PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UM MODELO DE INTEGRAÇÃO ENTRE A GERÊNCIA DE PROJETOS E O PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

MAURÍCIO COVOLAN ROSITO

ORIENTADOR: PROF. DR. RICARDO MELO BASTOS

DISSERTAÇÃO DE MESTRADO

PORTO ALEGRE
2008

MAURÍCIO COVOLAN ROSITO

UM MODELO DE INTEGRAÇÃO ENTRE A GERÊNCIA DE PROJETOS E O PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

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: Prof. Dr. Ricardo Melo Bastos

PORTO ALEGRE

2008

Dados Internacionais de Catalogação na Publicação (CIP)

R821m Rosito, Maurício Covolan
Um modelo de integração entre a gerência de projetos e o processo de desenvolvimento de software / Maurício Covolan Rosito. – Porto Alegre, 2008.
142 f.

Diss. (Mestrado) – Fac. de Informática, PUCRS
Orientador: Prof. Dr. Ricardo Melo Bastos

1. Informática. 2. Engenharia de Software. 3. Gerência de Projetos (Informática). 4. Software (Avaliação). I. Título.

CDD 005.1

Ficha Catalográfica elaborada pelo
Setor de Tratamento da Informação 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

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada **"Um Modelo de Integração entre a Gerência de Projetos e o Processo de Desenvolvimento de Software"**, apresentada por Maurício Covolan Rosito, como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Ciência da Computação, aprovada em 17/01/08 pela Comissão Examinadora:

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

Profa. Dra. Ana Paula Terra Bacelo — FACIN/PUCRS

Prof. Dr. Marcelo Blois Ribeiro — PPGCC/PUCRS

Homologada em 18/03/2008, conforme Ata No 05/08 pela Comissão Coordenadora.

Prof. Dr. Fernando Gehm Moraes
Coordenador.

Campus Central
Av. Ipiranga, 6681 – P32 – sala 507 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320-3621
E-mail: ppgcc@inf.pucrs.br
www.pucrs.br/facin/pos

AGRADECIMENTOS

Agradeço em primeiro lugar à minha noiva, Géssica Popow, minha fonte de inspiração e luz, por mais esta etapa cumprida em minha vida. Suas palavras de carinho e de incentivo foram muito importantes para amenizar minha ansiedade diante dos obstáculos.

Aos meus pais Mário e Laura Rosito pelo exemplo de vida, dedicação e amor. Obrigado por sempre me incentivarem a seguir em frente. Vocês contribuíram muito para meu crescimento pessoal. Esta conquista, sem dúvida, também é de vocês!

Ao meu irmão, Fernando Rosito, por sempre incentivar e acreditar no meu trabalho. Você, com certeza, contribuiu muito para esta vitória!

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

Aos membros da banca, professores Marcelo Blois Ribeiro e Ana Paula Terra Bacelo pela aceitação do convite de participação na avaliação deste trabalho. Ao professor Marcelo um agradecimento especial pelas sugestões e críticas dadas ao longo desta pesquisa.

Ao meu amigo e colega de pesquisa, professor Daniel Antonio Callegari, que conheci durante o mestrado. Você dividiu comigo momentos bons e ruins. Compartilhou experiências novas e foi um ótimo parcerio para passar horas discutindo sobre assuntos relacionados à esta pesquisa. Considero esta vitória sua também!

Ao bolsista e amigo Felipe Lima Foliatti da Silva. Sua participação neste trabalho foi muito importante. Obrigado pelo apoio e dedicação.

Ao professor Michael da Costa Móra por ser o primeiro incentivador, e talvez por isso o principal, para que eu realizasse o mestrado. Sem o seu incentivo, talvez eu não estivesse escrevendo neste momento minha dissertação.

Aos demais familiares e aos colegas de mestrado pelo apoio, torcida e profunda amizade. Graças à ajuda destas pessoas consegui superar diversos momentos de dificuldade que surgiram durante esta pesquisa.

Ao Programa de Pós-Graduação em Ciência da Computação e a todos os professores dos quais pude conviver durante estes dois anos.

“Nunca ande pelo caminho traçado,
pois ele conduz somente até onde os outros foram.”

(Alexander Grahan Bell)

RESUMO

As organizações de software estão constantemente procurando por processos de software bem definidos para o desenvolvimento de seus produtos e serviços. Entretanto, muitos processos de desenvolvimento de software existentes apresentam carências no quesito de gerência de projetos. Estes processos devem permitir que as organizações apliquem os conhecimentos gerenciais em conjunto com os aspectos técnicos. Neste contexto, esta pesquisa propõe um modelo que integra os conceitos da gerência de projetos e do processo de desenvolvimento de software, contribuindo não somente para a integração destes processos, mas também auxiliando os gestores no processo de tomada de decisões durante o planejamento de projetos. Ainda, em função da integração proposta, foi possível identificar um conjunto de regras (ou restrições) que visam, em sua maioria, apoiar a consistência do modelo de integração. Dessa forma, apresenta-se o modelo e os resultados de uma avaliação qualitativa de um protótipo que implementa o modelo proposto, realizado com gerentes de projetos de nove empresas.

Palavra-Chave: Gerência de Projetos. Processo de Desenvolvimento de Software. Modelo Integrado. Planejamento de Projetos.

ABSTRACT

Software organizations are constantly looking for better solutions when designing and using well-defined software processes for the development of their products and services. However, many existing software development processes lack for more project management skills in their models. These processes should allow organizations to apply management knowledge with technical aspects. In this context, this research proposes the definition of a model that integrates the concepts of project management and those available on the software development processes, helping not only process integration but also assisting managers in the decision making process during project planning. Still, through the proposed integration, it was possible to identify a set of rules (or restrictions) aimed, at the most, to facilitate the consistency of the integration model. We present the model and the results from a qualitative evaluation of a tool that implements the proposed model, conducted with project managers from nine companies.

Keywords: Project Management. Software Development Process. Integrated Model. Project Planning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CRC	*Class Responsibility Collaborator*
MSF	*Microsoft Solutions Framework*
OCL	*Object Constraint Language*
OMG	*Object Management Group*
OPEN	*Object-oriented Process, Environment and Notation*
OPF	*OPEN Process Framework*
PRINCE2	*Projects In Controlled Environments*
PMBOK	*Project Management Body of Knowledge*
PMI	*Project Management Institute*
RUP	*Rational Unified Process*
SMART	*Specific, Measurable, Agreed upon, Realistic, Time-Component*
SPEM	*Process Engineering Metamodel Specification*
SPIM	*Software Planning Integrated Model*
SPIT	*Software Planning Integrated Tool*
UML	*Unified Model Language*
WBS	*Work Breakdown Structure*
XP	*Extreme Programming*

SUMÁRIO

1 INTRODUÇÃO

A crescente preocupação relativa ao desenvolvimento de software pode ser observada devido à adoção de práticas de engenharia de software pelas empresas [PRE01]. Algumas das características mais desejadas destes modelos incluem a habilidade de capturar as melhores práticas de desenvolvimento de software, um bom nível de flexibilidade (a fim de lidar com diferentes tipos de projetos), bem como um bom apoio à gestão. A falta de metodologias específicas para a gerência de projetos de software e o aumento da complexidade e do número de projetos nas organizações contribui para o aumento dos problemas relacionados à gestão de projetos [KER00], [PRE01].

O desenvolvimento de produtos de software requer o planejamento e a execução de atividades definidas de acordo com o escopo do projeto, onde é necessário lidar tanto com assuntos gerenciais quanto técnicos. Em qualquer contexto deve-se considerar o fato de que os projetos sempre são esforços únicos e temporários, além de envolverem um elevado nível de incerteza [SCH02], [CAL07]. Entretanto, a maioria dos modelos ou guias voltados para a gerência de projetos não se dirigem especificamente a processos de desenvolvimento de software. Além disso, os processos de desenvolvimento de software, por sua vez, geralmente fornecem apenas um conjunto de práticas que atendem a determinadas atividades e fluxos de trabalho relacionados à gerência de projetos.

A fim de se obter um processo mais detalhado para o gerenciamento de projetos software, é necessário aplicar os conhecimentos de gestão de projetos aos processos de desenvolvimento do software. Portanto, se por um lado o *Project Management Body of Knowledge Guide* (PMBOK) [PMI00] pode fornecer uma perspectiva gerencial da solução, a visão sobre a produção deve ser obtida a partir de um modelo de processo de desenvolvimento de software, tal como os processos *Rational Unified Process* (RUP) [KRU00] e *Object-oriented Process, Environment and Notation* (OPEN) [GRA97].

Pesquisas anteriores apresentaram resultados interessantes, mas uma íntima integração da gerência de projetos e dos processos de software com resultados práticos ainda é uma questão em aberto [HEN00], [HEN01], [REH07], [SCH02]. Conseqüentemente, faz-se necessário mais estudo para uma solução que permita um melhor nível de integração entre os conceitos e modelos para estas duas áreas de conhecimento.

Esta pesquisa apresenta um modelo de integração entre a gerência de projetos e os processos de desenvolvimento de software denominado ***Software Planning Integrated Model* (SPIM)**. Este modelo inclui um conjunto de regras para o planejamento integrado de atividades gerenciais e produtivas no contexto de desenvolvimento de software. No SPIM são desenvolvidos os conceitos advindos da integração do PMBOK com o RUP em extensão à proposta apresentada em [CAL07]. Enquanto o PMBOK fornece uma perspectiva gerencial da solução, a visão sobre a produção é obtida a partir do RUP. Assim, ao aplicar o conhecimento de gerência de projetos em conjunto com um processo de desenvolvimento de software apropriado será possível obter um fluxo mais completo e integrado de atividades e suas dependências. A integração dos modelos também permitiu que um conjunto de regras (ou restrições) pudessem ser derivadas (Tabela 1). Tais características, em sua maioria, visam apoiar a consistência do modelo proposto.

Para demonstrar a viabilidade dos conceitos propostos pelo modelo integrado SPIM e pelo conjunto de regras derivadas deste modelo, foi desenvolvido um protótipo denominado ***Software Planning Integrated Tool* (SPIT)** que atua como um *Add-in* para uma ferramenta de gerenciamento de projetos já existente. Este software permite a avaliação do modelo SPIM em termos de sua funcionalidade e abrangência com relação à solução do problema proposto.

1.1 MOTIVAÇÃO

Segundo estudos empíricos, a eficácia de uma organização depende, em boa parte, do sucesso de seus projetos [KER00]. Desta forma, muitos pesquisadores trabalham na investigação dos fatores de sucesso dos projetos, tais como a definição de produto, qualidade de execução e técnicas de gerência de projetos [PIN87]. Além disso, diversos artigos na literatura salientam a importância de se utilizar processos de software bem definidos nas organizações. Entretanto, parece não haver estudos suficientes para suprir a carência no quesito de gerência de projetos destes processos.

De acordo com [KER00], o processo de desenvolvimento de software é um dos principais mecanismos responsáveis por gerenciar e controlar os projetos e produtos de software. Porém, o gerente de projetos pode não possuir todas as informações relevantes ao realizar planejamento de um projeto de software e, conseqüentemente, poderão ocorrer interações com outros departamentos da organização. O gerente de projetos, por exemplo, pode precisar contatar o setor de recursos humanos sobre a necessidade de contratação de pessoal para um determinado projeto de desenvolvimento de software. Percebe-se, dessa

forma, que o fluxo de atividades de um projeto de software pode interagir com os demais fluxos de atividades comuns da organização (aqui denominados **fluxos empresariais**).

Ambos os tipos de fluxos de trabalho são executados em paralelo, possuem recursos próprios e podem influenciar nos prazos das atividades e custos do projeto de software. Além disso, pode haver uma relação de dependência entre as atividades pertencentes a estes dois tipos de fluxos de trabalho. Por exemplo, a atividade de desenvolvimento de um *web site* (pertencente ao fluxo de trabalho do projeto de software) pode depender da aquisição de um servidor *web* pelo setor financeiro (pertencente ao fluxo empresarial da organização).

A desconsideração das dependências entre as atividades dos diferentes fluxos de trabalho pode resultar em distorções no planejamento de projetos de software. Os modelos de gerência de projetos e os processos de desenvolvimento de software atuais, porém, não apresentam uma solução que permita o planejamento de projetos de software considerando as interações do gerente de projetos com outros departamentos da organização. Esta solução deve considerar a complexidade de identificar a interdependência entre as atividades dos fluxos de trabalho da organização e o fluxo de trabalho do projeto.

Dessa forma, observou-se a necessidade de especificar um modelo que forneça algum tipo de suporte ao gerente de projetos para obter acesso as informações dos fluxos empresariais e acompanhar, de maneira integrada, o andamento das atividades destes tipos de fluxos de trabalho.

1.2 QUESTÃO DE PESQUISA

A questão de pesquisa abordada neste estudo foi a seguinte: **"Como realizar o planejamento integrado de atividades pertencentes ao fluxo de trabalho de um projeto de software e aos demais fluxos de trabalho da organização?"**.

1.3 OBJETIVOS

Uma vez definida a questão de pesquisa, definiu-se o objetivo geral e os objetivos específicos deste trabalho, os quais são apresentados a seguir.

1.3.1 OBJETIVO GERAL

O estudo realizado neste trabalho tem como objetivo geral propor um modelo que considera a integração dos conceitos da gerência de projetos e do processo de desenvolvimento de software, contribuindo não somente para a integração destes processos,

mas também auxiliando os gestores no processo de tomada de decisões durante o planejamento de projetos de software.

1.3.2 OBJETIVOS ESPECÍFICOS

Como objetivos específicos têm-se:

- Aprofundar o estudo teórico sobre a gerência de projetos e os processos de desenvolvimento de software;
- Permitir o planejamento de projetos considerando os elementos que compõem a gerência de projetos e o processo de desenvolvimento de software;
- Fazer a distinção dos tipos de atividades e produtos de trabalho;
- Permitir o planejamento integrado de atividades gerenciais e produtivas;
- Distinguir as possíveis relações entre uma atividade e um artefato (criar/atualizar/consultar);
- Permitir a integração dos conceitos de gerência de projeto fornecidos no PMBOK com os conceitos de desenvolvimento de software fornecidos pelo RUP e pelo OPEN.
- Definir um conjunto de regras (ou restrições) para apoiar a consistência do modelo de integração SPIM;
- Especificar uma ferramenta de software e desenvolver um protótipo que possibilite o uso do modelo SPIM; e finalmente
- Aplicar o modelo proposto, através do protótipo desenvolvido, em empresas e avaliar os resultados.

1.4 ETAPAS DA PESQUISA

O desenho da pesquisa se refere ao plano concebido para responder ao problema de pesquisa, onde aparecem os métodos a serem utilizados e a identificação das etapas do processo de 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: nesta etapa realizou-se o levantamento bibliográfico e o estudo do referencial teórico que permitiu aprofundar os conhecimentos sobre os processos de desenvolvimento de software. Esta análise resultou em um estudo comparativo dos processos de desenvolvimento de software RUP, OPEN e MSF em relação ao SPEM, permitindo

identificar as classes e relacionamentos comuns entre os metamodelos. Ainda nesta etapa, realizou-se um estudo sobre os conhecimentos de gerência de projetos advindos do PMBOK.

Etapa 2: após a análise dos resultados da etapa 1, selecionou-se dois processos de desenvolvimento de software (respectivamente o RUP e o OPEN) e identificou-se os elementos e relacionamentos comuns entre os metamodelos destes processos e os conceitos de gestão de projetos contidos no PMBOK.

Etapa 3: nesta etapa foi proposto um modelo de integração entre gerência de projetos e o processo de desenvolvimento de software que auxilia o planejamento integrado das atividades relacionadas a estas duas áreas de conhecimento.

Etapa 4: nesta etapa especificou-se uma ferramenta de software e desenvolveu-se um protótipo que possibilita o uso do modelo proposto. Assim, procurou-se comprovar a efetividade do modelo SPIM através do desenvolvimento de uma ferramenta integrada que auxilia o planejamento de projetos considerando os elementos que compõem o processo de desenvolvimento de software.

Etapa 5: após o desenvolvimento do protótipo, foi elaborado um questionário de avaliação do modelo e, posteriormente, aplicado em organizações de desenvolvimento de software. Ainda nesta fase, foi realizada a consolidação e avaliação dos resultados obtidos, permitindo a definição das conclusões desta pesquisa.

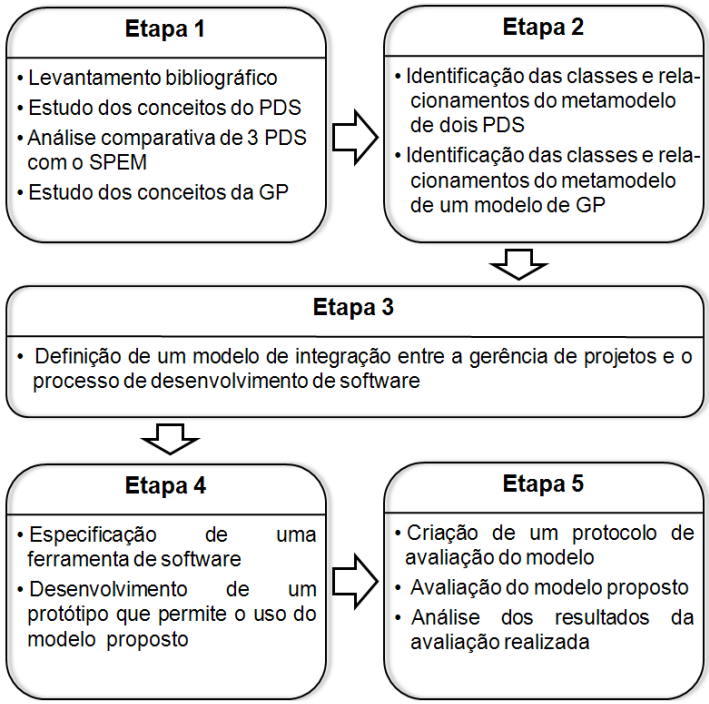


Figura 1. Desenho da pesquisa

1.5 ESTRUTURA DO TRABALHO

Este trabalho está dividido em três partes: fundamentação teórica, proposta do modelo integrado e sua avaliação em empresas de software. O Capítulo 2 apresenta a fundamentação teórica necessária para um bom entendimento do trabalho, onde são esclarecidos aspectos de processos de desenvolvimento de software, gerência de projetos, além das classes e relacionamentos dos metamodelos do PMBOK, RUP e OPEN.

No Capítulo 3 são apresentados alguns trabalhos encontrados na literatura que estão relacionados à análise da gestão de projetos nos processos de desenvolvimento de software. No Capítulo 4 são discutidos os problemas de gestão de projetos nos processos de desenvolvimento de software pesquisados neste trabalho. No Capítulo 5 são descritos os metamodelos base de integração oriundos da integração do PMBOK com o RUP e da integração do PMBOK com o OPEN (denominados PMBOK+RUP e PMBOK+OPEN, respectivamente). A partir da análise destes metamodelos é apresentado no Capítulo 6 um modelo de integração entre a gerência de projetos e o processo de desenvolvimento de software. Além disso, discorre-se sobre o conjunto de restrições identificadas para apoiar a consistência do modelo proposto. No Capítulo 7, é apresentado o protótipo desenvolvido para avaliar os conceitos propostos neste estudo.

No Capítulo 8 discorre-se sobre a pesquisa qualitativa realizada sobre a utilização do modelo proposto em empresas de desenvolvimento de software. Além disso, apresentam-se os resultados obtidos e as conclusões acerca destas informações. Por fim, no Capítulo 9 são apresentadas as conclusões, contribuições e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Para um bom entendimento do modelo proposto e dos trabalhos relacionados é necessário a compreensão dos fundamentos teóricos e metodologias básicas nos quais o modelo integrado se baseia. Assim, a seção 2.1 fornece uma visão geral sobre projetos e gerência de projetos. A seção 2.2 trata alguns dos conceitos relacionados aos processos de desenvolvimento de software. Já na seção 2.3 o interesse é fornecer uma visão geral do PMBOK, RUP e OPEN e apresentar os elementos e relacionamentos que constituem o metamodelo destes processos.

