para com o meta-modelo SPEM.

No capítulo 4 são apresentados os principais trabalhos relacionados. Com base no estudo de tais trabalhos e dos principais processos de desenvolvimento de software (capítulo 4), a extensão ao meta-modelo RUP é apresentada no capítulo 5.

O capítulo 6 descreve detalhadamente as assertivas estruturais para adaptação de processos propostas a partir do meta-modelo apresentado no capítulo 5.

No capítulo 7 um exemplo de uso do meta-modelo e assertivas estruturais realizado a partir do protótipo é apresentado.

Finalmente, no capítulo 8 apresentam-se as considerações finais sobre o tema e enfocam-se os aspectos relacionados às contribuições e limitações deste estudo. Concluí-se destacando rumos para futuras pesquisas na área.

2. PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Esse capítulo trata alguns de conceitos relacionados a processos de desenvolvimento de software. Aspectos referentes à criação de um processo padrão de desenvolvimento de software nas organizações são apresentados com o objetivo de demonstrar a importância de sua utilização. Por fim, são descritos resumidamente os métodos de adaptação de processos por parte de algumas normas e modelos de maturidade, relevantes para o contexto deste trabalho.

2.1. Processo de Desenvolvimento de Software: Definição e Aspectos Relacionados

De acordo com Jacobson *et al* (2001), um processo de desenvolvimento de software é o conjunto completo de atividades necessárias para transformar requisitos de usuários em produtos de software. Como as organizações geralmente consideram o desenvolvimento de um produto de software como um projeto, então um processo pode ser considerado como uma seqüência de passos que um projeto pode seguir para desempenhar alguma tarefa.

Para Pressman (2001), o processo de desenvolvimento de software é a estrutura para as tarefas que são necessárias à construção de software com alta qualidade. Ainda, segundo Fuggetta (2000), o processo de desenvolvimento de software pode ser definido como um conjunto coerente de políticas, estruturas organizacionais, tecnologias, procedimentos e artefatos que são necessários para compreender, desenvolver e manter um produto de software.

Definir um processo de desenvolvimento de software envolve várias informações como atividades a ser desempenhadas, recursos utilizados, artefatos consumidos e gerados, dentre outras (Werner, 1996) *apud* (Machado, 2000), (Falbo, 1998). Para Derniame *et al* (1999), os principais conceitos ligados à sua modelagem são:

- **Atividade:** operação atômica ou composta, ou uma etapa de um processo. As atividades visam gerar ou modificar um conjunto de artefatos, incorporando e executando procedimentos, regras e políticas organizacionais. Algumas

atividades podem ser decompostas em sub-atividades, embora isso não seja obrigatório;

- **Artefato:** informação desenvolvida e mantida em um projeto de software. Alguns artefatos podem ser decompostos em sub-artefatos, embora isso não seja obrigatório;
- **Direção:** são procedimentos, regras e políticas organizacionais que dirigem atividades e geralmente estão estruturados na forma de manuais;
- **Recurso:** é um fator necessário na execução de uma atividade. Os recursos devem ser divididos em: executores (agentes humanos do processo) e ferramentas (agentes computarizados que são usados tradicionalmente no desenvolvimento de software).

Outra informação importante ligada à definição de processos de software é o que se deseja alcançar a partir de sua utilização. Nesse sentido, Tyrrell (2000) define um conjunto de objetivos que os processos de software devem atender:

- **Efetividade –** processos de desenvolvimento de software devem ajudar a determinar as necessidades do cliente e verificar se o que foi produzido satisfaz o cliente.
- **Manutenibilidade –** o processo de desenvolvimento de software deve ser capaz de expor a maneira de pensar de projetistas e programadores de forma que suas intenções sejam claras. Assim, torna-se fácil encontrar e reparar falhas no produto.
- **Previsibilidade –** o processo de desenvolvimento de software deve ajudar a predizer quanto tempo será necessário para o desenvolvimento de cada parte do produto.
- **Repetível –** produzir um processo novo para cada projeto implica em grandes gastos para a organização. Dessa forma, é importante que o processo de desenvolvimento de software seja criado de forma a poder ser reutilizado em vários projetos.
- **Qualidade –** o processo de desenvolvimento de software deve permitir engenheiros de software assegurar um produto de qualidade elevada. Para isso, o processo deve fornecer uma ligação entre os desejos de um cliente e o produto de um desenvolvedor.

- **Melhoria –** o processo de desenvolvimento de software deve ser constantemente melhorado. Dessa forma, o próprio processo deve permitir a identificação de pontos de melhoria.
- **Rastreabilidade –** o processo de desenvolvimento de software deve permitir que a gerência, os desenvolvedores e o cliente sigam o status do projeto.

Embora, como visto anteriormente, existam conceitos e objetivos bem definidos para processos de software, esses podem apresentar grande complexidade e possibilitar diversas alternativas de execução de suas atividades (Machado, 2000). Assim, para as organizações é importante que o processo de desenvolvimento de software seja padronizado de forma a garantir uma maior qualidade dos produtos de software.

2.2. Processo Padrão de Desenvolvimento de Software

Um processo padrão de desenvolvimento de software deve guiar a execução de todos os projetos de desenvolvimento dentro de uma organização. De acordo com (Borges & Falbo, 2002), o processo padrão é a definição operacional do processo básico que guia o estabelecimento de um processo comum em uma organização. Ainda, para Ginsberg & Quinn (1995), esse é o meio pelo qual a organização expressa os requisitos que todos os processos de desenvolvimento de software dos projetos devem atender. Sua implantação apresenta vantagens como:

- Redução dos problemas relacionados a treinamento, revisões e suporte de ferramentas (Humphrey, 1989) *apud* (Coelho, 2003);
- Experiências adquiridas em cada projeto podem ser incorporadas ao processo padrão, contribuindo para sua melhoria (Humphrey, 1989) *apud* (Coelho, 2003);
- Maior facilidade em medições de processo e qualidade (Humphrey, 1989) *apud* (Coelho, 2003);
- Maior facilidade de comunicação entre os membros da equipe (Xu & Ramesh, 2005);
- Melhor adequação de novos membros na equipe de projeto (Xu & Ramesh, 2005);
- Melhor desempenho, previsibilidade e confiabilidade dos processos de trabalho (Xu & Ramesh, 2005).

Além das vantagens acima, outro importante aspecto a ser considerado é que a adoção de um processo padrão torna-se ainda mais importante em organizações que desejam implantar um modelo de qualidade como as normas ISO/IEC 15504 (ISO, 1998), ISO/IEC

12007 (ISO, 1995) ou *Software Capability Maturity Model* (SW-CMM) (Paulk *et al*, 1993). Isso porque, tais modelos possuem seus esforços centrados no uso de processos padronizados e na melhoria desses processos (Fuggetta, 2000). Segundo Borges (2001), tais modelos definem um processo padrão como um ponto base a partir do qual um processo especializado poderá ser obtido de acordo com as características de um projeto de software específico.

Atualmente, devido a importância da utilização de um processo padrão, um considerável esforço tem sido devotado para sua modelagem e, como forma de auxiliar a sua criação, diversos processos tais como *Rational Unified Process* (RUP) (Kruchten, 2000), *Extreme Programming* (XP) (Beck, 1999), *Object-oriented Process, Environment and Notation* (OPEN) (OPEN, 2004), estão surgindo. Eles podem ser adotados por completo, ou permitem ainda ser adaptados de acordo com algumas características da organização. É possível também que uma organização baseie-se em mais de um desses processos para a definição de seu processo padrão.

Entretanto, é importante considerar que, independente da estratégia escolhida para a adoção do processo padrão de desenvolvimento de software, seu uso é de suma importância, já que ele é um dos maiores mecanismos para gerenciar e controlar projetos e produtos de software (Humphrey, 1991).

2.3. Adaptação do Processo Padrão de Desenvolvimento de Software

Segundo Ginsberg & Quinn (1995), o processo padrão de desenvolvimento de software de uma organização deve ser utilizado a cada novo projeto. Entretanto, conforme Rocha *et al* (2001), o processo padrão não pode servir a qualquer tipo de projeto. Questões relacionadas ao porte da empresa e à cultura organizacional, objetivos de projetos específicos, recursos disponíveis, tecnologias de desenvolvimento, conhecimento e experiência da equipe impõem características aos processos.

Atualmente, é mundialmente aceito que um processo padrão de desenvolvimento de software deve sofrer adaptações para atender a necessidades no contexto dos projetos (Fitzgerald *et al*, 2003). Isso porque cada projeto é único em termos de domínio de negócio, requisitos de cliente, tecnologia etc. (Xu & Ramesh, 2005).

Um processo de adaptação consiste em excluir, modificar ou ainda, adicionar novos elementos ao processo padrão. Para (Ginsberg & Quinn,1995), (ISO, 1995), (Welzel *et al*, 1995), (ISO, 1998), (Yoon *et al*, 2001), (Jalote, 2002), o processo de adaptação deve gerar um processo de desenvolvimento específico para o projeto cada vez que executado. Ainda, de

acordo com Kellner (1996) e Yoon *et al* (2001), o processo de adaptação pode também ser considerado como uma atividade de reuso do processo padrão.

As normas ISO/IEC 15504, ISO/IEC 12007 e *Software Capability Maturity Model* (SW-CMM) também referenciam o processo de adaptação como um importante requisito para as organizações de software.

O SW-CMM, considerado o principal modelo para processos de software, prevê a definição de um processo padrão de desenvolvimento de software em seu nível 3 de maturidade. Ainda nesse nível, ele assume que tal processo deve ser adaptado para atender necessidades específicas de projetos de software.

O processo de adaptação proposto pelo SW-CMM é baseado no uso de guias e critérios de adaptação associados ao processo padrão (Ginsberg & Quinn, 1995). Essas guias e critérios devem ser desenvolvidos pela organização e devem conter quais elementos podem ser modificados a partir do processo padrão, bem como em que condições isso pode ocorrer. Dessa forma, permite-se que elementos sejam adicionados ou excluídos do processo padrão durante a execução de projetos específicos de software. Entretanto, é importante considerar que as guias de adaptação devem ser construídas de forma a manter os benefícios das práticas comuns estabelecidas no processo padrão.

De acordo com Ginsberg & Quinn (1995), para fornecer uma guia de adaptação consistente com as informações do SW-CMM é necessário um alto conhecimento dos elementos que estão envolvidos no processo padrão de desenvolvimento de software. Elementos como trabalhadores, critérios de entrada e saída, atividades e artefatos precisam ser antecipadamente identificados para o processo de construção dos guias.

Na última versão do SW-CMM, conhecida como *Capability Maturity Model Integration* (CMMI) (Chrissis *et al*, 2003), a menção à definição e adaptação do processo padrão de desenvolvimento de software é feita da mesma forma que o modelo anterior (SW-CMM), embora agora exista uma maior ênfase à criação de um conjunto de processos padrão para a organização.

Segundo Chrissis *et al* (2003), o estabelecimento de um conjunto de processos padrão dá-se na organização para permitir que cada um desses processos atenda a necessidades de diferentes aplicações de domínio, modelos de ciclo de vida, metodologias e ferramentas, embora muitas organizações possam optar pelo estabelecimento de apenas um processo padrão para toda organização.

Ainda, no CMMI é estabelecido que as guias de adaptação devem conter requisitos obrigatórios (atividades, artefatos, entre outros) a ser mantidos em todos os processos definidos para projetos. As normas ISO/IEC 15504 (ISO, 1998) e ISO/IEC 12007 (ISO, 1995) que constituem um *framework* para avaliação de processos e uma norma para descrição dos processos de ciclo de vida do software respectivamente, definem o processo de adaptação como um processo especializado para um projeto de software. Segundo (ISO, 1995) e (ISO, 1998), as atividades necessárias para adaptar um processo padrão são:

- Identificar o ambiente do projeto: verificar quais são as características do projeto que vão influenciar o processo de adaptação. Exemplos destas características são: políticas organizacionais, procedimentos e estratégias, tamanho, criticidade da aplicação, número de pessoas envolvidas etc;
- Solicitar entradas: envolver as entradas da organização que vão ser afetadas pelas decisões do processo de adaptação. Usuários, pessoal de suporte, contratantes e licitantes podem ser considerados como entradas;
- Selecionar processos, atividades e tarefas: decidir que processos, atividades e tarefas vão ser desempenhados no projeto. Isso inclui qual documentação será desenvolvida e a distribuição de responsabilidades. Para a seleção de processos, atividades e tarefas adequadas ao projeto é importante levar em consideração as informações coletadas anteriormente sobre ambiente do projeto e entradas da organização. É permitido que atividades sejam adicionadas ou apagadas desde que as mesmas sejam especificadas no contrato do projeto;
- Documentar decisões do processo de adaptação no contrato do projeto.

De acordo com (ISO, 1995) e (ISO, 1998), é ainda importante que a conformidade com o processo padrão seja mantida. Para isso, os processos, atividades, e tarefas selecionadas por meio do processo de adaptação para um projeto devem ser executados conforme documentado em contrato.

2.4. Considerações Finais

Esse capítulo apresentou conceitos referentes a processos de desenvolvimento de software objetivando demonstrar a importância de seu uso para organizações de software. Foi mostrado como a adaptação de processos é referenciada pelos modelos de processos SW-CMM e CMMI e pelas normas ISO/IEC 15504 e ISO/IEC 12207.

Realizando um comparativo entre as abordagens estudadas identificou-se que muitas das recomendações descritas pelos modelos SW-CMM e CMMI podem ser encontradas no processo descrito pelas normas ISO/IEC 15504 e ISO/IEC 12207; atividades como mapear papéis, entradas e saídas entre outros são descritos nas abordagens. Ainda é comum tanto para ambos os modelos SW-CMM e CMMI como para as normas ISO/IEC 15504 e ISO/IEC 12007 que durante o processo de adaptação elementos do processo padrão sejam adicionados ou excluídos de acordo com necessidades específicas de projetos.

Pode-se perceber que, o SW-CMM e CMMI possuem um processo de adaptação melhor definido apresentando algumas sugestões para a construção de suas guias de adaptação. Ainda, o CMMI define que um conjunto de processos padrão pode ser criado nas organizações. Isso confirma a importância de um processo de adaptação bem definido para as organizações, já que o conjunto de tais processos pode ser definido adaptando um único processo padrão estabelecido na organização.

Entretanto, embora tenha sido possível constatar a importância do processo padrão de desenvolvimento de software e do processo de adaptação nos modelos e normas descritos nesse capítulo, identificou-se que em nenhuma das abordagens estudadas apresentou-se um conjunto de regras para adaptação de processos que contemple os relacionamentos de dependência entre os elementos do processo padrão, como por exemplo, relação entre atividades, papéis e artefatos. Esse é o principal interesse dessa pesquisa.

Neste contexto, no próximo capítulo será descrito o meta-modelo SPEM (OMG, 2005a), existente na literatura para definição de processos de desenvolvimento de software, bem como o processo de desenvolvimento de software RUP originado a partir dele. O objetivo é a identificação de elementos e relacionamentos que constituem um processo padrão de desenvolvimento de software e a tentativa de identificar referências a relacionamentos que devem ser considerados em processos de adaptação.

3. MODELOS DE PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Este capítulo fornece uma visão geral do meta-modelo SPEM, definido pela OMG como o principal template para processos de desenvolvimento de software. Ainda apresenta o RUP, considerado atualmente como um dos principais processos de desenvolvimento de software. O interesse é descrever quais elementos e relacionamentos constituem cada um deles e ainda apresentar como o processo de adaptação é abordado. Além disso, é realizada também uma análise de conformidade do processo RUP para com o meta-modelo SPEM, já que ele é proveniente deste meta-modelo.

3.1. *Software Process Engineering Metamodel Specification* - SPEM

O *Software Process Engineering Metamodel Specification* (SPEM) (OMG, 2005a) foi desenvolvido pelo *Object Management Group* (OMG) e representa um meta-modelo para a definição de processos de software e seus componentes. Atualmente, segundo OMG (2005) existem vários modelos de processo cuja definição é baseada no SPEM, tais como, RUP, DRM da Macroscope, *Global Services Method* da IBM e *QualiCycle* da Unisys.

Embora não seja considerado um modelo de processo, o SPEM será descrito neste trabalho, pois como sugere um conjunto de elementos que devem constituir um processo de desenvolvimento de software, é considerado relevante no contexto desta pesquisa. Sua importância está relacionada ao fato de ser considerado pela OMG a principal base na definição de processos de software e ainda, por tratar-se do meta-modelo que originou o modelo de processo RUP, o qual é utilizado como principal referência neste trabalho.

O SPEM encontra-se presente na arquitetura de modelagem proposta pela OMG (OMG, 2005a) conforme mostra a Figura 2. A realização de um processo em um projeto de desenvolvimento de software corresponde ao nível M0. A definição do processo associado (por exemplo, RUP) aparece no nível M1. O meta-modelo, neste caso o SPEM, aparece no nível M2 e serve de "*template*" para o nível M1. A especificação SPEM é estruturada em UML e fornece um meta-modelo completo baseado no MOF (*Meta-Object Facility*).

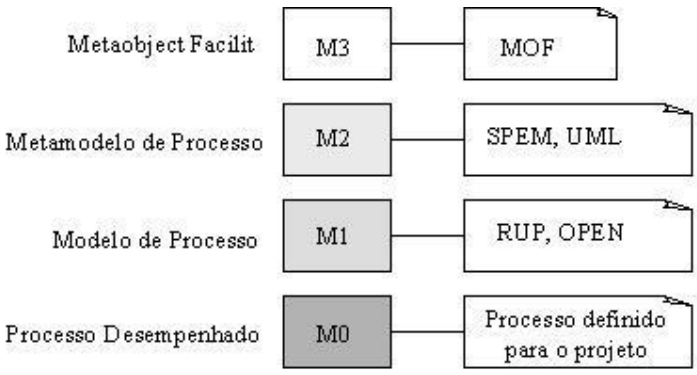

Figura 2 - Arquitetura de modelagem proposta pela OMG (adaptado de (OMG, 2005a))

A construção do meta-modelo SPEM foi realizada a partir da extensão de um subconjunto do meta-modelo da UML 1.4 chamado *SPEM_Foundation* (Figura 3). Essa extensão ao *SPEM_Foundation* é representada pelo pacote *SPEM_Extensions*, o qual adiciona os construtores e a semântica necessária à engenharia de processos de software. Os subpacotes constituintes do pacote *SPEM_Extensions* são: Elementos Básicos, Dependências, Estrutura do Processo, Componentes do Processo e Ciclo de Vida do Processo.

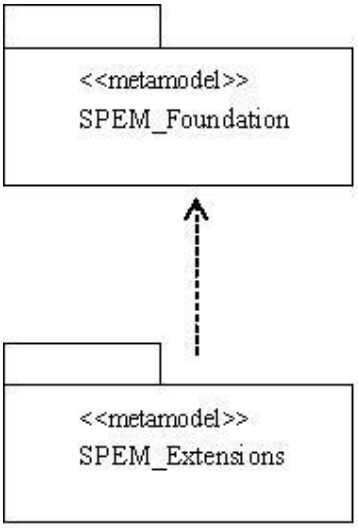

Figura 3 - Pacotes *SPEM_Foundation* e *SPEM_Extensions* (extraído de (OMG, 2005a))

A idéia principal do SPEM (Figura 4) é a de que um processo de desenvolvimento de software é uma colaboração entre entidades abstratas ativas designadas por *roles* do processo (papéis) que desempenham *activities* (atividades) em entidades concretas e tangíveis chamadas *workproducts* (produtos de trabalho). Embora essa idéia esteja centrada em *roles*,

workproducts e *activities*, outros elementos encontram-se presentes no meta-modelo. Tais elementos estão contidos nos subpacotes citados anteriormente do pacote *SPEM_Extensions*.

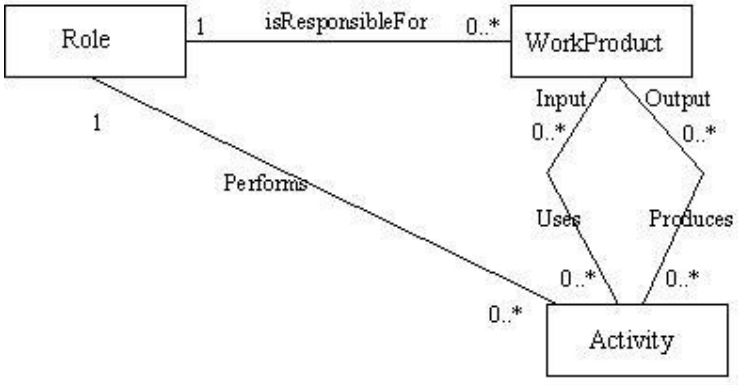

Figura 4 - Principais elementos do meta-modelo SPEM

3.1.1. Pacote Elementos Básicos

Nesse pacote encontram-se os elementos usados para a descrição do processo. Esses elementos são representados por duas metaclasses: *guidance* (orientação) e *externaldescription* (descrição). A metaclasse *guidance* representa os elementos de orientação, enquanto a metaclasse *externaldescription* os elementos de descrição. Ambos os tipos de elementos devem ser associados a outros elementos do processo. Os elementos de descrição descrevem elementos do processo e os de orientação são usados para fornecer informações detalhadas a estes elementos. Os tipos de orientação dependem da família de processo e podem ser, por exemplo: guias, técnicas, métricas, exemplos, UML *profiles*, mentores de ferramentas, *checklists* e modelos.

3.1.2. Pacote Dependências

Nesse pacote, algumas dependências definidas pelo SPEM são armazenadas. Essas dependências são utilizadas para relacionar outros elementos do meta-modelo e podem ser dos seguintes tipos:

Categorização

Fornece maneiras de associar elementos de processos a múltiplas categorias, atuando de um pacote até um elemento de processo de outro pacote. Geralmente é utilizado para categorizar todos os elementos em um alto nível em conjunção com o elemento disciplina.

Impacto

Atua entre um produto de trabalho e outro produto de trabalho para indicar que uma modificação feita em um deles pode invalidar o outro.

Importação

Demonstra que o conteúdo do pacote destino está no *namespace* do pacote de origem. Possui semântica parecida com o *Import* do UML.

Precedência

Indica dependências na forma fim-início ou fim-fim entre o trabalho descrito, atuando entre atividades para representar o conceito de seqüência.

Referência

Atua de um elemento de processo a outro, para garantir que eles pertencem ao mesmo componente de processo. Geralmente é aplicado quando o texto de um elemento de processo se refere a outro elemento, enfatizando de forma explícita na representação estrutural a relação entre os elementos.

Rastro

É utilizado principalmente para rastrear mudanças e requisitos através dos modelos, atuando entre elementos de modelo. Semanticamente parecido ao *Trace* da UML.

3.1.3. Pacote Estrutura do Processo

No pacote Estrutura do Processo (Figura 5) estão os elementos que servem de base à construção de um processo de software.

A metaclasse *workproduct* (produto de trabalho) representa algo que é produzido, consumido ou modificado em um processo. A metaclasse *workproductkind* (tipo do produto de trabalho) descreve sua categoria, ou seja, identifica se o produto de trabalho é um documento de texto, modelo UML, executável, biblioteca de classes ou outros.

Os elementos representados pela metaclasse *workdefinition* (definição de trabalho) são tipos de operações que descrevem os trabalhos executados no processo. Sua principal subclasse é *activity* (atividade), mas *phase* (fase), *iteration* (iteração) e *lyfecicle* (ciclo de vida) também são suas subclasses. Uma *workdefinition* pode ser instanciada para representar partes compostas de um trabalho que posteriormente serão decompostas. A metaclasse *activityparameter* (parâmetro de atividade) é utilizada para explicitar suas entradas e saídas em termos de *workproducts*. Ainda, tem-se que toda *workdefinition* deve estar sobre a responsabilidade de um *processperformer* (responsável de processo).

Os *processperformers* ficam associados a *workdefinitions*, enquanto *roles* representados pela subclasse *role* (papel) ficam associados como executores às *activities* e como responsáveis por *workproducts.*

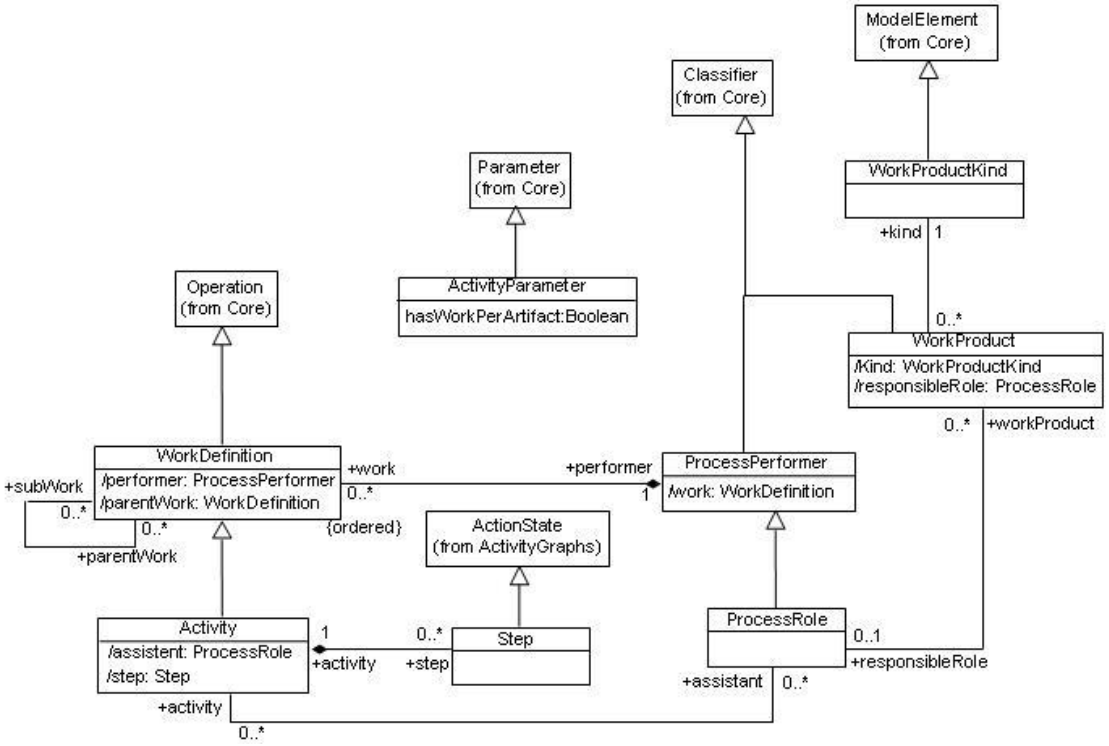

Figura 5 - Pacote Estrutura do Processo (extraído de (OMG, 2005a))

Por fim, tem-se no pacote Estrutura do Processo as *activities* (atividades) que são a principal subclasse de *workdefinition* e correspondem as tarefas, operações e ações desempenhadas por *roles*. Os elementos atômicos que constituem uma *activity* são representados pela classe *step* (passo).

3.1.4. Pacote Componentes do Processo

As classes desse pacote dividem um ou mais elementos do processo em partes "*self-contained*" que podem ser colocadas sobre gerência de configuração e controle de versão. Fazem parte do pacote Componentes do Processo (Figura 6) as classes p*ackage* (pacote), *processcomponent* (componente de processo), *process* (processo) e *discipline* (disciplina).

A classe *package* é um contâiner que pode tanto possuir como importar elementos de definição de processos, semelhante a UML (OMG, 2005b). Esse elemento e a dependência de categorização podem implementar o conceito de categoria para os elementos de processo.

Será criado um *package* para representar cada categoria e todos os elementos ligados por dependências de categorização serão membros dessa categoria. Um tipo de categorização de *activities* é implementado pelas *disciplines*.

Um *processcomponent* é uma descrição de processo internamente consistente que pode ser reutilizada por outros *processcomponents* para criar um processo mais completo.

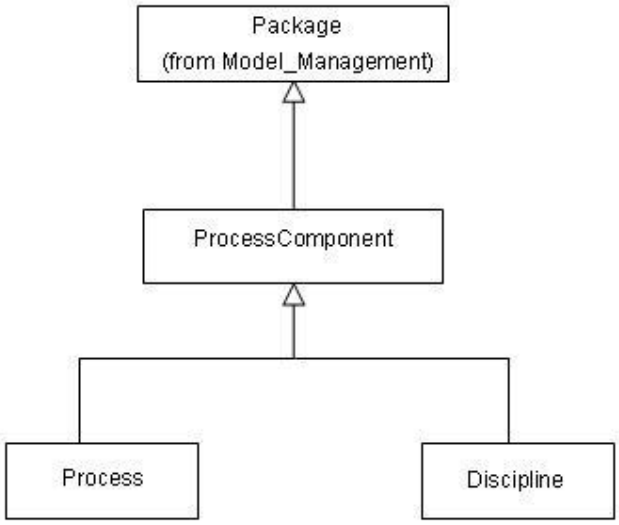

Figura 6 - Pacote Componentes do Processo (extraído de (OMG, 2005a))

A classe *process* representa um elemento que é um componente de processo completo, ou seja, não pode composto com outros componentes, o que o diferencia de um componente de processo convencional.

Discipline é uma especialização particular de *package* que agrupa as *activities* de processo de acordo com um tema comum. Esse tipo de separação implica que as instâncias das classes *guidance* e *workproduct* devem ser também categorizados por tema. A inclusão de uma *activity* em uma *discipline* é representada pela dependência de categorização e tem a restrição de que cada *activity* só pode ser categorizada por uma *discipline*.

3.1.5. Pacote Ciclo de Vida do Processo

O pacote ciclo de vida do processo contém elementos de processo que definem como um processo vai ser executado. Eles descrevem ou restringem o comportamento de um processo e são usados para assistir no planejamento, execução e monitoria de processo. São elementos deste pacote: *phase* (fase), *lifecycle* (ciclo de vida) e *iteration* (iteração).