2.1 GERÊNCIA DE PROJETOS

Antes de examinar a gerência de projetos, é importante primeiramente compreender o conceito de projeto. De acordo com [SCH02], um projeto é um empreendimento temporário com o objetivo de produzir um único produto ou serviço. Geralmente um projeto é direcionado a alcançar um resultado específico e envolve a execução coordenada de atividades inter-relacionadas. Mais do que isso, os projetos são planejados, executados e controlados por pessoas, e são restringidos por recursos limitados [PMI00]. Dessa forma, [SCH02] define um projeto de acordo com as seguintes características:

- **Único:** Os projetos envolvem o desenvolvimento de algo que nunca foi feito antes e, portanto, que é único. É importante considerar que um produto ou serviço pode ser único mesmo considerando que já tenha sido desenvolvida uma infinidade de produtos/serviços em sua categoria;
- **Temporário:** os projetos possuem início e fim bem definidos. O fato de um projeto ser considerado temporário não significa que a sua duração é curta, pois muitos projetos duram vários anos. A maioria dos projetos são empreendidos para criar um resultado duradouro. Assim, chega-se ao fim do projeto quando os seus objetivos foram alcançados ou quando se torna claro que os objetivos do projeto não serão ou não poderão mais ser atingidos ou a necessidade do projeto não existe mais;
- **Objetivo:** Todo projeto deve ter um ou mais objetivos, uma vez que é esperado um conjunto de resultados. Os objetivos devem seguir o conceito definido pela

sigla SMART (*Specific, Measurable, Agreed upon, Realistic, Time-Component*), ou seja, devem ser específicos, mensuráveis, acordado entre os interessados no projeto (*stakeholders*), realistas e possuírem uma data de fim;

- **Requerem recursos:** Os recursos incluem pessoas, hardware e software, considerando que estes não são ilimitados. Quando necessitam de recursos, os gerentes de projetos os requisitam para as demais áreas da organização, disputando estes recursos com essas áreas e com outros projetos. Ou seja, os recursos devem ser utilizados de maneira eficaz a fim atender aos objetivos do projeto e da empresa. Esta gestão é realizada pelo gerente de projetos;
- **Possuem um patrocinador principal:** Um projeto deve ter um patrocinador principal (*sponsor*) responsável por financiar os custos e estabelecer as principais necessidades do projeto;
- **Envolvem incertezas:** Projetos envolvem incertezas, pois são considerados únicos. Assim, é difícil definir de maneira clara e intuitiva os fatores importantes para o sucesso do projeto, tais como: determinar os objetivos do projeto, estimar a sua data de término e identificar os custos envolvidos.

Em contrapartida, a gerência de projetos é responsável pelo controle da realização dos objetivos do projeto através da aplicação de um conjunto de técnicas e ferramentas [SCH02]. A gerência de projetos tem como objetivo conseguir a conclusão do projeto dentro de um cronograma e orçamento definidos de acordo com um conjunto de especificações previamente determinadas [PMI00]. Estes elementos caracterizam a restrição tripla (*triple constraint*) da gerência de projetos, onde um projeto é composto por três componentes básicos: escopo, tempo e custo. Assim, um projeto pode ser considerado bem sucedido quando consegue atender a estes três objetivos e satisfazer aos seus patrocinadores (*sponsors*).

Segundo [PRE01], os principais problemas da gerência de projetos estão relacionados à falta de processos adequados e padronizados de gestão. Ainda de acordo com [PRE01], os principais motivos do insucesso nos projetos estão relacionados à falta de procedimentos, metodologias e padrões de gestão. O uso de conceitos relacionados ao sucesso é relativamente ambíguo na literatura. No contexto de projetos, o sucesso é freqüentemente conceituado por meio de critérios e fatores de sucesso. Com relação aos fatores críticos de sucesso, foram propostas diferentes listas e modelos na literatura, tais como em [PIN87], [GOB87], [BEL96] e [COO02]. Ainda, alguns pesquisadores acreditam que os fatores de sucesso são diferentes para tipos diferentes de projetos. Porém, é possível identificar que a

atividade de planejamento auxilia no gerenciamento de informações vitais para o projeto (tais como custo, cronograma, qualidade e recursos), de maneira colaborar para o bom andamento dos projetos.

A gerência de projetos em um ambiente de desenvolvimento de software é definida como a gerência das pessoas e de outros recursos por um gerente de projetos a fim de planejar, analisar, projetar, construir, testar e manter um sistema de informação [SCH02]. Para cumprir estes objetivos, um gerente de projetos precisa de algum tipo de suporte, geralmente baseado em uma metodologia de gerência de projetos, que permita lidar com diferentes variáveis de projeto, responsabilidades e tarefas. Para este fim, existem diversas propostas na literatura ou práticas já realizadas nas empresas. Entretanto, a maioria dos modelos ou guias voltados para a gerência de projetos, tal como o PMBOK Guide, não se dirigem especificamente a processos de desenvolvimento de software.

Além disso, apesar do processo de desenvolvimento de software ser considerado um dos principais mecanismos responsáveis por gerenciar e controlar os projetos e produtos de software, muitos destes processos existentes apresentam carências no quesito de gerência de projetos. Na próxima seção, discorre-se sobre o conceito de processo de desenvolvimento de software.

2.2 PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

De uma maneira simplificada, um processo do desenvolvimento de software é um conjunto de atividades e resultados relacionados que conduzem à produção de um software [JAC99]. O processo de desenvolvimento de software, segundo [PRE01], é o conjunto de atividades parcialmente ordenadas que um projeto deve seguir para desempenhar alguma tarefa. Esta tarefa deve ter como objetivo atingir uma determinada meta e geralmente está associada ao desenvolvimento de um ou mais produtos.

Os processos de desenvolvimento software podem apresentar grande complexidade e, dessa forma, possibilitar diversas alternativas de execução de suas atividades. Assim, é importante para as organizações que o processo de desenvolvimento de software seja padronizado de forma a garantir uma maior qualidade dos produtos de software. Segundo [DER99], os principais conceitos ligados à sua modelagem são os seguintes:

- **Atividade:** As atividades visam criar ou modificar um conjunto de artefatos, incorporando e executando procedimentos, regras e políticas organizacionais;

- **Artefato:** informação desenvolvida e mantida em um projeto de software;
- **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.

A importância da utilização de um processo de desenvolvimento de software padrão deve-se ao fato de que este atua como um guia para a execução de todos os projetos dentro de uma organização. Assim, muitos processos ou guias tais como RUP, *Extreme Programming* (XP) [BEC04], MSF e OPEN estão sendo utilizados como base no desenvolvimento de processos de desenvolvimento de software padrão nas organizações.

Entretanto, os processos de desenvolvimento de software, embora identifiquem a importância das atividades relacionadas à gestão, não tratam de forma adequada os aspectos relacionados à gerência de projetos. A seguir são a apresentados os modelos de gestão de projetos e de processo de desenvolvimento de software utilizados para realizar a integração proposta neste trabalho.

2.3 MODELOS DE GESTÃO DE PROJETOS E DE PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Nesta seção apresenta-se uma visão geral do PMBOK e dos processos de desenvolvimento de software RUP e OPEN. Além disso, são descritas as classes e relacionamentos dos metamodelos pesquisados para o desenvolvimento do modelo de integração proposto neste trabalho.

2.3.1 PMBOK - PROJECT MANAGEMENT BODY OF KNOWLEDGE

Reconhecido internacionalmente pelo seu esforço em definir normas e dar suporte aos profissionais de gerência de projetos, o *Project Management Institute* (PMI) [PMI06] publicou um guia geral de gerência de projetos, o PMBOK Guide. O *Project Management Body of Knowledge* reúne as melhores práticas aplicáveis à maioria dos projetos e sobre as quais há um amplo consenso sobre o seu valor e utilidade.

De acordo com [PMI00] e [PMI06], o principal objetivo do PMBOK Guide é identificar um subconjunto dos conhecimentos sobre gerência de projetos que seja reconhecido, genericamente, como sendo uma coleção de boas práticas. O PMBOK, porém, não é um processo sem seu sentido estrito, pois não determina quais são as ações e nem indica como estas devem ser executadas para o correto desenvolvimento de um projeto. Além disso,

o PMBOK é dito mais compatível com atividades industriais e, portanto, não aborda especificamente processos de desenvolvimento de software [SCH02], [CAL07].

2.3.1.1 CICLO DE VIDA DO PMBOK

O ciclo de vida de um projeto estabelece uma seqüência de fases, determinando o início e fim do projeto, com o objetivo de garantir um bom gerenciamento do mesmo. Dessa forma, as organizações geralmente dividem os projetos em várias fases visando um melhor controle gerencial. Porém, o número das fases varia de acordo com o escopo do projeto e o domínio da aplicação.

De acordo com [SCH02], o PMBOK não prescreve nenhum ciclo de vida específico para projetos. Especifica apenas que o ciclo de vida do projeto deve ser dividido em quatro fases principais: Iniciação, Desenvolvimento, Implementação e Encerramento. Estas fases estão descritas abaixo:

- **Iniciação:** consiste no levantamento inicial dos requisitos e análise da viabilidade dos projetos;
- **Desenvolvimento:** consiste no refinamento das informações levantadas na fase de concepção objetivando a execução do projeto;
- **Implementação:** consiste na realização das ações planejadas objetivando atingir os resultados esperados;
- **Encerramento:** consiste na entrega final dos produtos gerados do projeto e análise dos resultados obtidos.

2.3.1.2 DIMENSÕES DO PMBOK

O PMBOK sugere que o gerenciamento de projetos seja realizado através de processos gerenciais. Estes processos consistem em uma série de ações com o objetivo de alcançar os resultados do projeto. Dessa forma, um projeto é realizado através da execução de processos. De acordo com [CHA06], o PMBOK define a seguinte estrutura para o gerenciamento de processos: grupos de processos e áreas de conhecimento.

2.3.1.2.1 GRUPOS DE PROCESSOS

Os processos de gerência de projetos podem ser organizados em cinco grupos, cada um deles contendo um ou mais processos:

- **Iniciação:** reconhecer que um projeto deve começar e executá-lo.

- **Planejamento:** planejar e manter um esquema de trabalho viável para se atingir aqueles objetivos de negócios que determinaram a existência do projeto.
- **Execução:** coordenar pessoas e outros recursos para realizar o plano de desenvolvimento.
- **Monitoramento e Controle:** assegurar que os objetivos do projeto estão sendo atingidos através da monitoração e da avaliação do seu progresso, tomando ações corretivas quando necessárias.
- **Encerramento:** formalizar a aceitação do projeto ou fase e encerrá-lo de uma forma organizada.

Cada etapa ou fase do projeto contempla um conjunto de processos gerenciais. Entretanto, os processos de gerenciais não estão exatamente estanques no tempo; há uma sobreposição durante o desenvolvimento das fases. A Figura 2 representa o nível de interação entre esses processos ao longo de cada etapa do projeto.

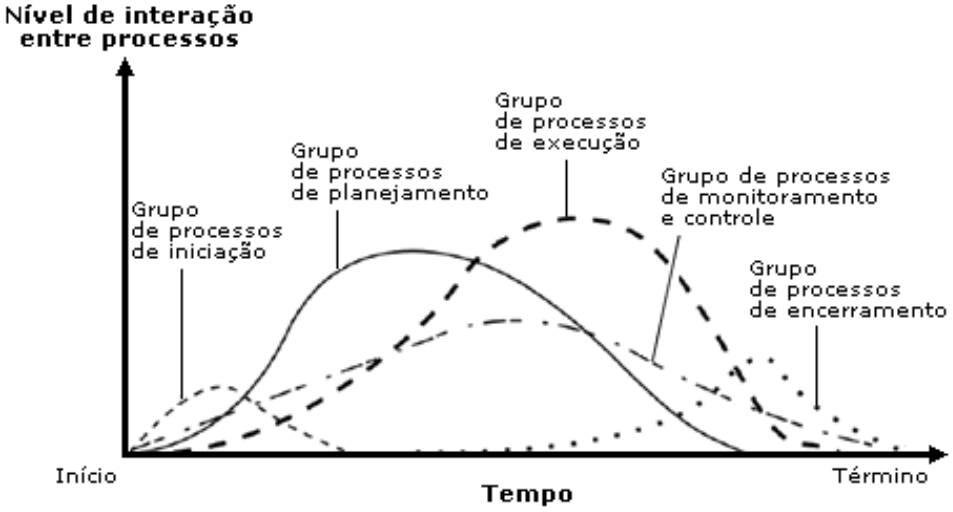


Figura 2. Interação de grupo de processos de um projeto [PMI06]

2.3.1.2.2 ÁREAS DE CONHECIMENTO DA GERÊNCIA DO PROJETO

As áreas de conhecimento do PMBOK descrevem as principais competências que os gerentes de projeto devem desenvolver para a execução dos processos gerenciais [PMI00]. Estes processos foram organizados em nove áreas de conhecimento conforme descrito abaixo:

- **Gerência de Integração:** Descreve os processos necessários para assegurar que os diversos elementos do projeto sejam adequadamente coordenados.

- **Gerência de Escopo:** Descreve os processos necessários para assegurar que o projeto contemple todo o trabalho requerido, e nada mais que o trabalho requerido, para completar o projeto com sucesso.
- **Gerência de Tempo:** Descreve os processos necessários para assegurar que o projeto termine dentro do prazo previsto.
- **Gerência de Custo:** Descreve os processos necessários para assegurar que o projeto seja completado dentro do orçamento previsto.
- **Gerência de Qualidade:** Descreve os processos necessários para assegurar que as necessidades que originaram o desenvolvimento do projeto serão satisfeitas.
- **Gerência de Recursos Humanos:** Descreve os processos necessários para proporcionar a melhor utilização das pessoas envolvidas no projeto.
- **Gerência de Comunicação:** Descreve os processos necessários para assegurar que a geração, captura, distribuição, armazenamento e pronta apresentação das informações do projeto sejam feitas de forma adequada e no tempo certo.
- **Gerência de Risco:** Descreve os processos que dizem respeito à identificação, análise e resposta aos riscos do projeto.
- **Gerência de Aquisição:** Descreve os processos necessários para a aquisição de mercadorias e serviços fora da organização que desenvolve o projeto.

As áreas de conhecimento são subdivididas em áreas centrais e áreas de apoio. Além disso, cada área de conhecimento abrange diversos processos de gerenciamento de projetos.

2.3.1.3 METAMODELO DE GERÊNCIA DE PROJETOS BASEADO NO PMBOK

Para efeito da discussão proposta por esta pesquisa, observa-se que os conceitos do PMBOK são, em sua maioria, representados por textos descritivos. Entretanto, objetivando comparar dois modelos e, posteriormente, executar a integração entre eles, deve-se representá-los sob estruturas compatíveis. Dessa forma, será utilizado o metamodelo previamente desenvolvido em [CAL07] que utiliza a notação da linguagem *Unified Modeling Language* (UML) [OMG07] para representar seus elementos.

Como podemos observar na Figura 3, este modelo foi desenvolvido observando os principais conceitos contidos no PMBOK, tais como programa, projeto, recursos, atividades, papéis, produtos e classes associadas. É importante salientar que este modelo foi desenvolvido com uma visão para atender as necessidades exclusivas dos projetos de software.

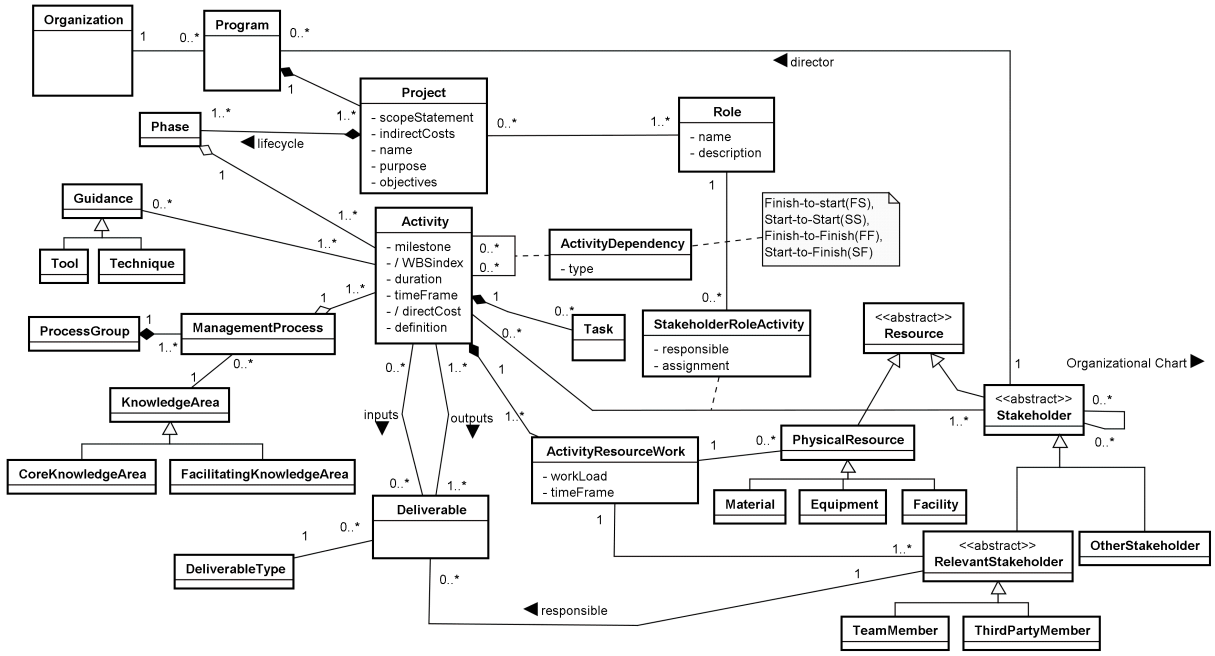


Figura 3. Metamodelo de gerência de projetos baseado no PMBOK Guide [CAL07]

Neste modelo, a classe *Organization* representa uma empresa que se organiza por programas (classe *Program*). Os programas são grupos de projetos (classe *Project*) designados a alcançar um objetivo estratégico. As organizações geralmente dividem os projetos em várias fases (classe *Phase*) visando um melhor controle gerencial.

Os recursos necessários para um projeto tais como pessoas, equipamentos e locais, são representados pela classe *Resource*. Estes recursos são divididos em recursos ativos (classe *Stakeholder*) e não-ativos (classe *PhysicalResource*). Os *stakeholders* correspondem às pessoas e organizações cujos interesses são afetados pelo projeto [PMI00].

Uma unidade particular de trabalho que é desempenhada por um papel durante o andamento do projeto, é representada pela classe *Activity*. Os *stakeholders*, por sua vez, podem desempenhar diversos papéis (classe *Role*) durante a execução das atividades do projeto. Assim, para cada associação entre um papel e uma atividade (representado pela classe associativa *StakeholderRoleActivity*) deve haver também uma associação dessa atividade com um *stakeholder* capaz de desempenhar aquele papel. A definição da carga de trabalho (atributo *workload*) dos recursos físicos (ao associar-se a diferentes atividades) é representada pela classe *ActivityResourceWork*.

O PMBOK Guide representa suas práticas em duas dimensões lógicas. Uma dimensão descreve as áreas de conhecimento (classe *KnowledgeArea*) enquanto que a outra dimensão descreve os processos gerenciais de um projeto (classe *ManagementProcess*), os quais estão contidos em cinco grupos de processo (classe *ProcessGroup*). As áreas de

conhecimento são responsáveis por descrever as principais competências que os gerentes de projeto devem desenvolver e derivam as áreas centrais (classe *CoreKnowledgeArea*) e as de apoio (classe *FacilitatingKnowledgeArea*). Assim, cada atividade gerencial pertence a um processo gerencial, sendo também relacionada a uma área de conhecimento.

Os produtos de trabalho que servem de entrada ou saída durante a execução das atividades são representados pela classe *Deliverable*. Além disso, a classe *DeliverableType* define o tipo de produto de trabalho, tais como um contrato, uma proposta comercial e assim por diante. Finalmente, a classe *Guidance* e representa os elementos de orientação, tais como ferramentas (classe *Tool*) e técnicas (classe *Technique*), para as atividades.

2.3.2 RUP - RATIONAL UNIFIED PROCESS

O *Rational Unified Process* (RUP) é um processo iterativo de desenvolvimento de software desenvolvido pela empresa *IBM Rational Software*, originado a partir do metamodelo SPEM [JAC99], [BEN06]. O RUP atua como um framework que pode ser adaptado e estendido de acordo com as características do processo de desenvolvimento de software da organização [RAT06]. Conforme a Figura 4, o RUP contém seus elementos em duas dimensões distintas: dinâmica e estática.

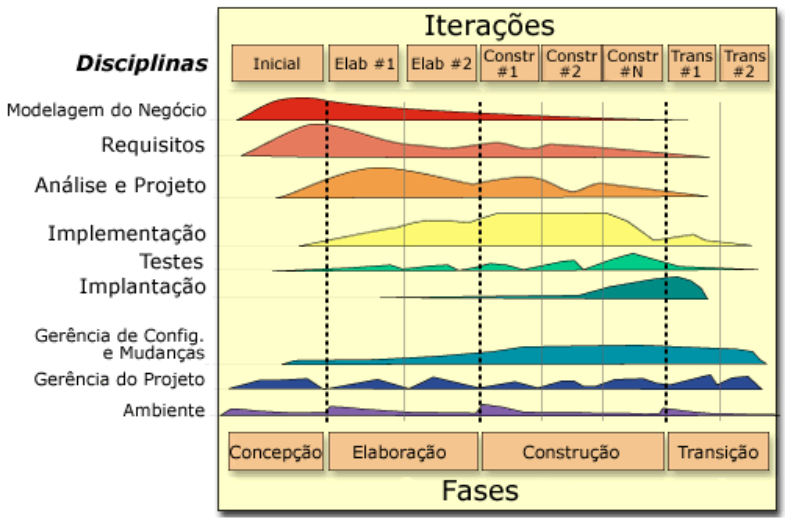


Figura 4. Visão geral da arquitetura do RUP [JAC99]

A dimensão dinâmica representa o tempo, mostrando os aspectos dinâmicos do processo através de ciclos, fases, iterações e *milestones*. Já a dimensão estática representa os aspectos estáticos descrevendo como os elementos do processo, atividades, disciplinas, artefatos e papéis, são agrupados em workflows.

2.3.2.1 DIMENSÃO DINÂMICA

De acordo com [JAC99], o ciclo de vida do software é divido em quatro fases (Concepção, Elaboração, Construção e Transição) e, através de sua característica iterativa, cada fase é realizada com base no resultado da fase anterior, de maneira a refinar o sistema até o momento em que o produto final esteja completo. Cada fase possui um *milestone* e um conjunto de objetivos, que devem ser utilizados para servir como guia no momento de decidir quais atividades desempenhar e quais artefatos devem ser produzidos [BEN06].

Cada fase no processo RUP atravessará diversas iterações. Uma iteração é um laço completo do desenvolvimento tendo por resultado uma liberação (interna ou externa) de um produto executável, um subconjunto do produto final sob o desenvolvimento, que cresce incremental de iteração em iteração até se transformar no sistema final [JAC99].

2.3.2.2 DIMENSÃO ESTÁTICA

Um processo deve ser capaz de descrever quem, como e quando uma determinada atividade está sendo desempenhada. Desta forma o RUP é representado através de quatro elementos primários de modelagem: papéis (quem), atividades (como) artefatos (o quê) e fluxos (quando).

Um papel define o comportamento e as responsabilidades de um indivíduo ou grupo de indivíduos que trabalham juntos como uma equipe [JAC99]. Os papéis são responsáveis pela execução das atividades do processo em cada uma das fases. Um indivíduo pode executar as tarefas de um ou mais papéis, da mesma forma que um grupo de indivíduos pode executar as atividades de um mesmo papel.

Uma atividade é uma unidade de trabalho que define como tarefas reais, atribuídas a um papel, devem ser executadas no contexto do projeto [RAT06]. Toda atividade deve ser atribuída a um papel. A atividade tem um propósito claro, normalmente expresso em termos de criar ou modificar artefatos.

Um artefato é um pedaço de informação que é produzida, modificada ou usada por um processo [RAT06]. Os artefatos são usados como entradas para executar uma atividade e são o resultado ou a saída de tais atividades. Os artefatos possuem diferentes formas de apresentação, tais como modelos, elementos de modelo, documentos, códigos-fonte e executáveis. Deve-se ainda considerar que um artefato pode ser composto de outros artefatos.

Os fluxos são as seqüências de atividades, que através das iterações entre os papéis, produzem um resultado de valor observável [RAT06]. No RUP podemos dividir os fluxos em dois grupos principais: fluxos centrais (*core workflows*), que são as disciplinas do processo, e os detalhes de fluxo (*workflow detail*), que são os fluxos internos de cada disciplina. Através da UML, um fluxo pode ser expresso como um diagrama de seqüência, colaboração ou de atividade [BOO00]. É importante ressaltar que um fluxo não pode ser interpretado literalmente como sendo um conjunto de passos no qual um indivíduo irá executar de forma automática e mecânica.

De acordo com [PEP06], cada disciplina é expressa através dos papéis (quem executa a tarefa), das atividades (como executa estas tarefas), e dos artefatos (o que a atividade consegue). Dessa forma, uma disciplina apresenta as atividades relacionadas de diferentes papéis em um fluxo de informação, assim, definindo como as atividades interagem com os artefatos.

2.3.2.3 METAMODELO SEMÂNTICO DO RUP

De acordo com [JAC99], o RUP utiliza os conceitos da linguagem UML, mantido pela organização *Object Management Group* (OMG), para especificar e documentar os modelos de sistemas de software. O RUP apresenta um modelo semântico, ilustrado na Figura 5, contendo seus principais elementos e relacionamentos [PEP06]. Este modelo determina como os elementos do processo são organizados e quais as relações válidas entre estes elementos.

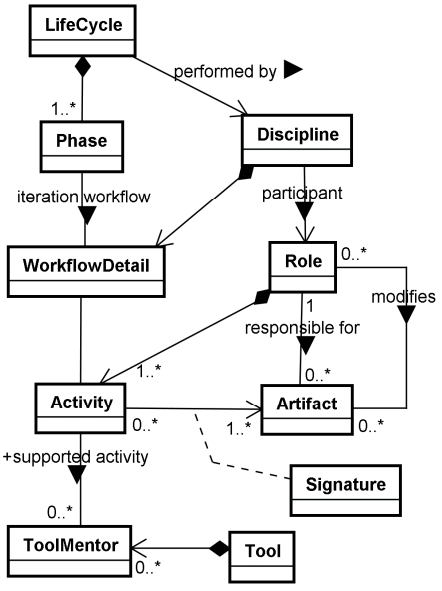


Figura 5. Modelo semântico do RUP [PEP06]

Na Figura 5, a classe *Lifecycle* representa o ciclo de vida de desenvolvimento de um software. Este conceito é particionado em um conjunto de quatro fases (classe *Phase*). A classe *Discipline* divide os elementos de processo em áreas de interesse. Um papel (classe *Role*) representa o elemento responsável por desempenhar atividades (*Activity*) para produzir ou modificar os artefatos (*Artifact*) do processo.

As informações de como os papéis devem colaborar entre si através de suas atividades são definidos pela classe *Workflow Detail*. A classe *Artifact* descreve os tipos de produtos de trabalho que são produzidos ou consumidos no desempenho de atividades. Assim, a classe associativa *Signature* indica que um artefato é utilizado como entrada ou saída de uma atividade. A classe *Tool* descreve as ferramentas que podem ser utilizadas auxiliando a produção ou modificação de um artefato. Finalmente, a classe *ToolMentor* descreve o uso de ferramentas no contexto de algumas atividades.

2.3.3 OPEN - OBJECT-ORIENTED PROCESS, ENVIROMENT AND NOTATION

O OPEN é uma metodologia de desenvolvimento de software orientado a objetos mantido pelo *OPEN Consortium Group* [GRA97], [OPE02], [OPF07]. Ele pode ser definido como um framework, denominado *OPEN Process Framework* (OPF), que fornece um metamodelo extensível e que pode ser configurado para processos de desenvolvimento de software distintos [GRA97]. O OPEN encapsula os conceitos e atividades relacionados ao negócio, qualidade, análise e reuso, e que são comuns a todo o processo de desenvolvimento de software, utilizando o paradigma de orientação a objetos.

Um processo é instanciado e customizado a partir do metamodelo do OPEN através da adição e remoção dos componentes de processo. Esta operação permite atender melhor as necessidades de uma organização em termos de tamanho, cultura, investimento e outras características, e envolve a escolha de atividades, tarefas, técnicas e configurações específicas para o negócio.

Uma unidade de processo (*Process Unit*) define um conjunto de atividades relacionadas que são executadas durante um projeto. Também define as entradas necessárias para gerar as saídas (*deliverables*), através da execução uma série de atividades [OPE02]. Além disso, o OPF define que as unidades do trabalho (*work units*) são componentes que modelam as operações que são executadas durante uma unidade de processo.

O OPF considera os seguintes tipos de unidades de trabalho:

- **Atividades:** definem o que precisa ser feito (fluxo);
- **Tarefas:** definem o que fazer de forma coesa;
- **Técnicas:** definem como tarefas serão realizadas.

O ciclo de vida de desenvolvimento de projetos do OPEN descreve o tempo de duração em que o projeto é construído [OPF07]. O ciclo de vida é formado por um conjunto de atividades que produzem e consomem produtos de maneira gradativa durante a realização de tarefas. É um processo baseado em entregas (*releases*) onde cada estágio envolve uma ou mais entregas. Em todo estágio do ciclo de vida do OPEN muitas tarefas podem ser executadas e, para cada tarefa, diferentes técnicas podem ser utilizadas (Figura 6).

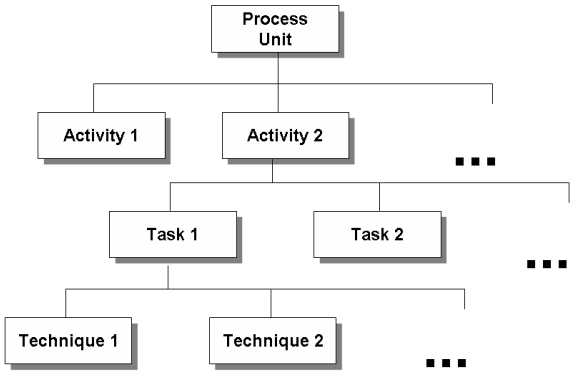


Figura 6. Relacionamento entre atividades, tarefas e técnicas [OPE02]

Cada atividade é definida como um conjunto de tarefas, que são a menor unidade de trabalho gerenciável. As tarefas são realizadas através do uso de técnicas. Dessa forma o OPEN inclui os conceitos de atividades com suporte ao ciclo de vida completo, além de tarefas e conjuntos de técnicas e artefatos.

2.3.3.1 COMPONENTES CENTRAIS DO OPEN

O framework do OPEN contém seu foco na interação cooperativa entre os produtores, suas unidades de trabalho e o que produzem [OPE02]. Dessa forma, o OPF reconhece as classes ilustradas na Figura 7 como sendo os componentes centrais de seu framework.

Neste modelo, a classe *Endeavor* refere-se ao componente que modela o esforço empreendido pelos produtores (*Producer*) que executam unidades de trabalho (*Work Unit*) durante um ou mais estágios (*Stage*). Os componentes produzidos durante o desenvolvimento do projeto são definidos pela classe *Work Product*. A classe *Language* modela o tipo de linguagem utilizada para documentar e produzir os produtos do projeto.

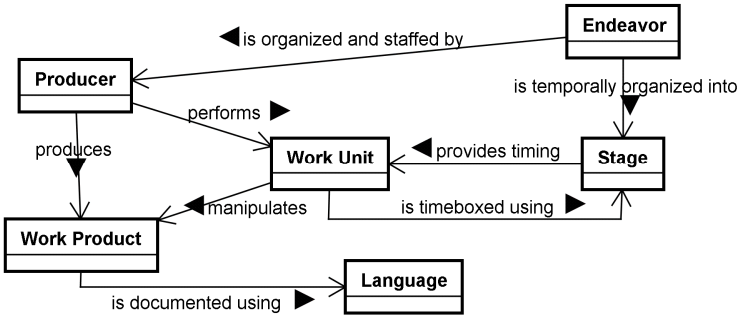


Figura 7. Componentes centrais ao framework do OPEN [OPF07]

Um produtor é o elemento responsável por produzir ou modificar – direta ou indiretamente – os artefatos do processo. Os produtores podem ser ferramentas ou pessoas definidas através de papéis. A classe *Work Unit* consiste de um conjunto de operações coesas executadas pelo produtor no desenvolvimento seu trabalho, e podem ser classificadas como tarefas, técnicas, fluxos de trabalho e atividades. Finalmente, a classe *Stage* determina as divisões de intervalos de tempo do processo, sendo dividida em estágios com duração (fases) e instantâneos (*milestones*).

2.4 CONSIDERAÇÕES SOBRE O CAPÍTULO

As seções anteriores apresentaram os modelos básicos utilizados na construção do modelo de integração proposto. Inicialmente, as definições apresentadas de gerência de projetos e processos de desenvolvimento de software esclarecem as características fundamentais relacionadas a estes conceitos, fornecendo indícios para a criação do modelo de integração que auxilia o planejamento de projetos considerando os elementos que compõem o processo de desenvolvimento de software.

Após, foi apresentado o PMBOK, documento redigido e mantido pelo PMI onde estão registradas as melhores práticas de gerência de projetos defendidas pela organização. É importante ressaltar que o PMI é uma instituição conhecida internacionalmente pela sua ação de apoio e fundamentação da área de gestão de projetos. Também foi apresentada uma visão geral dos processos de desenvolvimento RUP e OPEN, ressaltando a importância destes processos nos projetos de software.

O estudo do PMBOK, RUP e OPEN permitiu o levantamento das características básicas da estrutura de seus metamodelos, ajudando na posterior integração entre os conceitos de gerência de projetos e dos processos de desenvolvimento de software proposto neste trabalho.

Por fim, constatou-se que enquanto a maioria dos modelos ou guias voltados para a gerência de projetos, tal como o PMBOK Guide, não se dirigem especificamente a processos de desenvolvimento de software, os processos de desenvolvimento do software existentes possuem carências no quesito de gerência de projetos em suas metodologias ou modelos. Neste sentido, outras abordagens foram estudadas na tentativa de encontrar referências para solucionar estas carências. A seguir serão apresentadas as abordagens que de alguma forma contribuíram para a elaboração deste trabalho.

3 TRABALHOS RELACIONADOS

Este capítulo tem como principal objetivo apresentar os principais trabalhos relacionados à gestão de projetos nos processos de desenvolvimento de software.

Atualmente, alguns trabalhos vêm sendo desenvolvidos pela comunidade de engenharia de software salientando a importância de aplicar os conceitos de gestão de projetos nos processos de software.

Como o principal interesse desta pesquisa está relacionado à integração dos conceitos de gerência de projetos com os processos de desenvolvimento de software, nesse capítulo serão descritos os trabalhos propostos por [HEN00], [HEN01] e [REH07], os quais serviram como referência para esta pesquisa.

3.1 ABORDAGEM DE HENDERSON-SELLERS ET AL. 2000 [HEN00]

Em [HEN00], dois importantes processos de desenvolvimento de software, respectivamente o RUP e o OPEN, são analisados a partir um ponto de vista da gestão de projetos. Os autores concluem que ambos os processos são deficientes em certas áreas de conhecimento padrão da gestão de projetos, como a gerência de aquisição, de comunicação e de pessoas aquisição de gestão, gestão da comunicação e gestão de pessoal.

O gerenciamento de projetos no RUP é realizado através da execução de um conjunto de atividades de apoio definidas e agrupadas dentro de um fluxo de trabalho denominado “Fluxo de Gerenciamento de Projeto”. Este fluxo é executado em paralelo com os demais fluxos de trabalho do RUP. Porém, o RUP não define um conjunto de tarefas essenciais para o gerenciamento de projetos, ele somente indica a importância em se realizar este controle. Neste sentido, apesar do RUP auxiliar na execução das melhores práticas para o desenvolvimento de software, ele não cobre assuntos essenciais de gerência de projetos, tais como a gerência de pessoas e a gerência de contratos.

O OPEN, entretanto, possui um ciclo de vida dirigido a contrato onde as atividades são iniciadas somente quando um conjunto de pré-condições é aceito. Estas atividades são compostas por tarefas que devem ser realizadas para atingir o objetivo geral da atividade. Dessa forma, o OPEN apresenta um conjunto de tarefas responsáveis pelo planejamento e

gerenciamento dos projetos de software. No entanto, o OPEN mostrou-se deficiente em áreas essenciais de conhecimento da gerência de projetos, particularmente, a gerência de aquisição, de comunicação e de pessoas.

Dessa forma, os autores concluíram que a fim de apoiar ao conjunto completo de técnicas de gestão de projetos, novas extensões para estas áreas de conhecimento são desejáveis.

3.2 ABORDAGEM DE HENDERSON-SELLERS ET AL. 2001 [HEN01]

Uma avaliação qualitativa dos elementos componentes do RUP e do OPEN é realizada em [HEN01]. Os autores centralizam sua comparação sobre os aspectos do processo e da arquitetura do metamodelo, conceitos e terminologia utilizada, e o suporte a gestão de projetos.

O ciclo de vida do RUP baseia-se em quatro fases, através do qual o projeto evolui linearmente no tempo. Cada uma destas fases é composta por uma ou mais iterações. Cada iteração segue o padrão *waterfall* contendo atividades básicas de especificação de requisitos, análise, projeto, implementação, teste e implantação, que resultam em um *release* (interno ou externo) de um produto executável que cresce gradativamente a partir de cada iteração até se tornar o sistema final. Entretanto, os autores concluem que a arquitetura do metamodelo do RUP não permite o apoio a um verdadeiro desenvolvimento iterativo.

Em contrapartida, o metamodelo do OPEN apresenta um melhor suporte para desenvolver um processo que atenda as necessidades específicas de um determinado domínio de aplicação ou para adaptar um processo a projetos específicos. Além disso, o OPEN apresenta um conjunto de atividades e técnicas que contemplam diferentes áreas da gerência de projetos, tais como qualidade, estimativas de custo e métricas de gerenciamento.

3.3 ABORDAGEM DE REHMAN E HUSSAIN 2007 [REH07]

Em [REH07], quatro importantes metodologias/*frameworks* de gerência de projetos - *Projects In Controlled Environments* (PRINCE2) [OGC07], RUP, *Agile Development Methods* e MSF - foram comparados com o PMBOK.

No caso do PMBOK e do PRINCE2, suas documentações não informam como utilizar alguma das técnicas ou ferramentas descritas. Estas metodologias apenas estabelecem os processos e o conjunto de ferramentas e técnicas que podem ser adotadas em projetos.

Além disso, metodologias como RUP, PRINCE2 e MSF não fornecem informações relacionadas a algumas das áreas de conhecimento de gestão de projetos.

Os autores concluem que todas as metodologias/*frameworks* discutidas possuem ferramentas e procedimentos comuns. Eles também propõem uma abordagem que combina os conceitos destas metodologias objetivando obter os melhores resultados de cada abordagem.

3.4 CONSIDERAÇÕES SOBRE O CAPÍTULO

Como apresentado nas seções anteriores, diferentes são as abordagens propostas para dar apoio gerencial aos projetos de desenvolvimento de software, cada qual com suas contribuições e limitações.

Analisando o suporte oferecido pelos métodos descritos nesta seção do trabalho, pode ser visto que embora exista a preocupação acerca da carência de conceitos de gestão de projetos dos processos de desenvolvimento de software atuais, estes estudos não abordaram sobre como realizar planejamento de projetos considerando os elementos que compõem a gerência de projetos e os processos de desenvolvimento de software.

A seguir, serão apresentadas as principais necessidades observadas nesta pesquisa para o planejamento de projetos de software.

4 GESTÃO DE PROJETOS NOS PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

Este capítulo tem como principal objetivo dissertar sobre a carência no quesito de gerência de projetos dos processos de desenvolvimento de software e listar as principais necessidades observadas nesta pesquisa para o planejamento de projetos de software.

Conforme mencionado anteriormente, os processos de desenvolvimento do software existentes possuem carências no quesito de gerência de projetos em suas metodologias ou modelos. Tal fato é reforçado em [HEN00], onde é destacado que dois dos mais importantes processos de desenvolvimento de software respectivamente o RUP, por sua absorção no mercado, e o OPEN, por sua contribuição no meio acadêmico, necessitam de maior suporte no quesito de gerência de projetos. Tanto o RUP quanto o OPEN auxiliam na execução das melhores práticas para o desenvolvimento de software. No entanto, o RUP, por exemplo, não cobre assuntos essenciais de gerência de projetos, tais como a gerência de pessoas e a gerência de contratos. Em contraste, o OPEN apresenta um conjunto de atividades e técnicas que contemplam diferentes áreas da gerência de projetos, tais como qualidade, estimativas de custo e métricas de gerenciamento. Todavia, ambos os modelos mostraram-se deficientes em áreas essenciais de conhecimento da gerência de projetos, particularmente, a gerência de aquisição, de comunicação e de pessoas.

Analisando a literatura atual, não se identificou estudos que tratem em profundidade e apresentem soluções específicas para suprir a carência no quesito de gerência de projetos dos processos de desenvolvimento de software. Dessa forma, em seguida são listadas algumas das necessidades identificadas a partir da literatura examinada na seção anterior deste trabalho.

1) Acesso às informações pertencentes aos outros departamentos da organização

Ao realizar planejamento de um projeto de software, o gerente de projetos pode necessitar interagir com outros departamentos da organização a fim de obter informações relevantes para o projeto (contatar o setor de recursos humanos sobre a necessidade de contratação de pessoal, por exemplo). Além disso, estes outros setores da organização são responsáveis por atualizar as informações sobre custos e prazos destas atividades de apoio ao

projeto de software. Com o objetivo de obter estas informações, percebe-se que o fluxo de atividades de um projeto de software poderá interagir com os demais fluxos de atividades comuns da organização (**fluxos empresariais**). Conseqüentemente, o gerente de projetos precisa de uma solução que permita o acesso às informações dos fluxos empresariais durante a elaboração do planejamento de projetos de software.