Phase é uma especialização de *workdefinition* em que uma precondição define seu critério de entrada e seu *milestone* define o critério de saída.

Lifecycle é definido com uma seqüência de *phases* que tem um objetivo específico, definindo o comportamento de todo o processo a ser desempenhado em um projeto ou programa.

Finalmente, *iteration* é uma *workdefinition* composta que possui um *milestone* de menor porte.

Os elementos descritos acima constituem o meta-modelo completo sugerido pela OMG para definição de processos de desenvolvimento de software. Entretanto, como já dito anteriormente o SPEM aparece como um *template*, o que significa não ter nenhuma definição de processo associada a seus elementos. Isso implica não existir sugestões de atividades, papéis, disciplinas, ciclos de vida, entre outros, não possibilitando assim seu uso direto como processo de desenvolvimento de software pelas organizações. Ainda, embora apresente um conjunto de dependências consideradas importantes em processos de adaptação, não existem referências neste meta-modelo a esse assunto.

3.2. *Rational Unified Process* - RUP

O *Rational Unified Process* (RUP) é um processo de desenvolvimento de software originado a partir do meta-modelo SPEM (OMG, 2005a). Ele é considerado como um *framework* composto por vários tipos de elementos que, em conjunto, formam todo processo de desenvolvimento de software. Segundo Jabobson *et al* (2001), este *framework* pode ser adaptado e estendido de acordo com as necessidades da organização podendo ser aplicado a diferentes áreas, tipos de organização, níveis de competência e tamanho de projeto.

O RUP foi desenvolvido pela empresa *Rational Software*, como um produto de software e, como tal, é mantido, atualizado e aperfeiçoado de acordo com as experiências da *Rational* e de seus usuários (RAT, 1998). Segundo Kruchten (2000), seu processo atualmente é formado basicamente por quatro elementos primários de modelagem:

- Papéis: *quem*;
- Atividades: *como*;
- Artefatos: *o quê*;
- Fluxos: *quando*;

O *papel* é o conceito central no processo RUP. Um papel define o comportamento e as responsabilidades de um indivíduo ou grupo de indivíduos que trabalham juntos como uma

equipe. Os papéis são responsáveis pelo desempenho de atividades do processo e cada papel pode ter responsabilidades sobre certos artefatos do processo.

Uma *atividade* é uma unidade de trabalho que produz um resultado significante no contexto do projeto. A atividade tem um propósito claro, normalmente expresso em termos de criar ou modificar artefatos, como um modelo, uma classe ou um plano. Toda atividade deve ser atribuída a um papel.

Um *artefato* é um pedaço de informação que é produzida, modificada ou usada por um processo. Os artefatos são usados como entradas para desempenhar uma atividade e são o resultado ou a saída de tais atividades. No RUP, um artefato pode ser composto de outros artefatos. Por exemplo, o modelo de caso de uso possui muitos casos de uso, o modelo de projeto muitas classes, o plano de desenvolvimento de software contém vários outros planos etc. Deve-se ainda considerar que os artefatos podem ter várias formas e formatos: Modelos, Elemento de modelo, Documento, Código-fonte e Executáveis.

Os *fluxos* são seqüências de atividades que produzem um resultado de valor observável. Um fluxo pode ser expresso como um diagrama de seqüência, colaboração ou de atividade. Existem dois tipos de fluxos no RUP: *fluxos centrais* e *detalhes de fluxo*.

Os *fluxos centrais* são as disciplinas do processo e representam uma partição de todos os trabalhadores e atividades em áreas de interesse.

Cada disciplina do RUP é decomposta em *detalhes de fluxo* que nada mais são do que o agrupamento de atividades relacionadas em um fluxo de informação. Assim, uma disciplina apresenta vários detalhes de fluxo mostrando como as atividades interagem com os artefatos.

Embora os elementos acima sejam os principais elementos do RUP, deve-se considerar que seu desenvolvimento é realizado em fases que constituem um ciclo de vida, sendo estes importantes elementos do processo.

O ciclo de vida é o elemento que define o comportamento completo de um processo em um determinado projeto e é formado por um conjunto de fases. Uma fase representa uma parte do ciclo de vida e é constituída de iterações que são intervalos de tempo onde o sistema é construído. As fases do RUP são elementos seqüenciais e possuem objetivos diferentes na produção do software.

Para atender aos objetivos das fases e completar o ciclo de desenvolvimento do processo, temos uma distribuição das disciplinas com diferentes ênfases em cada uma das fases e iterações como mostra a Figura 7.

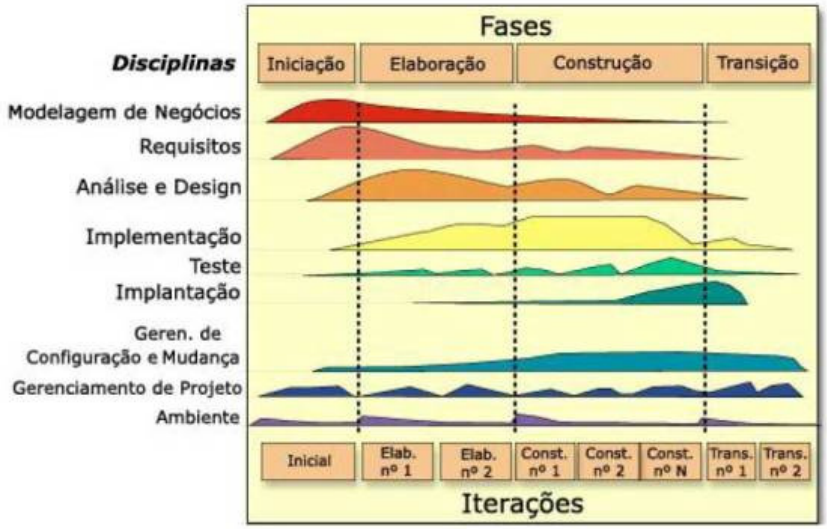

Figura 7 – Visão Geral das Disciplinas e Fases do RUP (adaptado de (IBM, 2005b))

É importante considerar que além de atividades, artefatos, papéis, fluxos e fases, o RUP apresenta ainda elementos de processos adicionais que devem ser associados às atividades e artefatos para facilitar o seu entendimento e uso. Esses elementos de processo são considerados arquivos de conteúdo e podem ser divididos em: diretrizes, *templates* e mentores de ferramentas.

3.2.1. Meta-modelo do Processo RUP

Embora na seção anterior todos os elementos de processo do RUP tenham sido apresentados e descritos é importante entendermos como os mesmos são organizados no processo. Para isso, o RUP apresenta um meta-modelo em termos de seus principais elementos e relacionamentos com objetivo de tornar possível a identificação de que elementos são permitidos e quais as relações válidas entre estes elementos (Figura 8).

Os elementos do processo são definidos no meta-modelo como classes e operações, sendo que elementos do tipo classe recebem o estereótipo <<*class*>> e elementos de operação são representados pelo estereótipo <<*operation*>>. Os elementos do tipo classe descrevem os "objetos" do meta-modelo e elementos do tipo operação descrevem o comportamento destes "objetos".

Lifecycle (ciclo de vida) é representado no meta-modelo por um elemento do tipo classe e define a partição do tempo num conjunto de fases. Essas são representadas pelo elemento do tipo operação ***phase*** (fase). Um elemento *lifecycle* é uma composição de elementos *phase*.

Discipline (disciplina) é um elemento do tipo classe e seu comportamento é definido por um conjunto de fluxos detalhados representados no meta-modelo pela classe do tipo operação ***workflowdetail*** (fluxo detalhado). Um elemento *discipline* é uma composição de elementos do tipo *workflowdetail*, sendo que um ou mais *workflowdetail* devem ser associados a uma ou mais *phases*.

Activity (atividade) é o elemento do meta-modelo do tipo operação que descreve uma unidade particular de trabalho que é desempenhada por um papel. Um elemento *activity* deve estar associado a um ou mais elementos do tipo *workflowdetail. U*m *workflowdetail* deve estar associado a um ou mais elementos *activity*. Ainda, podemos ter ferramentas associadas à execução de uma atividade e dessa forma podemos associar elementos denominados ***toolmentor*** (mentor de ferramenta) a um ou mais elementos do tipo *activity*.

O *toolmentor* é um tipo de orientação que demonstra como usar uma ferramenta específica. Cada instância da classe *toolmentor* aparece associada a uma instância da classe ***tool*** (ferramenta) e isto é representado no meta-modelo por uma associação do tipo composição da classe *tool* para a classe *toolmentor*.

A classe ***Role*** (papel) indica quem é responsável ou modificador de artefatos específicos e ainda define quem desempenha as atividades. Uma *role* é uma composição de um ou mais elementos *activity* e deve estar associado como responsável ou modificador de nenhum ou vários elementos do tipo ***artifact*** (artefato).

A classe *Artifact* mantém relações com *roles* e *activities* do processo. Um elemento *artifact* deve estar sobre a responsabilidade de uma *role* e pode ser modificado por nenhuma ou várias *roles*. Os *artifacts* são consumidos e produzidos por *activities* e dessa forma estão associados como entrada e saída desses elementos. Um *artifact* poderá ser produzido por nenhuma ou várias *activities* e deve ser consumido por uma ou várias *activities*. Por fim, deve-se ainda considerar que uma *activity* deve ter associado um ou mais *artifacts* na entrada e saída.

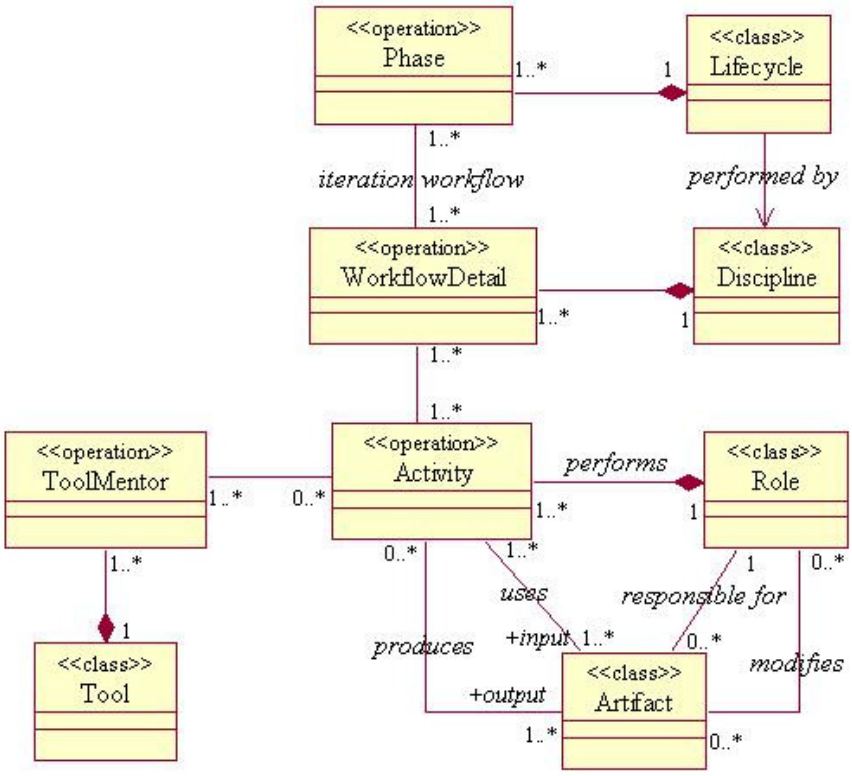

Figura 8 - Meta-modelo RUP (extraído de (Bencomo, 2005))

Como visto pela descrição do meta-modelo acima, o RUP é um modelo de processos que identifica exatamente quais elementos são permitidos em seu processo e ainda, define como os mesmos estarão relacionados. Deve-se considerar ainda, que o RUP apresenta uma ampla definição de processos associada oferecendo, por exemplo, disciplinas que cobrem desde a modelagem de negócios da organização até a entrega final do produto de software para seus clientes, incluindo suporte a gerência de projeto, gerência de requisitos, gerência de configuração e melhoria de processos.

A adaptação de processos no RUP é tratada especificamente em uma de suas disciplinas chamada Ambiente e será descrita na próxima seção deste trabalho.

3.2.2. Processo de Adaptação no RUP

O processo de adaptação no RUP é chamado "*configuration*" e diz respeito a modificar a estrutura do seu processo alterando elementos como artefatos, fases, disciplinas, fluxos detalhados, atividades, papéis, mentores de ferramenta e ferramentas. Em geral, existem dois níveis de adaptação previstos pelo RUP: o nível organizacional, no qual os engenheiros de processo modificam, melhoram ou adaptam o RUP para criar o processo

padrão de desenvolvimento de software que deve ser usado como base por todas as partes da organização e o nível de projeto, no qual os engenheiros de processo da organização adaptam o seu processo padrão para um processo específico de projeto.

No caso da construção de um processo padrão, deve-se levar em consideração questões como domínio da aplicação, reutilização de práticas e tecnologias centrais dominadas pela organização. Nesse caso, pode-se criar mais de um processo padrão para a organização, sendo que cada um deve ser adaptado para um tipo de desenvolvimento. Ainda, o próprio RUP sem adaptações pode ser usado como processo padrão.

Na adaptação do processo padrão para um processo específico de projeto que é o interesse desta pesquisa, leva-se em conta o tamanho do projeto, a reutilização de recursos da organização, entre outros. Nesse caso, a organização deve gerar um artefato chamado *Caso de Desenvolvimento* para todo projeto que for executado. Esse artefato descreve o fluxo do processo e como o projeto o aplicará especificando ainda, quais artefatos deverão ser produzidos durante o projeto.

Como dito anteriormente, todos os aspectos referentes à adaptação de processos no RUP são tratados na disciplina Ambiente. Nessa disciplina, são descritas as atividades, os papéis e os artefatos que dizem respeito a alterar a estrutura do RUP, seja para criação de um processo padrão ou para criação de um processo específico de projeto. Além disso, são descritos alguns comentários sobre características de projeto que devem guiar o processo de adaptação.

Entretanto, apesar da disciplina Ambiente tratar especificamente sobre configuração de processos, não é apresentado na sua descrição nenhuma explicação sobre como tratar os elementos do processo padrão e suas dependências durante a adaptação. Nesse contexto, o RUP apresenta duas ferramentas chamadas *RUP Builder* e *RUP Process Workbench* que embora não sejam citadas na disciplina Ambiente, tem como propósito apoiar o processo de adaptação.

3.2.2.1. *RUP Builder*

O *RUP Builder* é uma ferramenta disponibilizada pela *Rational* cuja principal funcionalidade é permitir a produção de configurações de processo a partir do processo RUP. De acordo com Kroll & Kruchten (2003), uma configuração de processo é uma instância que deve constituir um processo completo para um projeto atendendo todas as suas necessidades em específico.

Para gerar processos de projetos específicos a partir do *RUP Builder*, o RUP é usado como processo padrão. Para entender o funcionamento de tal ferramenta é necessário conhecer alguns conceitos que são utilizados pela mesma (Kroll & Kruchten, 2003):

- **Componente de processo RUP –** módulo quase independente de conhecimento de processo que pode ser nomeado, empacotado, trocado e montado com outros componentes do processo. Pode-se considerar que um componente de processo é uma parte do processo constituída de alguns elementos como atividades, papéis e artefatos.
- **Biblioteca RUP –** coleção de todos componentes de processo incluindo os da base RUP e *plug-ins* RUP. Novos componentes de processo podem ser adicionados para a biblioteca RUP por meio dos *plug-ins* RUP.
- **Base RUP –** é a coleção de componentes de processo referentes ao *framework* RUP. Tais componentes estão dispostos a ser estendidos por intermédio de *plug-ins* para gerar configurações de processos RUP. A base RUP reside em uma biblioteca RUP.
- ***Plug-in* RUP –** conjunto de componentes de processo RUP "précompilados", prontos para serem ligados em uma base RUP com objetivo de criar uma ou mais configurações do RUP. Um *plug-in* RUP pode ser compilado em um simples arquivo físico (com extensão ".cfu").

O funcionamento do *RUP Builder* está baseado na escolha dos componentes de processo que se deseja adicionar ou excluir em um processo. Quando iniciamos o uso da ferramenta é carregada a biblioteca RUP que apresenta a base RUP e alguns *plug-ins.* Os *plug-ins* apresentados no *RUP Builder* são desenvolvidos pela *Rational* e empresas terceiras para oferecer componentes de processo específicos para algum tipo de tecnologia como, por exemplo, o *plug-in Real Time* desenvolvido pela *Rational* para atender necessidades de projetos com desenvolvimento em tempo real. Ainda, outros *plug-ins* desenvolvidos no *RUP Process Workbench* (RPW) descrito na seção 3.2.2.2 deste trabalho podem ser importados para o *RUP Builder.*

Os responsáveis pelo uso do *RUP Builder* são engenheiros de processo ou gerentes de projeto, sendo estes livres para escolher trabalharem apenas com a base RUP quando desejam apenas os componentes de processo do *framework* RUP ou ainda, selecionar alguns dos *plug-ins* disponíveis quando um ou mais deles oferecer componentes de processo desejáveis.

Independente da escolha feita, toda base RUP e *plug-ins* selecionados serão apresentados pela ferramenta na forma de componentes de processo selecionáveis que deverão ser escolhidos de acordo com necessidades do projeto. Conforme o responsável escolhe os componentes de processo, o *RUP Builder* realiza a análise de quais componentes podem ter sido afetados pela sua decisão e mostra um alerta sobre quais atividades pertencentes aos componentes de processo afetados não poderão ser mais realizadas, sendo que tais atividades serão excluídas do processo específico de projeto.

Quando toda seleção de componentes é terminada, o *RUP Builder* permite que o processo resultante seja publicado em um *Website* que deve tornar-se disponível para todos os membros do projeto. Ainda, durante a criação do *Website* o responsável poderá personalizar a apresentação do conteúdo do processo gerando visões personalizadas para os vários membros do projeto. A partir dessas visões personalizadas é possível que um determinado papel do processo, como por exemplo, um analista, veja somente elementos de processo que dizem respeito a sua função no projeto.

3.2.2.2. *Rup Process Workbench*

O *RUP Process Wokbench* (RPW) é uma ferramenta utilizada para criação de *plug-ins* para o RUP. Como dito anteriormente, um *plug-in* é um conjunto de componentes de processo RUP "précompilados", prontos para serem ligados em uma base RUP com objetivo de criar uma ou mais configurações do RUP.

Em outras palavras, um *plug-in* pode ser considerado a maneira pela qual organizações podem estender ou modificar a base RUP para atender suas necessidades específicas. Durante sua criação, é possível que novos elementos de processo como papéis, atividades, artefatos, entre outros sejam criados e conectados a base RUP para que sejam utilizados posteriormente no *RUP Builder*. Ainda, elementos de processo existentes na base RUP podem ser modificados.

O RPW é dividido em quatro componentes que permitem em conjunto a criação dos *plug-ins* RUP (Kroll & Kruchten, 2003):

- **Rup *Modeler*:** ferramenta que possibilita a visualização dos elementos de processo da base RUP tais como atividades, artefatos, papéis, mentores de ferramenta e seus relacionamentos. O RUP *Modeler* oferece suporte para que elementos do processo existentes sejam modificados ou ainda, novos elementos

sejam criados. Para seu uso, é necessária a utilização da ferramenta *Rational* XDE, pois o RUP *Modeler* se encontra na forma de *add-in* para esta ferramenta;

- **Rup *Organizer*:** essa ferramenta permite que arquivos de conteúdo sejam associados com elementos de processo tais como atividades, artefatos, papéis, fluxos centrais (disciplinas), fluxos detalhados e mentores de ferramenta. Os arquivos associados podem ser, por exemplo, diretrizes, *templates*, exemplos, entre outros;
- **Biblioteca RUP:** coleção de todos componentes de processo incluindo os da base RUP e *plug-ins* RUP. Isso contém todos os elementos de processo tais como atividades, artefatos, papéis, fluxos centrais (disciplinas), fluxos detalhados e mentores de ferramenta. Ainda, deve-se considerar que todos os arquivos de conteúdo associados com elementos de processo residem na biblioteca RUP.
- **Processo de Engenharia do Processo (PEP):** o PEP consiste de um guia de processo para customizar, implementar e melhorar o processo.

Através dos componentes acima, dois tipos de *plug-ins* podem ser criados:

- ***Plug-ins* RUP finos –** esse tipo de *plug-in* permite que arquivos de conteúdos associados com elementos de processo existentes sejam adicionados, modificados ou excluídos. Isso permite, por exemplo, mudanças nos guias associados a artefatos ou atividades, ou ainda nos *templates* de certos artefatos. Este tipo de mudança deve ser realizada a partir do componente RUP *Organizer*.
- ***Plug-ins* RUP estruturais –** esse tipo de *plug-in* permite que novos elementos de processo como artefatos, atividades, papéis, fluxos centrais (disciplinas), fluxos detalhados e mentores de ferramenta sejam criados e conectados a base RUP. É possível ainda, modificar ou excluir elementos de processo existentes na base RUP. Esses *plug-ins* devem ser criados com o componente RUP *Modeler* do RPW. Ainda, o RUP *Organizer* pode ser usado caso haja alguma mudança em arquivos associados a algum elemento de processo.

Embora os *plug-ins* estruturais permitam modificações e exclusões nos elementos de processo da base RUP, isso deve ser evitado pois quando uma nova versão do produto é liberada tais modificações precisarão ser novamente realizadas. Ainda, a criação de *plug-ins* deste tipo não é indicada para todas as organizações que utilizam o RUP. Isto porque, para usar o *RUP Modeler* é necessário certo nível de especialidade e bom conhecimento em modelagem orientada a objetos, *Rational XDE* e o próprio *framework* RUP.

Segundo Kroll & Kruchten (2003), para criar um *plug-in* estrutural uma organização deverá gastar algumas semanas ou até meses dependendo de sua complexidade e deverá antes atender aos seguintes requisitos:

- Entender o RUP;
- Entender quais mudanças são necessárias ao RUP;
- Criar um esboço contendo quais elementos de processo deseja-se adicionar, modificar ou excluir;
- Produzir o *plug-in* RUP sem o conteúdo de processo associado;
- Desenvolver seu conteúdo de processo associado;
- Atualizar o *plug-in* com o conteúdo de processo associado;

3.3. Relacionamento entre o Meta-modelo RUP e o Meta-modelo SPEM e Análise de Conformidade

Como o RUP teve sua origem a partir do meta-modelo SPEM é importante relacionar seus elementos com este meta-modelo. Isso porque, embora terminologias diferentes sejam usadas verifica-se (Tabela 1) a existência de muitos elementos comuns em ambos os meta-modelos.

Tabela 1 - Relacionamento entre Elementos dos Meta-modelos RUP e SPEM

SPEM	*ProcessPerformer*	*ProcessRole*	*LifeCycle*	*Phase*	*Discipline*	*Iteration*
RUP	Não Implementa	*Role*	*LifeCycle*	*Phase*	*Discipline*	Não Implementa

SPEM	*WorkDefinition*	*Activity*	*Step*	*WorkProduct*	*Guidance*	*External Description*
RUP	Não Implementa	*Activity*	Não Implementa	*Artifact*	*Tool Mentors*	Não Implementa

Como pode ser visto, o RUP não implementa muitos dos elementos previstos pelo meta-modelo SPEM. Isso ocorre porque na maioria das vezes, durante a construção de um processo de desenvolvimento de software a partir do SPEM, algumas adequações a seus elementos e relacionamentos precisam ser realizadas, fazendo com que alterações sejam encontradas no processo resultante. Nesse sentido, é definido pela OMG que as alterações realizadas a partir do SPEM devem ser especificadas na descrição dos processos como uma

declaração de conformidade SPEM, sendo encontrado informações detalhadas em (OMG, 2005a).

O ponto de conformidade a ser mantido relacionado ao meta-modelo SPEM, principal interesse desta pesquisa, define que todos os elementos de tal meta-modelo devem ser implementados com exceção apenas dos elementos opcionais que são: *guidance, step* e *discipline*.

Como visto na Tabela 1, o RUP realizou vários ajustes em seus elementos a partir do SPEM, sendo muitos deles em elementos não opcionais como *ProcessPerformer*, *WorkDefinition* e *Iteration*. Nesse sentido, na definição de seu processo, estes ajustes são descritos em (IBM, 2005a), embora esteja explícito não se tratar de uma declaração de conformidade SPEM conforme previsto por (OMG, 2005a) e sim, apenas de uma descrição geral sobre o relacionamento entre o meta-modelo RUP e o meta-modelo SPEM.

De acordo com (IBM, 2005a), as não conformidades do meta-modelo RUP em relação ao meta-modelo SPEM são:

- Pacote Dependência:
  - Não implementado no RUP.
- Pacote Estrutura do Processo:
  - Atributo *isDeliverable* da classe *WorkProduct* não é suportado no RUP.
  - No RUP somente atividades podem ter parâmetros.
  - Atributo *hasWorkPerArtifact* da classe *WorkDefinition* não é suportado.
  - A classe *Step* não é suportada no RUP.
  - A classe *ProcessPerformer* não é suportada no RUP.
- Pacote Componentes de Processo:
  - Dependência de categorização não é suportada no RUP.
  - No RUP, a classe *Discipline* não representa um pacote.
  - No RUP, a classe *Activity* pode ser associada a várias instâncias da classe *Discipline*, por meio dos *WorkflowDetails.*
- Pacote Ciclo de Vida:
  - A classe *WorkDefinition* não suporta restrições de metas ou pré-condições.
  - No RUP, a classe *LyfeCicle* (Ciclo de Vida) não é definida como uma *WorkDefinition*.
  - A classe *Iteration* (Iteração) não é suportada no RUP.

Além das não conformidades acima, (IBM, 2005a) define que alguns pontos do meta-modelo RUP são apenas diferentes em relação ao meta-modelo SPEM, não significando estas diferenças pontos de inconformidade. Tais diferenças são:

- A classe *External Description* e alguns tipos de instâncias da classe *Guidance* (*Templates* e *Guidelines*) não são definidos no RUP como elementos da UML e sim, como parte do modelo de processo.
- Decomposição de *WorkDefinitions* usando *subWorks* não é suportado no RUP. Isso porque sua decomposição é restrita somente a *Phase* → *WorkflowDetail* → *Activity*.
- A classe *Discipline* contém *WorkflowsDetails* no RUP, sendo que as atividades ficam associadas indiretamente às disciplinas. Isto é diferente no SPEM, onde uma atividade é associada a apenas uma disciplina.

3.4. Considerações Finais

Este capítulo apresentou um dos principais processos de desenvolvimento de software, bem como o meta-modelo SPEM considerado pela OMG o meta-modelo padrão para construção de processos de desenvolvimento de software. Foram mostrados quais elementos e relacionamentos os constituem, e ainda considerado quais deles possuem definição de processo associada. Por fim, analisou-se como a adaptação de processos é abordada, isto devido a ser o principal interesse desta pesquisa.

Em relação aos elementos apresentados tanto no meta-modelo SPEM como no RUP pode-se perceber que os mesmos apresentam muitas semelhanças diferenciando-se na maioria das vezes pela nomenclatura usada e forma de relacionamento entre os elementos propostos. Elementos como ciclo de vida, fase, atividade, disciplina, papel, artefato (considere produto de trabalho no SPEM) e mentores de ferramenta (considere guias no SPEM) fazem parte do RUP e igualmente do SPEM conforme já descrito na seção 3.3 deste trabalho. Além disso, verificou-se que tanto o RUP quanto o SPEM apresentam elementos particulares que também os diferenciam, seja no caso do RUP por ter realizado adequações a partir do meta-modelo SPEM ou no caso do próprio meta-modelo que por se tratar de um padrão deve oferecer um conjunto maior de elementos abrangendo todas as situações possíveis em processos de desenvolvimento de software.

Quanto a possuir definição de processo associada, verificou-se que o meta-modelo SPEM não a possui isto devido a ser considerado um "*template*" para os demais modelos de

processo de desenvolvimento de software. Ainda, pode-se constatar que referências ao processo de adaptação são encontradas apenas no RUP, sendo completamente inexistentes no meta-modelo SPEM.

Ainda, apesar de o RUP apresentar referências ao processo de adaptação, pode-se constatar algumas inconsistências relativas a este assunto em seu processo. A primeira delas é que a disciplina Ambiente, responsável por assuntos ligados à adaptação de processos não faz menção às ferramentas *RUP Builder* e RPW, propostas também pela *Rational* para apoiar o processo de adaptação. Desta forma, o artefato *Caso de Desenvolvimento* (previsto na disciplina Ambiente) que deve ser gerado para todo processo específico de projeto não possui nenhuma ligação com a ferramenta *RUP Buider*, responsável no RUP por gerar tal processo. Outro ponto é que a ferramenta *RUP Builder* permite apenas a exclusão de componentes de processo não permitindo que elementos como artefatos, atividades, ferramentas, entre outros sejam incluídos ou excluídos em um projeto individualmente. A ferramenta que oferece tais funcionalidades (RPW) não é indicada pelo RUP como uma ferramenta a ser utilizada em todos os projetos e em muitos casos seu uso torna-se restrito até mesmo para algumas organizações, isto devido a exigir certo nível de especialidade em algumas tecnologias.

Por fim, constatou-se que não existem referências explícitas aos relacionamentos entre elementos que devem ser considerados durante um processo de adaptação. Neste sentido, outras abordagens foram estudadas na tentativa de encontrar referências a estes relacionamentos. A seguir serão descritas as que de alguma forma contribuíram para a elaboração deste trabalho.

4. ABORDAGENS PARA ADAPTAÇÃO DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

Este capítulo tem como principal objetivo apresentar os principais trabalhos relacionados ao tratamento de elementos e relacionamentos de um processo padrão durante sua adaptação utilizados no contexto desta pesquisa.

Analisando a literatura atual, é possível observar que o uso combinado de um processo padrão de desenvolvimento de software e sua posterior adaptação tem sido objeto de pesquisa na comunidade de engenharia de software. No que diz respeito à adaptação de processos de software para projetos em específico que é o principal interesse deste trabalho, diferentes aspectos tem sido abordados em (Welzel *et al*, 1995), (Machado, 2000), (Yoon *et al*, 2001), (Borges & Falbo, 2002), (Coelho, 2003), (Xu & Ramesh, 2003). Exemplos destes aspectos são: análise das principais características organizacionais e de projetos que influenciam decisões de adaptação de processos (Machado, 2000), (Borges & Falbo, 2002), (Coelho, 2003), (Xu & Ramesh, 2003), identificação e tratamento dos elementos e relacionamentos do processo padrão durante sua adaptação (Welzel *et al*, 1995), (Yoon *et al*, 2001), (Coelho, 2003) e apoio à reutilização de processos a partir da adaptação do processo padrão (Machado, 2000), (Coelho, 2003).

Como o principal interesse desta pesquisa está relacionado à identificação e tratamentos dos elementos e relacionamentos do processo padrão, nesse capítulo serão descritos os trabalhos propostos por (Welzel *et al*, 1995), (Yoon *et al*, 2001) e (Coelho, 2003), os quais serviram como referência para esta pesquisa.

4.1. Abordagem de Yoon, Min e Bae (Yoon *et al*, 2001)

Neste trabalho, um método para formalização e adaptação de processos é apresentado. Conforme o autor, o processo padrão deve ser baseado em módulos de processo inter-relacionados que consistem de atividades, artefatos e o relacionamento entre eles. Uma atividade é uma entidade do processo e possui as seguintes propriedades:

- *N*: Nome da atividade;
- *T*: Tipo da atividade;

- *IR*: Conjunto de artefatos de entrada da atividade;
- *OR*: Conjunto de artefatos de saída da atividade.

Similarmente, um artefato também é considerado uma entidade do processo e consiste das seguintes propriedades:

- *N*: Nome do artefato;
- *T*: Tipo do artefato;
- *CA*: Conjunto de atividades que consomem o artefato;
- *PA*: Atividade que produz o artefato;
- *SR*: Conjunto de artefatos que tem forte relação com o artefato;
- *II*: Valor representado para o artefato se ele é uma interface de entrada
- *OI*: Valor representado para o artefato se ele é uma interface de saída
- *P*: Valor representado para o artefato se ele é preexistente
- *D*: Valor representado para o artefato se ele é um produto de entrega

Os módulos do processo padrão são conectados por meio de artefatos especiais que são os artefatos de interface de entrada II e artefatos de interface de saída OI. Cada artefato de interface de entrada é um artefato requerido por um módulo do processo e cada artefato de interface de saída é produzido pelo módulo. O conjunto desses módulos forma o processo padrão de desenvolvimento de software da organização. Ainda, atividades de adaptação devem ser previstas para adequar este processo padrão a necessidades de projetos em particular. Para ele, o processo de adaptação é definido como todas as atividades de apagar ou modificar uma parte do processo padrão, ou ainda adicionar novas entidades sem violar a dependência entre as entidades. Os procedimentos de adaptação propostos neste trabalho são:

Adicionar Entidades ao Processo

A operação de adição é classificada como adição de entidades padrão e adição de entidades definidas pelo usuário. Adição de entidades padrão significa adicionar tipos de entidades que já são descritas no processo padrão enquanto que adição de entidades definidas pelo usuário significa adicionar entidades que foram descritas pelo projetista levando em consideração alguma necessidade particular do projeto. É possível adicionar atividades e artefatos e ainda relacioná-los com outras atividades e artefatos existentes. Segundo (Yoon *et al*, 2001), a operação de adição não viola a conformidade do processo padrão.

Apagar Entidades do Processo

A operação de apagar entidades é vista como uma das mais comuns em um processo de adaptação e também como uma das operações que mais afeta outras entidades do processo. Por exemplo, quando uma atividade é apagada do processo padrão de desenvolvimento de software, os artefatos gerados por aquela atividade serão apagados também. Nesse contexto, quando um artefato é apagado de um módulo do processo, deve-se considerar quando o mesmo é artefato de interface de entrada para algum módulo checando os atributos *II* e *OI* do artefato. Similarmente, quando uma atividade é apagada deve-se verificar se a mesma não produz ou usa artefatos de interface de entrada ou saída. Caso um artefato de interface de entrada ou saída ou uma atividade que o produz seja apagado deve-se remover todas as dependências existentes nos módulos do processo que utilizam este artefato. Se isso não for possível, a remoção do artefato ou atividade deve ser abortada.

Dividir Entidades do Processo

Atividades podem ser divididas em sub-atividades. Para essa operação, a atividade original deve ser apagada e novas atividades vão ser adicionadas ao módulo do processo. Pode-se também dividir artefatos e nesse caso, deve-se ligar os novos artefatos como artefatos de entrada do PA (atividade que produz o artefato) e como artefatos de saída dos CAs (conjunto de atividades que consomem o artefato) do artefato original. É importante lembrar que toda a informação sobre divisão de atividades e artefatos deve ser mantida durante o processo de adaptação.

Unir Entidades do Processo

Essa operação permite que duas atividades do processo padrão sejam unidas formando uma nova atividade, isto é na maioria das vezes utilizado em pequenos projetos ou no desenvolvimento de aplicações rápidas. Assim, duas atividades originais devem ser removidas e uma nova atividade será adicionada ao escopo do projeto. Durante a união de atividades todos os artefatos de entrada e saída devem ser conectados à nova atividade. União de artefatos não é permitida, pois um artefato é sempre produzido por uma atividade. Se tentarmos unir dois artefatos de atividades distintas o mesmo teria dois PA (atividade que produz o artefato) e isto não é possível. Assim, se necessário que dois artefatos sejam unidos deve-se criar uma nova atividade que gere o novo artefato integrado.

A idéia principal deste trabalho é que durante o processo de adaptação uma ou mais atividades descritas acima sejam usadas, embora seja importante serem conduzidas de forma cuidadosa, pois é importante que o novo processo gerado mantenha conformidade com o

processo padrão. Para o autor, se as atividades sugeridas acima for realizadas de forma correta a conformidade com o processo padrão é mantida.

4.2. Abordagem de Welzel, Hausen e Schmidt (Welzel *et al*, 1995)

Os autores deste trabalho propõem um método denominado *ProcePT* (*Process Programming and Testing*) para adaptação de processos. O método consiste de uma ferramenta desenvolvida em PROLOG para adaptação, sendo esta baseada em um modelo de processo denominado *GV-Model* (*German Process Model).*

O modelo *GV-Model* é um padrão internacional reconhecido no desenvolvimento de sistemas de TI e consiste em um modelo de processos que descreve atividades para o desenvolvimento de software e ainda atividades relacionadas à garantia de qualidade, gerência de configuração e gerência de projeto. O modelo foi desenvolvido na Alemanha e é utilizado no desenvolvimento de sistemas na República Federal da Alemanha.

O modelo *GV-Model* serve como um guia com instruções para o desenvolvimento de sistemas com descrições detalhadas das atividades e de produtos. Organizações podem utilizar o modelo em seus projetos e o mesmo indica que para sua utilização em um determinado projeto deve-se primeiro decidir quais atividades são requeridas e quais produtos devem ser gerados no escopo de tal projeto. A partir dessas informações o modelo permite que atividades e produtos sejam apagados do processo padrão de desenvolvimento de software consistindo assim em uma atividade de adaptação. Para que isso seja possível de forma consistente, o modelo fornece um conjunto de condições que vão dar suporte a operação de apagar uma atividade ou produto, ou seja, se uma condição validar a operação será permitido remover a atividade ou produto. Essas condições podem ainda influenciar outras condições, como por exemplo, se um produto pode ser apagado de um conjunto de produtos, a atividade que o produz também deve ser apagada. O *GV-Model* é formado por mais de cem atividades, 50 produtos e 90 condições. É importante considerar que somente operações de exclusão são permitidas. O modelo ainda considera que todas as decisões tomadas durante atividades de adaptação devem ser documentadas e constar no manual do projeto.

O Procept consiste na implementação de uma ferramenta baseada no modelo apresentado anteriormente desenvolvida em PROLOG. A ferramenta propõe-se a configurar a partir do modelo de processo *GV-Model* todas as atividades e produtos a ser utilizados por um projeto de desenvolvimento de software. Foi implementado um conjunto de regras que dará suporte às atividades de adaptação e conformidade com o processo padrão de

desenvolvimento de software. O banco de dados da ferramenta suporta um conjunto de atividades, documentos e as condições como previsto no *GV-Model*, ou seja, suporta a descrição de todo processo padrão. Toda vez que for necessário configurar um ciclo de vida para um projeto deve-se utilizar a ferramenta e analisar as atividades e produtos necessários no escopo do projeto podendo assim excluir atividades e produtos desde que a operação seja validada por uma das condições como previsto no *GV-Model*.

4.3. Abordagem de Coelho e Perreli (Coelho, 2003)

O trabalho realizado por Coelho e Perreli embora não trate explicitamente de um método para adaptação de processos, apresenta um estudo interessante sobre configuração de processos padrão para projetos específicos. A proposta é chamada pelos autores de *PConfig* e é parte integrante de um modelo para adaptação de processos de software denominado MAPS.

PConfig trata da configuração de processos de software para projetos específicos que baseia-se na escolha dos artefatos do processo padrão para gerar o processo adaptado. A escolha dos artefatos deve ser realizada a partir da análise das características do projeto. Assim, a partir dos artefatos escolhidos, são derivados os papéis e atividades necessários.

O *PConfig* é dependente do processo padrão que a organização utiliza por estar intrinsecamente relacionado com os artefatos do processo. Para sua utilização, é necessário primeiramente que a organização identifique todos seus artefatos e configure se serão ou não adaptáveis nos projetos. A partir daí, de acordo com um conjunto de características sugeridas pelo MAPS, deve-se construir uma matriz para cada característica com os níveis de classificação da característica e os artefatos, identificando, para cada nível, os artefatos que serão produzidos, não serão produzidos, serão produzidos com restrições (informalmente, apenas algumas seções do documento, entre outros) ou tem sua produção indiferente.

Assim, durante a utilização do *PConfig* em projetos específicos estas matrizes serão utilizadas para que suas colunas sejam selecionadas de acordo com as características dos projetos, gerando assim, ao final desta seleção o conjunto de artefatos a serem produzidos pelo projeto em questão.

De acordo com os autores, é importante ainda que a organização conheça todas as relações de dependência entre os artefatos de seu processo padrão, isso porque tais relações podem impedir que determinados artefatos sejam excluídos do processo enquanto outros artefatos fizerem parte do mesmo.

O processo padrão utilizado para avaliação do *PConfig* foi o RUP. Apenas a disciplina de planejamento e gerência de projetos foi analisada.

4.4. Considerações Finais

Como visto, podem existir diferentes formas para a adaptação de processos de desenvolvimento de software. Um método pode se mostrar mais completo que outro e estabelecer atividades que são ou não permitidas em seu processo como é o caso das abordagens propostas por (Welzel *et al*, 1995), (Yoon *et al*, 2001) e (Coelho, 2003).

Para Yoon *et al* (2001), várias operações são permitidas quando um processo de adaptação é aplicado para um projeto; o mesmo prevê operações como adicionar, remover, dividir e unir atividades e artefatos. Para isso, o mesmo possui algumas regras que devem ser seguidas, na maioria delas dizendo respeito a novas propriedades que artefatos e atividades devem receber. Segundo Yoon *et al* (2001), a prioridade é que o novo processo esteja em conformidade com o processo padrão, e é por esta razão que as regras sobre as propriedades devem ser seguidas. Já em (Welzel *et al*, 1995) e (Coelho, 2003) o processo de adaptação é baseado apenas na exclusão de elementos, sendo em (Welzel *et al*, 1995) permitido a exclusão de atividades e produtos e em (Coelho, 2003) somente a exclusão de artefatos.

Analisando o suporte oferecido pelos métodos descritos nesta seção do trabalho, pode ser visto que embora exista a preocupação acerca do tratamento dos elementos do processo padrão durante sua adaptação, algumas limitações são encontradas. Estas limitações dizem respeito principalmente à falta de referências a alguns elementos e relacionamentos do processo padrão a serem considerados em processos de adaptação. Por exemplo, nenhum dos trabalhos descritos faz menção ao relacionamento de precedência entre atividades de um processo, sendo esta uma dependência prevista em (OMG, 2005a) e considerada importante para processos de adaptação. Também constata-se a inexistência de um meta-modelo definindo quais elementos e relacionamentos constituem um processo padrão de desenvolvimento de software. Nesse sentido, existe a falta de referência a importantes elementos como ferramentas, papéis, disciplinas, fases e ciclo de vida. Estes estão previstos em (Bencomo, 2005) e (OMG, 2005a).

Outro problema é o tratamento de alguns relacionamentos do processo durante sua adaptação. Por exemplo, quando uma atividade é apagada (Welzel *et al*, 1995) e (Yoon *et al*, 2001) prevêem que todos os artefatos produzidos por ela devem ser apagados e desconectados das atividades que os consomem, embora não considerem a hipótese de algumas destas

atividades tornarem-se inválidas por não consumirem mais determinados artefatos. Em (Coelho, 2003) o problema passa a ser maior, isto porque não existe menção a exclusão de atividades.

Ainda, deve-se considerar que somente o trabalho proposto por Coelho (2003) faz referências à obrigatoriedade de alguns elementos em todos os processos, sendo este um dos requisitos para a compatibilidade com o nível 3 do modelo CMMI (Chrissis *et al*, 2003).

5. META-MODELO PARA ADAPTAÇÃO DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE BASEADOS NO RUP

Este capítulo apresenta a descrição de um meta-modelo para adaptação de processos de desenvolvimento de software proposto no contexto desta pesquisa. Ainda relata como o mesmo foi definido explanando a origem de seus elementos e relacionamentos.

A adaptação de um processo padrão de desenvolvimento de software para um projeto específico faz necessária a garantia de consistência do processo resultante, em termos da integração entre seus vários elementos (atividades, artefatos, papéis etc.). Nesse sentido, torna-se importante o entendimento sobre como os elementos de um processo padrão estão estruturados, além da identificação das várias dependências entre estes elementos.

Neste trabalho está sendo proposta uma extensão ao meta-modelo RUP com a inclusão de novas classes e associações. Tal extensão tem como propósito descrever o conjunto de elementos e relacionamentos necessários para adaptação de processos de desenvolvimento de software compatíveis com o RUP, evitando assim possíveis inconsistências no processo adaptado.

Na Figura 9, um diagrama de classes UML (OMG, 2005b) foi utilizado para representar o meta-modelo proposto. Na seqüência, uma descrição de suas classes (seção 5.1) e associações (seção 5.3) é apresentada fazendo a devida relação com o meta-modelo RUP. Ainda, na seção 5.5 é apresentada a organização do meta-modelo proposto em termos de pacotes UML.

5.1. Classes e Atributos do Meta-modelo

Esta seção apresenta uma documentação detalhada das classes e atributos do meta-modelo. As classes apresentadas representam os elementos que constituem um processo de desenvolvimento de software. O conjunto de atributos definidos para cada uma delas é baseado em estudos realizados em (IBM, 2005b) e representam as informações que necessitam ser mantidas sobre tais classes.

A seguir, em subseções cada uma das classes do meta-modelo (em **negrito**) será apresentada em termos de sua finalidade e atributos. Os atributos (em sublinhado) são descritos através do nome de cada atributo e uma descrição sobre o mesmo.

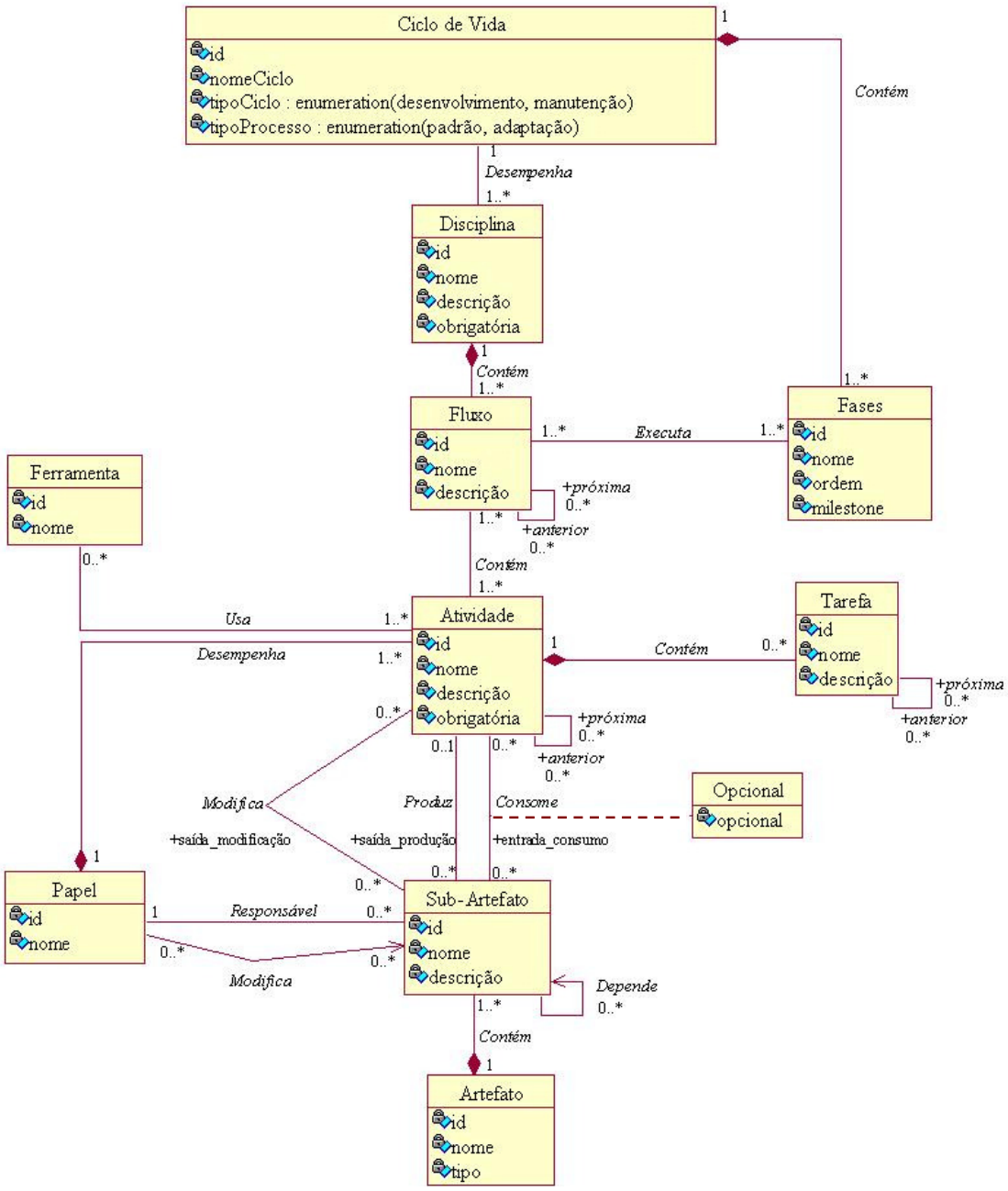

Figura 9 - Meta-modelo para adaptação de processos

5.1.1. Ciclo de Vida

A classe **Ciclo de Vida** (*LifeCyle* no RUP) representa o elemento de processo ciclo de vida. Este elemento é considerado o próprio processo de desenvolvimento de software, isto

porque é o elemento que define o comportamento completo de um processo em um determinado projeto. Para este elemento os atributos id, nome, tipoCiclo e tipoProcesso são definidos, sendo estes detalhados na Tabela 2.

Tabela 2 – Atributos da Classe **Ciclo de Vida**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento ciclo de vida.
nome	Atributo que mantém o nome do ciclo de vida.
tipoCiclo	Atributo que mantém informações sobre o tipo de ciclo de vida do processo. Este atributo é definido como tipo *Enumeration* restringindo que só os valores Manutenção e Desenvolvimento podem ser assumidos neste atributo.
tipoProcesso	Atributo que mantém informações sobre o tipo do processo. Este atributo é definido como tipo *Enumeration* restringindo que só os valores Padrão e Adaptação podem ser assumidos neste atributo.

Deve-se considerar que os atributos tipoCiclo e tipoProcesso não são previstos em (IBM, 2005b) e foram utilizados na classe **ciclo de vida** devido a sua importância no contexto desta pesquisa. O atributo tipoProcesso permite distinguir no meta-modelo os processos padrão de desenvolvimento de software dos processos que foram adaptados a partir deles. Já o atributo tipoCiclo permite classificar o tipo de ciclo de vida do processo em desenvolvimento ou manutenção. Esta classificação permite que restrições diferentes sejam implementadas de acordo com características específicas de cada ciclo. Inicialmente, uma restrição foi definida para ciclos de manutenção e será explicada no próximo capítulo deste trabalho.

5.1.2. Fase

A classe **Fase** (*Phase* no RUP) representa o elemento fase que é parte do ciclo de vida do processo e possui uma meta específica. Um ciclo de vida pode conter um conjunto de fases, sendo que a execução de todas elas indica que o ciclo de vida está completo. A classe **fase** contém os atributos id, nome, ordem e *milestone*, detalhados na Tabela 3.

Tabela 3 – Atributos da Classe **Fase**

Nome	Descrição
Id	Atributo que mantém um identificador para o elemento fase.
nome	Atributo que mantém o nome da fase.
ordem	Atributo que define a ordem de execução da fase no elemento ciclo de vida.
milestone	Atributo que mantém informações sobre a meta específica do elemento fase.

5.1.3. Disciplina

O elemento de processo disciplina é representado no modelo pela classe **Disciplina** (*Discipline* no RUP). Estes elementos representam uma divisão de elementos de processo em áreas de interesse. A classe **disciplina** possui os atributos id, nome, descrição e obrigatório, detalhados na Tabela 4.

Tabela 4 – Atributos da Classe **Disciplina**

Nome	Descrição
Id	Atributo que mantém um identificador para o elemento disciplina.
nome	Atributo que mantém o nome da disciplina.
descrição	Atributo que mantém informações resumidas sobre disciplina.
obrigatório	Atributo usado para estabelecer se uma disciplina é obrigatória ou não em um processo.

É importante considerar que o atributo obrigatório não é utilizado em (IBM, 2005b) e seu uso em várias classes do meta-modelo foi definido no contexto deste trabalho para permitir que determinados elementos não possam ser excluídos durante um processo de adaptação.

5.1.4. Fluxo

Fluxo (*WorkflowDetail* no RUP) representa o elemento de processo fluxo detalhado. Um fluxo detalhado é o agrupamento de atividades relacionadas a uma determinada disciplina. A relação entre as atividades que são agrupadas em um fluxo detalhado pode estabelecer-se, por exemplo, quando as atividades são desempenhadas seqüencialmente, em paralelo ou ainda, por serem desempenhadas pelo mesmo papel. A classe **fluxo** mantém o atributo id, nome e descrição, detalhados na Tabela 5.

Tabela 5 – Atributos da Classe **Fluxo**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento fluxo.
nome	Atributo que mantém o nome do fluxo.
Descrição	Atributo que mantém informações resumidas sobre fluxo.

5.1.5. Atividade

A classe **Atividade** (*Activity* no RUP) mantém informações referentes ao elemento de processo atividade. Uma atividade é uma unidade de trabalho que produz um resultado

significante no contexto de um projeto. Este elemento possui um propósito claro expresso em termos de produzir ou modificar sub-artefatos, sendo que toda atividade deve ser atribuída a um papel. A classe **atividade** possui os atributos id, nome, descrição e obrigatório, detalhados na Tabela 6.

Tabela 6 – Atributos da Classe **Atividade**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento atividade.
nome	Atributo que mantém o nome da atividade.
descrição	Atributo que mantém informações resumidas sobre atividade.
obrigatório	Atributo usado para estabelecer se uma atividade é obrigatória ou não em um processo.

5.1.6. Tarefa

O elemento de processo tarefa não existe no meta-modelo RUP é representado no modelo proposto pela classe **Tarefa**. Uma tarefa é definida como uma parte do elemento atividade e é considerada a menor ação a ser desempenhada, ou seja, é um elemento atômico no processo. Seus atributos são id, nome e descrição, detalhados na Tabela 7.

Tabela 7 – Atributos da Classe **Tarefa**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento tarefa.
nome	Atributo que mantém o nome da tarefa.
descrição	Atributo que mantém informações resumidas sobre tarefa.

5.1.7. Ferramenta

O elemento de processo ferramenta é um elemento que pode ser usado como auxílio para execução de algumas atividades. Na maioria das vezes, auxiliam a produção ou modificação de um artefato. Este elemento é representado no modelo pela classe **Ferramenta** (*Tool* no RUP) e seus atributos são id, nome e descrição, detalhados na Tabela 8.

Tabela 8 – Atributos da Classe **Ferramenta**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento ferramenta.
nome	Atributo que mantém o nome da ferramenta.
descrição	Atributo que mantém informações resumidas sobre ferramenta.

5.1.8. Papel

A classe **Papel** (*Role* no RUP) representa o elemento de processo papel que é definido como elemento responsável por desempenhar atividades para produzir e/ou modificar os sub-artefatos do processo. Um papel define o comportamento e as responsabilidades de um indivíduo ou grupo de indivíduos que trabalham juntos como uma equipe. Para este elemento somente os atributos id e nome são definidos, sendo estes detalhados na Tabela 9.

Tabela 9 – Atributos da Classe **Papel**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento papel.
nome	Atributo que mantém o nome do papel.

5.1.9. Sub-Artefato

A classe **Sub-Artefato** representa o elemento de processo sub-artefato e não é utilizado no meta-modelo RUP. Um sub-artefato é informação produzida, consumida ou modificada em um processo de software. Os sub-artefatos são definidos como partes de artefato e podem ser, por exemplo, uma seção de um documento, partes de um modelo (por exemplo, os atores em um modelo de casos de uso), entre outros. A classe **sub-artefato** possui os atributos id, nome e descrição, detalhados na Tabela 10.

Tabela 10 – Atributos da Classe **Sub-Artefato**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento sub-artefato.
nome	Atributo que mantém o nome do sub-artefato.
descrição	Atributo que mantém informações resumidas sobre sub-artefato.

5.1.10. Artefato

O elemento de processo artefato é representado no modelo pela classe **Artefato** (*Artifact* no RUP). Um artefato é definido com um conjunto de sub-artefatos e pode ser dos seguintes tipos: modelo, documento, plano, repositório, relatório e executável. A classe **artefato** possui os atributos id, nome e tipo, detalhados na tabela 11.

Tabela 11 – Atributos da Classe **Artefato**

Nome	Descrição
id	Atributo que mantém um identificador para o elemento artefato.
nome	Atributo que mantém o nome do artefato.
tipo	Atributo que mantém informações sobre o tipo do artefato.

5.1.11. Opcional

A classe **Opcional** não representa um elemento de processo e também não é previsto em (IBM, 2005b). Esta classe está relacionada com a associação *Consome* entre Atividade e Sub-Artefato e mantém informações se um determinado sub-artefato é opcional ou não em uma atividade. Apenas o atributo opcional é definido para esta classe, sendo este detalhado na Tabela 12.

Tabela 12 – Atributos da Classe **Opcional**

Nome	Descrição
opcional	Atributo que mantém informações sobre a opcionalidade dos sub-artefatos nas atividades.

5.2. Classes Incluídas no Meta-modelo RUP

Como visto, as classes do meta-modelo representam elementos de processo de software e seus atributos as informações referentes a cada um destes elementos. Os conceitos apresentados para cada um dos elementos de processo são os mesmos apresentados em (IBM, 2005b), sendo que os seguintes elementos foram incorporados ao modelo: **Tarefa** e **Sub-Artefato**.

Tarefa é um elemento de processo previsto no processo RUP embora não seja modelado em seu meta-modelo. Em (Kruchten, 2000), tem-se que uma atividade deve ser dividida em etapas que representam suas partes. Desta forma, no meta-modelo proposto criou-se a classe **Tarefa** para que seja possível decompor uma atividade em partes.

A criação do elemento **Sub-Artefato** é a principal mudança incorporada ao meta-modelo RUP em termos de elementos de processo. Através deste elemento permite-se que qualquer artefato seja dividido em partes e que as atividades sejam modeladas em termos de quais destas partes específicas irão produzir, consumir e/ou modificar.

O uso de **sub-artefatos** traz uma importante contribuição ao processo de adaptação que é o principal interesse deste trabalho. Isto porque, torna possível que durante a exclusão

de uma atividade sejam identificadas quais sub-artefatos não serão mais produzidos. Isto não é possível se o processo modela suas atividades em termos de consumo e produção de artefatos como é o caso do meta-modelo RUP.

Por fim, é importante ressaltar que o meta-modelo proposto não considera o elemento de processo ***toolmentor*** do meta-modelo RUP. Isto porque este elemento é visto apenas como um elemento de orientação do processo, sendo utilizado para descrever o uso de ferramentas no contexto de algumas atividades. Neste sentido, este elemento aparece apenas associado a ferramentas do processo e não é considerado importante para processos de adaptação. Além disso, em (OMG, 2005a) os elementos de orientação são considerados como opcionais para os processos de desenvolvimento de software como já citado na seção 3.3 deste trabalho.