2) **Identificação das relações de dependência entre as atividades dos fluxos de trabalho da empresa e do projeto de software**

As atividades pertencentes a um fluxo de trabalho de empresarial não são exclusivas de um projeto de software específico, mas de um fluxo comum da empresa que é compartilhado pelos projetos em andamento. Neste instante, percebe-se que há uma dissociação entre o fluxo de atividades de um projeto de software e os demais fluxos de atividades de suporte ao projeto da organização. Durante o planejamento de atividades do projeto, por exemplo, o gerente de projetos informa o setor de recursos humanos sobre a necessidade de contratação de um analista de testes. Neste caso, constata-se a existência de uma relação de dependência entre as atividades do projeto (tais como, a modelagem dos casos de teste) com as atividades pertencentes ao fluxo de trabalho do setor de recursos humanos referentes à contratação do profissional requerido para executar a atividade do projeto de software.

A dificuldade para identificar a interdependência dos fluxos de trabalho da empresa e do projeto de software durante o planejamento do projeto pode resultar, por exemplo, no aumento dos custos e em atrasos nos prazos do projeto. Assim, percebe-se a necessidade de identificação das relações de dependência entre as atividades pertencentes a estes dois tipos de fluxos de trabalho, permitindo a antecipação das necessidades advindas das áreas de apoio da organização durante o planejamento de projetos de software.

3) **Capacidade de evitar distorções no planejamento de projetos (tais como, o aumento dos custos e atrasos nos prazos do projeto)**

Conforme mencionado anteriormente, os fluxos de trabalho da empresa e do projeto de software são executados de forma distinta. Além disso, as atividades pertencentes aos fluxos de trabalho de empresariais utilizam recursos não alocados diretamente ao projeto de software. Estes recursos podem influenciar tanto nos prazos das atividades quanto nos custos do projeto de software. Ao realizar o planejamento de um projeto de software, por exemplo, o gerente de projetos identifica a necessidade de contratação de um analista de banco de dados.

Esta empresa possui uma política de admissão de profissionais que exige a realização de exames médicos antes da contratação. Dessa forma, o gerente de projetos faz uma previsão de quando poderá utilizar este novo recurso no projeto. Entretanto, caso o médico responsável pelo exame de admissão precise ficar ausente por alguns dias, este atraso poderá impactar negativamente no cronograma do projeto de software em questão.

Dessa forma, o gerente de projetos precisa de um suporte que o auxilie a evitar distorções no planejamento de projetos (tais como, o aumento dos custos e atrasos nos prazos do projeto) pela desconsideração de que as atividades de apoio pertencentes aos fluxos de trabalho de empresariais utilizam recursos que não são alocados diretamente ao projeto de software.

4) Auxílio na identificação e mensuração dos custos indiretos do projeto

O gerente de projetos precisa lidar tanto com assuntos gerenciais quanto técnicos durante o planejamento e a execução de atividades. Ele deve considerar, durante o planejamento de projetos de software, tanto as atividades que produzem um resultado significativo no contexto de um projeto de software quanto as atividades que pertencem exclusivamente aos demais fluxos de atividades de suporte ao projeto da organização. Dessa forma, a distinção explícita entre as atividades pertencentes aos fluxos empresariais e as de um projeto de software específico permite a identificação e a mensuração dos custos indiretos do projeto de software advindos das atividades de apoio da organização.

O modelo proposto considera a integração dos conceitos da gerência de projetos e do processo de desenvolvimento de software para o auxílio no processo de tomada de decisões dos gerentes de projetos durante o planejamento de projetos de software. Dessa forma, serão descritas a seguir as abordagens pesquisadas na literatura que contribuíram de alguma forma para a elaboração deste trabalho.

4.1 CONSIDERAÇÕES SOBRE O CAPÍTULO

Nesta seção pode-se observar que apesar da existência de artigos na literatura salientando a importância de se utilizar processos de software bem definidos nas organizações, parece não haver estudos suficientes para suprir a carência no quesito de gerência de projetos destes processos.

Este estudo aborda exclusivamente a realização do planejamento de projetos considerando os elementos que compõem a gerência de projetos e os processos de

desenvolvimento de software. Assim foram listados os principais pontos observados nesta pesquisa, identificados na literatura da área de gestão de projetos de software, relacionados ao planejamento de projetos.

A seguir, serão apresentados os metamodelos base de integração entre os conceitos advindos da gerência de projetos e dos processos de desenvolvimento de software.

5 METAMODELOS BASE DE INTEGRAÇÃO

Esse capítulo apresenta os modelos base de integração desenvolvidos nesta pesquisa, respectivamente o PMBOK+RUP e o PMBOK+OPEN. A integração entre o RUP e o PMBOK consiste no estágio inicial do estudo sobre a integração do processo de desenvolvimento de software e a gerência de projetos. Dessa forma, é apresentado um estudo detalhado deste metamodelo através da análise de suas classes e relacionamentos. Posteriormente é apresentada a integração entre o PMBOK e o OPEN, que teve como objetivo o refinamento do modelo SPIM.

O estudo detalhado dos modelos do PMBOK e RUP, reforçado posteriormente com a inclusão do OPEN para seu refinamento, permitiu identificar como são organizados e quais as relações válidas entre os elementos de cada modelo. Através da integração entre a gerência de projetos e os processos de desenvolvimento de software foi possível identificar as principais características e discrepâncias entre os elementos de tais modelos. Conforme citado anteriormente, o metamodelo de referência do PMBOK [CAL07] inclui os elementos necessários para a gerência de projetos de software, enquanto que os conceitos de processos de desenvolvimento de software são obtidos pelo RUP ou pelo OPEN.

O critério de integração entre os modelos PMBOK+RUP e PMBOK+OPEN seguiu um conjunto de regras identificado por [CAL07], que afirma que ao se realizar uma integração entre dois modelos, as seguintes situações podem ocorrer:

- Uma sobreposição de conceitos (duas classes com o mesmo conceito em cada modelo): neste caso, pode-se transformar e unir estas duas classes em um único conceito dentro de um pacote comum;
- Relação entre conceitos (uma classe de um dos modelos se relaciona com alguma outra classe de outro modelo, mas estas classes não representam exatamente o mesmo conceito): deve-se manter as classes em seus modelos originais e criar uma associação entre elas; e
- Conceitos independentes (classes com conceitos independentes e distintos): deve-se deixar cada classe em seu próprio pacote.

A proposta de integração entre a gerência de projetos e os processos de desenvolvimento de software apresentada neste trabalho é constituída de três pacotes: um para os conceitos de gerência de projetos, outro para os relacionados aos processos de desenvolvimento de software e, finalmente, um pacote comum ("*Common*") que une os conceitos que ocorrem em ambos os modelos. As seções a seguir apresentam uma discussão mais elaborada, que envolve o desenvolvimento dos dois metamodelos de integração desenvolvidos nesta pesquisa.

5.1 METAMODELO INTEGRADO ENTRE O PMBOK E O RUP

De modo a explicar o metamodelo de integração PMBOK+RUP (Figura 8), seus principais elementos serão descritos. Posteriormente, serão apresentadas as principais extensões realizadas para o desenvolvimento do SPIM ao metamodelo integrado entre o PMBOK e o RUP proposto em [CAL07].

É importante ressaltar que alguns conceitos relacionados à gerência de projetos que estão contidos nos processos de desenvolvimento de software (neste caso, no RUP e no OPEN) foram propositadamente deslocados para o pacote de classes gerenciais (pacote PMBOK) com o objetivo de deixar mais explícita a classificação dos conceitos de gerência de projetos e do processo de desenvolvimento de software. Também optou-se por manter os conceitos na língua inglesa para facilitar a comparação com os modelos originais.

Neste modelo, a classe *Organization* representa uma empresa que se organiza por programas (classe *Program*). Os programas são grupos de projetos (classe *Project*) designados a alcançar um objetivo estratégico. As organizações geralmente dividem os projetos em várias fases (classe *Phase*) visando um melhor controle gerencial.

Os recursos necessários para um projeto são explicitamente descritos no sub-pacote *Resources*. Sendo assim, pessoas, equipamentos e locais são representados pela classe *Resource*. Estes recursos são divididos em recursos ativos (classe *Stakeholder*) e não-ativos (classe *PhysicalResource*). Os s*takeholders* correspondem às pessoas e organizações cujos interesses são afetados pelo projeto [PMI00]. A classe associativa *ProjectStakeholder* foi adicionada ao modelo para informar o nível de interesse e influência de um *stakeholder* para um projeto.

O sub-pacote *Resources* também contém informações sobre a disponibilidade de cada recurso ao atribuí-lo às atividades, mesmo que realizado de forma manual ou automática, através da classe *ResourceAvailability*. Esta classe permite automatizar os processos de

alocação de recursos em projetos de software. Foi adicionado ao modelo a classe *AvailableTime*, que informa quando um recurso está disponível. Paralelamente, a definição da carga de trabalho (atributo *workload*) dos recursos físicos (ao associar-se a diferentes atividades) é representada pela classe *ActivityPhysicalResourceWork*.

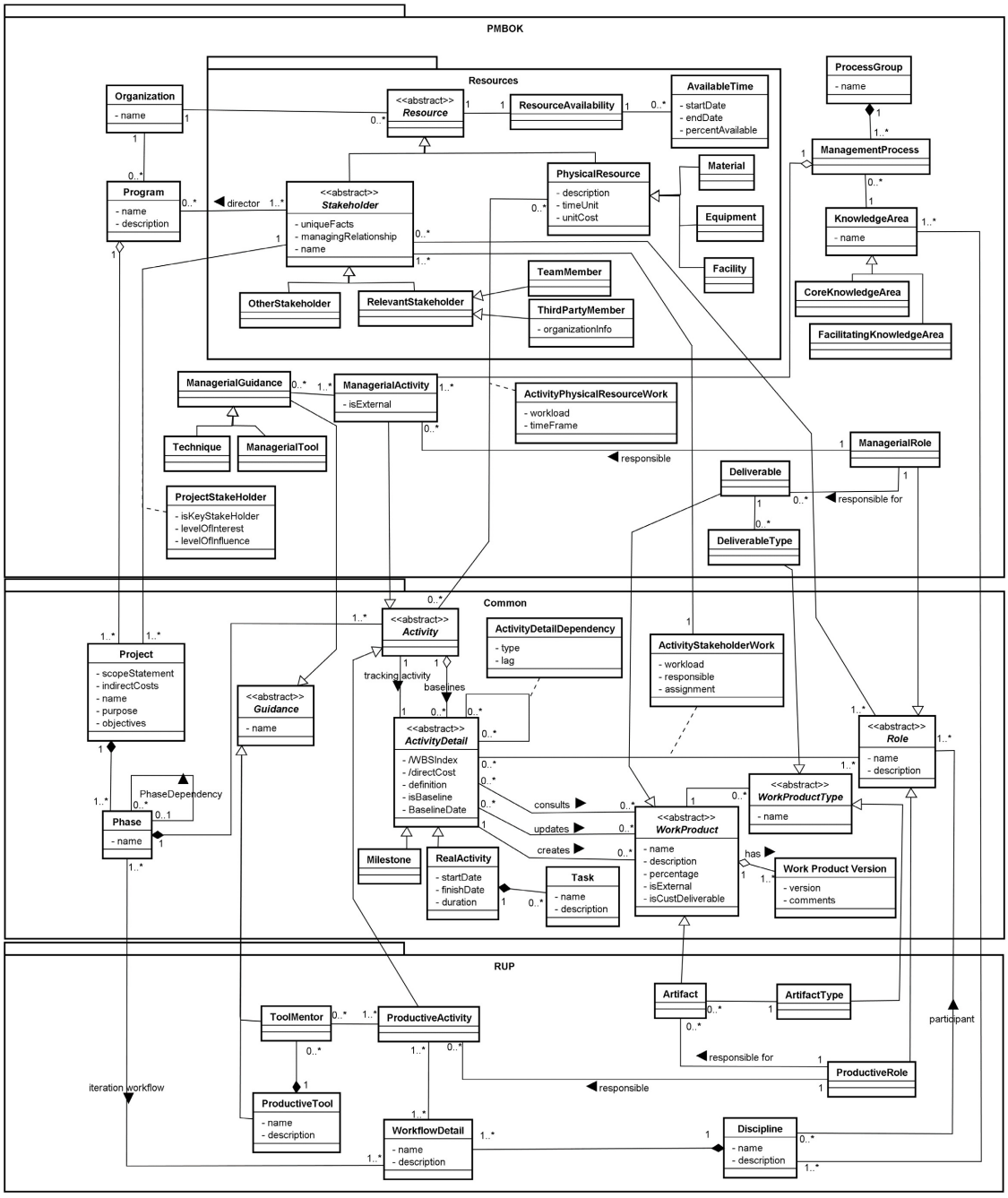


Figura 8. Metamodelo de integração PMBOK+RUP [ROS08]

Este metamodelo define que a classe *Activity* pode ser especializada como atividade produtiva (*ProductiveActivity*) ou atividade gerencial (*ManagerialActivity*), relacionadas ao pacote RUP e ao pacote PMBOK respectivamente. Uma atividade produtiva representa uma unidade de trabalho, desempenhada por um papel produtivo, que produz um resultado

significativo no contexto de um projeto de software (por exemplo, a modelagem do banco de dados). As atividades gerenciais, entretanto, podem pertencer tanto ao fluxo de desenvolvimento de software quanto aos fluxos empresariais da organização. Esta distinção é possível através do atributo denominado *isExternal* da classe *ManagerialActivity,* de maneira que *isExternal=false* define uma atividade gerencial como sendo pertencente ao projeto de software enquanto que *isExternal=true* refere-se a uma atividade gerencial de apoio da organização. Seguindo esta nomenclatura, a atividade de organizar e conduzir uma reunião de acompanhamento do projeto é um exemplo de uma atividade gerencial que pertence exclusivamente ao projeto de desenvolvimento de software. Em contrapartida, a atividade de contratar um administrador de banco de dados é um exemplo de uma atividade gerencial que pertence exclusivamente aos demais fluxos de atividades de apoio ao projeto da organização (neste caso, esta atividade é realizada pelo setor de recursos humanos).

Adicionalmente, cada atividade pode pertencer a um ou mais *baselines*. Em cada geração da *baseline*, uma atividade deve manter os relacionamentos com os papéis e produtos de trabalho. Assim, objetivando-se manter estes relacionamentos das atividades com outras entidades do modelo durante a geração de *baselines*, decidiu-se assumir que a classe *Activity* conterá uma agregação de uma ou mais instâncias da classe *ActivityDetail*. Dessa forma, foi adicionado ao modelo a classe *ActivityDetail* objetivando-se manter estes relacionamentos das atividades com outras entidades do modelo durante a geração de *baselines*. Assim, decidiu-se assumir que a classe Activity conterá uma agregação de uma ou mais instâncias da classe *ActivityDetail*. Também, uma atividade pode possuir um tempo de duração definido e ser atribuída a um papel (classe *RealActivity*) ou pode não ter um tempo de duração (classe *Milestone*).

Os *stakeholders* podem desempenhar diversos papéis (classe *Role*) durante a execução das atividades do projeto. Assim, para cada associação entre um papel e uma atividade (representado pela classe associativa *ActivityStakeholderWork*) deve haver também uma associação dessa atividade com um *stakeholder* capaz de desempenhar aquele papel. Além disso, como o conceito de papéis (classe *Role*) aparece em ambos os modelos, estes foram divididos em papéis gerenciais (classe *ManagerialRole*) e papéis produtivos (classe *ProductiveRole*), tal como ocorreu com as atividades.

Um produto de trabalho, por sua vez, (classe *WorkProduct*) pode ser classificado como um produto gerencial (classe *Deliverable*) ou produtivo (classe *Artifact*), o qual deve estar associado a um tipo de produto (classes *DeliverableType* e *ArtifactType*,

respectivamente). Cabe salientar que o modelo faz distinção das possíveis relações entre uma atividade e um artefato (criar/atualizar/consultar). Isto é importante para apoiar a consistência do modelo a partir de regras, como a de número 16 na Tabela 1 apresentada mais adiante no Capítulo 6.

Em relação à gerência de projetos, o PMBOK Guide representa suas práticas em duas dimensões lógicas. Uma dimensão descreve as áreas de conhecimento (classe *KnowledgeArea*) enquanto que a outra dimensão descreve os processos gerenciais de um projeto (classe *ManagementProcess*), os quais estão contidos em cinco grupos de processo (classe *ProcessGroup*). As áreas de conhecimento são responsáveis por descrever as principais competências que os gerentes de projetos devem desenvolver e derivam as áreas centrais (classe *CoreKnowledgeArea*) e as de apoio (classe *FacilitatingKnowledgeArea*). Assim, cada atividade gerencial pertence a um processo gerencial, sendo também relacionada a uma área de conhecimento.

Em contrapartida, o RUP define uma disciplina (classe *Discipline*) como sendo a divisão de elementos de processo em áreas de interesse. Cada disciplina é composta por um ou mais fluxos de trabalho (classe *WorkflowDetail*). Os fluxos de trabalho definem como os papéis produtivos devem colaborar entre si através de suas atividades.

5.1.1 PRINCIPAIS EXTENSÕES AO METAMODELO PMBOK+RUP

Nesta subseção serão apresentadas as principais extensões realizadas ao metamodelo integrado entre o PMBOK e o RUP proposto em [CAL07]. O conjunto adicional de classes e relacionamentos propostos ao metamodelo PMBOK+RUP baseou-se nos estudos realizados em [PMI00], [SCH02] [BEN06], [JAC99], [RAT06] e [PEP06].

De acordo com [SCH02], um projeto deve ter apenas um recurso responsável por direcionar e fundamentar o projeto. A importância da correta identificação dos principais *stakeholders* deve-se ao fato de que o sucesso do projeto depende, entre outros fatores, da capacidade de atender as necessidades e expectativas dos *stakeholders*. Dessa forma, foi adicionado ao metamodelo a classe associativa *ProjectStakeholder* que informa, explicitamente, qual é o *stakeholder* responsável por um determinado projeto. Também, define o nível de interesse e o nível de influência deste *stakeholder* no projeto.

A classe *AvailableTime* foi adicionada ao metamodelo PMBOK+RUP com o objetivo de informar o tempo que um recurso (tais como pessoas ou salas) tem disponível para ser utilizado nos projetos da organização. Esta classe contém atributos que informam a

data inicial, data final e o percentual de alocação de um recurso a uma determinada atividade. É importante ressaltar que esta disponibilidade é independente do projeto, mas não é independente da organização em que atua.

De acordo com este modelo, a classe *Activity* pode ser especializada como atividade produtiva ou atividade gerencial. A classe *ManagerialActivity* descreve as atividades desempenhadas por papéis gerenciais em um projeto. Entretanto, observou-se que estas atividades gerenciais podem pertencer tanto ao fluxo de desenvolvimento de software quanto aos fluxos empresariais da organização. Dessa forma, foi adicionado à classe *ManagerialActivity* o atributo *isExternal* objetivando-se permitir a distinção entre estes dois tipos de atividades gerenciais que ocorrem em projetos de software. Conseqüentemente, neste modelo identifica-se três tipos de atividades: **produtivas**, **gerenciais** e **gerenciais de apoio**.

Outra colaboração ao modelo está relacionada à geração de *baselines*. Uma atividade pode pertencer a um ou mais *baselines* durante o desenvolvimento do projeto. Na geração de cada *baseline* é necessário manter os relacionamentos das atividades com os papéis e produtos de trabalho relacionados. Assim, a classe *ActivityDetail* (pacote *Common*) foi definida como responsável por manter estes relacionamentos, enquanto que a classe *Activity* foi definida como sendo uma agregação de uma ou mais classes *ActivityDetail*. Dessa forma, enquanto que o relacionamento denominado *Baselines* permite que uma atividade pertença a uma ou mais baselines, o relacionamento *TrackingActivity* diferencia a atividade atual daquelas pertencentes às *baselines*. Também foi adicionado ao pacote *Common* a classe *RealActivity* (representa uma atividade que tem tempo de execução) e a classe *Milestone* (é uma atividade que não tem tempo de duração).

Conforme mencionado anteriormente, os conceitos de gerência de projetos que existem tanto no RUP quanto no OPEN foram atribuídos ao pacote gerencial (neste caso, o PMBOK) a fim permitir uma distinção explícita entre os conceitos produtivos e gerenciais. Por exemplo, no RUP um papel interage com ambas as tarefas gerenciais e produtivas, mas no metamodelo integrado foram definidas, anteriormente, as classes especializadas *ProductiveRole* (pacote RUP) e *ManagerialRole* (pacote PMBOK). Assim, a classe *Role* pertence ao pacote comum. Considerando que uma atividade pode ser derivada em atividade gerencial (classe *ManagerialActivity*) ou produtiva (classe *ProductiveActivity*), foi necessário duplicar o relacionamento que existia entre as classes *Role* e *Acticity* (denominado *responsible*) de maneira a respeitar a divisão entre gerência de projetos e desenvolvimento de software que está sendo proposta nesta pesquisa. Além disso, por este mesmo motivo foi

necessário duplicar o relacionamento que existia entre as classes *Role* e *Workproduct* (denominado *responsible for*). Dessa forma, foi explicitamente adicionado ao metamodelo o conceito que apenas um papel gerencial é responsável por um produto de trabalho gerencial (classe *Deliverable*) e que apenas um papel produtivo é responsável por um produto de trabalho produtivo (*Artifact*).

Também, foram incluídos novos relacionamentos entre as classes que pertencem ao pacote de PMBOK e pacote *Common*, tais como o auto-relacionamento da classe *Phase* (nomeado *PhaseDependency*) que permite o seqüenciamento das fases em um projeto e o relacionamento entre as classes *Stakeholder* e *Project* que permite um *stakeholder* de participar de um ou mais projetos.

5.2 METAMODELO INTEGRADO ENTRE O PMBOK E O OPEN

O metamodelo de integração PMBOK+OPEN (Figura 9) foi inteiramente desenvolvido nesta pesquisa e apresenta uma estrutura similar ao apresentado na seção anterior, substituindo apenas o pacote referente ao processo de desenvolvimento de software (neste caso o OPEN). Assim, as classes dos pacotes PMBOK e *Common* são as mesmas que as apresentadas no metamodelo de integração PMBOK+RUP. Também permanecem inalterados os relacionamentos entre as classes pertencentes a estes dois pacotes.

Os dois processos de desenvolvimento de software, porém, possuem características particulares que são refletidas em classes distintas e em diferentes relacionamentos com os pacotes PMBOK e Common. Porém, a integração com o OPEN permitiu tanto a confirmação quanto a adequação dos conceitos propostos no modelo final. A seguir, o conjunto de classes do pacote OPEN será apresentado. Cabe salientar que a Figura 7 apresenta apenas os componentes centrais (classes *Producer*, *WorkUnit*, *WorkProduct*, *Stage* e *Language*) do framework do OPEN. Estes componentes representam classes abstratas e derivam num conjunto maior de sub-classes (algumas destas sub-classes são ilustradas no pacote OPEN).

A análise das classes e relacionamentos do metamodelo de integração PMBOK+OPEN baseou-se nos estudos realizados em [PMI00], [SCH02], [OPE02], [OPF07] e [GRA97] e tiveram como objetivo evitar possíveis inconsistências no metamodelo.

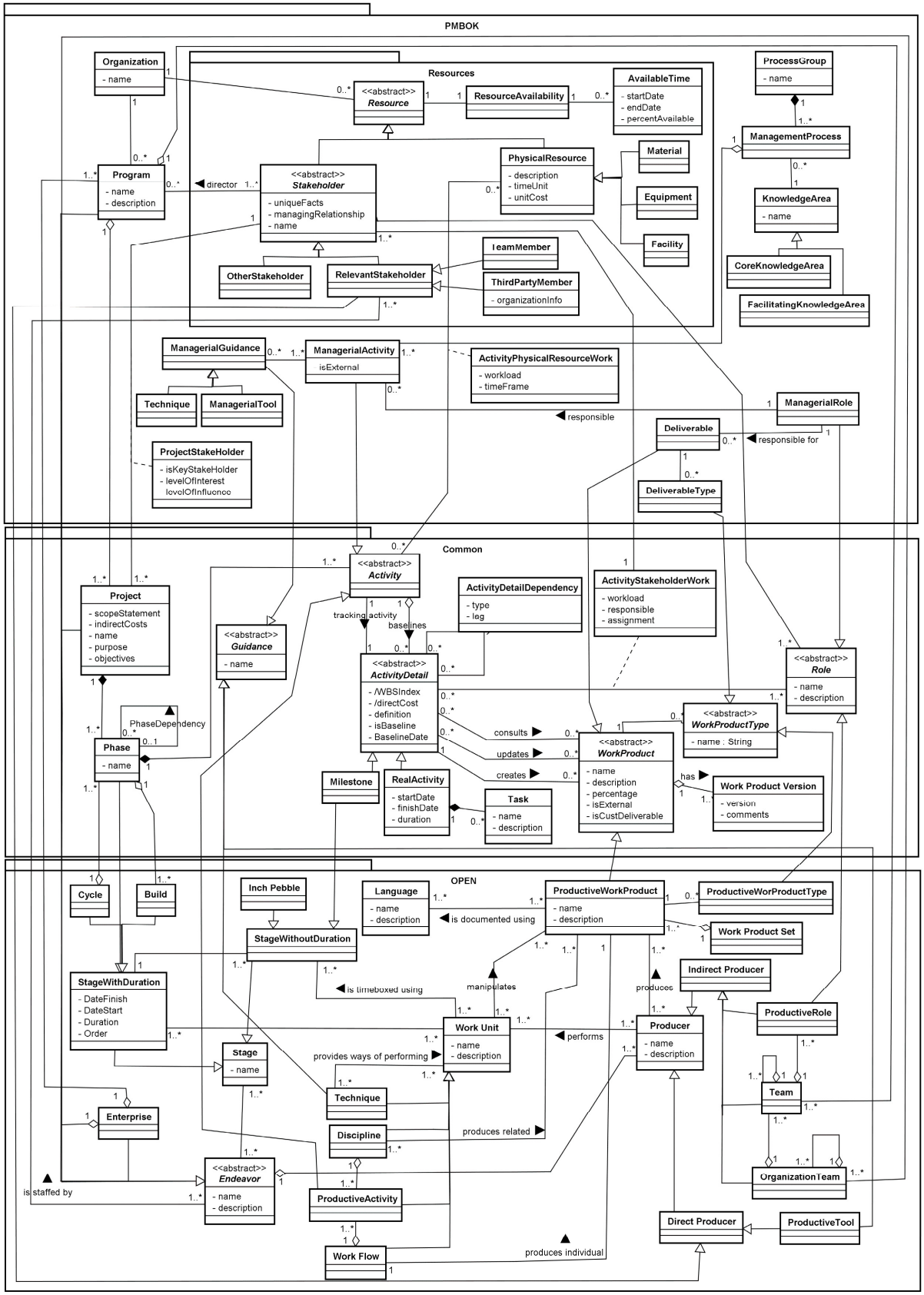


Figura 9. Metamodelo de integração PMBOK+OPEN

Neste modelo, a classe *Endeavor* representa o esforço empreendido pelos produtores durante o desenvolvimento do projeto. Esta classe possui como responsabilidade desenvolver

e/ou manter um ou mais produtos e serviços relacionados ao esforço empreendido. O OPEN define que a classe *Enterprise* representa o nível mais elevado de esforço, consistindo em uma coleção de programas relacionados que são controlados como uma única unidade. Possui as seguintes subclasses: *Enterprise, Program e Project.*

A classe *ProductiveWorkProduct* (originalmente denominada *WorkProduct* no metamodelo OPF) representa um produto do trabalho que é produzido, consumido ou modificado durante a execução de atividades produtivas por papéis produtivos. O conjunto de produtos produzidos pelas tarefas de uma ou mais atividades é representado pela classe *Work Product Set*, enquanto que a classe *Work Product Version* corresponde a uma versão específica do produto obtido através do processo de desenvolvimento incremental e iterativo.

A classe *Producer* é responsável por produzir ou modificar, diretamente ou indiretamente, os produtos de trabalho, além de realizar um ou mais serviços. É subdividida em produtores diretos (pessoas e ferramentas) e produtores indiretos (organização, equipe e papel).

Segundo [OPF07], as unidades de trabalho (classe *WorkUnit*) modelam as operações coesas que são executadas pelos produtores durante o processo de entrega do projeto. Estas são classificadas como tarefas, técnicas, fluxos de trabalho e atividades.

A classe *Stage* representa os intervalos de tempo que fornecem uma organização macro às unidades de trabalho, sendo subdividida em estágios com duração (ciclos, fases, fluxos, projeto, desenvolvimento, versões de entregas e entregas) e *milestones*.

5.3 CONSIDERAÇÕES SOBRE O CAPÍTULO

Como apresentado nas seções anteriores, o estudo detalhado dos modelos do PMBOK, RUP e OPEN permitiu identificar como são organizados e quais as relações válidas entre os elementos de cada modelo, permitindo identificar as principais características e discrepâncias entre os elementos de tais modelos.

Dessa forma, foi possível identificar que os metamodelo de integração PMBOK+RUP e PMBOK+OPEN apresentam uma estrutura de classes similar, substituindo apenas o pacote referente ao processo de desenvolvimento de software. Assim, tanto as classes dos pacotes PMBOK e *Common* como os relacionamentos entre as classes pertencentes a estes dois pacotes permanecem inalterados. Porém, como os dois processos de desenvolvimento de software possuem características particulares, estas diferenças são

refletidas em classes distintas e em diferentes relacionamentos com os pacotes PMBOK e *Common*.

Apesar do modelo SPIM implementar exclusivamente os conceitos advindos da integração do PMBOK com o RUP, foi realizado também uma integração do PMBOK com o OPEN objetivando-se mostrar que a estrutura constituída de três pacotes (um para os conceitos de gerência de projetos, outro para os relacionados aos processos de desenvolvimento de software e um pacote comum que une os conceitos que ocorrem em ambos os modelos) funciona com diferentes processos de desenvolvimento de software. Dessa forma, a integração do PMBOK com o RUP e o OPEN permitiu tanto a confirmação quanto a adequação dos conceitos propostos no modelo final.

No próximo capítulo é apresentado o modelo de integração entre a gerência de projetos e o processo de desenvolvimento de software desenvolvido neste trabalho.

6 SPIM - SOFTWARE PLANNING INTEGRATED MODEL

Neste capítulo é apresentado o modelo de integração entre a gerência de projetos e o processo de desenvolvimento de software proposto nesta pesquisa. Também, esse capítulo fornece uma visão geral sobre o planejamento integrado de atividades pertencentes ao projeto de software e aos fluxos de trabalho da organização. Ainda, são apresentadas as restrições definidas para apoiar a consistência do modelo proposto.

6.1 MODELO DE INTEGRAÇÃO ENTRE A GERÊNCIA DE PROJETOS E O PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

O estudo da integração com os dois processos distintos de desenvolvimento de software (RUP e OPEN) permitiu elaborar uma visão mais ampla de como a gerência de projetos pode positivamente contribuir no desenvolvimento de um produto de software. Assim, os metamodelos definidos durante esta pesquisa (PMBOK+RUP e PMBOK+OPEN) fornecem a estrutura conceitual necessária para o desenvolvimento de um modelo que auxilie o planejamento de projetos considerando os elementos que compõem o processo de desenvolvimento de software. A atual versão do modelo de integração entre a gerência de projetos e o processo de desenvolvimento de software, denominado **Software Planning Integrated Model (SPIM)**, implementa exclusivamente os conceitos advindos da integração do PMBOK com o RUP. Uma implementação que aborda também o OPEN está prevista como um trabalho futuro desta pesquisa.

Conforme mencionado anteriormente, como o gerente de projetos pode não possuir todas as informações relevantes para o projeto, poderão ocorrer interações com outros departamentos da organização durante o planejamento de um projeto de software. Assim, o fluxo de atividades de um projeto de software pode interagir com os demais fluxos de atividades da organização (**fluxos empresariais**). Conforme ilustrado na Figura 10, ambos os tipos fluxos de trabalho são executados em paralelo e podem influenciar nos prazos das atividades e custos do projeto de software.

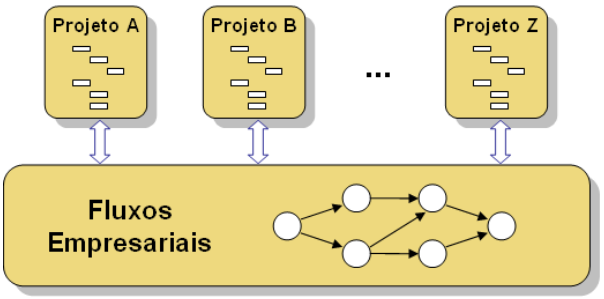


Figura 10. Relação entre projetos e fluxos empresariais [ROS08]

A preparação técnica e a liberação de uma sala ou equipamento testes são outros exemplos cujas atividades não são exclusivas de um projeto em especial, mas de um fluxo comum da empresa, compartilhado pelos projetos em andamento e que utiliza recursos não alocados diretamente ao projeto de software. Dessa forma, pode haver também uma relação de dependência entre as atividades pertencentes a estes dois tipos de fluxos de trabalho. A dificuldade para identificar esta interdependência dos fluxos de trabalho durante o planejamento de atividades pode afetar negativamente o projeto, resultando, por exemplo, no aumento dos custos e em atrasos nos prazos do projeto.

O estudo dos modelos do PMBOK e RUP permitiu identificar características relevantes da gerência de projetos e dos processos de desenvolvimento de software. Este estudo resultou no desenvolvimento de um modelo de integração PMBOK+RUP. O modelo SPIM foi desenvolvido implementando os conceitos advindos desta integração. Também identificou-se a dissociação entre o fluxo de atividades produtivas de um projeto e os demais fluxos empresariais da organização. Assim, o modelo SPIM tem como objetivo apoiar o processo decisório do gerente de projetos permitindo o planejamento integrado de atividades gerenciais e produtivas em projetos de software (Figura 11).

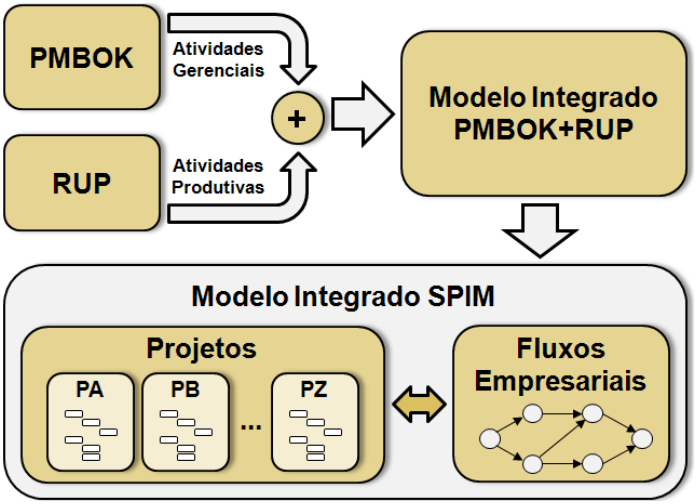


Figura 11. Etapas do desenvolvimento do modelo de integração SPIM

O modelo SPIM foi concebido considerando a necessidade do gerente de projetos obterem acesso às informações pertencentes aos outros departamentos da organização durante o planejamento de projetos de software. Para suportar esta funcionalidade, o modelo proposto estabelece a distinção explícita entre os tipos de atividades relacionados aos projetos de software. As atividades diretamente relacionadas com a construção do produto de software são chamadas de **atividades produtivas**. As atividades que são necessárias apenas para coordenar a construção do produto de software são definidas como **atividades de gerenciais**. Finalmente, todas as outras atividades que não pertençam ao fluxo de atividades de um projeto particular (as quais podem ser compartilhadas com outros projetos) são chamadas de **atividades gerenciais de apoio** (estas fazem parte do componente denominado **fluxo empresarial** do modelo - Figura 10). Estas últimas são atividades gerenciais que pertencem exclusivamente aos demais fluxos de atividades de suporte ao projeto da organização. Seguindo esta definição, é possível distinguir em um cronograma quais atividades devem ter suas informações atualizadas pelos outros setores da organização (utilizando um mecanismo tal como um *workflow*) e quais devem ser atualizadas diretamente pelo gerente de projetos.

Além disso, através da distinção proposta entre os tipos de atividades é possível identificar as relações de dependência potenciais entre as atividades pertencentes aos fluxos de trabalho da organização e ao fluxo de trabalho de um projeto de software específico. Como exemplo, a atividade de implantar um sistema de banco de dados (a qual se enquadra no fluxo de atividades do projeto de software) pode depender da aquisição de um servidor pelo departamento responsável por esta ação na organização (esta atividade se enquadra no fluxo de trabalho compartilhado da empresa). Logo, o modelo SPIM fornece um tipo de suporte, baseado em uma metodologia de gerência de projetos, que permite lidar com estes tipos de dependências (aqui, implantar o sistema de banco de dados é uma atividade produtiva, enquanto que a aquisição deste servidor é uma atividade de gerencial de apoio).

Conforme mencionado anteriormente, as atividades gerenciais de apoio fazem parte dos fluxos empresariais da organização. Cada fluxo de trabalho de empresarial é um conjunto de atividades de instanciadas que podem ser consumidas por um ou mais projetos. Partindo dessa premissa, o SPIM permite registrar cada instância de um fluxo de trabalho de empresarial como uma atividade gerencial de apoio em projetos de desenvolvimento de software. Assim, o modelo proposto procura ajudar na identificação das dependências entre as atividades em ambos os fluxo de trabalho (Figura 12). Isto permite a integração com uma ferramenta de workflow, de maneira que este último seja responsável por atualizar os prazos

das atividades gerenciais de apoio ao projeto. Um fluxo empresarial pode atender múltiplos projetos simultaneamente, daí a necessidade de um mecanismo tal como um workflow. O exemplo apresentado na Figura 12 é representado na notação de redes PERT [BUR01]: setas representam atividades e círculos representam eventos entre atividades.

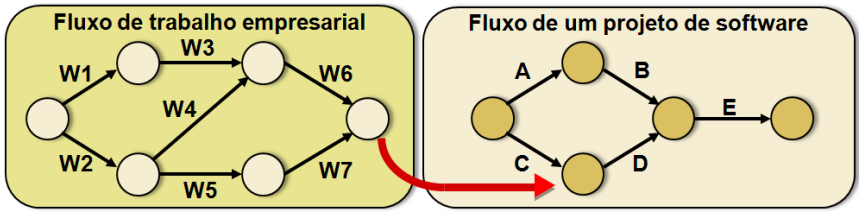


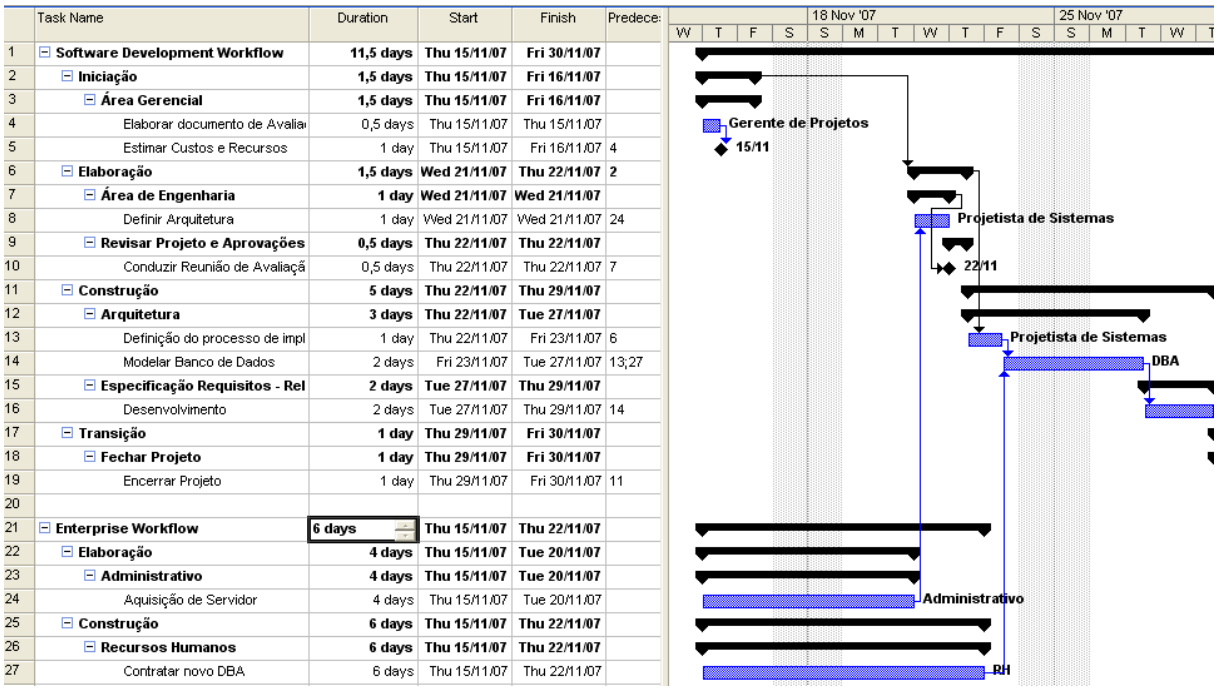
Figura 12. Interdependência entre os fluxos de trabalho empresarial e de um projeto de software específico.