5.3. Associações do Meta-modelo

Esta seção descreve as associações existentes entre as classes do meta-modelo proposto. Cada uma das associações é descrita devido a sua importância em processos de adaptação. Isto porque, um processo de adaptação permite que elementos sejam adicionados, excluídos ou ainda modificados a partir de um processo de software (Ginsberg & Quinn, 1995), (Yoon *et al*, 2001), (Jalote, 2002) e (Fitzgerald *et al*, 2003). Assim, é preciso respeitar as relações entre estes elementos para garantir a conformidade com o processo padrão durante a sua adaptação.

Após a descrição de todas as relações uma comparação ao meta-modelo RUP será realizada de modo a justificar a inclusão e modificação de algumas associações.

Ciclo de Vida / Disciplina: As disciplinas são sempre desempenhadas durante o ciclo de vida do processo. Desta forma, tem-se uma relação de associação entre ciclo de vida e disciplina. Nesta relação, uma disciplina está sempre associada a um ciclo de vida, sendo que um ciclo de vida pode ser associado a uma ou várias disciplinas.

Ciclo de Vida / Fase: O ciclo de vida define a partição do tempo num conjunto de fases. Assim, tem-se uma associação do tipo agregação entre ciclo de vida e fase, pois um ciclo de vida é um conjunto de uma ou várias fases.

Fase / Fluxo: A relação de associação entre fase e fluxo é estabelecida visto que os fluxos devem ser executados nas fases. Em uma fase temos a execução de pelo menos um ou mais fluxos, enquanto que um fluxo deve ser executado em pelo menos uma ou mais fases.

Disciplina / Fluxo: Uma disciplina é um conjunto de um ou vários fluxos. Desta forma, é estabelecida uma associação do tipo composição entre disciplina e fluxo. Esta associação indica que a existência de um fluxo depende da existência da disciplina a que pertence.

Fluxo / Fluxo: A relação de associação criada entre fluxos é estabelecida visto que estes são elementos seqüências ou paralelos em uma disciplina e sendo assim apresentam relações de precedência.

Fluxo / Atividade: A relação de associação entre fluxo e atividade é estabelecida visto que as atividades devem ser executadas em um ou mais fluxos. Desta forma, em um fluxo temos a execução de pelo menos uma ou mais atividades, enquanto que uma atividade deve ser executada em pelo menos um ou mais fluxos.

Atividade / Atividade: A relação de associação criada entre atividades é estabelecida visto que estes são elementos seqüências ou paralelos em um fluxo e sendo assim apresentam relações de precedência.

Atividade / Tarefa: Tem-se uma associação do tipo composição entre atividade e tarefa, pois uma atividade é um conjunto de tarefas. Esta associação indica que a existência de uma tarefa depende da existência da atividade a que pertence.

Tarefa / Tarefa: A relação de associação entre tarefas é estabelecida visto que estes são elementos seqüências ou paralelos em uma atividade e sendo assim apresentam relações de precedência.

Atividade / Ferramenta: A relação de associação entre atividade e ferramenta é estabelecida pois atividades usam ferramentas durante sua execução. Nesta relação, uma ferramenta deve ser associada a uma ou mais atividades, enquanto que uma atividade pode ser associada a nenhuma ou várias ferramentas.

Atividade / Papel: Atividades e papéis relacionam-se pois papéis executam atividades em um processo. No modelo, tem-se uma associação do tipo composição entre papel e atividade, pois um papel é um conjunto de atividades. Esta associação indica que a existência de uma atividade depende da existência do papel a que pertence.

Sub-Artefato / Atividade: Sub-artefatos podem ser produzidos, consumidos ou modificados por uma atividade estabelecendo assim três associações diferentes.

Na relação de associação de uso representado pelo papel *+entrada_consumo* tem-se que uma atividade pode estar associada a nenhum ou vários sub-artefatos, enquanto que um sub-artefato associa-se a nenhuma ou várias atividades.

A associação estabelecida de produção entre atividade e sub-artefato é representada pelo papel *+saída_produção* e define que um sub-artefato deve ser produzido por exatamente uma atividade, enquanto que uma atividade pode produzir nenhum ou vários sub-artefatos.

Por fim, temos uma associação de modificação entre atividade e sub-artefato representada pelo papel *+saída_modificação*. Esta associação estabelece que um sub-artefato pode ser modificado por nenhuma ou várias atividades, enquanto que uma atividade poderá modificar nenhum ou vários sub-artefatos.

Sub-Artefato / Papel: Sub-artefatos estão sobre a responsabilidade de um papel e podem também ser modificados por papéis em um processo. Desta forma, temos associações de responsabilidade e modificação ente sub-artefato e atividade.

Na associação de responsabilidade tem-se que um sub-artefato deve estar associado a exatamente um papel, enquanto que um papel pode ser responsável por nenhum ou vários sub-artefatos.

A associação de modificação entre sub-artefato e papel indica que um sub-artefato pode ser modificado por nenhum ou vários papéis, enquanto que um papel pode ser modificador de nenhum ou vários sub-artefatos.

Sub-Artefato / Sub-Artefato: Temos que sub-artefatos de saída de uma atividade podem depender dos sub-artefatos de entrada desta atividade. Assim, pode-se criar uma relação de associação de um sub-artefato para outro(s). Esta relação pode ser estabelecida de sub-artefatos de um artefato para outros sub-artefatos do mesmo artefato, ou ainda, com sub-artefatos de outros artefatos.

Artefato / Sub-Artefato: Um artefato é um conjunto de um ou vários sub-artefatos. Desta forma, é estabelecida uma associação do tipo composição entre artefato e sub-artefato. Esta associação indica que a existência de um sub-artefato depende da existência do artefato a que pertence.

As associações descritas acima são de ampla importância em qualquer processo de desenvolvimento de software, isto porque, conforme (Bencomo, 2005) são elas que expressam as relações válidas entre seus elementos. Em processos de adaptação estas

associações tornam-se ainda mais importantes, visto que representam dependências entre elementos do processo. Segundo Yoon *et al* (2001), dependências entre elementos do processo padrão devem ser preservadas durante um processo de adaptação para que inconsistências não sejam geradas no processo resultante.

5.4. Associações Incluídas e Modificadas a partir do Meta-modelo RUP

Em razão de o meta-modelo proposto ser usado exclusivamente para adaptação de processos algumas associações precisaram ser incluídas e modificadas a partir do meta-modelo RUP para expressar as devidas relações de dependência existente entre os elementos do processo. A seguir tais associações serão resumidamente descritas.

Os auto relacionamentos criados para fluxo, atividade e tarefa não existem no meta-modelo RUP e foram definidos no modelo proposto pois estes elementos possuem uma determinada ordem de execução em processos de software. Esta ordem de execução precisa ser devidamente verificada durante um processo de adaptação visto que possivelmente sofrerá alterações caso um destes elementos seja excluído, adicionado ou modificado. Em (OMG, 2005a), esta relação de dependência é prevista e denominada "precedência", embora seja considerada apenas para atividades.

O auto relacionamento criado para sub-artefato foi definido visto que os sub-artefatos podem depender de outros sub-artefatos para poderem ser produzidos no processo. Em (OMG, 2005a), esta relação de dependência é prevista e denominada "impacto". Ainda, em (Kellner, 1996), (Yoon *et al*, 2001) e (Kroll & Kruchten, 2003) este tipo de dependência também é citada, embora seja importante considerar que todas as referências estudadas consideram esta dependência entre os artefatos do processo. A motivação para que neste trabalho tal relacionamento tenha sido criado para sub-artefato é em virtude das atividades agora serem modeladas em termos de quais sub-artefatos produzem, consomem ou modificam como já explicado na seção anterior deste trabalho.

Outras associações modificadas a partir do meta-modelo RUP foram entre as classes atividade e artefato. No RUP, artefatos são produzidos e usados por atividades e desta forma, duas associações são estabelecidas entre estas classes. A associação de uso é representada pelo papel *+input* e define que uma atividade usa um ou mais artefatos, enquanto que um artefato é usado por uma ou mais atividades. A associação de produção é representada pelo papel *+output* e define que uma atividade produz um ou mais artefatos, enquanto que um artefato pode ser produzido por nenhuma ou várias atividades.

No meta-modelo proposto estas associações são representadas entre as classes atividade e sub-artefato, isto porque, como explicado na seção anterior deste trabalho, o modelo proposto modela as atividades em termos de quais partes específicas de artefatos devem ser produzidas, consumidas ou modificadas. Ainda, uma nova associação foi estabelecida para modificação de sub-artefatos e é representada pelo papel *+saída_modificação*.

A motivação para criar uma nova associação de modificação é para que seja possível diferenciar quando um sub-artefato esta sendo produzido ou modificado no processo. Desta forma, no modelo proposto tem-se três associações entre as classes atividade e sub-artefato: *produção*, *modificação* e *uso*.

Por fim, uma última relação de associação que embora não tenha sido modificada a partir do meta-modelo RUP mereça devida atenção, é a relação de associação entre fluxo e atividade. A associação entre estas classes é definida no RUP como uma associação de muitos para muitos. Entretanto, o que se pode perceber através de um estudo detalhado em suas atividades e fluxos é que muitas vezes cada fluxo possui um propósito diferente em termos de executar atividades do processo. Desta forma, embora uma atividade esteja presente em vários fluxos, esta é executada de maneira diferente em cada um deles, sendo muitas vezes executada por partes.

Para entender melhor como algumas atividades são diferentes nos fluxos do processo, exemplos de execução das atividades *Desenvolver Documento de Visão* e *Desenvolver Caso Empresarial* serão descritos a seguir.

A atividade *Desenvolver Documento de Visão* é associada no RUP aos seguintes fluxos da disciplina Requisitos: *Analisar o problema*, *Entender necessidades dos stakeholders*, *Definir o sistema* e *Gerenciar o escopo do sistema*. O primeiro fluxo a ser executado é *Analisar o problema* onde uma versão inicial (parte do documento) do *Documento de Visão* é produzida. Os demais fluxos refinam tal documento em termos de outras informações (outras partes do documento) que devem ser documentadas sobre o produto a ser desenvolvido.

A atividade *Desenvolver Caso Empresarial* é associada aos fluxos *Conceber novo projeto*, *Avaliar escopo e risco do projeto* e *Planejar para a próxima iteração*. Tais fluxos pertencem à disciplina Gerenciamento de Projeto e tem propósitos diferentes quanto à produção do *Caso Empresarial*. O fluxo *Conceber novo projeto* é responsável pela produção

do *Caso Empresarial* e os fluxos *Avaliar escopo e risco do projeto* e *Planejar para a próxima iteração* são responsáveis apenas por atualizá-lo com informações sobre escopo de projeto e mudanças em orçamento e/ou prazo respectivamente.

Os exemplos acima mostram que as atividades executam ações diferentes nos fluxos. Neste contexto, para que em um processo de adaptação apenas algumas destas ações possam ser excluídas ou incluídas é necessário que sejam consideradas como atividades únicas nos fluxos e não agrupadas em apenas uma atividade que é executada em muitos fluxos como é previsto no RUP.

Por outro lado, existem também atividades que são executadas em vários fluxos e realmente possuem o mesmo propósito em todos eles. Exemplos destas atividades são as atividades *Capturar um vocabulário comum* e *Gerenciar dependências*. A atividade *Capturar um vocabulário comum* é executada nos fluxos *Analisar o problema*, *Entender necessidades dos stakeholders* e *Definir o sistema*. No fluxo *Analisar o problema* o propósito é produzir o artefato *Glossário* e nos fluxos *Entender necessidades dos stakeholders* e *Definir o sistema* o propósito é atualizar tal artefato, sendo exatamente a mesma atualização nos dois fluxos.

A atividade *Gerenciar dependências* possui o mesmo propósito em todos os fluxos onde é executada, sendo estes *Entender necessidades dos stakeholders*, *Definir o sistema*, *Gerenciar o escopo do sistema* e *Gerenciar mudança dos requisitos*. O propósito da atividade em todos os fluxos tem a mesma descrição e é a atualização dos artefatos *Plano de Gerência de Requisitos*, *Documento de Visão* e *Atributos de Requisitos*.

Os exemplos descritos acima mostram que embora a associação entre as classes fluxo e atividade seja definida como uma associação muitos para muitos é necessário analisar o propósito das atividades em cada fluxo onde estas são executadas. Isto porque, muitas vezes uma mesma atividade possui propósitos diferentes nos diversos fluxos, caracterizando assim novas atividades. Neste sentido, para que em processos de adaptação as ações corretas sejam incluídas e/ou excluídas do processo durante a inclusão e/ou exclusão de atividades é necessário que estas sejam devidamente diferenciadas nos fluxos onde são executadas, sendo consideradas a mesma atividade somente quando possuem o mesmo propósito.

5.5. Estrutura do Meta-modelo

As classes (seção 5.1) e associações (seção 5.3) do meta-modelo proposto adotam a mesma estrutura em termos de pacotes sugerida pelo meta-modelo RUP. Isto porque, como já dito anteriormente, trata-se de uma extensão a este meta-modelo.

O RUP apresenta uma divisão de três pacotes para seu meta-modelo conforme mostra a Figura 10:

- ***Process Structure*** **(Pacote Estrutura de processo):** usado para definir os conceitos que representam os elementos de processo e como eles se relacionam uns com os outros.
- ***Process Components*** **(Pacote Componentes de processo):** usado para definir conceitos que suportam extensão e empacotamento do processo.
- ***External Descriptions*** **(Pacote Descrições Externas):** usado para definir a taxonomia dos arquivos que fornecem descrições para os elementos do processo.

O pacote relativo aos elementos e associações do meta-modelo RUP é o pacote Estrutura de processo. Este pacote determina os conceitos que definem os elementos individuais de um processo.

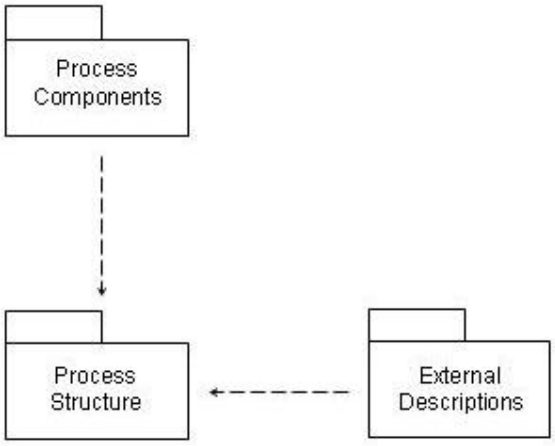

Figura 10 - Pacotes do meta-modelo RUP (extraído de (IBM, 2005a))

Em (IBM, 2005a), uma classificação quanto aos tipos de elementos que compõem o pacote Estrutura de Processo é proposta dividindo-os em elementos do tipo classe e elementos do tipo operação. A perspectiva na modelagem do processo é que o tipo classe represente as "coisas" do processo, e o tipo operação seja usado para descrever o "comportamento" dos elementos classe (Figura 11).

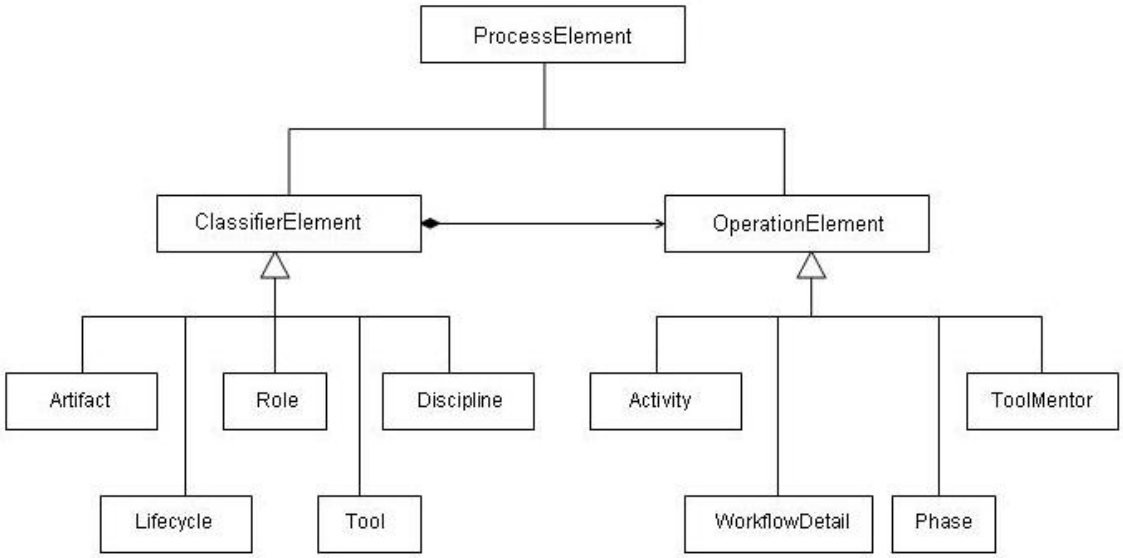

Figura 11 - Pacote Estrutura de Processo (extraído de (IBM, 2005a))

No meta-modelo proposto, as classes e associações seguem a mesma classificação apresentada em (IBM, 2005a), embora uma extensão ao pacote Estrutura de Processo (Figura 12) tenha sido realizada para a inclusão e exclusão de alguns elementos.

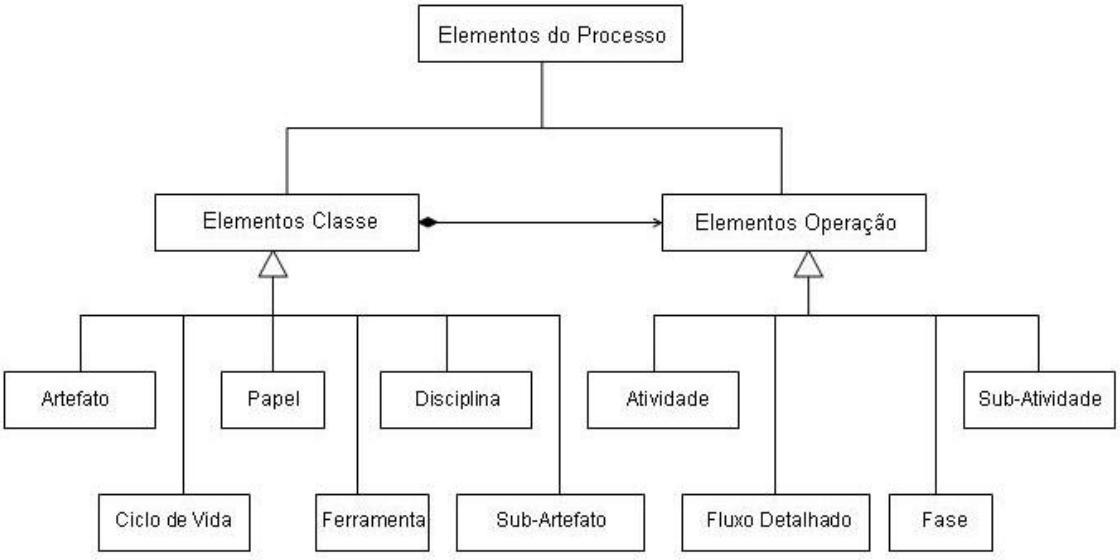

Figura 12 - Extensão ao Pacote Estrutura de Processo para o Meta-modelo Proposto

Como pode ser visto, os elementos sub-artefato e tarefa foram incluídos ao pacote Estrutura de processo, enquanto o elemento *toolmentor* (mentor de ferramenta) foi excluído.

A inclusão de sub-artefato e tarefa foi realizada visto que neste trabalho estes elementos são considerados como parte do processo de desenvolvimento de software, conforme já descrito na seção 5.1. Quanto à classificação destes elementos, procurou-se utilizar a mesma lógica empregada pelo RUP, sendo sub-artefatos classificados como elementos do tipo classe e tarefas como elementos do tipo operação.

A classificação de sub-artefato como um elemento do tipo classe deve-se ao fato deste ser considerado uma parte do elemento artefato, classificado pelo RUP como um elemento do tipo classe. Já a classificação de tarefa como um elemento do tipo operação também deve-se ao fato deste ser considerado uma parte do elemento atividade, classificado pelo RUP como um elemento do tipo operação.

Por fim, a exclusão do elemento *toolmentor* foi realizada no pacote Estrutura de Processo devido ao fato deste não ter sido considerado no meta-modelo proposto como um elemento de processo, conforme já explicado na seção 5.2.

5.6. Considerações Finais

Este capítulo apresentou uma proposta de extensão ao meta-modelo RUP para suporte ao processo de adaptação. O meta-modelo descrito é relevante devido ao fato de apresentar os principais elementos e relacionamentos de um processo padrão a serem considerados durante sua adaptação.

No próximo capítulo, um conjunto de assertivas estruturais para adaptação de processos definidas a partir deste meta-modelo será apresentado. A contribuição principal das assertivas será o apoio à adaptação de processos de software, visando manter conformidade com o processo padrão.

6. ASSERTIVAS ESTRUTURAIS PARA ADAPTAÇÃO DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

Este capítulo apresenta um conjunto de assertivas estruturais para adaptação de processos de desenvolvimento de software definidas a partir do meta-modelo proposto neste trabalho. Ainda, descreve como estas foram definidas e como devem ser utilizadas durante o processo de adaptação.

Neste trabalho, para tornar possível a adaptação de processos de desenvolvimento de software a partir do meta-modelo proposto (capítulo 5), um conjunto de assertivas estruturais para este propósito foi definido. Estas assertivas estruturais estabelecem como os elementos e relacionamentos do processo padrão devem ser tratados durante sua adaptação, de forma a garantir consistência no processo resultante.

As operações de exclusão e inclusão de elementos estão sendo consideradas para o processo de adaptação. Desta forma, o conjunto de assertivas estruturais para adaptação de processos de desenvolvimento de software diz respeito a estas operações.

As operações foram baseadas nas abordagens de (Welzel *et al*, 1995), (Ginsberg & Quinn, 1995), (Yoon *et al*, 2001), (Coelho, 2003) e definem quais elementos do meta-modelo podem ser incluídos e/ou excluídos durante a adaptação do processo padrão. Cada operação apresenta um conjunto de assertivas estruturais, sendo que estas devem ser devidamente seguidas de forma a garantir que inconsistências não sejam geradas no processo resultante. É através destas operações e suas regras que o uso do meta-modelo proposto se torna possível na adaptação de um processo padrão de desenvolvimento de software.

As assertivas estruturais para cada operação foram definidas de maneira a analisar o impacto que cada elemento do processo sofre durante sua execução. A seguir, o conjunto das assertivas estruturais para as operações de inclusão e exclusão de instâncias da classe atividade são mostradas na Tabela 13.

Tabela 13 - Regras para Exclusão e Inclusão de Atividades

Elemento do Processo	Exclusão	Inclusão
Atividade	- Atividades relacionadas (anterior e próxima) devem ser conectadas; - Atividades que consomem sub-artefatos (somente os não opcionais) excluídos pela atividade devem ser eliminadas.	- Definir atividades relacionadas (anterior e próxima).
Ferramenta	- Atividade deve ter suas relações de associação com ferramentas eliminadas.	**-** Se necessário, associar uma ou mais ferramentas a atividade.
Fluxo	- Atividade deve ter suas relações de associação com fluxos eliminadas, sendo que fluxos que não possuem associações com atividades devem ser excluídos.	- Associar a atividade a um ou mais fluxos.
Papel	- Atividade deve ter sua relação de associação com o papel eliminada.	- Associar a atividade a um papel.
Sub-artefato	- Sub-artefatos produzidos pela atividade devem ser excluídos juntamente com seus sub-artefatos dependentes (isto pode implicar em exclusão de outras atividades); - Sub-artefatos consumidos pela atividade devem ter sua relação de associação eliminada; - Sub-artefatos modificados pela atividade devem ter sua relação de associação eliminada.	- Se necessário, associar sub-artefatos a serem consumidos (definindo os opcionais), produzidos ou modificados pela atividade.
Tarefa	- Tarefas pertencentes à atividade devem ser excluídas.	- Se necessário, tarefas devem ser incluídas.

Conforme se pode observar na Tabela 13, a exclusão de instâncias da classe atividade impacta em muitos outros elementos de um processo padrão durante sua adaptação. Já sua inclusão ocasiona a definição de várias informações como fluxo(s) onde a nova atividade deve ser executada, suas tarefas, o papel responsável, ferramentas para sua execução e a definição de suas atividades relacionadas. Entretanto, embora todos os impactos estejam sendo previstos, é importante considerar que a exclusão de uma instância da classe atividade só pode ser realizada se a mesma não é obrigatória no processo padrão.

Em verdade, a obrigatoriedade de um elemento sempre cancela a operação de exclusão, isto porque, todos os elementos definidos como obrigatórios em um processo padrão devem ser mantidos em todos os processos adaptados (Chrissis *et al*, 2003). Como já mostrado no capítulo 5, os elementos do meta-modelo proposto que podem ser obrigatórios

são as disciplinas e atividades. Neste sentido, deve-se considerar que em todas as operações onde um destes elementos for obrigatório, esta deve ser cancelada.

A seguir, as assertivas estruturais das operações de inclusão e exclusão de instâncias das classes disciplina, fluxo, ferramenta, artefato e sub-artefato serão mostradas e discutidas em detalhes.

Na tabela 14, se pode perceber que a exclusão de elementos do tipo disciplina possui impacto em poucos elementos do processo, embora seu impacto sobre o elemento fluxo leve a execução de outras operações de exclusão relacionadas com as regras apresentadas na Tabela 15. Ainda, apenas assertivas estruturais para exclusão de disciplinas são mostradas na Tabela 14, não sendo referenciadas regras para sua inclusão. Isto acontece porque estes elementos não podem ser incluídos durante a adaptação de um processo padrão de desenvolvimento de software. De acordo com (IBM, 2005a), o responsável pela definição das disciplinas deve ser o engenheiro de processo e isto deve ser realizado quando um novo processo de desenvolvimento de software é definido em uma organização. Neste sentido, considera-se que se uma nova disciplina precisar ser incluída isto deve ser realizado no âmbito da manutenção dos processos padrão da organização e não no contexto de seus projetos de software.

Por fim, como já dito anteriormente, disciplinas obrigatórias levam ao cancelamento da exclusão.

Tabela 14 - Regras para Exclusão de Disciplinas

	Exclusão
Ciclo de Vida	- Excluir a relação de associação da disciplina com o ciclo de vida a qual esta associada, sendo que se a disciplina é única no ciclo de vida este deve ser também excluído.
Fluxo	- Excluir todos os fluxos pertencentes à disciplina.

A Tabela 15 mostra que a exclusão de instâncias da classe fluxo ocasiona a exclusão de instâncias da classe atividade, disciplina e fase do processo. Isto leva ainda a possíveis exclusões de outros elementos como tarefas, ferramentas, artefatos e sub-artefatos considerando que estas exclusões podem propagar-se pelo processo. A operação de inclusão implica na escolha dos elementos que ficarão relacionados ao nova instância de fluxo como atividades, disciplina, fases e fluxos relacionados.

Tabela 15 - Regras para Exclusão e Inclusão de Fluxos

	Exclusão	Inclusão
Atividade	- Excluir a relação de associação das atividades associadas ao fluxo, sendo que atividades que não possuem associação com outros fluxos devem ser excluídas.	- Incluir pelo menos uma atividade no novo fluxo.
Disciplina	- Excluir o fluxo da disciplina a que pertence, sendo que se o fluxo é único na disciplina esta deve ser também excluída.	- Associar o novo fluxo a uma disciplina.
Fase	- Excluir a relação de associação do fluxo com a(s) fase(s) a qual está associado, sendo que fases que não possuem associações com outros fluxos devem ser excluídas.	- Associar o novo fluxo às fases onde ele será executado, sendo que é obrigatória a associação em pelo menos uma fase.
Fluxo	- Conectar fluxos relacionados (anterior e próximo).	- Definir fluxos relacionados (anterior e próximo).

Os elementos do tipo ferramenta também podem ser incluídos e excluídos de um processo específico de projeto. Para isto, as seguintes assertivas estruturais devem ser respeitadas (Tabela 16):

Tabela 16 - Regras para Exclusão e Inclusão de Ferramentas

	Exclusão	Inclusão
Atividade	- Excluir a relação de associação da ferramenta com as atividades a qual está associada, sendo que ferramentas que não possuem associações com outras atividades devem ser excluídas.	- Associar a nova ferramenta a pelo menos uma atividade.

As operações mostradas na Tabela 16 são as com menor impacto sobre outros elementos do processo. É possível constatar que a inclusão e exclusão de instâncias da classe ferramentas afeta apenas o elemento atividade sendo que nenhum outro elemento do processo pode ser impactado.

Por fim, outros elementos que podem ser incluídos e excluídos do processo padrão durante sua adaptação são artefatos e sub-artefatos. Desta forma, as assertivas estruturais para estas operações são mostradas respectivamente nas Tabelas 17 e 18.

A Tabela 17 mostra que a inclusão e exclusão de instâncias da classe artefato afeta apenas o elemento de processo sub-artefato. Como visto, sua exclusão origina a exclusão de todos seus sub-artefatos, sendo esta regra de boa formação definida para respeitar a associação de composição entre as classes artefato e sub-artefato definida no meta-modelo proposto. Ainda, a regra de boa formação para inclusão de pelo menos uma instância da classe

sub-artefato no novo artefato também foi definida devido ao meta-modelo restringir que artefatos só podem existir no processo quando possuem pelo menos um sub-artefato.

Tabela 17 - Regras para Exclusão e Inclusão de Artefatos

	Exclusão	**Inclusão**
Sub-artefato	- Excluir todos sub-artefatos pertencentes ao artefato.	- Incluir pelo menos um sub-artefato no novo artefato.

A exclusão de instâncias da classe sub-artefato durante um processo de adaptação pode ser realizada para contemplar algumas mudanças necessárias em algum artefato do processo padrão. Isto ocorre pois muitas vezes versões mais simplificadas de alguns artefatos precisam ser produzidas no contexto de alguns projetos dispensando algumas de suas partes. Embora esta seja uma operação bastante comum em processos de adaptação, conforme previsto por (Welzel *et al*, 1995), (Yoon *et al*, 2001) e (Coelho, 2003), a Tabela 18 mostra que ela ocasiona a exclusão de outros elementos do processo como sub-artefatos e atividades. A exclusão de outros sub-artefatos é devido à dependência que estes podem ter com o sub-artefato a ser excluído, enquanto que a exclusão de outras atividades refere-se a aquelas que de alguma forma tornaram-se invalidadas por esta operação.

Tabela 18 - Regras para Exclusão e Inclusão de Sub-Artefatos

	Exclusão	**Inclusão**
Artefato	- Excluir o sub-artefato do artefato a que pertence, sendo que artefatos que não possuem sub-artefatos devem ser excluídos.	- Associar o novo sub-artefato a um artefato.
Atividade	- Excluir a atividade de produção do sub-artefato. - Desconectar os sub-artefatos excluídos da entrada das atividades aos quais eles estão associados, sendo que onde o consumo é obrigatório as atividades também devem ser excluídas.	- Associar os sub-artefatos necessários para criação do novo sub-artefato como entrada na sua atividade de produção.
Papel	- Excluir a relação de associação do sub-artefato com os papéis a qual está associado.	- Associar um papel como responsável do novo sub-artefato. - Verificar se é necessário associar papéis como modificares do novo sub-artefato.

Sub-artefato	- Excluir os sub-artefatos que mantém relação de dependência com o sub-artefato a ser excluído.	- Definir atividade de produção para o novo sub-artefato. - Verificar se o novo sub-artefato é dependente de outros sub-artefatos para que tais relacionamentos sejam criados. - Verificar se é necessário criar relação de dependência de algum sub-artefato existente para o novo sub-artefato;

A inclusão de instâncias da classe sub-artefato é a operação que permite partes de um artefato serem incluídas a um processo específico de projeto atendendo assim a suas necessidades particulares. Esta operação implica na escolha dos elementos que ficarão relacionados ao novo sub-artefato como artefato ao qual ele pertence, papel responsável, papéis para modificação, atividade de produção, atividades que irão consumir e/ou modificar o novo sub-artefato, sub-artefatos dependentes no novo sub-artefato e ainda, de quais sub-artefatos o novo sub-artefato irá depender. Entretanto, esta regra de boa formação em específico apresenta uma importante restrição em se tratando de projetos de manutenção. A restrição é que em projetos deste tipo não é necessário a definição de uma atividade de produção. Isto acontece porque projetos de manutenção utilizam na maioria das vezes documentação originada a partir de outros projetos. Neste sentido, a restrição para esta regra de boa formação define que processos com valor de atributo tipoCiclo da classe Ciclo de Vida igual a Manutenção não necessitam obrigatoriamente definir atividade de produção para os sub-artefatos.

6.1. Considerações Finais

Este capítulo teve como principal objetivo apresentar o conjunto de assertivas estruturais definidas nesta pesquisa para inclusão e exclusão de elementos do processo padrão durante sua adaptação. Através destas regras é possível adaptar um processo padrão mantendo a integridade do processo resultante. Isto garante a consistência entre os elementos e relacionamentos do processo resultante, garantindo a conformidade com o processo padrão, bem como evitando as falhas de execução pelas inconformidades geradas durante a adaptação do processo, conforme identificado em (Jalote, 2002).

No próximo capítulo como forma de avaliar o meta-modelo e as assertivas estruturais para inclusão e exclusão de atividades propostas neste trabalho, um exemplo de uso destes

será apresentado. Ainda, serão descritos o processo de condução da avaliação e seus resultados.

7. AVALIAÇÃO DO META-MODELO E ASSERTIVAS ESTRUTURAIS PROPOSTAS

Este capítulo descreve o uso do protótipo (apêndice I) em um cenário exemplo criado a partir do processo RUP. O objetivo é avaliar o meta-modelo e assertivas estruturais propostas neste trabalho.

Para avaliar a consistência e a viabilidade do meta-modelo e assertivas estruturais propostas neste trabalho, um cenário exemplo criado a partir do processo RUP é utilizado. O cenário consiste na criação de um processo padrão de desenvolvimento de software utilizando o meta-modelo proposto e a geração de um processo específico de projeto proveniente da adaptação deste processo padrão, sendo esta realizada a partir da aplicação das assertivas estruturais. Nas seções seguintes, a abordagem utilizada para construção do cenário exemplo é descrita em maiores detalhes, assim como seu processo de condução e os resultados encontrados.

7.1. Abordagem Utilizada

O cenário utilizado para avaliação do meta-modelo e das assertivas estruturais envolve as disciplinas de Requisito e Análise e Projeto do processo RUP. Toda a sua avaliação é conduzida utilizando o protótipo apresentado no Apêndice I, sendo sua construção composta pelas seguintes etapas:

Etapa 1: Criação do Processo Padrão a partir do Meta-modelo. Inicialmente, um estudo aprofundado em todo o processo RUP foi realizado. Este estudo permitiu a definição de instâncias para todos os elementos do meta-modelo, bem como o estabelecimento das associações entre estes elementos. Para o cenário exemplo, foram considerados apenas elementos relativos às disciplinas de Requisito e Análise e Projeto do RUP, tendo esta o objetivo de avaliar o meta-modelo e assertivas estruturais. Entretanto, deve-se considerar que a escolha de tais disciplinas do RUP foi feita de maneira aleatória, sendo que se outras disciplinas tivessem sido selecionadas a avaliação poderia ser realizada da mesma forma.

Etapa 2: Caracterização de um Projeto de Desenvolvimento de Software. Para permitir que o processo padrão definido na Etapa 1 fosse adaptado gerando um processo

específico de projeto, estudos foram realizados no RUP com objetivo de caracterizar um projeto de desenvolvimento de software. Estes estudos foram baseados principalmente nos guias sugeridos pelo RUP para desempenho de suas disciplinas nos projetos de software, sendo estes guias encontrados em (IBM, 2005b). Assim, toda a adaptação do processo padrão para um projeto de desenvolvimento de software conduzida nesta avaliação é baseada nas decisões propostas pelos guias do RUP, sendo estes guias discutidos na seção 7.2.2 deste trabalho.

Etapa 3: Adaptação do Processo Padrão para o Processo Específico de Projeto. Com base nas decisões sugeridas pelos guias do RUP para o desempenho de suas disciplinas de Requisito e Análise e Projeto a adaptação do processo padrão criado na Etapa 1 foi realizada. Para isto, foram utilizadas as assertivas estruturais para inclusão e exclusão de atividades do processo padrão.

Etapa 4: Avaliação dos Resultados. Finalmente, foram analisados os resultados encontrados com o uso do cenário exemplo a partir do processo resultante da adaptação do processo padrão de desenvolvimento de software. Esta análise consiste em determinar se a utilização do meta-modelo e assertivas estruturais propostas realmente garantiram a conformidade do processo padrão durante sua adaptação.

7.2. Processo de Condução

O processo de condução para criação e avaliação do cenário exemplo é realizado através das quatro etapas identificadas na seção anterior. A seguir, estas etapas são descritas detalhadamente.

7.2.1. Etapa 1: Criação do Processo Padrão a partir do Meta-modelo

Nesta etapa, a criação do processo padrão de desenvolvimento de software no protótipo é realizada. Para isto, inicialmente, o conjunto de elementos e associações definidas para este processo são incluídos no protótipo, de forma a permitir sua posterior seleção. Isto porque, conforme detalhado no Apêndice I, a criação de um processo padrão no protótipo é realizada a partir de um conjunto de informações previamente cadastradas. A Figura 13 ilustra a interface de visualização de processos do protótipo contendo o processo padrão cadastrado.

As disciplinas de Requisito e Análise e Projeto do RUP foram detalhadas de maneira a contemplar todo o conjunto de informações previstas pelo meta-modelo proposto. Isto se refere à definição dos elementos previstos pelo processo RUP (fases, disciplinas, fluxos

detalhados, atividades, papéis, artefatos e ferramentas), além da definição de tarefas e sub-artefatos. Ainda, definições para ordem de execução de fluxos, atividades e tarefas foram criadas. Por fim, as relações de dependências entre os sub-artefatos também foram incluídas.

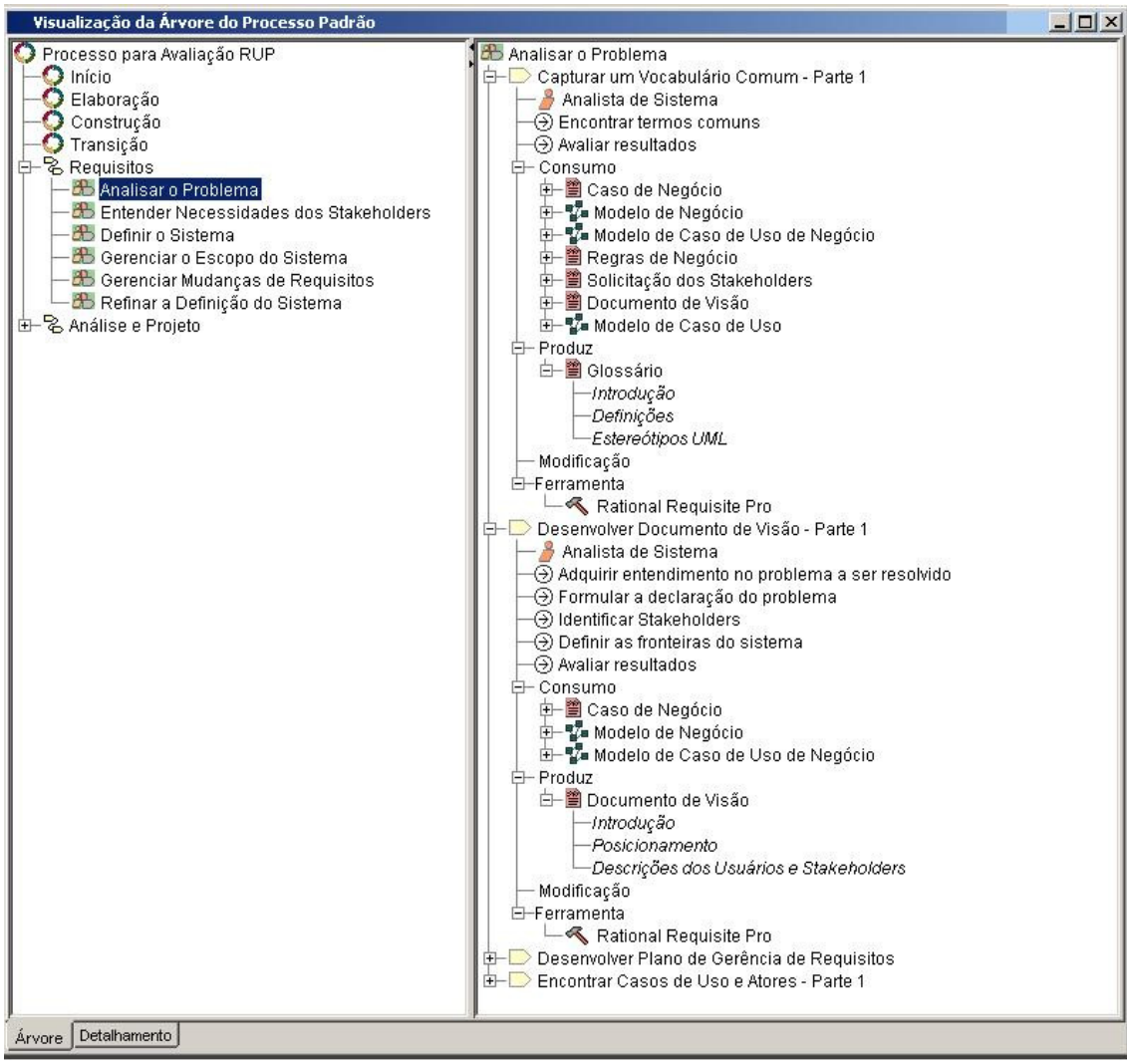

Figura 13 – Processo padrão definido para avaliação

7.2.2. Etapa 2: Caracterização de um Projeto de Desenvolvimento de Software

A partir da definição do processo padrão, a caracterização de um projeto de desenvolvimento de software é realizada com o objetivo de definir quais atividades devem constituir seu processo de desenvolvimento. A caracterização do projeto é feita com base nos guias propostos pelo RUP para a execução de suas disciplinas em projetos de desenvolvimento de software.

Relacionados aos guias das disciplinas de Requisito e Análise e Projeto é possível identificar que muitos de seus artefatos ou parte deles são opcionais, implicando portanto na

opcionalidade de suas atividades de produção. Com base nestas informações, considerou-se para o cenário exemplo que a adaptação do processo padrão para a geração do processo específico de projeto seria baseada na exclusão de atividades que produzem artefatos opcionais do RUP (IBM, 2005b).

Desta forma, para a disciplina de Requisito, definiu-se a exclusão de todas as atividades com produção de artefatos opcionais. Ainda, para a disciplina de Análise e Projeto, considerou-se a exclusão das atividades com produção de artefatos opcionais para desenvolvimento em tempo real e para o *Modelo de Deployment*. A redução do número de atividades escolhidas para exclusão na disciplina de Análise e Projeto foi feita apenas como forma de delimitar um escopo apropriado para o cenário. Na Tabela 19, tem-se a relação das atividades definidas para exclusão do processo padrão de desenvolvimento de software durante sua adaptação.

Tabela 19 – Atividades a serem Excluídas durante a Adaptação do Processo Padrão.

Atividades da Disciplina Requisito
Desenvolver Plano de Gerência de Requisitos (Produz artefato opcional Plano de Gerência de Requisitos)
Produzir *StoryBoard* (Produz artefato opcional *StoryBoard*)
Encontrar Atributos de Requisitos (Produz artefato opcional Atributos de Requisitos)
Detalhar os Requisitos de Software (Produz artefato opcional Especificação de Requisitos de Software)
Atividades da Disciplina Análise e Projeto
Identificar Elementos do Projeto - Parte 2 (Produz os sub-artefatos opcionais Cápsula, Protocolo e *Signal* do artefato Modelo de Projeto)
Análise dos Casos de Uso – Parte 2 (Produz o sub-artefato opcional Evento do artefato Modelo de Projeto)
Análise Arquitetural – Parte 2 (Produz o artefato opcional Modelo de *Deployment* e sub-artefato opcional Visão de *Deployment* do artefato Documento de Arquitetura deSoftware)

Todas as atividades mostradas na Tabela 19 fazem parte do processo padrão inserido no protótipo. Entretanto, é importante considerar que algumas atividades do RUP foram decompostas com objetivo de criar um processo padrão com alta coesão e baixo acoplamento, permitindo sua adaptação de forma consistente e mantendo a integridade do processo resultante.

A decomposição de atividades a partir do RUP é necessária, pois conforme já citado na seção 5.4, muitas vezes uma mesma atividade possui propósitos diferentes nos diversos fluxos, caracterizando assim novas atividades. No cenário exemplo (Tabela 19), estas atividades receberam apenas um identificador no fim de seus nomes, como por exemplo,

atividade *Identificar Elementos do Projeto - Parte 2*, caracterizando que são atividades diferentes nos diversos fluxos onde são executadas. Ainda, algumas atividades com produção de artefatos obrigatórios e opcionais do processo também foram decompostas. Isto tornou possível dividi-las em atividades obrigatórias e atividades opcionais.

7.2.3. Etapa 3: Adaptação do Processo Padrão para o Processo Específico de Projeto.

Neste ponto, tendo completado as etapas 1 e 2, a adaptação do processo padrão de desenvolvimento de software para o processo específico do projeto é realizada, consistindo assim a etapa 3. Nesta etapa, o uso do protótipo é realizado do início ao fim. Primeiramente, a escolha do processo padrão definido na etapa 1 é feita a partir de todos os processos padrão disponíveis no protótipo, conforme mostra a Figura 14.

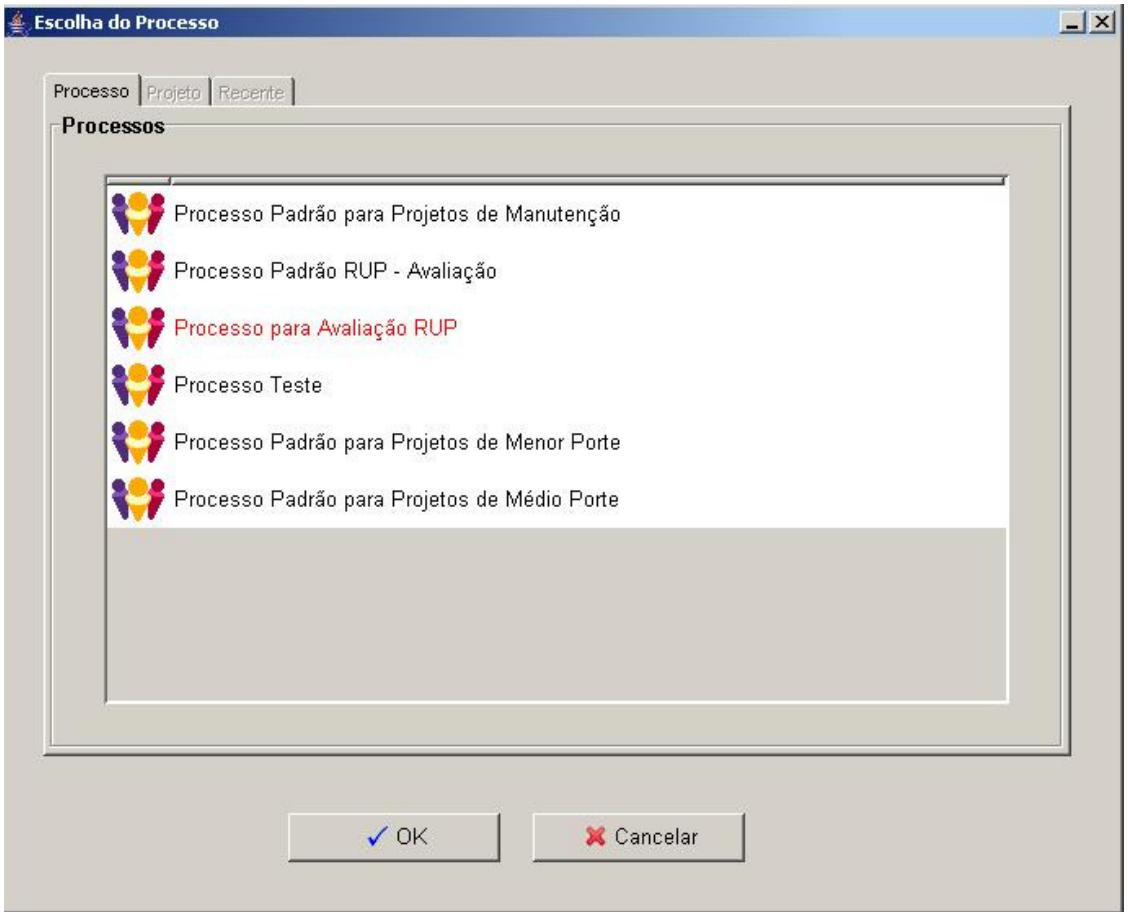

Figura 14 – Escolha do processo padrão definido para a avaliação

A partir disto, o processo selecionado torna-se disponível para adaptação, bem como as operações para exclusão e inclusão de atividades. Ainda, através da operação de exclusão disponibiliza-se a funcionalidade de análise de impacto nos elementos do processo padrão

durante a exclusão de atividades. Esta funcionalidade está baseada na ferramenta RUP *Builder* (seção 3.2.2.1) e permite conhecer todo o efeito causado nos outros elementos do processo durante a exclusão de uma atividade, permitindo ao responsável pela adaptação prosseguir ou cancelar a exclusão.

Por exemplo, quando a atividade *Desenvolver Plano de Gerência de Requisitos* (Tabela 19) foi excluída, mostrou-se o impacto sofrido por todos os outros elementos do processo alertando quais deles seriam afetados por esta exclusão. Neste momento, tanto a disciplina Requisito como os fluxos *Analisar o Problema, Definir o Sistema, Gerenciar o Escopo do Sistema, Entender Necessidades dos Stakeholders* e *Gerenciar Mudanças nos Requisitos* receberam ícones de alerta indicando que alguns de seus elementos, atividades e/ou artefatos, foram impactados pela exclusão da atividade. O impacto apresentado é baseado nas assertivas estruturais (capítulo 6) definidas para a operação de exclusão de atividades. A Figura 15 mostra parte do impacto causado pela exclusão da atividade *Desenvolver Plano de Gerência de Requisitos,* bem como apresenta os botões de acesso as funcionalidades de exclusão e inclusão de atividades.

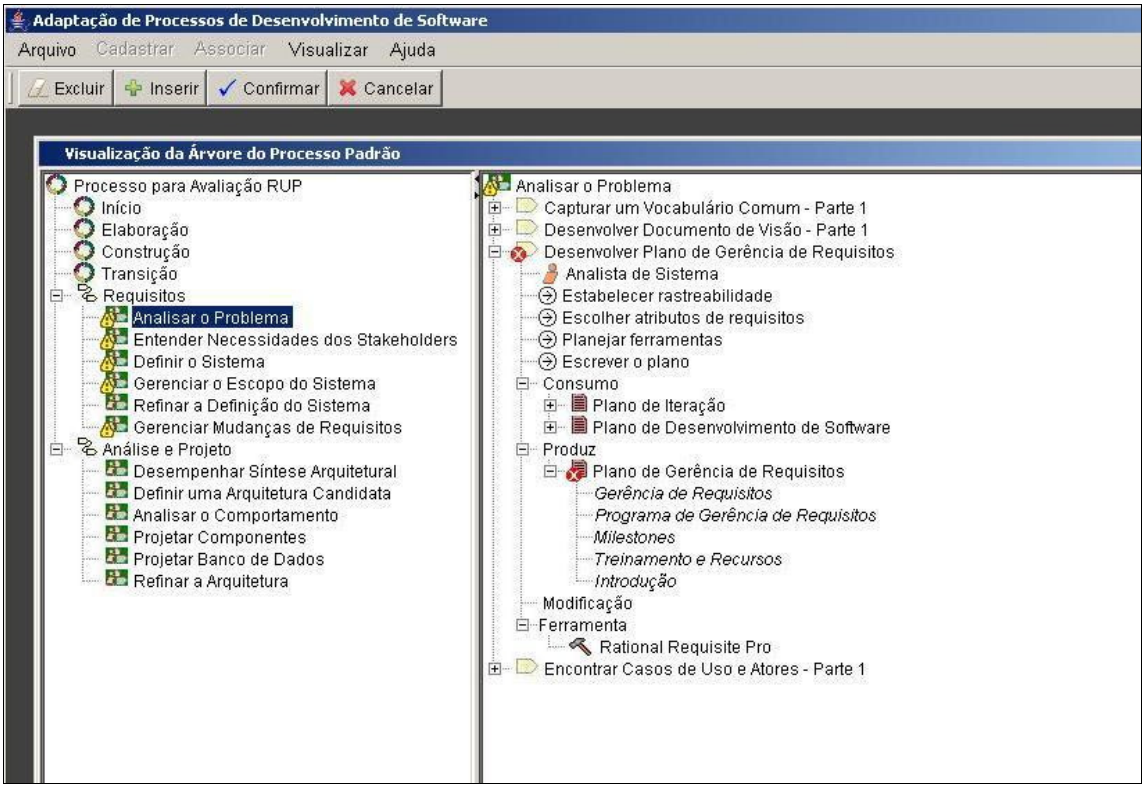

Figura 15 – Processo Padrão para Avaliação e Funcionalidades de Exclusão e Inclusão de Atividades

Os impactos em todo o processo puderam ser verificados através dos fluxos com ícones de alerta, sendo estes resumidamente descritos a seguir. A descrição diz respeito a demonstrar a relação de seus elementos impactados, bem como o efeito e a causa (assertivas estruturais) para os mesmos.

7.2.3.1. Impactos do Fluxo Analisar o Problema

No fluxo Analisar o Problema encontra-se a atividade excluída *Desenvolver Plano de Gerência de Requisitos.* Os elementos afetados neste fluxo referem-se somente a esta atividade, sendo estes mostrados na Tabela 20.

Tabela 20 – Elementos Impactados pela Exclusão da Atividade Desenvolver Plano de Gerência de Requisitos no Fluxo Analisar o Problema

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefatos de: - Plano de Gerência de Requisitos	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-artefatos de: - Plano de Iteração - Plano de Desenvolvimento de Software	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
Fluxo	Analisar o Problema	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	*Rational Requisite Pro*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Analista de Sistema	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Analista de Sistema	Relação de associação com sub-artefatos do Plano de Gerência de Requisitos excluída	Sub-artefatos excluídos devem ter suas relações de associação com papéis eliminadas.
Tarefa	Tarefas de Desenvolver Plano de Gerência de Requisitos	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

Ainda, referente às informações da Tabela 20, deve-se considerar que no fluxo Analisar o Problema todas as atividades são executadas em paralelo. Desta forma, as

atividades relacionadas (anteriores e próximas) para a atividade *Desenvolver Plano de Gerência de Requisitos* não precisaram ser conectadas, necessitando apenas removê-la da estrutura de paralelismo no fluxo.

7.2.3.2. Impactos do Fluxo Entender Necessidades dos *Stakeholders*

A exclusão da atividade *Desenvolver Plano de Gerência de Requisitos* causou a exclusão das atividade *Gerenciar Dependências* e *Encontrar Atributos dos Requisitos* pertencentes a este fluxo. Ambas as exclusões deveram-se ao fato destas atividades consumirem obrigatoriamente o artefato *Plano de Gerência de Requisitos* excluído do processo no fluxo Analisar o Problema. As Tabelas 21 e 22 apresentam todos elementos afetados pela exclusão das atividades *Gerenciar Dependências* e *Encontrar Atributos dos Requisitos* respectivamente.

Tabela 21 - Elementos Impactados pela Exclusão da Atividade Gerenciar Dependências no Fluxo Entender Necessidades dos *Stakeholders*

Elemento	Instância	Impacto	Regra de boa formação
Sub-Artefato	Sub-Artefatos de: - Solicitação dos *Stakeholders* - Documento de Visão - Plano de Gerência de Requisitos - Modelo de Caso de Uso - *Change Request* - Atributos de Requisito - Lista de Riscos - Especificação Complementar	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
	Sub-Artefatos de: - Documento de Visão - Plano de Gerência de Requisitos - Atributos de Requisito	Relação de saída (modificação) com atividade excluída	Sub-artefatos modificados pela atividade devem ter relação de associação eliminada
Fluxo	- Entender Necessidades dos *Stakeholders*	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	- *Rational Requisite Pro* - *Rational ClearQuest* - *Rational SoDA*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Analista de Sistema	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com

			papéis eliminadas.
Tarefa	Tarefas de Gerenciar Dependências	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

Tabela 22 - Elementos Impactados pela Exclusão da Atividade Encontrar Atributos de Requisitos no Fluxo Entender Necessidades dos *Stakeholders*

Elemento	Instância	Impacto	Regra de boa formação
Sub-Artefato	Sub-Artefatos de: - Atributos de Requisito	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-Artefatos de: - Modelo de Negócio - Modelo de Caso de Uso de Negócio - Documento de Visão - Solicitação dos *Stakeholders*	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
Fluxo	- Entender Necessidades dos *Stakeholders*	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	- *Rational Requisite Pro*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Analista de Sistema	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Analista de Sistema	Relação de associação com sub-artefatos de Atributos de Requisitos excluída	Sub-artefatos excluídos devem ter suas relações de associação com papéis eliminadas.
Tarefa	Tarefas de Encontrar Atributos de Requisitos	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

Ambas as atividades excluídas *Gerenciar Dependências* e *Encontrar Atributos dos Requisitos* são executadas em paralelo a outras atividades no fluxo Entender Necessidades dos *Stakeholders*. Desta forma, atividades relacionadas (anteriores e próximas) não precisaram ser conectadas, necessitando apenas removê-las da estrutura de paralelismo.

7.2.3.3. Impactos dos Fluxos Definir o Sistema, Gerenciar o Escopo do Sistema e Gerenciar Mudanças nos Requisitos

Os impactos causados pela exclusão da atividade *Desenvolver Plano de Gerência de Requisitos* são os mesmos nos fluxos *Definir o Sistema*, *Gerenciar o Escopo do Sistema* e *Gerenciar Mudanças nos Requisitos*. Este impacto refere-se a exclusão da atividade *Gerenciar Dependências* já justificado na seção anterior. Desta forma, todos elementos afetados nestes fluxos estão também relacionados à Tabela 21.

Neste ponto, tendo sido analisado todos os impactos da exclusão da atividade *Desenvolver Plano de Gerência de Requisitos,* o responsável pela adaptação decide pela sua confirmação ou cancelamento. A confirmação da operação implica na exclusão ou modificação de elementos de forma a respeitar as assertivas estruturais. Já seu cancelamento apenas remove os ícones de alerta dos elementos e nenhuma exclusão ou alteração é realizada. Ainda, ambas as operações podem ser acessadas pelos botões *Confirmar* e *Cancelar* mostrados na interface da Figura 15.

Para o cenário exemplo, confirmou-se a exclusão da atividade em questão, verificando assim todos os impactos descritos nas Tabelas 20, 21 e 22 ser executados. Desta forma, se pode perceber uma reconfiguração automática em vários elementos do processo, como por exemplo, a retirada das atividades excluídas nos fluxos do processo e na seqüência de execução das atividades destes fluxos.