Neste exemplo, a atividade D em um projeto de software depende tanto da atividade C (que pode ser produtiva ou gerencial) quanto do término de um dos fluxos de trabalho empresariais. A seta curvada é uma atividade gerencial de apoio que indica que devem ser finalizadas todas as atividades naquele específico fluxo de trabalho empresarial (W1, W2, W3 e assim por diante) antes de ser possível passar para atividade D (assumindo que a atividade C também terminou). Então, uma atividade gerencial de apoio é de fato um tipo de atividade virtual que representa uma execução inteira de um dos fluxos empresariais (atributo *isExternal* da classe *ManagerialActivity* - Figura 8). Ilustrando este exemplo com atividades pertencentes a um projeto de software, pode-se considerar que a atividade C refere-se à definição da arquitetura de hardware do projeto e a atividade D refere-se à implantação do sistema de banco de dados. Ainda, a atividade gerencial de apoio (relacionada ao conjunto de atividades W) refere-se à aquisição de um servidor de banco de dados pelo setor responsável por esta ação na organização.

O modelo SPIM foi concebido considerando a complexidade de identificar as relações de dependência entre as atividades de existentes dos fluxos de trabalho empresariais e o fluxo de trabalho de um projeto de software particular. Dessa forma, a empresa pode definir um conjunto reutilizável de fluxos de trabalho empresariais, tais como "aquisição de hardware", "contratação de novas pessoas", "configuração do ambiente de trabalho", e assim por diante. Cada referência a um fluxo de trabalho empresarial representa novas instâncias das atividades "W" correspondentes. Cabe lembrar que os outros departamentos da organização são responsáveis por atualizar os prazos das atividades gerenciais de apoio. Assim, um dos benefícios do SPIM é a antecipação das necessidades que surgem de diferentes fontes da organização (por exemplo, entre os projetos de software e um departamento de apoio, tal

como o setor de recursos humanos), através do envio de alertas que permite o re-planejando das atividades afetadas por esta relação de dependência.

A Figura 13 apresenta um cronograma utilizado para explicar como o modelo SPIM pode auxiliar o gerente de projetos na antecipação das necessidades dos projetos de software advindas das integrações com os diferentes setores da organização. Enquanto que as atividades (produtivas e gerenciais) do projeto de software pertencem ao agrupamento **Software Development Workflow**, as atividades dos fluxos empresariais (denominadas atividades gerenciais de apoio) pertencem ao agrupamento **Enterprise Workflow**. Neste exemplo, o início da atividade produtiva "Definir Arquitetura" depende do término da atividade gerencial de apoio "Aquisição de Servidor". Este mesmo tipo de dependência acontece entre a atividade produtiva "Modelar Banco de Dados" e a atividade gerencial de apoio "Contratar novo DBA".



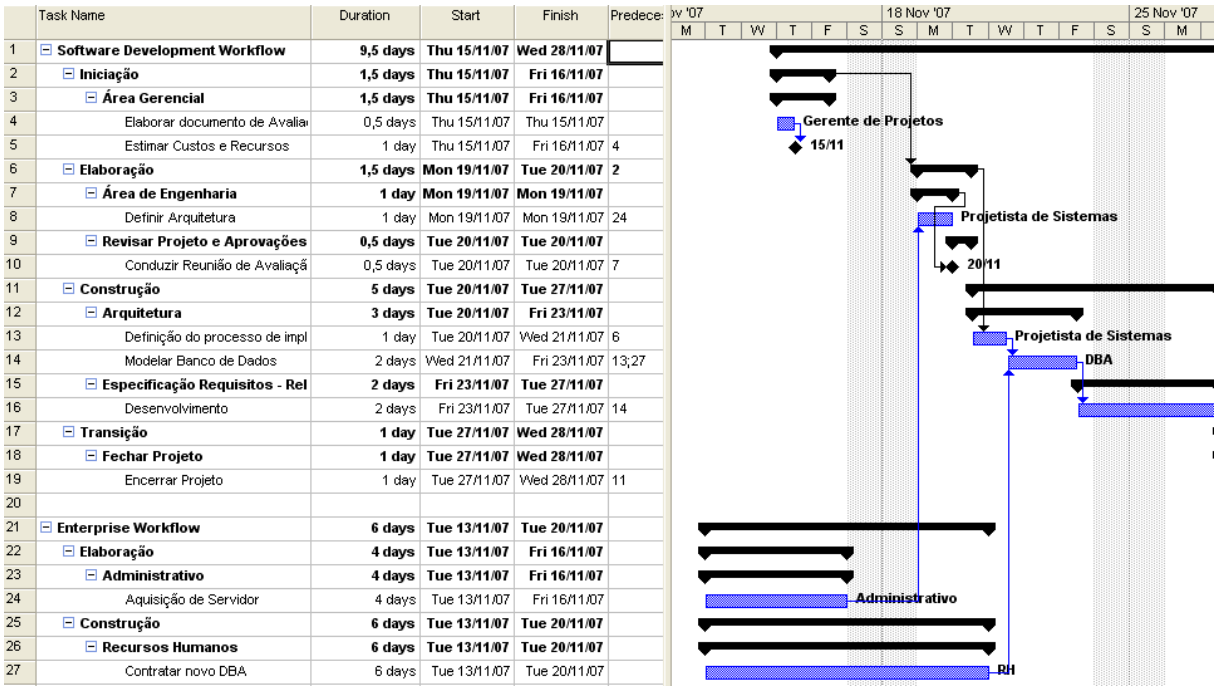
	Task Name	Duration	Start	Finish	Predece:
1	**Software Development Workflow**	**11,5 days**	**Thu 15/11/07**	**Fri 30/11/07**	
2	**Iniciação**	**1,5 days**	**Thu 15/11/07**	**Fri 16/11/07**	
3	**Área Gerencial**	**1,5 days**	**Thu 15/11/07**	**Fri 16/11/07**	
4	Elaborar documento de Avalia	0,5 days	Thu 15/11/07	Thu 15/11/07	
5	Estimar Custos e Recursos	1 day	Thu 15/11/07	Fri 16/11/07	4
6	**Elaboração**	**1,5 days**	**Wed 21/11/07**	**Thu 22/11/07**	**2**
7	**Área de Engenharia**	**1 day**	**Wed 21/11/07**	**Wed 21/11/07**	
8	Definir Arquitetura	1 day	Wed 21/11/07	Wed 21/11/07	24
9	**Revisar Projeto e Aprovações**	**0,5 days**	**Thu 22/11/07**	**Thu 22/11/07**	
10	Conduzir Reunião de Avaliaçã	0,5 days	Thu 22/11/07	Thu 22/11/07	7
11	**Construção**	**5 days**	**Thu 22/11/07**	**Thu 29/11/07**	
12	**Arquitetura**	**3 days**	**Thu 22/11/07**	**Tue 27/11/07**	
13	Definição do processo de impl	1 day	Thu 22/11/07	Fri 23/11/07	6
14	Modelar Banco de Dados	2 days	Fri 23/11/07	Tue 27/11/07	13;27
15	**Especificação Requisitos - Rel**	**2 days**	**Tue 27/11/07**	**Thu 29/11/07**	
16	Desenvolvimento	2 days	Tue 27/11/07	Thu 29/11/07	14
17	**Transição**	**1 day**	**Thu 29/11/07**	**Fri 30/11/07**	
18	**Fechar Projeto**	**1 day**	**Thu 29/11/07**	**Fri 30/11/07**	
19	Encerrar Projeto	1 day	Thu 29/11/07	Fri 30/11/07	11
20					
21	**Enterprise Workflow**	**6 days**	**Thu 15/11/07**	**Thu 22/11/07**	
22	**Elaboração**	**4 days**	**Thu 15/11/07**	**Tue 20/11/07**	
23	**Administrativo**	**4 days**	**Thu 15/11/07**	**Tue 20/11/07**	
24	Aquisição de Servidor	4 days	Thu 15/11/07	Tue 20/11/07	
25	**Construção**	**6 days**	**Thu 15/11/07**	**Thu 22/11/07**	
26	**Recursos Humanos**	**6 days**	**Thu 15/11/07**	**Thu 22/11/07**	
27	Contratar novo DBA	6 days	Thu 15/11/07	Thu 22/11/07	

Figura 13. Cronograma sem o re-planejando das atividades pelas regras do modelo SPIM

Ao realizar o planejamento deste projeto, o gerente de projetos identificou a necessidade de aquisição de um servidor e a contratação de pessoal (ambas são atividades pertencem ao fluxo empresarial da empresa). Além disso, o gerente de projetos definiu que ambas estas atividades gerenciais de apoio seriam iniciadas na data definida para o início do projeto, ou seja, dia 17/11/2007. Esta definição, porém, ocasiona o atraso no início de algumas atividades do projeto de software. De acordo com o fluxo de atividades deste projeto de software, por exemplo, a atividade produtiva "Definir Arquitetura" depende do término das atividades da fase de "Iniciação", que ocorre no dia 16/11/2007 (sexta-feira). Logo, esta

atividade poderia iniciar no dia 19/11/2007 (segunda-feira). Entretanto, devido à relação de dependência desta atividade do projeto com o fluxo empresarial responsável pela aquisição de uma máquina servidora (que tem seu término previsto para a terça-feira dia 20/11/2007), ela tem seu início postergado para o dia 21/11/2007 (quarta-feira). Dessa forma, pode-se observar que a dificuldade de identificar a distinção entre estes dois tipos de fluxos de trabalho está resultando em distorções no planejamento deste projeto (ambas as atividades gerenciais de apoio estão afetando negativamente no prazo das atividades do projeto de software).

Entretanto, o modelo SPIM possui um conjunto de regras (apresentadas na próxima subseção) que auxilia o gerente de projetos na antecipação das atividades gerenciais de apoio e, conseqüentemente, no re-planejando do projeto de software devido à existência de relações de dependência destas atividades com as atividades do projeto. A Figura 14 apresenta o cronograma utilizado no exemplo anterior após o re-planejando das atividades afetadas por estas relações de dependência. Dessa forma, as atividades gerenciais de apoio "Aquisição de Servidor" e "Contratar novo DBA" foram antecipadas para o dia 13/11/2007 (quinta-feira). Conseqüentemente, as atividades pertencentes ao fluxo deste projeto de software também tiveram seus prazos modificados, por exemplo, a atividade "Definir Arquitetura" teve sua data de início antecipada para o dia 19/11/2007 (segunda-feira).



	Task Name	Duration	Start	Finish	Predece
1	**Software Development Workflow**	**9,5 days**	**Thu 15/11/07**	**Wed 28/11/07**	
2	**Iniciação**	**1,5 days**	**Thu 15/11/07**	**Fri 16/11/07**	
3	**Área Gerencial**	**1,5 days**	**Thu 15/11/07**	**Fri 16/11/07**	
4	Elaborar documento de Avalia	0,5 days	Thu 15/11/07	Thu 15/11/07	
5	Estimar Custos e Recursos	1 day	Thu 15/11/07	Fri 16/11/07	4
6	**Elaboração**	**1,5 days**	**Mon 19/11/07**	**Tue 20/11/07**	**2**
7	**Área de Engenharia**	**1 day**	**Mon 19/11/07**	**Mon 19/11/07**	
8	Definir Arquitetura	1 day	Mon 19/11/07	Mon 19/11/07	24
9	**Revisar Projeto e Aprovações**	**0,5 days**	**Tue 20/11/07**	**Tue 20/11/07**	
10	Conduzir Reunião de Avaliaçã	0,5 days	Tue 20/11/07	Tue 20/11/07	7
11	**Construção**	**5 days**	**Tue 20/11/07**	**Tue 27/11/07**	
12	**Arquitetura**	**3 days**	**Tue 20/11/07**	**Fri 23/11/07**	
13	Definição do processo de impl	1 day	Tue 20/11/07	Wed 21/11/07	6
14	Modelar Banco de Dados	2 days	Wed 21/11/07	Fri 23/11/07	13;27
15	**Especificação Requisitos - Rel**	**2 days**	**Fri 23/11/07**	**Tue 27/11/07**	
16	Desenvolvimento	2 days	Fri 23/11/07	Tue 27/11/07	14
17	**Transição**	**1 day**	**Tue 27/11/07**	**Wed 28/11/07**	
18	**Fechar Projeto**	**1 day**	**Tue 27/11/07**	**Wed 28/11/07**	
19	Encerrar Projeto	1 day	Tue 27/11/07	Wed 28/11/07	11
20					
21	**Enterprise Workflow**	**6 days**	**Tue 13/11/07**	**Tue 20/11/07**	
22	**Elaboração**	**4 days**	**Tue 13/11/07**	**Fri 16/11/07**	
23	**Administrativo**	**4 days**	**Tue 13/11/07**	**Fri 16/11/07**	
24	Aquisição de Servidor	4 days	Tue 13/11/07	Fri 16/11/07	
25	**Construção**	**6 days**	**Tue 13/11/07**	**Tue 20/11/07**	
26	**Recursos Humanos**	**6 days**	**Tue 13/11/07**	**Tue 20/11/07**	
27	Contratar novo DBA	6 days	Tue 13/11/07	Tue 20/11/07	

Figura 14. Cronograma após o re-planejando das atividades pelas regras do modelo SPIM

Além disso, a distinção explícita entre as atividades pertencentes aos fluxos empresariais e de um projeto de software específico permite a identificação e a mensuração

dos custos indiretos do projeto de software (advindos das atividades de apoio da organização) pelo gerente de projetos.

Percebe-se, dessa forma, que alguns tipos de atividades gerenciais são inerentes ao processo e não aparecem no momento do planejamento do projeto. São justamente estas atividades (ou suas dependências) que na maioria das vezes causam um atraso considerável no cronograma e não são consideradas na definição dos riscos do projeto. Conseqüentemente, o modelo SPIM pretende ajudar os gerentes a resolver problemas relacionados à definição inadequada e a inter-relação das atividades em um projeto de software. A análise de como o conhecimento sobre a gerência de projetos permite aperfeiçoar os processos de desenvolvimento de software atuais torna possível o desenvolvimento de novas ferramentas capazes de suportar diferentes níveis de automatização no planejamento e na execução de atividades num projeto de software.

A análise das classes e relacionamentos do SPIM originou dois conjuntos de restrições para a validação de seus elementos: regras implícitas e explícitas. O conjunto de restrições para o modelo SPIM é apresentado na seção a seguir.

6.2 CONJUNTO DE RESTRIÇÕES PARA O SPIM

A integração dos modelos também permitiu que um conjunto de regras implícitas ou explícitas pudessem ser derivadas (Tabela 1). As regras implícitas são aquelas que podem ser obtidas diretamente do modelo integrado, por meio da semântica do diagrama UML. Um conjunto de regras explícitas foi adicionado objetivando, em sua maioria, para apoiar a consistência do modelo. Porém, as restrições explícitas não puderam ser expressas no diagrama devido à falta de expressividade do diagrama de classes da UML para este fim.

Algumas destas regras podem, contudo, ser definidas através de uma linguagem para especificação formal de restrições, tal como a *Object Constraint Language* (OCL) [WAR99]. Entretanto, nesta pesquisa optou-se por grafá-las em linguagem natural, para facilitar a compreensão.

Considerando que cada projeto é único, não é possível construir um *template* de projeto de software universal; ao invés disso, pode-se prover ferramentas que ajudem na execução e validação de um plano de projeto, baseado nos três tipos de atividades, recursos e produtos de trabalho relacionados

Tabela 1: Conjunto de Restrições Adicionais para o Modelo Integrado

Restrições Adicionais ao Modelo Integrado
• Um programa deve possuir um diretor. Logo, um *stakeholder* que é diretor de um programa deve possuir um papel gerencial;
• Um projeto deve ter apenas um *stakeholder*-chave, ou seja, um recurso responsável por direcionar e fundamentar o projeto (atributo *isKeyStakeholder* da classe *ProjectStakeholder*);
• Uma fase não pode ter ela mesma como predecessora ou antecessora;
• As fases do projeto não podem ocorrer em paralelo, de maneira que uma fase deve ser totalmente finalizada antes de iniciar outra;
• Uma atividade não pode ter ela mesma como predecessora ou antecessora;
• O fluxo de atividades não pode resultar em um ciclo; por exemplo, a atividade A é pré-requisito para a atividade B e a atividade B é pré-requisito para a atividade A;
• Uma mesma atividade não pode criar, modificar ou consultar um mesmo artefato. Para realizar tais operações devem ser criadas atividades distintas;
• Uma atividade gerencial deve ter pelo menos um papel gerencial como um de seus papéis;
• Uma atividade produtiva deve ter pelo menos um papel produtivo como um de seus papéis;
• O *stakeholder* responsável por uma atividade gerencial deve possuir um papel gerencial;
• O *stakeholder* responsável por uma atividade produtiva deve possuir um papel produtivo;
• Cada atividade do projeto deve ser de responsabilidade de apenas um indivíduo, mesmo que muitas pessoas venham a trabalhar naquela atividade;
• Uma atividade gerencial não pode produzir ou modificar um produto de trabalho produtivo, somente um produto de trabalho gerencial. Porém, esta atividade pode consultar um produto de trabalho produtivo;
• Uma atividade produtiva não pode produzir ou modificar um produto de trabalho gerencial, somente um produto de trabalho produtivo. Porém, esta atividade pode consultar um produto de trabalho gerencial;
• Para cada associação entre um papel e uma atividade (representado pela classe *ActivityStakeholderWork*) deve haver também uma associação dessa atividade com um *stakeholder* capaz de desempenhar aquele papel;
• Uma atividade somente pode atualizar ou consultar um produto de trabalho que já tenha sido criado por uma atividade antecessora;
• Uma atividade pode ou não ter *baselines*. Se tiver, a atividade original deve ter o atributo *IsBaseline=false* e todas as outras atividades (relacionadas via a associação *Baselines*) devem manter *IsBaseline* como *true*.
• Na classe associativa *ActivityStakeholderWork*, o *stakeholder* associado deve possuir o papel que se está associando à atividade.
• O papel do *stakeholder* envolvido deve ser compatível com o tipo de atividade.

De todas as regras definidas para o modelo SPIM, oito foram avaliadas neste trabalho (veja a Tabela 30). As regras foram implementadas em uma ferramenta chamada SPIT (***Software Planning Integrated Tool***), descrita na próxima seção, como um *Add-in* para um software de gerenciamento de projetos já existente e muito utilizado. A seguir serão apresentados maiores detalhes sobre o protótipo desenvolvido nesta pesquisa.

6.3 CONSIDERAÇÕES SOBRE O CAPÍTULO

Este capítulo apresentou os aspectos gerais sobre o planejamento integrado de atividades pertencentes ao projeto de software e aos fluxos de trabalho da organização. Objetivando-se obter um processo mais detalhado para o gerenciamento de projetos software é necessário aplicar os conhecimentos de gestão de projetos aos processos de desenvolvimento do software. Conseqüentemente, faz-se necessário mais estudo para uma solução que permita um melhor nível de integração entre os conceitos e modelos para estas duas áreas.

Dessa forma, após uma análise individual de cada metamodelo base de integração (vistos no Capítulo 5), foi apresentado neste capítulo o modelo de integração SPIM. O modelo SPIM foi concebido considerando a complexidade de identificar as relações de dependência entre as atividades de existentes dos fluxos de trabalho empresariais e o fluxo de trabalho de um projeto de software particular. Além disso, a análise das classes e relacionamentos do SPIM originou dois conjuntos de restrições, implícitas e explícitas, para a validação de seus elementos.

O próximo capítulo apresenta o protótipo desenvolvido para demonstrar a viabilidade dos conceitos propostos pelo modelo integrado SPIM e pelo conjunto de regras derivadas deste modelo.

7 SPIT - SOFTWARE PLANNING INTEGRATED TOOL

Esse capítulo fornece uma visão detalhada sobre o protótipo desenvolvido para demonstrar os conceitos propostos pelo modelo integrado SPIM.

7.1 DESCRIÇÃO DO PROTÓTIPO

O protótipo denominado ***Software Planning Integrated Tool*** **(SPIT)** tem como objetivo demonstrar os conceitos propostos pelo modelo integrado SPIM e pelo seu conjunto de restrições. O SPIT foi desenvolvido utilizando a linguagem de programação orientada a objetos Microsoft Visual C#.Net [WIL02]. Além disso, este protótipo acessa o banco de dados *Microsoft SQL 2005 Express* [MIC07] *por meio de aproximadamente 30 stored procedures*. Nesta base de dados são armazenadas as informações propostas pelas classes do modelo integrado SPIM, tais como fases, papéis e produtos de trabalho.

Conforme ilustrado na Figura 15, o protótipo *Software Planning Integrated Tool* é composto por dois módulos: *BackOffice* e *Add-in*.