Dando andamento ao processo de adaptação, as outras atividades da Tabela 19 foram excluídas. Para cada uma delas a análise de impacto nos fluxos foi verificada, sendo estas apresentadas no Apêndice II deste documento. Embora estas análises tenham mostrado que vários elementos do processo seriam afetados pelas exclusões, todas estas puderam ser realizadas, o que já constituiu um ponto positivo para os resultados desta avaliação. Isto porque, demonstrou que o processo padrão construído para a mesma foi bem modelado, isto em termos de dependências, relações de consumo, produção e modificação. Considera-se isto, pois após a exclusão de várias atividades sugeridas pelos guias do RUP como opcionais chegou-se ao mesmo conjunto de elementos restantes em seu processo, ou seja, nenhum outro elemento considerado importante (recomendado) no processo RUP foi excluído pelas assertivas estruturais propostas neste trabalho. Apenas o que se garantiu foi a consistência entre os elementos no processo resultante.

Por fim, antes de finalizar o processo de adaptação criando o projeto ao qual o processo resultante ficaria associado, percebeu-se que nenhuma atividade havia sido incluída no processo padrão durante sua adaptação, isto porque o RUP não propõe em seus guias inclusão de novas atividades. Assim, apenas como forma de tornar possível a avaliação das

assertivas estruturais propostas para esta operação e a explicação desta funcionalidade a partir do protótipo, sugeriu-se a inclusão de uma nova atividade na disciplina de Requisito.

A nova atividade estabelece uma reunião informal para aprovação do sub-artefato *Caso de Uso* com todos *stakeholders* do projeto após sua produção no processo. Desta forma, considerou-se que esta atividade deve ser executada no mesmo fluxo onde os casos de uso são produzidos e o papel responsável por sua execução será o mesmo responsável pelo sub-artefato *Caso de Uso*. Ainda, definiu-se que nenhum documento de aprovação necessitaria ser produzido na nova atividade, sendo a aprovação dos casos de uso realizada informalmente durante a reunião. Por fim, duas tarefas foram definidas para a nova atividade, sendo que nenhuma ferramenta foi associada a sua execução. A Tabela 23 mostra em mais detalhes todas as informações da nova atividade, sendo estas baseadas nas assertivas estruturais definidas para inclusão de atividades.

Tabela 23 – Informações da Nova Atividade

Nome	Aprovar Casos de Uso.
Descrição	Uma reunião informal deve ser realizada para aprovação dos casos de uso. A reunião deve contar com a participação de todos os *stakeholders* do projeto e deve acordar sobre a completude dos casos de uso. Caso estes estejam incompletos deve-se retornar a atividade de produção do sub-artefato Casos de Uso.
Obrigatória (S / N)	S
Responsável	Analista de Sistema.
Tarefa(s)	- Convocar todos *stakeholders* para uma reunião; - Conduzir reunião de aprovação dos Casos de Uso com *stakeholders.* - Enviar e-mail comunicando *stakeholders* do projeto sobre resultado da reunião.
Fluxo(s)	- Entender Necessidades dos *Stakeholders*
Ferramenta	–
Artefato(s) Produzido(s)	–
Artefato(s) Consumido(s)	- Casos de Uso (sub-artefato do artefato Modelo de Caso de Uso)
Artefato(s) Modificado(s)	–
Atividade Anterior	Encontrar Casos de Uso e Atores – Parte 2
Próxima Atividade	- Se Casos de Uso Completo (Aprovado): Fim - Se Casos de Uso Incompletos (Reprovado): Encontrar Casos de Uso e Atores – Parte 2

Com base nas informações da Tabela 23 a nova atividade foi incluída, sendo isto realizado através da interface mostrada na Figura 16.

É importante considerar que a inclusão de atividades pode dar-se de duas formas no protótipo: a inclusão de atividades não selecionadas para o processo padrão corrente (já cadastradas) ou atividades novas, definidas pelo engenheiro de processo ou gerente de projeto durante o processo de adaptação.

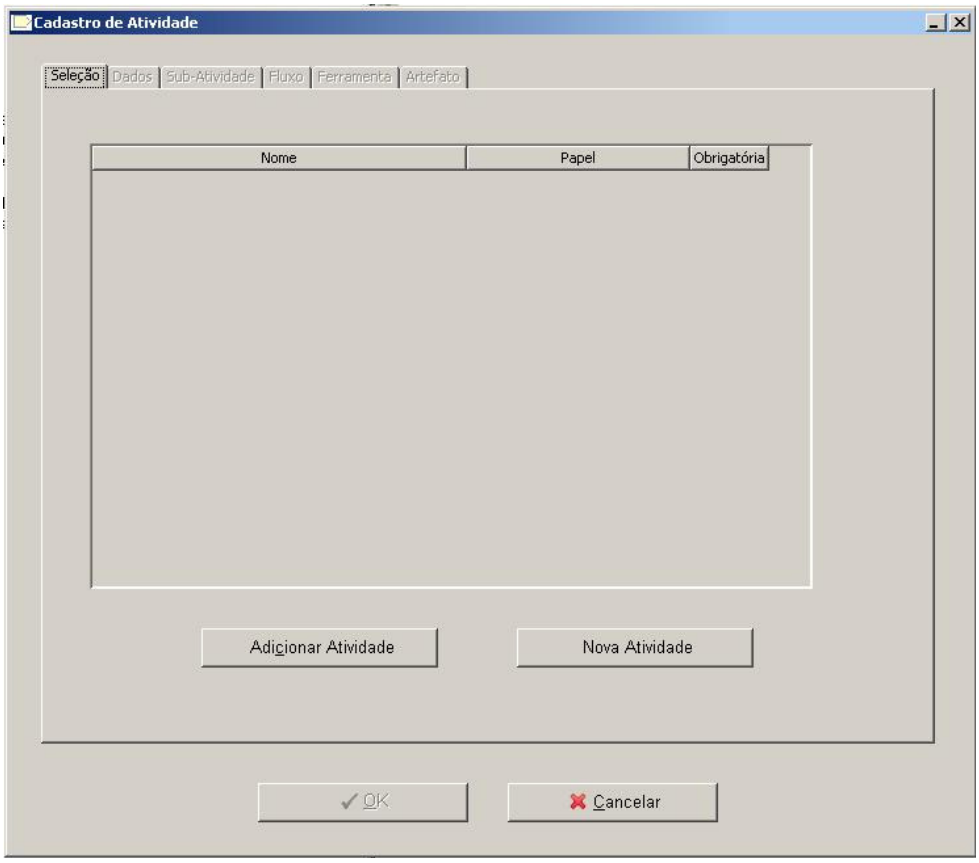

Figura 16 – Interface para Inclusão de Novas Atividades

No cenário utilizado, foi considerada a inclusão de uma atividade nova definida no contexto da adaptação. Neste caso, para sua inclusão, o responsável pela adaptação seleciona o botão *Nova Atividade* iniciando a inserção de suas informações mostradas na Tabela 23. Após entrar com todas informações da atividade e confirmar sua inclusão pelo botão *OK* a mesma tornou-se disponível no processo como mostra a Figura 17.

Por fim, concluída a inclusão da nova atividade e exclusão de todas as atividades desejadas do processo padrão, criou-se neste momento o projeto ao qual o processo adaptado foi associado. Sua criação foi realizada através da opção Salvar Processo localizada no menu Arquivo, sendo necessário informar seu nome e uma breve descrição opcional.

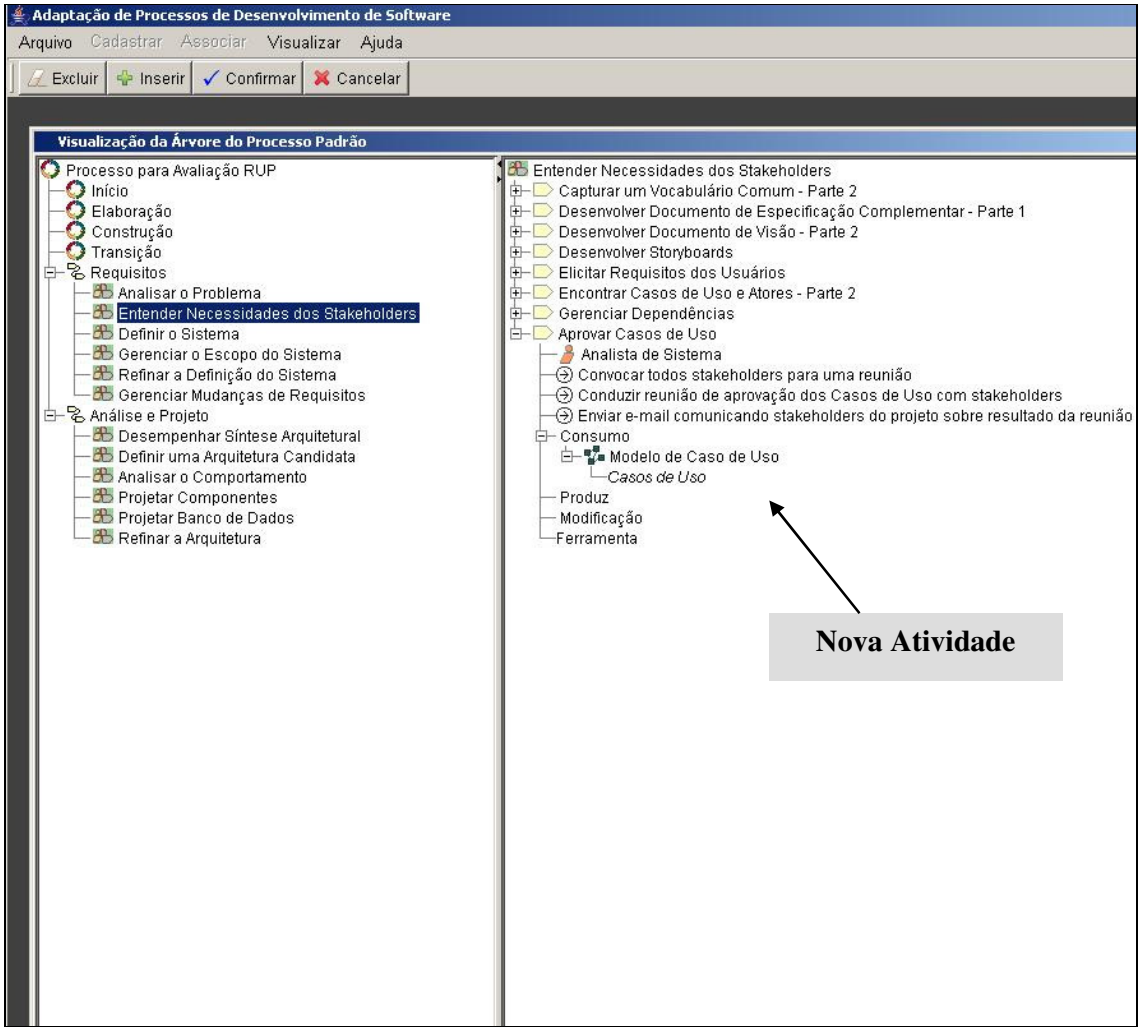

Figura 17 - Nova Atividade no Processo

7.2.3.4. Replicação da Avaliação Proposta na Ferramenta *Rup Builder*

Nesta seção, será apresentada brevemente a replicação da avaliação proposta neste trabalho a partir da ferramenta *Rup Builder,* descrita na seção 3.2.2.1. A replicação foi realizada devido ao fato desta ser a principal ferramenta de configuração para processos de projeto do RUP.

O processo de condução, utilizado na avaliação a partir *do Rup Builder,* não considerou as etapa 1 e 2 propostas na seção 7.1, isto porque esta ferramenta não possui ambiente para manutenção de processos padrão disponibilizando sempre o *framework* RUP para adaptação em três versões: clássico, para projetos de menor porte e projetos de médio porte. Ainda, o conjunto de atividades selecionadas para exclusão do processo padrão já

estavam previamente determinadas. Desta forma, esta avaliação consistiu basicamente na tentativa de exclusão de atividades.

Durante as exclusões das atividades propostas, somente algumas foram possíveis de ser realizadas, isto porque como já explicado na seção 3.2.2.1 o *Rup Builder* não permite a exclusão de atividades e sim de componentes de processo. Na disciplina de Requisito, a exclusão do componente *Requirements Management* (Gerência de Requisitos) permitiu a exclusão das atividades *Desenvolver Plano de Gerência de Requisitos*, *Gerenciar Dependências* e embora não tenha excluído a atividade *Desenvolver Visão*, retirou do processo resultante o artefato *Atributos de Requisito* produzido pela mesma. Ainda, referente às outras atividades desta disciplina, nenhuma pode ser excluída devido à falta de componentes onde estas estivessem agrupadas. Desta forma, através do *Rup Builder* os artefatos *Storyboards* e *Especificação de Requisitos de Software* sempre são produzidos embora sejam considerados opcionais pelo RUP.

Para a disciplina de Análise e Projeto, nenhuma das atividades selecionadas para exclusão puderam ser eliminadas, isto porque para estas atividades não existem componentes relacionados.

Ainda, a análise de impactos durante o processo de adaptação também foi oferecido pela ferramenta *Rup Builder*. Esta análise funciona da mesma forma que o protótipo desenvolvido neste trabalho. Além disso, outras funcionalidades são oferecidas adicionalmente pelo *Rup Builder* como publicação de projetos em *WebSites* e configuração de diferentes visões para geração do processo do projeto.

Entretanto, para o propósito da adaptação de processos, principal interesse deste trabalho, a ferramenta encontrou limitações por não permitir a exclusão de atividades individualmente e também por não fazer referências a inclusão de atividades.

7.2.4. Etapa 4: Avaliação dos Resultados

Os resultados encontrados a partir do cenário exemplo descrito anteriormente evidenciaram aspectos da aplicação prática do meta-modelo e assertivas estruturais propostas.

O uso de parte do processo RUP demonstrou sua compatibilidade com o meta-modelo proposto, embora algumas alterações tenham sido necessárias a partir da descrição deste processo para contemplar sua adaptação. Esta compatibilidade já era um resultado

esperado, isto porque o meta-modelo proposto estende os elementos e relacionamentos do meta-modelo RUP complementado-o apenas para suporte a adaptação de processos.

As alterações realizadas a partir da descrição do processo RUP referiram-se a decomposição de algumas de suas atividades, isto para torná-lo um processo com maior coesão e menor acoplamento. A necessidade da decomposição destas atividades tornou-se evidente, pois estas são executadas em vários fluxos com propósitos diferentes, o que caracteriza atividades distintas. Ainda, percebeu-se que diversas atividades do RUP são responsáveis por produzir artefatos obrigatórios e opcionais do processo ao mesmo tempo, o que dificulta o seu processo de adaptação, isto devido a dificuldade de exclusão destas atividades que em parte são opcionais e em parte são obrigatórias.

Ainda, um ponto positivo evidenciado na avaliação em relação ao meta-modelo proposto, é que os novos elementos incorporados a este em relação ao meta-modelo RUP puderam ser utilizados no cenário exemplo. Estes elementos dizem respeito a tarefas que no RUP são representados pelos passos das atividades e sub-artefatos não referenciados no processo RUP. O elemento sub-artefato tornou possível a configuração das atividades em termos de quais sub-artefatos produzem, consomem e/ou modificam. A importância disto para processos de adaptação é a possibilidade de inclusão e/ou exclusão de apenas partes de artefatos a partir do processo padrão.

Um exemplo que evidenciou a contribuição destes elementos (sub-artefatos) a partir do cenário exemplo foi durante a exclusão da atividade *Análise Arquitetural – Parte 2,* atividade responsável por produzir entre outros o sub-artefato *Visão de Deployment,* pertencente ao artefato *Documento de Arquitetura de Software.* A exclusão desta atividade implicou na exclusão de todos seus sub-artefatos de produção e neste caso em apenas uma seção do arteafato *Documento de Arquitetura de Software* representada no processo pelo sub-artefato *Visão de Deployment.* Em processos onde as atividades são modeladas somente em termos de artefato para consumo, produção e/ou modificação não seria possível identificar exatamente qual parte do artefato *Documento de Arquitetura de Software* não estaria mais sendo produzida, o que poderia gerar inconsistências no processo resultante.

Em relação às assertivas estruturais para inclusão e exclusão de atividades também constatou-se um resultado satisfatório durante a avaliação. Especificamente, as assertivas estruturais para exclusão de atividades, operação considerada de maior impacto em processos de adaptação, tornaram possível a garantia de conformidade do processo resultante em relação

ao processo padrão. Isto porque, após a exclusão de cada atividade verificou-se que todas as dependências no processo referentes a esta também haviam sido eliminadas.

Ainda, as assertivas estruturais para a operação de inclusão de atividades também puderam ser utilizadas e avaliadas para demonstrar sua contribuição neste trabalho. Esta contribuição diz respeito principalmente a definir para novas atividades relações de associação com outros elementos do processo, o que muitas vezes tornar-se-ão relações de dependência a serem verificadas em posteriores operações de exclusões.

Por fim, deve-se considerar que o protótipo utilizado para construção e avaliação do cenário exemplo também foi considerado com uma contribuição deste trabalho, isto porque este facilita o uso do meta-modelo e assertivas estruturais propostas. Considera-se que todas as análises necessárias durante a exclusão de atividades constituiriam uma tarefa exaustiva com maior probabilidade de erros. Ainda, outro aspecto fundamental abordado através de funcionalidades do protótipo é a análise de impacto promovida durante a exclusão de uma atividade, o qual permite ao responsável pela adaptação a identificação de situações indesejadas para o processo resultante.

8. CONSIDERAÇÕES FINAIS

Este capítulo apresenta as considerações finais deste trabalho, descrevendo suas principais contribuições e limitações. Ainda, destaca rumos para futuras pesquisas na área.

A definição e uso de um processo padrão de desenvolvimento de software têm se tornado uma prática bastante comum nas organizações desenvolvedoras de software. Entretanto, frente à diversidade das necessidades específicas encontradas em projetos de desenvolvimento de software faz-se necessária a utilização de processos de adaptação. Neste sentido, desafios são encontrados em garantir a conformidade do processo padrão nos diversos processos definidos na organização.

Partindo deste contexto, este trabalho realizou a definição de um meta-modelo para adaptação de processos de desenvolvimento de software. O meta-modelo proposto estende o meta-modelo RUP para identificação de relações de dependências do processo padrão, permitindo assim que durante sua adaptação todos os elementos afetados sejam devidamente identificados. Se tais relações são devidamente respeitadas garantimos que a conformidade com o processo padrão é mantida, sendo isto de ampla importância, pois conforme Jalote (2002), inconformidades geradas por processos de adaptação provocam falhas na execução de um processo.

Ainda, o desenvolvimento de um conjunto de assertivas estruturais para adaptação de processos de desenvolvimento de software realizado a partir do meta-modelo proposto, tornou possível o auxílio à configuração do ciclo de vida dos projetos de desenvolvimento de software garantindo a consistência entre os elementos e relacionamentos do processo padrão.

8.1. Contribuições

A proposta de um meta-modelo para adaptação de processos de desenvolvimento de software visa contribuir para área de engenharia de software, ajudando a preencher uma lacuna existente especificamente na área de adaptação de processos. A utilização do meta-modelo definido neste trabalho permite as organizações de desenvolvimento de software definir um ou mais processos padrão, com base no processo RUP, de forma a garantir a

identificação de importantes elementos e relacionamentos para processos de adaptação. Além disso, a definição de um conjunto de assertivas estruturais para uso do meta-modelo proposto permite que os processos padrão definidos a partir deste sejam adaptados de forma a garantir consistência nos processos resultantes.

Partindo deste contexto, este trabalho contribui também com a prática ao atender uma demanda organizacional crescente por melhorias nos processos de desenvolvimento de software.

8.2. Limitações do Estudo

Uma das principais limitações desta pesquisa refere-se ao meta-modelo proposto estender especificamente o meta-modelo RUP, restringindo seu uso para processos derivados do RUP.

Outra limitação deste trabalho deve-se ao fato de não terem sido considerados os aspectos referentes à construção de um processo padrão de desenvolvimento de software a partir do meta-modelo proposto. Esta limitação dificultou o processo de avaliação do meta-modelo e das assertivas estruturais, não possibilitando inicialmente sua utilização em um ambiente de desenvolvimento real.

8.3. Trabalhos Futuros

Identifica-se potencial de crescimento nesta linha de pesquisa, centrada no tema de adaptação de processos de desenvolvimento de software para projetos específicos de projetos. Como pesquisas futuras, identifica-se:

- Desenvolvimento de um estudo para avaliação dos métodos para adaptação de processos atualmente existentes, visando a proposição de um método que de suporte a todo processo de adaptação;
- Ampliação do estudo sobre os modelos de processo de desenvolvimento de software (XP, OPEN, MSF, entre outros) para avaliar quais suas relações com processos de adaptação;
- Extensão ao meta-modelo SPEM para suporte ao processo de adaptação. A principal motivação para a escolha deste meta-modelo como objeto de estudo é tornar esta pesquisa genérica a todos os processos de desenvolvimento de software que tiveram ou venham a ter sua origem a partir deste meta-modelo.

A continuidade deste trabalho indica uma contribuição para a área de engenharia de software no sentido de avançar as pesquisas sobre processos de desenvolvimento de software, considerando a afirmativa que muitos autores fazem sobre a qualidade do produto de software estar fortemente relacionada com a qualidade do processo utilizado na sua construção.

REFERÊNCIAS BIBLIOGRÁFICAS

Beck, K. Extreme Programming Explained: Embrace Change. Addison Wesley, 1999, 190p.

Bencomo, A. Extending the RUP, Part 1: Process Modeling. Capturado em: http://www-128.ibm.com/developerworks/rational/library/05/323_extrup1/, Julho 2005.

Borges, L. M. S., Falbo R. A. Gerência de Conhecimento sobre Processos de Software. In: Anais do VIII Workshop de Qualidade de Software, XV Simpósio Brasileiro de Engenharia de Software, Outubro, 2001, pp. 27-38.

Borges, L. M. S., Falbo R. A. Uma Ferramenta de Apoio à Instanciação de Processos de Software com Gerência de Conhecimento. In: Anais do I Simpósio Brasileiro de Qualidade de Software, I Concurso de Teses e Dissertações em Qualidade de Software, Outubro, 2002, pp. 237 –248.

Chrissis, M. B., Korad, M., Shrum, S. CMMI Guidelines for Process Integration and Product Improvement. Addison-Wesley, 2003, 663p.

Coelho, C. MAPS: Um Modelo para Adaptação de Processos de Software. Dissertação de Mestrado. Universidade Federal de Pernambuco, Pernambuco, Brasil, 2003, 162p.

Derniame, J.-C., Kaba B. A., Wastell D. Software Process: Principles, Methodology, and Technology. Lecture Notes in Computer Science, Volume 1500, 1999.

Falbo, R. A. Integração de Conhecimento em um Ambiente de Desenvolvimento. Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro, Brasil, 1998.

Fitzgerald B., Russo N. L., O'Kane T. Software Development Method Tailoring at Motorola. Communications of the ACM, Volume 46, Issue 4, April, 2003, pp. 65-70.

Fuggetta, A. Software Process: A Roadmap. In: Proceedings of the Conference on The Future of Software Engineering, International Conference on Software Engineering, 2000, pp.25-34.

Ginsberg, M. P., Quinn, L. H. Process Tailoring and the Software Capability Maturity Model. Technical Report CMU/SEI-94-TR-024, Carnegie Mellon Software Engineering Institute, November, 1995.

Humphrey, W. S. Managing the Software Process. Addison-Wesley, 1989, 512p.

Humphrey, W. S., Snyder, T. R., Willis, R. R. Software Process Improvement at Hughes Aircraft. IEEE Software, Vol.8, Issue 4, 1991, pp. 11-23.

International Business Machines (IBM). PEP - Process Engineering Process. Capturado em: http://www-128.ibm.com/developerworks/rational/library/6001.html, Agosto 2005a.

International Business Machines (IBM). RUP - Rational Unified Process Evaluation Assembly V2003.06.13 for Windows. Capturado em: http://www14.software.ibm.com/webapp/download/product.jsp?KER&s=z&cat=&S_TACT=104AH +W42&S_CMP, Julho 2005b.

International Organization for Standardization. ISO/IEC 12207 – Software Life-Cycle Processes. Technical Report, August, 1995.

International Organization for Standardization. ISO/IEC TR 15504 – Software Process Assessment. Technical Report, 1998.

Jacobson, I., Booch G., Rumbaugh J. The Unified Software Development Process. Upper Saddle River, Addison Wesley, 2001, 463p.

Jalote, P. CMM in Practice. Processes for Executing Software Projects at Infosys, The SEI Series in Software Engineering, 2002.

Kellner, M. I. Connecting reusable software process elements and components. In: Proceedings of the 10th International Software Process Workshop, International Software Process Workshop, 1996, pp. 8-11.

Kroll, P., Kruchten, P. The Rational Unified Process Made Easy: A Practitioner's Guide to the Rup. Addison-Wesley, 2003, 416p.

Kruchten, P. The Rational Unified Process: An Introduction. Upper Saddle River, Addison-Wesley, 2000, 298p.

Machado, L. F. C. Modelo para Definição de Processos de Software na Estação TABA. Dissertação de Mestrado, COPPE/UFRJ, Rio de Janeiro, Brasil, 2000, 124p.

Object Management Group. Software Process Engineering Metamodel Specification, Version 1.1. Capturado em: http://www.omg.org/technology/documents/formal/spem.htm, Março 2005a.

Object Management Group. Unified Modeling Language Specification. Capturado em: http://www.uml.org/, Julho 2005b.

Open Web Site. Capturado em: www.open.org.au, Fevereiro 2004.

Paulk, M., Curtis, B., Chrissis, M., Weber, C. Capability Maturity Model, Version 1.1, Technical Report CMU/SEI-93-TR-24, Vol. 10. No. 4, 1993, 64p.

Pressman, R. S. Software Enginnering: A practitioner's approach. Makron Books, 2001, 888p.

Rational Software Corporation. Rational Unified Process: Best Practices for Software Development Teams, White Paper. Capturado em: http://www.rational.com/products/rup/whitepapers.jsp, Julho 2005.

Rocha, A. R. C., Maldonado, J. C., Weber, K. C. Qualidade de Software: Teoria e Prática. Prentice Hall, 2001, 303p.

Tyrrell S. The Many Dimensions of the Software Process. Crossroads - The ACM Student Magazine, Volume 6, Issue 4, Junho, 2000, pp.22-26. Capturado em: http://www.acm.org/crossroads/xrds6-4/software.html, Agosto 2005.

Welzel, D., Hausen, H. L., Schmidt W. Tailoring and Conformance Testing of Software Processes: The ProcePT Approach. In: Proceedings of the 2nd IEEE Software Engineering Standards Symposium, 1995.

Werner, C. M. L., Travassos, G. H., Rocha, A. R. C., Werneck, V. M. Memphis: Um Ambiente para Desenvolvimento de Software Baseado em Reutilização. Relatório Técnico, COPPE/UFRJ, 1996.

Xu, P., Ramesh, B. A Tool for the Capture and Use of Process Knowledge in Process Tailoring. In: Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS), 2003**.** pp. 96-102

Xu, P., Ramesh, B. Knowledge Support in Software Process Tailoring. In: Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS), 2005**,** pp. 87-95

Yoon, I., Min, S., Bae, D. Tailoring and Verifying Software Process. In: Eighth Asia-Pacific Software Engineering Conference (APSEC'01), 2001, pp. 202-209.

APÊNDICE I - PROTÓTIPO PARA APOIO AUTOMATIZADO À ADAPTAÇÃO DE PROCESSOS

Este apêndice apresenta a descrição de um protótipo para apoio automatizado à adaptação de processos de desenvolvimento de software desenvolvido no contexto desta pesquisa.

1. Descrição do Protótipo

Visando auxiliar a aplicabilidade do meta-modelo e assertivas estruturais para adaptação de processos propostos neste trabalho um protótipo de software foi desenvolvido. O protótipo foi desenvolvido utilizando a linguagem de programação Java e o bando de dados *Oracle*, atendendo às funcionalidades de controle de acesso, manutenção de processos padrão, manutenção de processos específicos para projetos e assertivas estruturais para inclusão e exclusão de atividades. A escolha das assertivas estruturais a serem implementadas levaram em consideração sua importância e complexidade em processos de adaptação, permitindo definir um escopo apropriado para o protótipo.

As funcionalidades do protótipo estão representadas no diagrama de casos de uso UML da Figura 2. Os atores principais correspondem ao engenheiro de processo e gerente de projeto que farão uso do ambiente de apoio à adaptação de processos. A Tabela 1 descreve brevemente estas funcionalidades.

Tabela 1 – Funcionalidades do protótipo

Funcionalidade	Ator	Descrição
Manter disciplina	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de disciplinas.
Manter fase	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de fases.
Manter fluxo	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de fluxos.
Manter ferramenta	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de ferramentas.
Manter atividade	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de atividades.
Manter tarefa	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de tarefas.
Manter artefato	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de artefatos.
Manter sub-artefato	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de sub-artefatos.