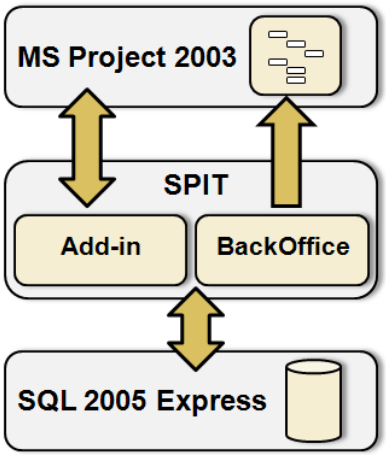


Figura 15. Arquitetura da solução

O módulo *BackOffice* é responsável por cadastrar as informações requeridas pelo modelo de integração SPIM, tais como a definição de papéis, tipos de atividades, *guidances* e produtos de trabalho associados. Posteriormente, estas informações podem ser exportadas para uma ferramenta comercial de gerenciamento de projetos através de campos customizados (*custom fields*). Uma maneira de realizar esta exportação é através da criação de projetos *template*. Assim, as informações necessárias para realizar a validação do projeto de acordo com as restrições

definidas pelo modelo integrado (tais como a diferenciação de atividades produtivas, gerenciais e gerenciais de apoio) serão exportadas para o software comercial juntamente com as atividades cadastradas no projeto *template*. Outra forma de realizar esta exportação é selecionar especificamente quais as informações serão cadastradas como campos customizados para um projeto específico.

O módulo *Add-in* contém as principais funcionalidades do protótipo SPIT. Ele atua como um *Add-in* para a ferramenta de gestão de projetos *Microsoft Office Project Professional 2003* [CHA03]. Conforme mencionado anteriormente, a definição do modelo SPIM resultou em um conjunto de restrições identificadas para apoiar a sua consistência. Dessa forma, este módulo é responsável por realizar as validações dos conceitos propostos pelo modelo integrado SPIM com o auxílio das informações que estão contidas nos campos customizados da ferramenta comercial utilizada. A Figura 16 apresenta um *snapshot* do SPIT em ação.

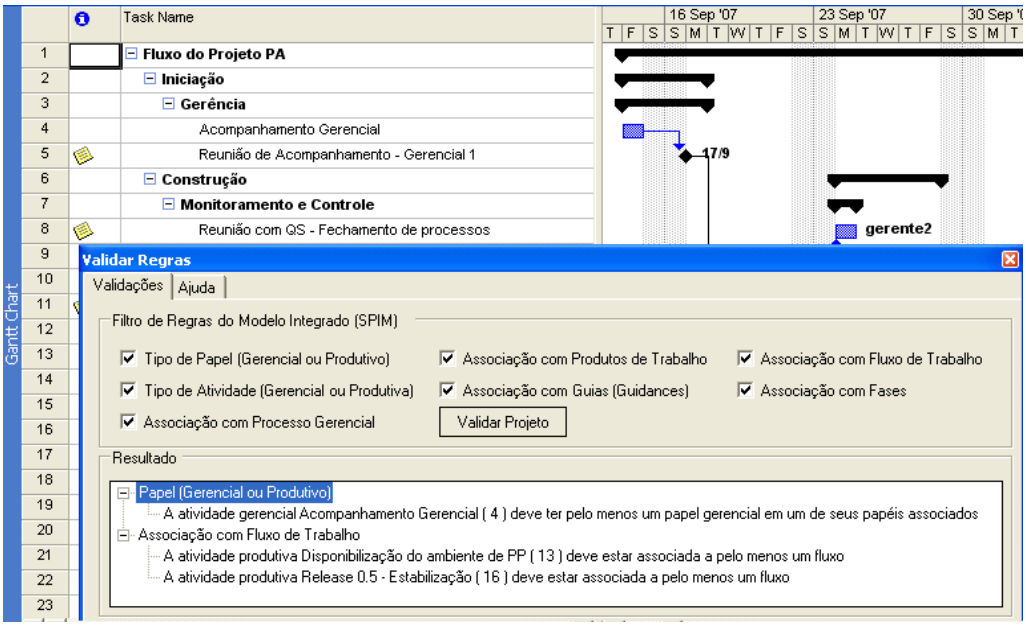


Figura 16. Módulo *Add-in* do SPIT [ROS08]

O protótipo desenvolvido realiza uma integração com o *Microsoft Office Project* objetivando aproveitar as funcionalidades que esta ferramenta já implementa e que estão de acordo com o modelo de integração proposto. Porém, o SPIM propõem alguns conceitos e restrições que não são implementados por esta ferramenta comercial. De acordo com [CHA03], este software comercial possui campos extras na sua base de dados que permitem o armazenamento de informações personalizadas às atividades e recursos dos projetos. Assim, foi possível adicionar à ferramenta comercial, por exemplo, a informação sobre o tipo de papel de um *stakeholder*. Além disso, conforme ilustrado na Figura 17, foram especificadas as seguintes informações customizadas para as atividades do projeto: identificador da atividade,

tipo de atividade (produtiva, gerencial ou gerencial de apoio), associação com um ou mais fluxos de trabalho do RUP, associação com um processo gerencial do PMBOK, associação com um tipo de *guidance* e associação com os produtos de trabalho que são criados, consultados ou modificados pela execução desta atividade.

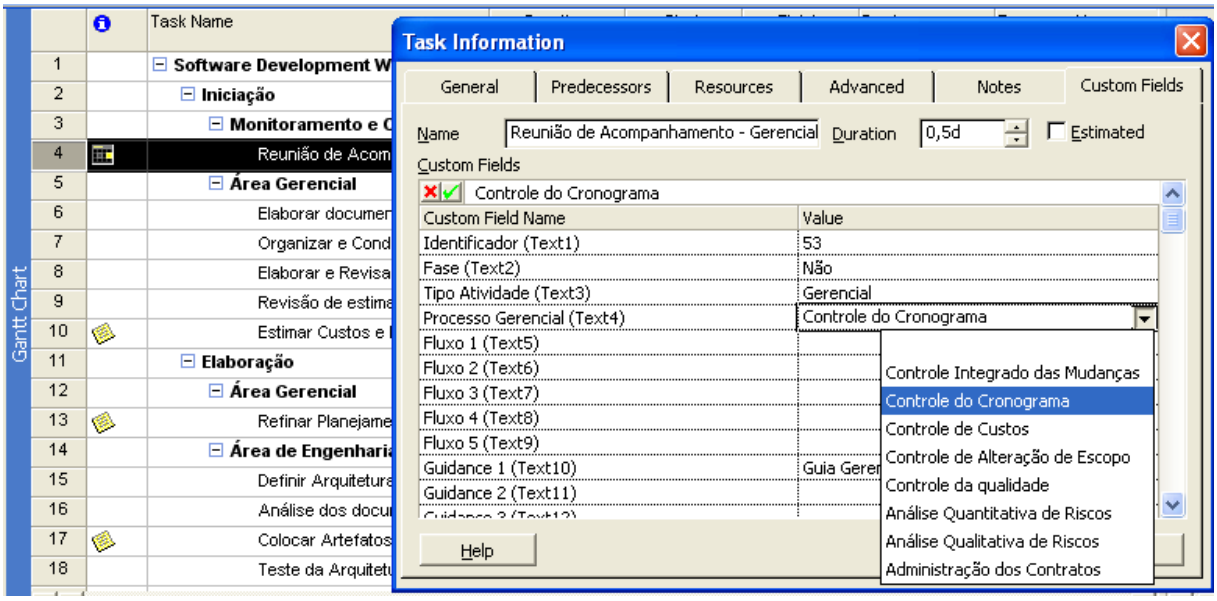


Figura 17. Módulo *Add-in* do SPIT [ROS08]

A seguir é apresentada a modelagem desenvolvida para esta ferramenta. Em seguida, são apresentadas as interfaces do protótipo SPIT.

7.2 MODELAGEM DO SISTEMA

De acordo com [BOO00], a UML (*Unified Modeling Language*) é uma linguagem para especificação, documentação, visualização e desenvolvimento de sistemas orientados a objetos. Por meio de seus diagramas é possível representar sistemas de softwares sob diversas perspectivas de visualização.

Dessa forma, nesta seção serão apresentados os principais diagramas desenvolvidos para modelar o SPIT.

7.2.1 ATORES

Foi identificado o seguinte ator de sistema:

- **Gerente de Projetos:** profissional responsável pela análise e definição das atividades do cronograma.

7.2.2 MODELOS DE CASOS DE USO

7.2.2.1 MÓDULO ADD-IN

Para compor a modelagem de sistema para o módulo *Add-in* do SPIT foram identificados os casos de uso ilustrados na Figura 18. Conforme mencionado anteriormente, o este módulo contém as principais funcionalidades do protótipo SPIT. Dessa forma, a seguir serão identificados os casos de uso desenvolvidos para este módulo.

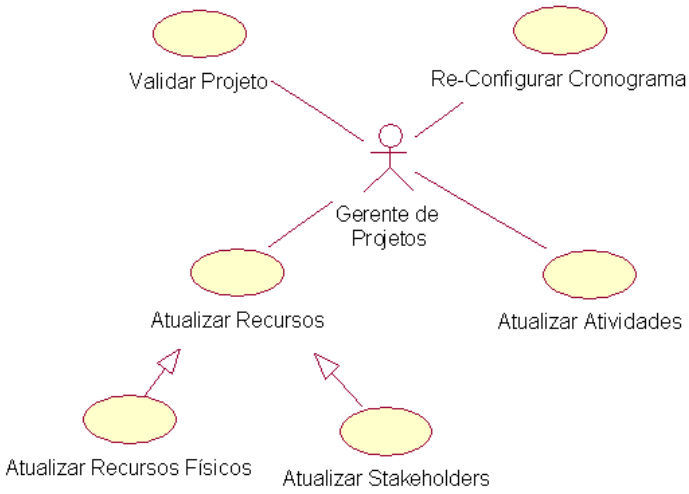


Figura 18. Diagrama de casos de uso do módulo *Add-in* do SPIT

A Tabela 2 descreve brevemente estas funcionalidades identificadas para cada caso de uso identificado para o módulo *BackOffice* do protótipo. A descrição detalhada dos casos de uso deste módulo é apresentada no Apêndice A deste volume.

Tabela 2: Funcionalidades do módulo Add-in do SPIT.

Caso de Uso	Funcionalidade
• Validar Projeto	Validar o projeto de software de acordo com os conceitos propostos pelo modelo de integração SPIM.
• Re-Configurar Cronograma	Re-configurar os prazos das atividades do projeto de maneira a auxiliar o gerente de projetos a antecipar as necessidades advindas das áreas de apoio durante o planejamento do projeto.
• Atualizar Atividades	Manutenção das informações correspondentes as atividades do cronograma, associando a recursos, fases, produtos e *stakeholders* do projeto.
• Atualizar Recursos	Manutenção das informações correspondentes aos *stakeholders* e recursos físicos do projeto.
• Atualizar *Stakeholders*	Manutenção das informações correspondentes especificamente aos *stakeholders*.
• Atualizar Recursos Físicos	Manutenção das informações correspondentes especificamente aos recursos físicos.

7.2.2.2 MÓDULO BACKOFFICE

As funcionalidades do módulo de *BackOffice* do protótipo estão representadas no diagrama de casos de uso UML da Figura 19.

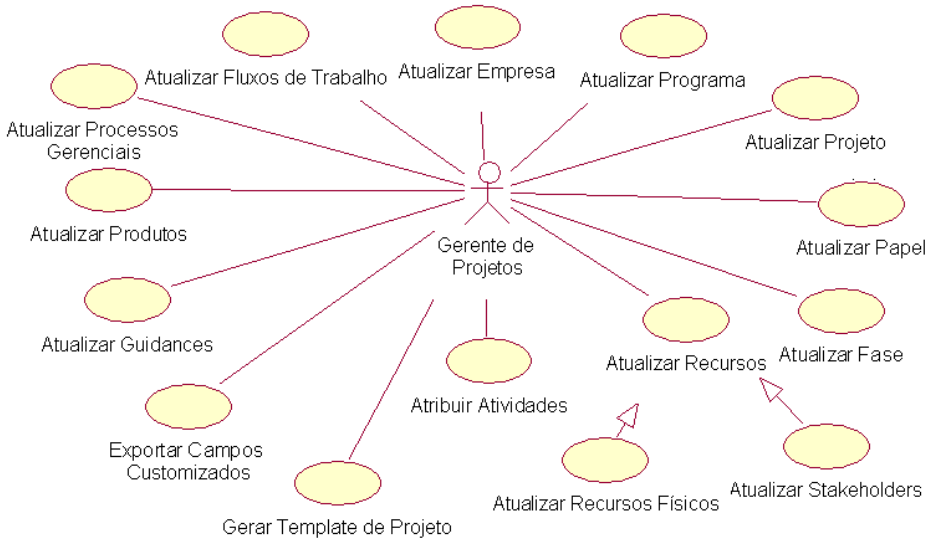


Figura 19. Diagrama de casos de uso do módulo *BackOffice* do SPIT

A Tabela 3 descreve brevemente estas funcionalidades identificadas para cada caso de uso identificado para o módulo *BackOffice* do protótipo.

Tabela 3: Funcionalidades do módulo *BackOffice* do SPIT.

Caso de Uso	Funcionalidade
• Atualizar Empresa	Manutenção das informações correspondentes as organizações que participam de projetos de software.
• Atualizar Programa	Manutenção das informações correspondentes aos programas de desenvolvimento de software de uma empresa.
• Atualizar Projeto	Manutenção das informações correspondentes especificamente ao projeto de software.
• Atualizar Papel	Manutenção das informações correspondentes aos papéis desempenhados pelos *stakeholders* nos projetos.
• Atualizar Fase	Manutenção das informações correspondentes às fases de um projeto de software.
• Atualizar Recursos	Manutenção das informações correspondentes aos *stakeholders* e recursos físicos do projeto.
• Atribuir Atividades	Manutenção das informações correspondentes as atividades do cronograma, associando a recursos, fases, produtos e *stakeholders* do projeto de acordo com o processo de desenvolvimento de software adotado pela organização.
• Gerar *Template* de Projeto	Gerar os arquivos de *template* que contenham as configurações necessárias para realizar a validação das regras do SPIM.

Caso de Uso	Funcionalidade
• Exportar Campos Customizados	Exportar os campos customizados para um arquivo, sem utilizar um *template* de atividades previamente cadastradas.
• Atualizar *Guidances*	Manutenção das informações de (*guidances*) para um projeto.
• Atualizar Produtos	Manutenção das informações sobre os produtos gerados, desenvolvidos ou consultados durante a execução de atividades.
• Atualizar Processos Gerenciais	Manutenção das informações correspondentes aos processos gerenciais do PMBOK
• Atualizar Fluxos de Trabalho	Manutenção das informações correspondentes aos fluxos de trabalho definidos pelo RUP

7.2.3 MODELO DE CLASSES

Conforme mencionado anteriormente, a atual versão do modelo SPIM implementa exclusivamente os conceitos advindos da integração do PMBOK com o RUP. Dessa forma, foram definidas as classes ilustradas na Figura 20 para o desenvolvimento do protótipo SPIT.

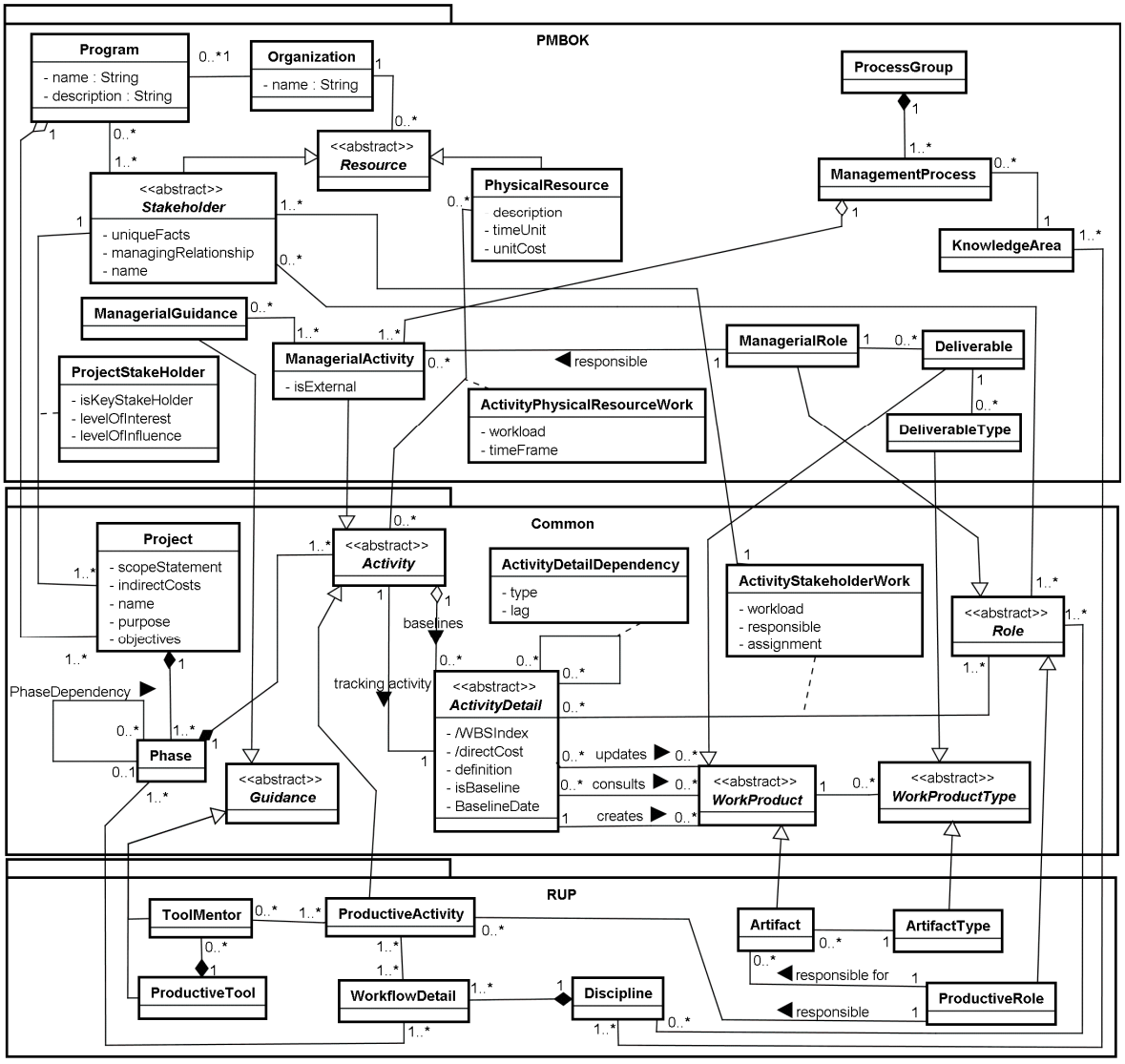


Figura 20. Diagrama de classes de sistema

A seguir, cada pacote do modelo SPIM e seu conjunto de classes serão detalhados.

7.2.3.1 PACOTE PMBOK

Nesta seção, cada uma das classes pertencentes ao pacote PMBOK são apresentadas em termos de sua finalidade.

7.2.3.1.1 CLASSE ORGANIZATION

Esta classe representa uma organização que se organiza por programas. A classe *Organization* possui o atributo detalhado na Tabela 4.

Tabela 4: Atributos da Classe *Organization*.

Campo	Tipo	Descrição
Name	String	Atributo que contém o nome da organização.

7.2.3.1.2 CLASSE PROGRAM

Os programas são grupos de projetos designados a alcançar um objetivo estratégico. Dessa forma, um programa é um conjunto de projetos gerenciados de um modo coordenado. A classe *Program* contém os atributos detalhados na Tabela 5.

Tabela 5: Atributos da Classe *Program*.

Campo	Tipo	Descrição
Name	String	Atributo que contém o nome do programa
Description	String	Atributo que contém informações resumidas sobre o programa.

7.2.3.1.3 CLASSE RESOURCE

A classe *Resource* representa um recurso necessário para o projeto (tais como pessoas, equipamentos e local). Estes recursos são divididos em recursos ativos (pessoas) e não-ativos (equipamentos e materiais).

7.2.3.1.4 CLASSE PHYSICALRESOURCE

A classe *PhysicalResource* herda da classe *Resource* e representa um recurso não-ativo em um projeto, como um equipamento, um local ou materiais. A classe *PhysicalResource* possui os atributos detalhados na Tabela 6.

Tabela 6: Atributos da Classe *PhysicalResource*.