Manter papel	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração de papéis.
Associar fase e fluxo	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de associação entre fases e fluxos.
Associar atividade e fluxo	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de associação entre atividades e fluxos.
Associar atividade e ferramenta	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de associação entre atividades e ferramentas.
Associar sub-artefatos p/ consumo nas atividades	Engenheiro de Processo	O engenheiro de processo dever ser capaz de definir para as atividades quais os seus sub-artefatos de consumo.
Associar sub-artefatos p/ modificação nas atividades	Engenheiro de Processo	O engenheiro de processo dever ser capaz de definir para as atividades quais os seus sub-artefatos de modificação.
Definir sub-artefato opcional	Engenheiro de Processo	O engenheiro de processo dever ser capaz de definir sub-artefatos opcionais no consumo das atividades.
Visualizar processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de visualizar o processo padrão durante sua inclusão.
Definir ordem p/ tarefas	Engenheiro de Processo	O engenheiro de processo deve ser capaz de definir a ordem de execução das tarefas.
Manter processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar operações de inclusão, exclusão e alteração nos processos padrão.
Associar fase no processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar de associar fases em processo padrão.
Associar fluxo no processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar de associar fluxos em processo padrão.
Associar disciplina no processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar de associar disciplinas em processo padrão.
Associar atividade no processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de realizar de associar atividades em processo padrão.
Definir ordem p/ fluxos no processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de definir a ordem de execução dos fluxos em um processo padrão.
Definir ordem p/ atividades no processo padrão	Engenheiro de Processo	O engenheiro de processo deve ser capaz de definir a ordem de execução das atividades em um processo padrão.
Manter processo específico de projeto	Engenheiro de Processo Gerente de Projeto	O engenheiro de processo / gerente de projeto deve ser capaz de realizar operações de inclusão, exclusão e alteração nos processos específicos de projetos.
Escolher processo padrão	Engenheiro de Processo Gerente de Projeto	O engenheiro de processo / gerente de projeto deve ser capaz de escolher o processo padrão para gerar processos específicos de projetos.
Incluir atividade no processo específico de projeto	Engenheiro de Processo Gerente de	O engenheiro de processo / gerente de projeto deve ser capaz de incluir atividades nos processos específicos de projetos.

	Projeto	
Excluir atividade no processo específico de projeto	Engenheiro de Processo Gerente de Projeto	O engenheiro de processo / gerente de projeto deve ser capaz de excluir atividades nos processos específicos de projetos.

2. Interfaces Principais

Esta seção apresenta as principais interfaces gráficas do protótipo explicando seu funcionamento.

2.1. Controle de Acesso

A Figura 1 apresenta a interface de *login* do usuário, a qual permite o acesso dos usuários ao protótipo. A interface apresenta campos para escolha do usuário, entrada de senha e botões para envio das informações de validação do acesso ou cancelamento da operação.

Figura 1 - *Login* do usuário

As funcionalidades são disponibilizadas conforme as permissões de acesso de cada usuário. Estes usuários e suas permissões são os mesmos definidos no RUP, acordando com (IBM, 2005a). A Tabela 2 relaciona quais funcionalidades estão disponíveis para cada tipo de papel.

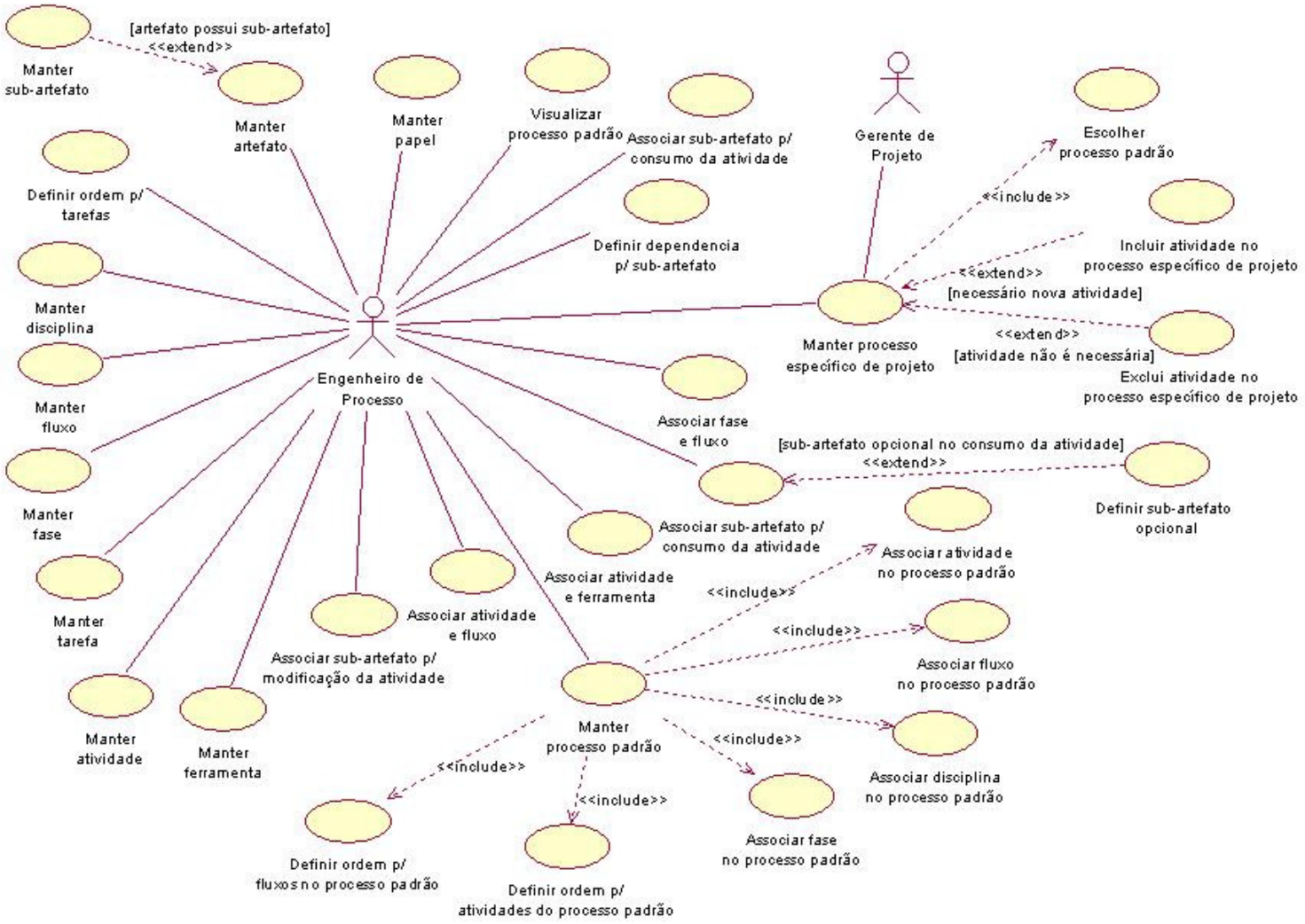

Figura 2 - Diagrama de caso de uso

Tabela 2 - Controle de Acesso

Usuário / Funcionalidade	Engenheiro de Processo	Gerente de Projeto
Manutenção de Elementos	**X**	
Manutenção de Processos Padrão	**X**	
Manutenção de Processos Específicos para Projetos	**X**	**X**
Exclusão e Inclusão de Atividades	**X**	**X**

A Figura 3 apresenta os menus da interface principal para usuários com papel de engenheiro de processo, o qual possui acesso a todas as funcionalidades disponibilizadas através do protótipo.

Figura 3 - Menus da interface principal para usuário engenheiro de processo

A Figura 4 apresenta as operações oferecidas através do menu Arquivo, o qual disponibiliza acesso a maioria das funcionalidades oferecidas pelo protótipo. Tais operações incluem: criação de um novo processo padrão, manutenção de processos padrão existentes, criação de um novo processo específico de projeto (*Novo Projeto*), manutenção de processos específicos projetos existentes (*Manutenção de Projetos*), salvamento de processos e manutenção dos elementos.

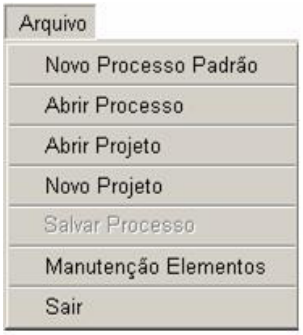

Figura 4 - Menu Arquivo

A Figura 5 apresenta as operações oferecidas através do menu Cadastrar, o qual só é disponibilizado a partir da opção Manutenção de Elementos do menu Arquivo. As operações disponibilizadas a partir do menu Cadastrar permitem a inserção, exclusão ou alteração de todos elementos como fases, disciplinas, atividades, etc. Estes elementos

serão utilizados no protótipo para a criação de um processo padrão novo ou ainda, para manutenção dos existentes.

Figura 5 - Menu Cadastrar

A Figura 6 apresenta as operações oferecidas através do menu Associar, o qual possui tanto operações para associações entre elementos inseridos no menu Cadastrar como associações destes elementos em processos padrão. As associações entre elementos dizem respeito a relações definidas no meta-modelo e são representadas pelas opções do menu "Fase .:. Fluxo", "Fluxo .:. Atividade", "Atividade .:. Artefato", "Dependência de Artefatos" e "Ordem Tarefa".

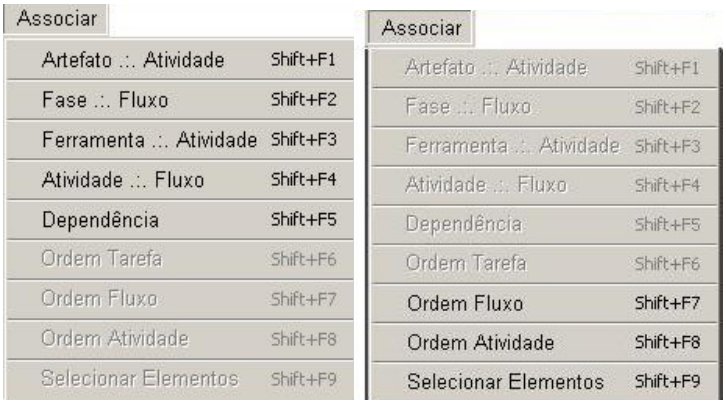

Figura 6 – Menu Associar

Já as opções "Selecionar Elementos", "Ordem Fluxo" e "Ordem Atividade" dizem respeito às operações que permitem associação de elementos em novos ou já existentes processos padrão.

O menu Visualizar, mostrado na Figura 7 traz apenas uma opção que permite ao engenheiro de processo visualizar o processo padrão enquanto este está sendo criado. O processo é apresentado em uma estrutura de árvore contendo todos os elementos já escolhidos pelo engenheiro de processo, sendo estes organizados de forma hierárquica.

Figura 7 – Menu Visualizar

2.2. Manutenção de Elementos

A manutenção de elementos deve ser a primeira funcionalidade a ser utilizada no protótipo. Isto porque, é através dela que a inclusão, exclusão e alteração dos elementos a serem utilizados na criação dos processos padrão são permitidas. Desta forma, enquanto estes elementos não são inseridos no protótipo nenhuma outra funcionalidade pode ser acessada. Estes elementos são: fase, disciplina, fluxo, atividade, tarefa, ferramenta, papel e artefato (sub-artefatos são incluídos, excluídos ou alterados através dos artefatos). Ainda, é através da manutenção de elementos que várias associações definidas no meta-modelo devem ser realizadas. Estas associações são: associações entre fases e fluxos, fluxos e atividades, atividades e artefatos, associação de dependência entre artefatos e ordem de execução de tarefas. As outras associações definidas no meta-modelo (ordem de execução de fluxos e atividades) são definidas na funcionalidade *Novo Processo Padrão* e serão explicadas na próxima seção deste documento.

A seguir, como forma de demonstrar o funcionamento das operações oferecidas pela funcionalidade manutenção de elementos será detalhado as operações de inclusão, exclusão e alteração de um artefato e seus sub-artefatos e também a operação de associação entre atividades e artefatos, sendo que todas as outras operações têm o mesmo funcionamento.

2.2.1. Manter Artefato

A figura 8 ilustra a interface inicial para as operações de inclusão, exclusão ou alteração de artefatos. Neste momento, uma lista dos artefatos já incluídos no protótipo é apresentada e o usuário pode escolher através dos botões qual operação deseja realizar. Para as operações de alteração ou exclusão deve-se primeiramente selecionar o artefato a partir da lista e indicar a opção desejada através dos botões. A operação de alteração direciona o usuário para a tela de edição que pode ser vista na Figura 9.

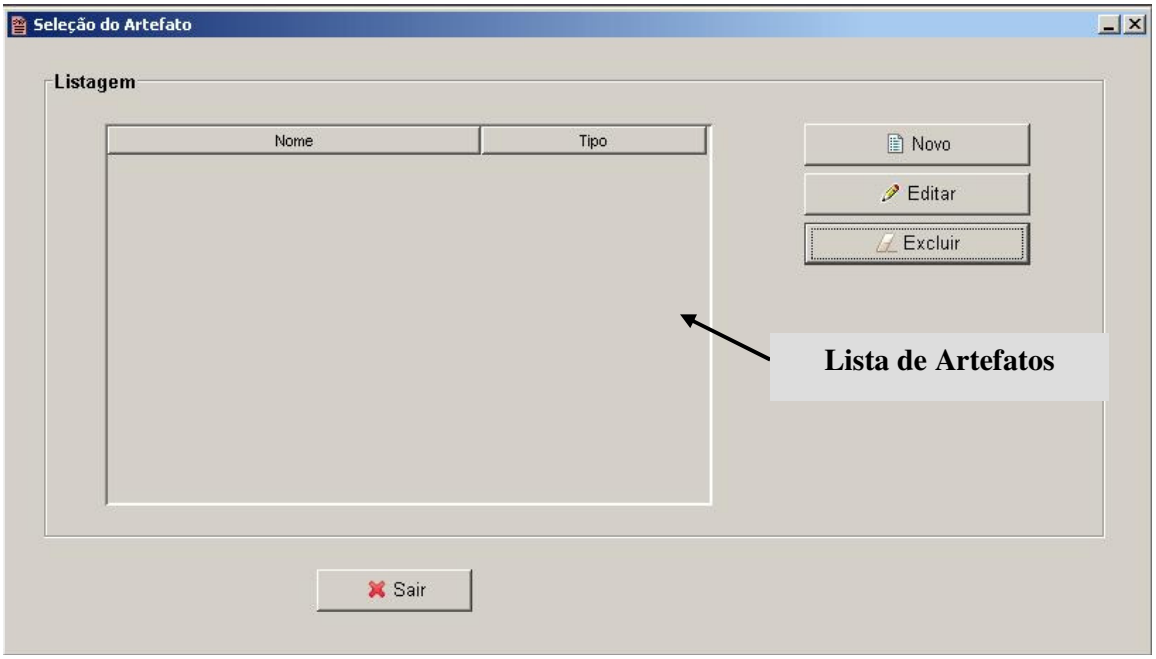

Figura 8 - Interface para inserção, exclusão ou alteração de artefatos

A Figura 9 também representa a interface que pode ser usada para inclusão de novos artefatos. Nela são apresentados os campos para a inserção do novo artefato como nome, descrição e tipo. Para prosseguir, é necessário informar se o artefato possui ou não sub-artefatos. Caso a opção *Não Incluir Sub-artefatos* seja selecionada o usuário deverá (Figura 10) informar a atividade de produção e o responsável pelo artefato.

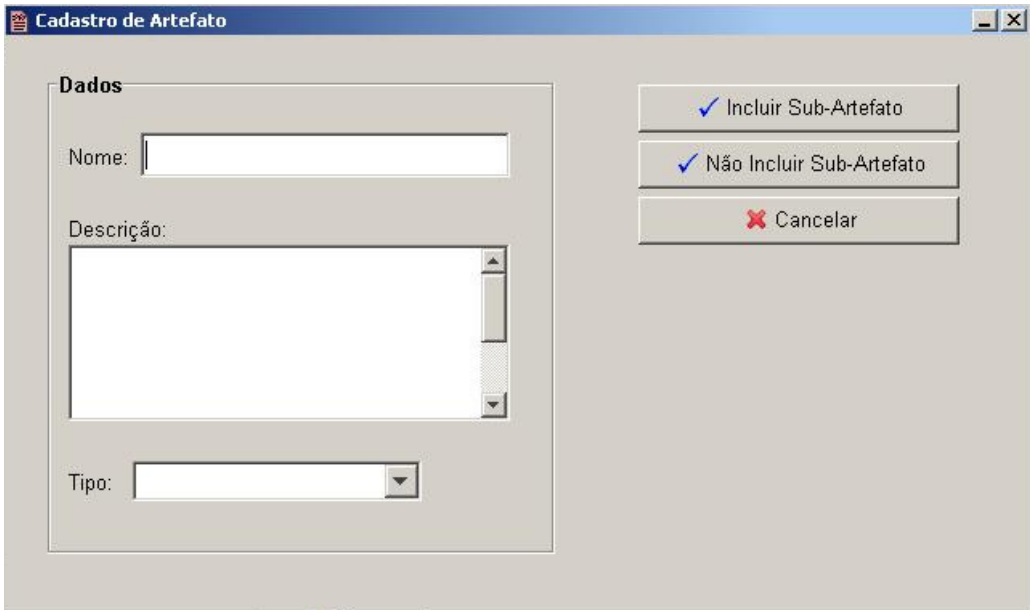

Figura 9 - Interface para inclusão ou edição de um artefato

Caso a opção Incluir Sub-Artefatos seja selecionada o usuário terá acesso às operações de inclusão, alteração e exclusão de sub-artefatos podendo incluir todos sub-artefatos pertencentes ao artefato em questão.

É importante considerar que embora o meta-modelo defina que todos os artefatos são compostos por sub-artefatos, como pode ser visto, o protótipo permite se necessário a inclusão de apenas artefatos. Isto acontece porque, no protótipo os artefatos que não possuem sub-artefatos são considerados o próprio sub-artefato, ou seja, quando um artefato é incluído e este não possui sub-artefatos é criado na base de dados do protótipo um sub-artefato com o mesmo nome do artefato e este é associado como seu sub-artefato. Apesar destas ações serem realizadas sem o conhecimento do usuário, isto é feito para facilitar o seu trabalho, pois muitas vezes devido a simplicidade de um determinado artefato este não precisa ser decomposto em sub-artefatos. A decisão final fica por conta do engenheiro de processo responsável pela definição de todos artefatos e sub-artefatos.

Figura 10 - Tela para inclusão de artefatos que não possuem sub-artefatos

2.2.2. Associar Atividades e Artefatos

A operação de associação entre atividades e artefatos permite ao usuário indicar quais artefatos e/ou sub-artefatos são consumidos e/ou modificados nas atividades. A Figura 11 ilustra a interface que permite estas associações.

Para realizar as associações o usuário deve primeiramente utilizar os filtros de disciplina, fluxo e atividade para encontrar a atividade desejada. Após isto, o mesmo deve escolher os artefatos e/ou sub-artefatos na lista *Sub-artefatos Disponíveis* associando-os na lista *Sub-Artefatos Consumidos* e/ou na lista *Sub-Artefatos Modificados*. Ainda, como os artefatos e/ou sub-artefatos podem ser considerados opcionais no consumo de uma atividade, é possível na lista *Sub-Artefatos Consumidos* marcar quais artefatos e/ou sub-artefatos devem ser opcionais através do botão direito do mouse.

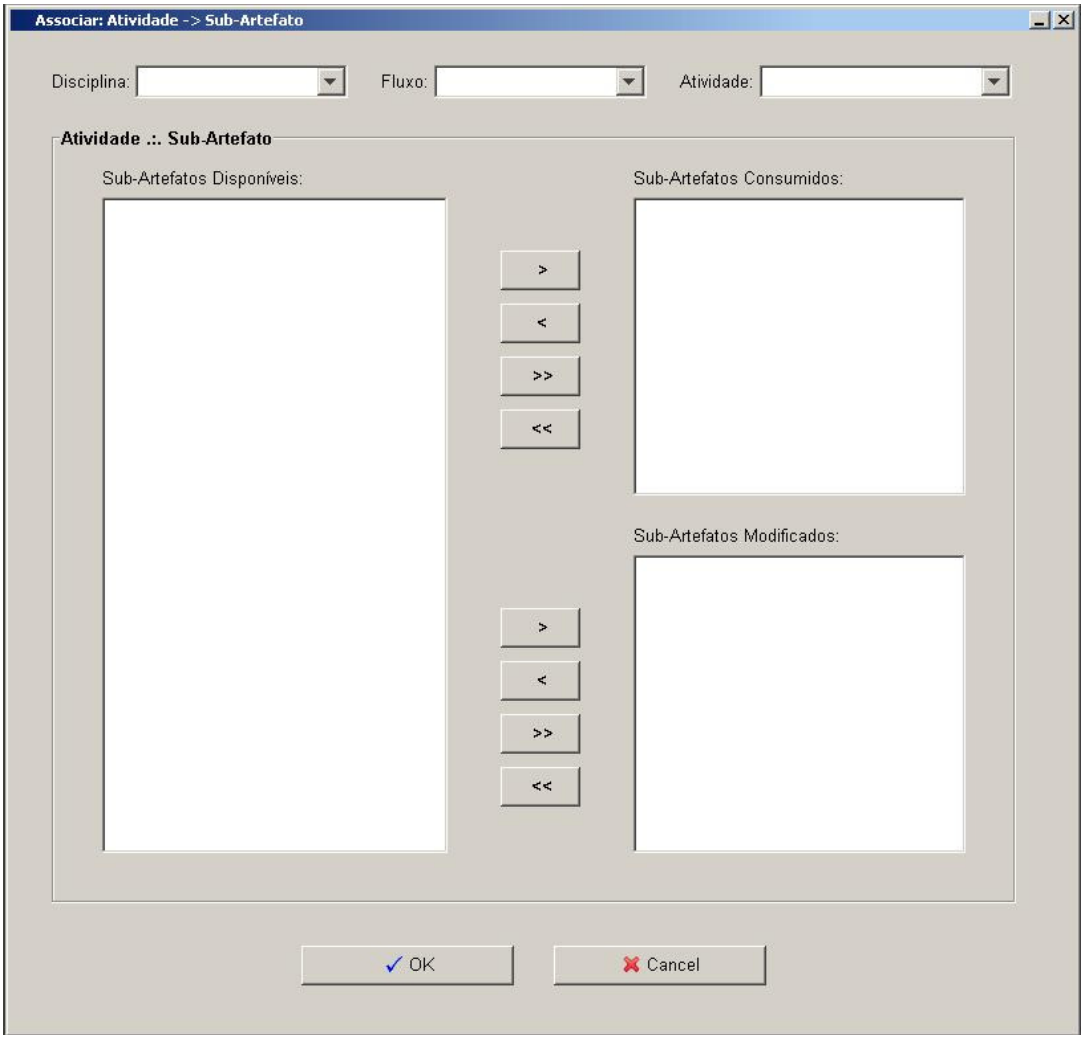

Figura 11 - Tela de associação entre atividades e artefatos

2.3. Novo Processo Padrão

A inserção de um ou mais processos padrão pode ser feita através desta funcionalidade. Tais processos ficarão disponíveis para adaptação de forma a gerar processos específicos de projetos através da funcionalidade *Novo Projeto*.

A utilização de vários processos padrão no protótipo é baseada nas referências de (Chrissis *et al*, 2003) e (Kroll & Kruchten, 2003) e permite ao usuário responsável por gerar o processo específico de projeto escolher qual dos processos padrão deseja adaptar. A Figura 12 apresenta os campos para iniciar a inserção de um novo processo padrão. Devem ser informados neste momento nome, descrição e tipo do ciclo de vida para o processo, sendo que o ciclo de vida representa o próprio processo padrão. Para prosseguir a inserção o usuário deverá ainda associar fases, disciplinas, fluxos e atividades através da interface mostrada na Figura 13.

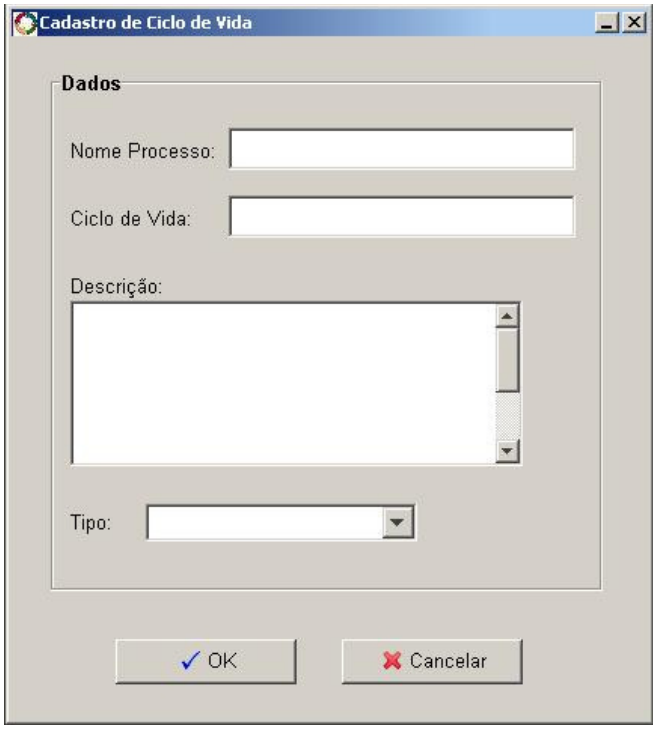

Figura 12 - Tela inicial de cadastro do processo padrão

Para as associações do processo padrão, algumas restrições foram implementadas na interface de associação (Figura 13). Estas restrições dizem respeito a disponibilizar para o usuário só elementos válidos para associação ao processo. Por exemplo, uma das restrições é que apenas fases que possuem ao menos um fluxo

associado devem estar disponíveis para associação. As outras restrições dizem respeito a disciplinas só serem disponibilizadas em casos onde possuam pelo menos um fluxo, sendo este associado a pelo menos uma fase e fluxos só serem disponibilizados se possuírem pelo menos uma atividade associada. Entretanto, é importante considerar que nenhuma restrição referente a seleção de atividades foi implementada. Isto faz com que a consistência durante a seleção destas para o processo padrão fique sobre a responsabilidade do engenheiro de processo. Esta consistência diz respeito a verificar possíveis relações de dependência que uma atividade possa ter quando esta sendo selecionada.

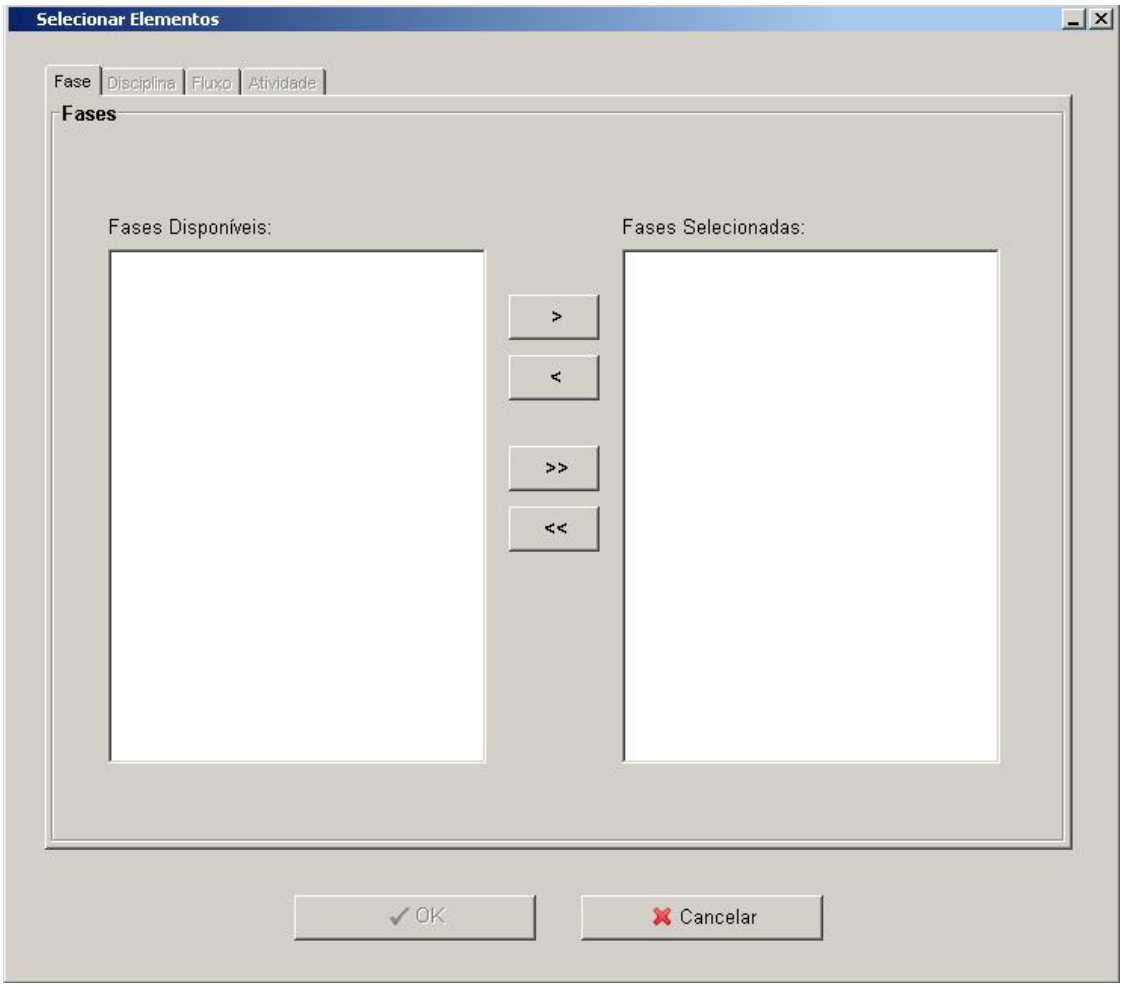

Figura 13 - Tela para associação de elementos no novo processo padrão

Por fim, para finalizar a inserção do novo processo padrão deve-se definir a ordem de execução para os fluxos e atividades selecionadas, sendo isto realizado através de interfaces do protótipo que oferecem várias opções do diagrama de atividades da

UML (Figura 14), ou seja, permitem a utilização de estruturas de união, decisão, paralelismo e seqüência.

2.4. Manutenção de Processos Padrão

Esta funcionalidade permite que alterações sejam realizadas em um processo padrão existente no protótipo. Para isto, o usuário deve escolher o processo a ser alterado a partir de uma lista disponibilizada através da interface mostrada na Figura 15.

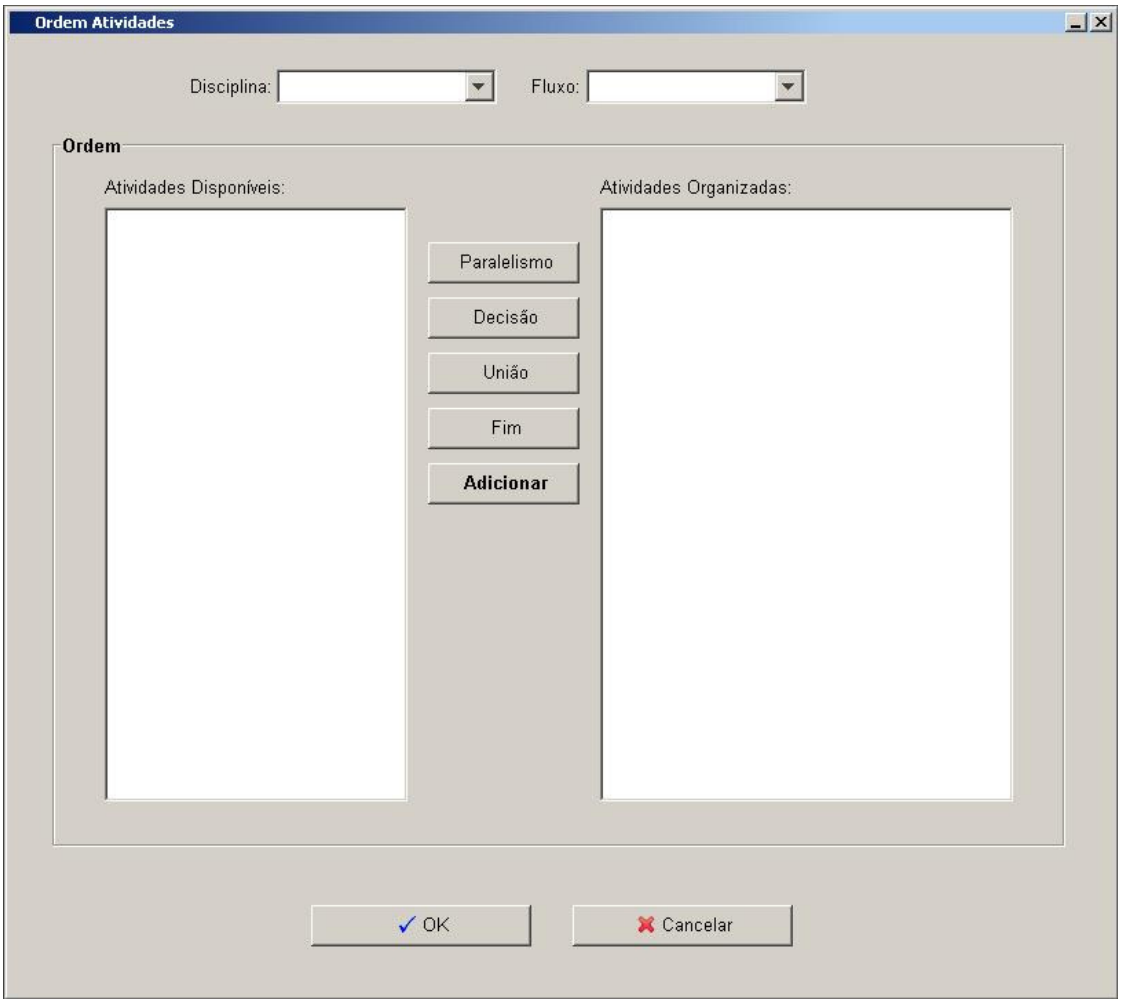

Figura 14 - Exemplo de tela para definição de ordem de execução dos elementos

Após a escolha do processo, é disponibilizado para o usuário acesso às interfaces mostradas na Figura 13 e Figura 14 cujas funcionalidades já foram explicadas anteriormente. A idéia é que o usuário possa alterar associações de fases, disciplinas, fluxos, atividades ou ainda, ordem de execução para fluxos e atividades. Por fim, quando do término de todas as alterações o usuário deve novamente salvar o processo através do menu Arquivo.

2.5. Novo Projeto

As operações referentes à criação de projetos podem ser acessadas tanto por engenheiros de processo como gerentes de projeto, sendo esta a principal funcionalidade disponibilizada pelo protótipo. A Figura 15 ilustra a interface inicial para criação de um novo projeto, sendo esta a tela de abertura do protótipo para os usuários gerente de projeto. Nela, o usuário deve primeiramente escolher o processo padrão a ser adaptado. Após a escolha, o processo padrão selecionado é apresentado em uma estrutura de árvore facilitando sua visualização. Esta estrutura contém todos os elementos do processo organizados de forma hierárquica e permite ao usuário visualizar todas as informações do processo. A Figura 16 mostra um exemplo de árvore para um processo padrão.

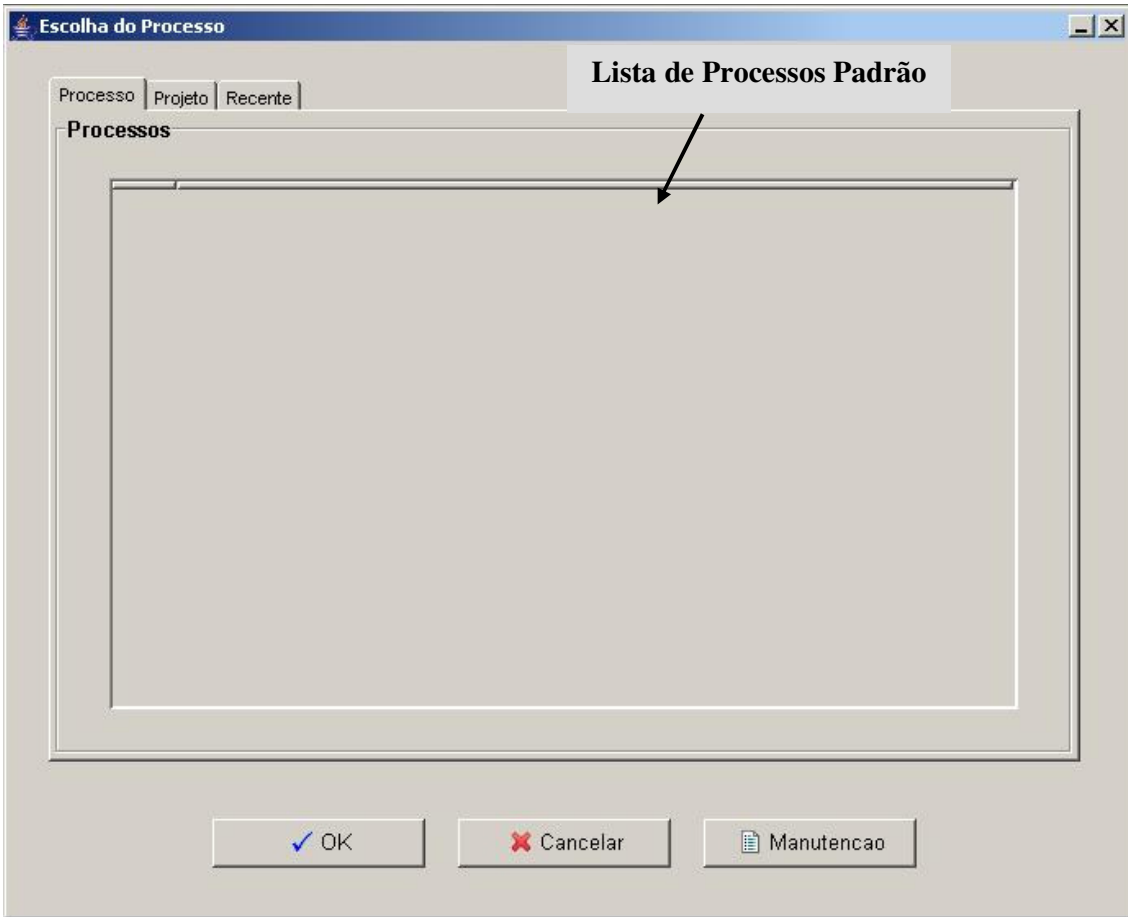

Figura 15 - Tela que lista todos os processos padrão

No lado esquerdo da janela são apresentados ciclo de vida, fases, disciplinas e fluxos do processo, sendo os fluxos apresentados dentro das disciplinas a que pertencem. Ainda, caso o usuário deseje ver o detalhamento de um determinado elemento (disciplina, fluxo ou atividade) o mesmo deve selecionar o elemento desejado e verificar tais informações no lado direito da janela selecionando para isto a aba *detalhamento* localizada no canto inferior da tela.

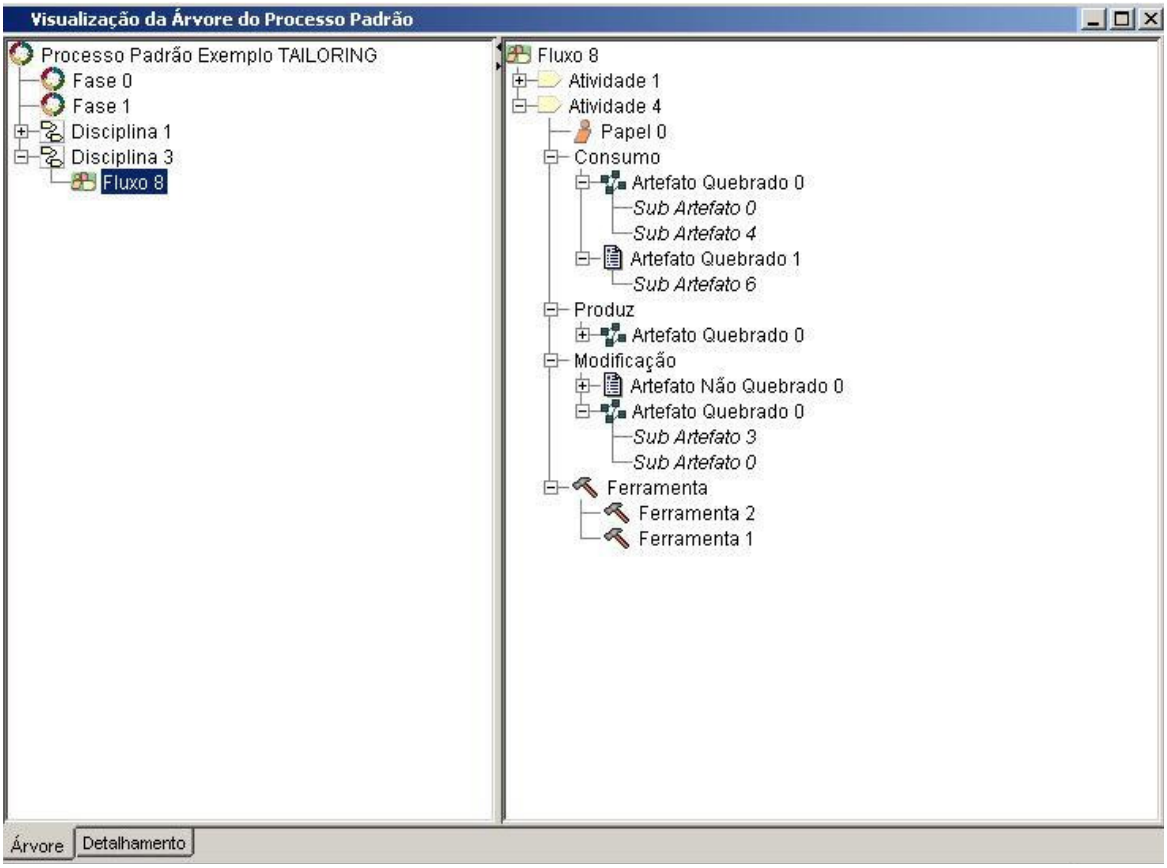

Figura 16 - Tela para visualização de um processo padrão e pequeno exemplo de um processo padrão de desenvolvimento de software

Ainda, deve-se considerar que é através da árvore de um processo padrão que as operações de inclusão e exclusão de atividades podem ser acessadas. O usuário responsável pela criação do projeto escolhe quais atividades ele deseja excluir do processo e ainda, verifica a necessidade de incluir novas atividades. Estas operações estão explicadas em detalhe no capítulo 7 deste documento, através de um exemplo de uso do protótipo.

Por fim, quando o usuário finaliza a inclusão e exclusão de todas as atividades desejadas, o mesmo deve salvar o processo criando neste momento o projeto que ficará associado a tal processo. A Figura 17 apresenta os campos para criação de um projeto, sendo estes referentes ao nome e descrição do projeto.

Figura 17 - Tela para inclusão de um novo projeto

2.6. Manutenção de Projetos

Esta funcionalidade permite que alterações sejam realizadas em qualquer um dos processos específicos de projetos existentes no protótipo. Para isto, o usuário deve escolher o projeto a ser alterado a partir de uma lista disponibilizada através da interface mostrada na Figura 18.

Após a escolha do projeto, é disponibilizado para o usuário acesso à interface já mostrada na Figura 16. A idéia é que o usuário possa continuar a inclusão e exclusão de atividades até gerar o processo desejado para o projeto em questão. Por fim, quando do término de todas as alterações o usuário deve novamente salvar o projeto através do menu Arquivo. Neste momento, alterações nos campos nome e descrição do projeto também podem ser realizadas.

2.7. Salvar Processo

A operação Salvar Processo está disponível através do menu Arquivo e permite o salvamento dos processos, projetos e elementos de processo no protótipo. As funcionalidades “manutenção de elementos”, “novo processo padrão”, “manutenção de processos padrão”, “novo projeto” e “manutenção de projetos” devem utilizar esta

operação sempre que finalizadas para que as informações geradas por elas sejam persistidas no banco de dados e fiquem disponíveis para posterior acesso.

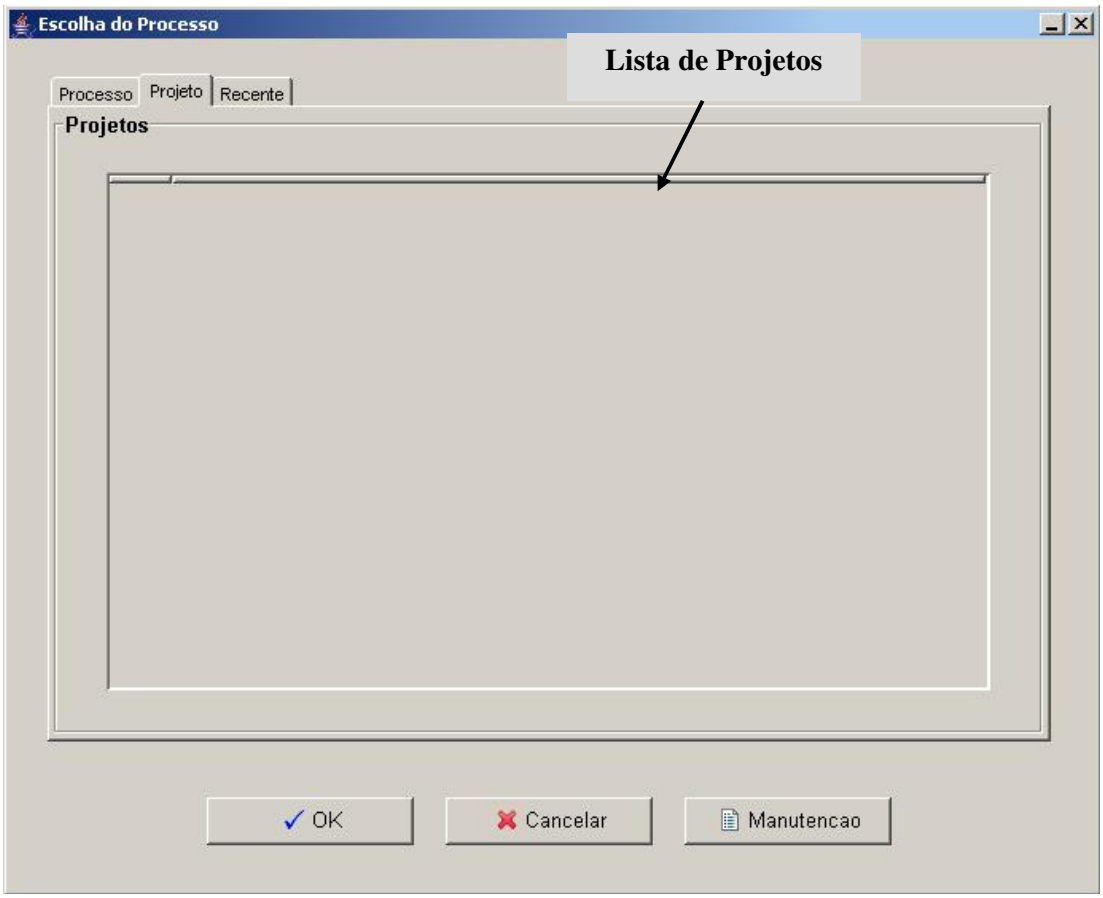

Figura 18 - Tela para abertura de projetos

2.8. Visualizar Processo Padrão

Esta funcionalidade permite apenas que o usuário engenheiro de processo visualize o processo padrão enquanto este está sendo criado. A Figura 16 (seção 2.5) mostrou um exemplo de visualização para um processo padrão.

APÊNDICE II – ANÁLISE DE IMPACTO DE EXCLUSÕES DAS ATIVIDADES

Este apêndice apresenta a análise de impacto realizado pelo protótipo durante as exclusões de atividades da disciplina de Requisito e Análise e Projeto do RUP.

1. Exclusão da Atividade Produzir *StoryBoard* (Requisito)

O fluxo impactado pela exclusão da atividade *Produzir StoryBoard* é somente o fluxo *Entender Necessidades dos Stakeholders.*

1.1. Impactos do Fluxo Entender Necessidades dos *Stakeholders*

Neste fluxo, a atividade excluída *Produzir Storyboard* é desempenhada. Os elementos afetados neste fluxo referem-se somente a esta atividade, sendo estes mostrados na Tabela 1.

Tabela 1 - Elementos Impactados pela Exclusão da Atividade Produzir *Storyboard* no Fluxo Entender Necessidades dos *Stakeholders*

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefatos de: - *Storyboard*	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-artefatos de: - *Change Request* - Caso de Negócio - Modelo de Negócio - Documento de Visão - Plano de Iteração	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
Fluxo	Entender Necessidades dos *Stakeholders*	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	*Rational Requisite Pro* *Rational ClearQuest* *Rational XDE Developer* *Rational Rose* *Rational SoDA*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Analista de Sistema	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Analista de Sistema	Relação de associação com sub-	Sub-artefatos excluídos devem ter suas relações

		artefatos do *Storyboard* excluída	de associação com papéis eliminadas.
Tarefa	Tarefas de Produzir *StoryBoard*	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

Ainda, referente as informações da Tabela 1, deve-se considerar que no fluxo *Entender Necessidades dos Stakeholders* a atividade excluída é paralela a atividade *Elicitar Requisitos dos Usuários*. Desta forma, as atividades relacionadas (anteriores e próximas) a atividade *Produzir StoryBoard* não precisam ser conectadas, necessitando apenas eliminar esta estrutura de paralelismo do fluxo mantendo a conexão de suas atividades relacionadas com a atividade *Elicitar Requisitos dos Usuários*.

2. Exclusão da Atividade Detalhar os Requisitos de Software (Requisito)

A atividade *Detalhar os Requisitos de Software* é desempenhada no fluxo *Refinar a Definição do Sistema*. Sua exclusão da disciplina Requisito causa impacto somente neste fluxo, sendo estes impactos referentes aos próprios elementos da atividade excluída.

2.1. Impactos do Fluxo Refinar a Definição do Sistema

Os elementos impactados pela exclusão da atividade Detalhar os Requisitos de Software são apresentados na Tabela 2.

Tabela 2 - Elementos Impactados pela Exclusão da Atividade Detalhar os Requisitos de Software no Fluxo Refinar a Definição do Sistema

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefatos de: - Especificação de Requisitos de Software	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-artefatos de: - Especificação Complementar - Plano de Gerência de Requisitos - Glossário - Documento de Visão - Plano de Iteração	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
	Sub-artefatos de: - Especificação Complementar - Modelo de Caso de Uso	Relação de saída (modificação) com atividade excluída	Sub-artefatos modificados pela atividade devem ter relação de associação eliminada
Fluxo	Refinar a Definição do Sistema	Relação com	Atividades excluídas

		atividade excluída	devem ter suas relações de associação com fluxos eliminadas.
Papel	Especificador de Requisitos	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Especificador de Requisitos	Relação de associação com sub-artefatos de Especificação de Requisitos de Software excluída	Sub-artefatos excluídos devem ter suas relações de associação com papéis eliminadas.
Tarefa	Tarefas de Detalhar os Requisitos de Software	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

O fluxo *Refinar a Definição do Sistema* possui a execução de suas atividades em paralelo. Desta forma, as atividades relacionadas (anteriores e próximas) para a atividade *Detalhar os Requisitos de Software* não precisaram ser conectadas, necessitando apenas removê-la da estrutura de paralelismo no fluxo.

3. Exclusão da Atividade Identificar Elementos de Projeto – Parte 2 (Análise e Projeto)

A exclusão da atividade *Identificar Elementos de Projeto – Parte 2* impacta os fluxos *Analisar o Comportamento* e *Projetar Componentes* da disciplina Análise e Projeto. A seguir, estes impactos são detalhados através de tais fluxos.

3.1. Impactos do Fluxo Analisar o Comportamento

A atividade *Identificar Elementos de Projeto – Parte 2* é executada no fluxo *Analisar o Comportamento*. Os elementos impactados neste fluxo pela sua exclusão são mostrados na Tabela 3.

Tabela 3 - Elementos Impactados pela Exclusão da Atividade Identificar Elementos de Projeto – Parte 2 no Fluxo Analisar o Comportamento

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefatos Cápsula / Protócolo / *Signal* de: - Modelo de Projeto	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-artefatos de:	Relação de entrada (consumo) com	Sub-artefatos consumidos pela

	- Modelo de Caso de Uso - Especificação Complementar - Modelo de Análise - Modelo de Projeto - Guias Específicas de Projeto	atividade excluída	atividade devem ter relação de associação eliminada
	Sub-artefatos de: - Modelo de Projeto	Relação de saída (modificação) com atividade excluída	Sub-artefatos modificados pela atividade devem ter relação de associação eliminada
Fluxo	Analisar o Comportamento	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Papel	Arquiteto de Software	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Arquiteto de Software Projetista de Cápsulas	Relação de associação com sub-artefatos do Modelo de Projeto excluída	Sub-artefatos excluídos devem ter suas relações de associação com papéis eliminadas.
Tarefa	Tarefas de Identificar Elementos de Projeto – Parte 2	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

Referente a Tabela 3, deve-se considerar que no fluxo *Analisar o Comportamento* a atividade excluída é executada em paralelo a atividade *Identificar Elementos de Projeto – Parte 1*. Desta forma, as atividades relacionadas (anteriores e próximas) a atividade *Identificar Elementos de Projeto – Parte 2* não precisam ser conectadas, necessitando apenas eliminar esta estrutura de paralelismo do fluxo mantendo a conexão de suas atividades relacionadas com a atividade *Identificar Elementos de Projeto – Parte 1*.

3.2. Impactos do Fluxo Projetar Componentes

A exclusão da atividade *Identificar Elementos de Projeto – Parte 2* do fluxo *Analisar o Comportamen*to causou a exclusão da atividade *Projetar Cápsulas* no fluxo *Projetar Compontentes*. A exclusão deve-se ao fato desta atividade consumir obrigatoriamente o sub-artefato *Cápsula* do artefato *Modelo de Projeto* excluído no fluxo *Analisar o Comportamento*. A seguir, a Tabela 4 indica os impactos da exclusão da atividade *Projetar Cápsulas* no fluxo *Projetar Componentes.*

Tabela 4 - Elementos Impactados pela Exclusão da Atividade Projetar Cápsulas no Fluxo Projetar Componentes

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefatos de: - Modelo de Projeto	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
	Sub-artefatos de: - Modelo de Projeto	Relação de saída (modificação) com atividade excluída	Sub-artefatos modificados pela atividade devem ter relação de associação eliminada
Fluxo	Projetar Componentes	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	*Rational Rose*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Projetista de Cápsula	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Arquiteto de Software Projetista de Cápsula	Relação de associação com sub-artefatos do Modelo de Projeto excluída	Sub-artefatos excluídos devem ter suas relações de associação com papéis eliminadas.
Tarefa	Tarefas de Identificar Elementos de Projeto – Parte 2	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

No fluxo *Projetar Componentes,* a atividade *Projetar Cápsulas* é executada em paralelo a outras atividades do fluxo. Desta forma, não é necessário conectar atividades relacionadas (anteriores e próximas) durante a exclusão desta atividade, sendo necessário apenas retirá-la da estrutura de paralelismo.

4. Exclusão da Atividade Análise dos Casos de Uso – Parte 2 (Análise e Projeto)

A exclusão da atividade *Análise dos Casos de Uso – Parte 2* causa impactos ao fluxo *Definir uma Arquitetura Candidata*. A seguir, estes impactos são detalhados neste fluxo.

4.1. Impactos do Fluxo Definir uma Arquitetura Candidata

Os impactos causados pela exclusão da atividade *Análise dos Casos de Uso – Parte 2* podem ser vistos na Tabela 5.

Tabela 5 - Elementos Impactados pela Exclusão da Atividade Análise dos Casos de Uso – Parte 2 no Fluxo Definir uma Arquitetura Candidata

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefatos Evento de: - Modelo de Projeto	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-artefatos de: - Modelo de Caso de Uso - Especificação Complementar - Glossário - Documento de Arquitetura do Sofware - Guias Específicas de Projeto	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
Fluxo	Definir uma Arquitetura Candidata	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	*Rational XDE Developer* *Rational Rose*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Projetista	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Arquiteto de Software Projetista	Relação de associação com sub-artefatos do Modelo de Projeto excluída	Sub-artefatos excluídos devem ter suas relações de associação com papéis eliminadas.
Tarefa	Tarefas de Análise dos Casos de Uso – Parte 2	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

Referente a Tabela 5, deve-se considerar que no fluxo *Definir uma Arquitetura Candidata* a atividade excluída é executada em paralelo a atividade *Análise dos Casos de Uso – Parte 1*. Desta forma, as atividades relacionadas (anteriores e próximas) a atividade *Análise dos Casos de Uso – Parte 2* não precisam ser conectadas, necessitando apenas eliminar esta estrutura de paralelismo do fluxo mantendo a conexão de suas atividades relacionadas com a atividade *Identificar Análise dos Casos de Uso – Parte 1.*

.

5. Exclusão da Atividade Análise Arquitetural – Parte 2 (Análise e Projeto)

A atividade *Análise Arquitetural – Parte 2* é desempenhada no fluxo *Desempenhar Síntese Arquitetural*. A exclusão desta atividade do processo causa impactos somente neste fluxo.

5.1. Impactos do Fluxo Desempenhar Síntese Arquitetural

Os impactos causados pela exclusão da atividade *Análise Arquitetural – Parte 2* podem ser vistos na Tabela 6.

Tabela 6 - Elementos Impactados pela Exclusão da Atividade Análise Arquitetural – Parte 2 no Fluxo Desempenhar Síntese Arquitetural

Elemento	Instância	Impacto	Regra de boa Formação
Sub-artefato	Sub-artefato Visão de *Deployment* de: - Documento de Arquitetura do Software Sub-Artefatos de: - Modelo de *Deployment*	Exclusão	Sub-artefatos produzidos pela atividade devem ser excluídos
	Sub-artefatos de: - Modelo de Caso de Uso - Especificação Complementar - Glossário - Documento de Visão - Lista de Riscos - Prova de Conceito Arquitetural - Modelo de Análise	Relação de entrada (consumo) com atividade excluída	Sub-artefatos consumidos pela atividade devem ter relação de associação eliminada
Fluxo	Desempenhar Síntese Arquitetural	Relação com atividade excluída	Atividades excluídas devem ter suas relações de associação com fluxos eliminadas.
Ferramenta	*Rational WebPublisher* *Rational SoDA* *Rational Rose* *Rational XDE Developer*	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com ferramentas eliminadas.
Papel	Arquiteto de Software	Relação de associação com atividade excluída	Atividades excluídas devem ter suas relações de associação com papéis eliminadas.
	Arquiteto de Software	Relação de associação com sub-artefatos do Modelo	Sub-artefatos excluídos devem ter suas relações de associação com

		de *Deployment* e Documento de Arquitetura de Software excluída	papéis eliminadas.
Tarefa	Tarefas de Análise dos Casos de Uso – Parte 2	Exclusão	Atividades excluídas devem ter suas tarefas eliminadas.

No fluxo *Desempenhar Síntese Arquitetural,* a atividade *Análise Arquitetural – Parte 2* é executada em paralelo a atividade *Análise Arquitetural – Parte 1*. Desta forma, não é necessário conectar atividades relacionadas (anteriores e próximas) durante a exclusão desta atividade, sendo necessário apenas remover a estrutura de paralelismo mantendo a conexão de suas atividades relacionadas com a atividade *Análise Arquitetural – Parte 1.*

.