Campo	Tipo	Descrição
Description	String	Atributo que contém informações resumidas sobre o recurso.
TimeUnit	String	Atributo que contém o tipo de unidade de tempo para este recurso
UnitCost	Double	Contém o custo unitário deste recurso

7.2.3.1.5 CLASSE STAKEHOLDER

A classe *Stakeholder* herda da classe *Resource* e corresponde a todas as pessoas e organizações cujos interesses são afetados do projeto. A classe *Stakeholder* possui os atributos detalhados na Tabela 7.

Tabela 7: Atributos da Classe *Stakeholder*.

Campo	Tipo	Descrição
Name	String	Atributo que contém nome do *stakeholder*
UniqueFacts	String	Atributo que contém informações gerais sobre o *stakeholder*. Por exemplo: Ph.D. em Biologia tem um bom senso de humor e é casado com dois filhos.
ManagingRelationship	String	Atributo que contém informações importantes sobre como se relacionar com o *stakeholder*. Por exemplo: Mantenha este *stakeholder* o mais informado possível e permita que ele coordene as reuniões.

7.2.3.1.6 CLASSE PROJECTSTAKEHOLDER

Esta classe associativa representa o relacionamento do *stakeholder* com o projeto. Ela informa se é um *stakeholder* chave do projeto (atributo *isKeyStakeHolder*), o nível de interesse (atributo *levelOfInterest*) e o seu nível de influência deste *stakeholder* no projeto (atributo *levelOfInfluence*). Assim, por exemplo, um *stakeholder* pode estar associado a um ou mais projetos, porém exatamente um *stakeholder* é considerado como responsável do projeto. Os atributos desta classe são detalhados na Tabela 8.

Tabela 8: Atributos da Classe *ProjectStakeholder*.

Campo	Tipo	Descrição
IsKeyStakeHolder	Boolean	Atributo que informa se é um *stakeholder* chave do projeto
LevelOfInterest	Int	Atributo que informa o nível de interesse deste *stakeholder* no projeto.
LevelOfInfluence	Int	Atributo que informa o nível de influência deste *stakeholder* no projeto.

7.2.3.1.7 CLASSE MANAGERIALGUIDANCE

Esta classe herda da classe *Guidance* e representa os elementos de orientação, tais como ferramentas e técnicas, necessárias para executar as atividades gerenciais. As ferramentas auxiliam gerentes de projetos e suas equipes a lidar com fatores de escopo, tempo e custo do projeto, além de outros assuntos gerenciais (tais como, gerências de recursos

humanos, qualidade e riscos). As técnicas gerenciais, por sua vez, ajudam a definir as habilidades necessárias para executar as atividades gerenciais.

7.2.3.1.8 CLASSE MANAGERIALACTIVITY

A classe *ManagerialActivity* ela herda da classe *Activity* e descreve as atividades gerenciais desempenhadas por papéis gerenciais durante o andamento do projeto. Estas atividades podem pertencer ao fluxo particular de um projeto (atributo *isExternal=false*) ou podem pertencer a um fluxo de atividades comum na organização (atributo *isExternal=true*). A classe *ManagerialActivity* contém o atributo detalhado na Tabela 9.

Tabela 9: Atributos da Classe *ManagerialActivity*.

Campo	Tipo	Descrição
IsExternal	Boolean	Atributo que informa se uma atividade é dita gerencial ou gerencial de apoio.

7.2.3.1.9 CLASSE MANAGERIALROLE

A classe *ManagerialRole* herda da classe *Role* e representa os papéis gerenciais desempenhados pelos *stakeholders* na execução de suas atividades.

7.2.3.1.10 CLASSE ACTIVITYPHYSICALRESOURCEWORK

A classe *ActivityPhysicalResourceWork* associa os recursos físicos às atividades de um projeto de software. Ela estabelece a carga de trabalho destes recursos físicos (atributo *workload*) naquela atividade. Este relacionamento permite a automatização, pois permite que uma atividade, por exemplo, utilize um computador (recurso) sem a necessidade da iteração com uma pessoa. A classe *ActivityPhysicalResourceWork* contém os atributos detalhados na Tabela 10.

Tabela 10: Atributos da Classe *ActivityPhysicalResourceWork*.

Campo	Tipo	Descrição
Workload	Double	Atributo que contém a informação sobre a carga de trabalho atribuída ao *stakeholder* para o desempenho de uma atividade.
TimeFrame	Int	Atributo que contém a informação sobre a seqüência temporal das atividades

7.2.3.1.11 CLASSE DELIVERABLE

A classe *Deliverable* herda da classe *WorkProduct* e representa um produto de trabalho que é produzido, consumido ou modificado durante a execução das atividades gerenciais.

7.2.3.1.12 CLASSE DELIVERABLETYPE

A classe *DeliverableType* herda da classe *WorkProductType* e define o tipo de um produto de trabalho gerencial (tais como um contrato e uma proposta comercial).

7.2.3.1.13 CLASSE PROCESSGROUP

A classe *ProcessGroup* representa os grupos de processos do PMBOK, tais como iniciação, planejamento, execução, controle e fechamento.

7.2.3.1.14 CLASSE MANAGEMENTPROCESS

A classe *ManagementProcess* representa cada um dos trinta e nove processos gerenciais contidos no PMBOK Guide.

7.2.3.1.15 CLASSE KNOWLEDGEAREA

A classe *KnowledgeArea* representa uma área de conhecimento do PMBOK e descreve as principais competências que os gerentes de projeto devem desenvolver. Desta classe derivam as áreas centrais (*core*) e as de apoio (*facilitating*).

7.2.3.2 PACOTE COMMON

Esta seção descreve as classes do pacote *Common* do modelo integrado SPIM.

7.2.3.2.1 CLASSE PROJECT

A classe *Project* representa um projeto único dentro de um programa. Cabe lembrar que as organizações (classe *Organization*) são responsáveis por executar trabalhos, que geralmente envolvem projetos (classe *Projects*). Estes projetos são agrupados em programas (classe *Program*). A classe *Project* contém os atributos detalhados na Tabela 11.

Tabela 11: Atributos da Classe *Project*.

Campo	Tipo	Descrição
ScopeStatement	String	Atributo que contém informações resumidas sobre o escopo do projeto
Name	String	Atributo que contém nome do projeto
Purpose	String	Atributo que contém informações sobre qual é o propósito deste projeto
Objectives	String	Atributo que contém uma lista de objetivos do projeto
IndirectCost	Double	Atributo que contém informações sobre os custos indiretos do projeto

7.2.3.2.2 CLASSE PHASE

A classe *Phase* representa uma fase do ciclo de vida de um projeto. As organizações, geralmente, dividem os projetos em várias fases visando um melhor controle gerencial. Segundo [SCH02], um projeto deve terminar cada fase com sucesso antes de iniciar outra fase. Dessa forma, existe uma relação de dependência entre as fases (relacionamento *PhaseDependency*).

7.2.3.2.3 CLASSE ACTIVITY

A classe *Activity* descreve uma unidade particular de trabalho que é desempenhada por um papel. Cada atividade deve possuir um produto de trabalho bem definido, uma pessoa responsável pela atividade e os recursos (humanos e materiais) necessários para executar a atividade. Uma atividade (classe *Activity*) pode ser classificada como atividade produtiva (*ProductiveActivity*) ou atividade gerencial (*ManagerialActivity*). Além disso, as atividades gerenciais podem pertencer ao fluxo de desenvolvimento de software (*isExternal=false*) ou aos fluxos empresariais da organização (*isExternal=true*). Dessa forma, identificam-se três tipos de atividades: produtivas, gerenciais e gerenciais de apoio.

7.2.3.2.4 CLASSE ACTIVITYDETAIL

Cada atividade pode pertencer a um ou mais *baselines*. Em cada geração da *baseline*, uma atividade deve manter os relacionamentos com os papéis e produtos de trabalho. Dessa forma, a classe *ActivityDetail* é responsável por armazenar a informação pertinente sobre a execução de uma atividade e, principalmente, por manter o relacionamento entre as atividades. A classe *ActivityDetail* contém os atributos detalhados na Tabela 12.

Tabela 12: Atributos da Classe *ActivityDetail*.

Campo	Tipo	Descrição
/WBSIndex	Int	Atributo que contém o identificador desta atividade no WBS
/DirectCost	Double	Atributo derivado que contém informações sobre os custos diretamente relacionados a execução desta atividade.
Definition	String	Atributo que contém a definição de uma atividade
isBaseline	Boolean	Atributo que informa se esta é a atividade que faz parte do *baseline*
BaselineDate	Date	Atributo que informa a data de adição desta classe ao baseline.

7.2.3.2.5 CLASSE ACTIVITYDEPENDENCY

A classe *ActivityDependency* representa uma dependência entre duas atividades. Uma relação de dependência define a seqüência das atividades de um projeto. Desta forma, é definido se uma atividade deve terminar antes de outra atividade ser iniciada, se atividades podem ser realizadas em paralelo, ou então se uma atividade pode sobrepor outra. A classe *ActivityDependency* possui os atributos detalhados na Tabela 13.

Tabela 13: Atributos da Classe *ActivityDependency*.

Campo	Tipo	Descrição
Type	Int	Atributo que contém o tipo de dependência entre atividades
Lag	Int	Atributo que informa o tempo de retardo, ou seja, um atraso entre tarefas que apresentam uma dependência.

A definição dos relacionamentos e dependências entre as atividades tem um impacto significativo no desenvolvimento e controle do cronograma do projeto. Segundo [SCH02], o tipo da dependência reflete uma das quatro possibilidades:

- Finish-to-Start (FS): uma atividade (B) não pode iniciar até que outra atividade (A) termine;
- Start-to-Start (SS): uma atividade (B) não pode iniciar até que outra atividade (A) também inicie;
- Finish-to-Finish (FF): uma atividade (B) não pode terminar até que outra atividade (A) termine;
- Start-to-Finish (SF): uma atividade (B) não pode terminar até que outra atividade (A) não tenha iniciado.

7.2.3.2.6 CLASSE ACTIVITYSTAKEHOLDERWORK

Uma atividade pode estar associada a um ou mais papéis, mas ela deve ser de responsabilidade de apenas um indivíduo. Dessa forma, a classe associativa *ActivityStakeholderWork* informa o *stakeholder* que é o único responsável pela atividade através do atributo *responsability*. Além disso, o tipo de responsabilidade atribuída ao *stakeholder* para uma determinada atividade pode ser definido através do atributo *assignment*. Finalmente, o atributo *workload* contém a informação sobre a carga de trabalho atribuída ao *stakeholder* para a atividade. Os atributos desta classe são detalhados na Tabela 14..

Tabela 14: Atributos da Classe *ActivityStakeholderWork*.

Campo	Tipo	Descrição
Responsability	Boolean	Atributo que informa se o *stakeholder* é o responsável pela atividade.
Assignment	String	Atributo que contém o tipo de responsabilidade atribuída ao *stakeholder* para a atividade.
Workload	Double	Atributo que contém a informação sobre a carga de trabalho atribuída ao *stakeholder* para o desempenho de uma atividade.

7.2.3.2.7 CLASSE ROLE

Esta classe descreve os papéis desempenhados por *stakeholders*. Assim, um *stakeholder* pode desempenhar diversos papéis e diversos *stakeholders* podem fazer parte de um único papel. Os papéis podem ser divididos em papéis gerenciais (classe *ManagerialRole*) e papéis produtivos (classe *ProductiveRole*).

7.2.3.2.8 CLASSE WORKPRODUCT

A classe *WorkProduct* representa algo que é produzido, consumido ou modificado (tais como documentos, modelos ou códigos fonte) durante a execução das atividades. Um produto do trabalho deve ser associado a um papel, que é formalmente responsável pela produção deste produto do trabalho. Assim, um produto do trabalho pode ser subdividido em produtos gerenciais (classe *Deliverable*) ou produtos produtivos (classe *Artifact*).

7.2.3.2.9 CLASSE WORKPRODUCTTYPE

A classe *WorkProductType* contém as informações sobre o tipo de um produto de trabalho específico em um projeto de software (tais como, código fonte e diagrama UML).

7.2.3.2.10 CLASSE GUIDANCE

A classe *Guidance* representa os elementos de orientação do processo. Ela descreve o uso das técnicas e ferramentas necessárias para a execução de algumas atividades.

7.2.3.3 PACOTE RUP

Esta seção apresenta cada uma das classes pertencentes ao pacote RUP.

7.2.3.3.1 CLASSE DISCIPLINE

A classe *Discipline* define a divisão de elementos de processo em áreas de interesse. Uma disciplina é composta por uma ou mais fluxos de trabalho.

7.2.3.3.2 CLASSE WORKFLOWDETAIL

A classe *WorkflowDetail* representa o agrupamento de atividades relacionadas a uma determinada disciplina. Ela define como os papéis devem colaborar entre si através de suas atividades para alcançar objetivos específicos do processo.

7.2.3.3.3 CLASSE PRODUCTIVEACTIVITY

A classe *ProductiveActivity* é denominada originalmente de *Activity* no metamodelo do RUP. Representa uma unidade de trabalho que produz um resultado significativo no contexto de um projeto. Ela herda da classe *Activity* pois descreve as atividades produtivas realizadas por papéis produtivos.

7.2.3.3.4 CLASSE PRODUCTIVETOOL

A classe *ProductiveTool* é denominada originalmente de *Tool* no metamodelo do RUP e descreve as ferramentas que podem ser utilizadas no auxílio da produção ou modificação de um artefato.

7.2.3.3.5 CLASSE TOOLMENTOR

A classe *ToolMentor* descreve o uso de ferramentas no contexto de algumas atividades produtivas.

7.2.3.3.6 CLASSE PRODUCTIVEROLE

A classe *ProductiveRole* é denominada originalmente de *Role* no metamodelo do RUP. Representa os papéis produtivos, desempenhados por stakeholders, responsáveis por desempenhar atividades para produzir ou modificar os artefatos do projeto.

7.2.3.3.7 CLASSE ARTIFACT

A classe *Artifact* descreve os de produtos de trabalho que são produzidos, consumidos ou modificados durante o projeto por papéis produtivos.

7.2.3.3.8 CLASSE ARTIFACTTYPE

A classe *ArtifactType* define o tipo de artefato técnico, tais como um modelo UML, executável, biblioteca, e assim por diante.

7.2.4 CRC CARDS

A partir da leitura realizada em cada caso de uso de sistema, descritos na seção acima, foram identificadas as responsabilidades e colaborações das principais classes do

modelo SPIM através da técnica de utilização de CRC (*Class Responsibility Collaborator*) Cards [WIL95].

7.2.4.1 CLASSE ORGANIZATION

ATabela 15 apresenta as responsabilidades e colaborações identificadas para a classe *Organization*.

Tabela 15: *Organization* CRC Card.

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre as empresas; • Manter um conjunto de programas associados; • Conter um conjunto de recursos ativos (pessoas) e não ativos (recursos físicos) que possam ser utilizados em projetos pertencentes aos seus programas; • Conter um grupo de papéis que serão utilizados no desempenho das atividades dos seus projetos.	• *Program* • *Resource* • *Role* • *Guidance*

7.2.4.2 CLASSE PROGRAM

A Tabela 16 apresenta as responsabilidades e colaborações da classe *Program*.

Tabela 16: *Program* CRC Card.

Responsabilidades	Colaborações
• Manter as informações básicas sobre os programas; • Uma organização é gerenciada por programas; • Manter um conjunto de projetos associados.	• *Organization* • *Project*

7.2.4.3 CLASSE PROJECT

A Tabela 17 apresenta as responsabilidades e colaborações identificadas para a classe *Project*.

Tabela 17: *Project* CRC Card.

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre os Projetos; • Os projetos são agrupados em programas; • Um projeto tem seu ciclo de vida dividido em fases; • Um projeto tem um ou mais papéis associados os quais executam as suas atividades; • Um projeto tem produtos de trabalho que serão criados, consultados e atualizados no desempenho de atividades	• *Program* • *Phase* • *Role* • *WorkProduct*

7.2.4.4 CLASSE PHASE

A Tabela 18 apresenta as responsabilidades e colaborações da classe *Phase*.

Tabela 18: *Phase* CRC Card.

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre as fases do projeto; • Detalhar as fases do ciclo de vida de um projeto; • Cada fase é composta por um conjunto de atividades • Manter uma ordem de precedência entre outras fases	• *Project* • *Activity* • *Phase*

7.2.4.5 CLASSE RESOURCE

A Tabela 19 apresenta as responsabilidades e colaborações da classe *Resource*.

Tabela 19: *Resource* CRC Card.

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre os recursos necessários para o projeto, por exemplo: pessoas, equipamentos, local, etc. • Dividir os recursos em recursos ativos (pessoas) e não-ativos (equipamentos, materiais, etc.); • Pertencer a uma empresa; • Definir informações de taxa de custo;	• *Organization* • *Stakeholder* • *Physical Resource*

7.2.4.6 CLASSE STAKEHOLDER

A Tabela 20 apresenta as responsabilidades e colaborações identificadas para a classe *Stakeholder*.

Tabela 20: *Stakeholder* CRC Card.

Responsabilidades	Colaborações
• Manter as informações básicas sobre os *stakeholders*; • Informar o nível de interesse do *stakeholder* para cada projeto selecionado; • Informar o nível de influência para cada projeto selecionado; • Informar se é o *stakeholder* chave para um projeto; • Realizar as atividades de um projeto; • Pertencer a um conjunto de recursos disponíveis de uma empresa e especificar as informações de custo para desempenhar atividades do projeto; • Assumir certo papel no projeto no desempenho das atividades;	• *Resource* • *ProjectStakeholder* • *Project* • *ActivityStakeholderWork* • *Program* • *Role* • *Activity* • *WorkProduct*

7.2.4.7 CLASSE PHYSICALRESOURCE

A Tabela 21 contém as responsabilidades e colaborações da classe *PhysicalResource*.

Tabela 21: *PhysicalResource* CRC Card.

Responsabilidades	Colaborações
• Manter as informações básicas sobre os recursos do projeto; • Pertencer a um conjunto de recursos disponíveis de uma empresa e especificar as informações de custo para desempenhar atividades do projeto; • Ser utilizado no desempenho de atividades do projeto	• *Resource* • *Project* • *Activity*

7.2.4.8 CLASSE ROLE

A Tabela 22 apresenta as responsabilidades e colaborações da classe *Role*.

Tabela 22: *Role* CRC Card.

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre os Papéis; • Permitir que os *stakeholders* assumam papéis para desempenhar as atividades (gerenciais ou técnicas) nos projetos de uma determinada empresa; • Indica qual o papel de um *stakeholder* em uma atividade, definindo sua carga de trabalho; • Ser responsável pelo desenvolvimento de produtos	• *Organizarion* • *Stakeholder* • *ActivityStakeholderWork* • *WorkProduct*

7.2.4.9 CLASSE ACTIVITY

A Tabela 23 apresenta as responsabilidades e colaborações da classe *Activity*.

Tabela 23: *Activity* CRC Card.

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre as atividades do projeto; • Manter uma ordem de precedência entre as atividades; • Pertencer a um projeto; • Pertencer a uma fase do projeto; • Estar associados à *guidances* do projeto; • Especificar recursos (ativos e não ativos) que auxiliam no desempenho da atividade, informando dados de tempo e custo para o desempenho de tais atividades; • Especificar produtos criados, modificados ou consultados no desempenho esta atividade; • Estar atribuída a um *stakeholder* do projeto; • Pertencer a zero ou mais *baselines*.	• *Activity* • *ActivityDetailDependency* • *Phase* • *ActivityStakeholderWork* • *Role* • *PhysicalResource* • *WorkProduct* • *Guidelines* • *ActivityDetail*

7.2.4.10 CLASSE ACTIVITYDETAILDEPENDENCY

A Tabela 24 apresenta as responsabilidades e colaborações da classe *ActivityDetailDependency*.

Tabela 24: *ActivityDetailDependency* CRC Card.

Responsabilidades	Colaborações
• Manter as informações de dependência entre as atividades;	• *Activity*

7.2.4.11 CLASSE ACTIVITYRESOURCEWORK

A Tabela 25 apresenta as responsabilidades e colaborações da classe *ActivityResourceWork*.

Tabela 25: *ActivityResourceWork* CRC Card.

Responsabilidades	Colaborações
• Determina o papel de cada *stakeholder* envolvido em uma atividade indicando sua carga de trabalho e se é o único responsável pela atividade.	• *Stakeholder* • *Activity*

7.2.4.12 CLASSE PHYSICALRESOURCEWORK

A Tabela 26 apresenta as responsabilidades e colaborações da classe *PhysicalResourceWork*.

Tabela 26: *PhysicalResourceWork* CRC Card.

Responsabilidades	Colaborações
• Determina a carga de trabalho de um recurso físico atribuída a uma determinada atividade.	• *Physical Resource* • *Activity*

7.2.4.13 CLASSE WORKPRODUCT

A Tabela 27 apresenta as responsabilidades e colaborações da classe *WorkProduct*.

Tabela 27: *WorkProduct CRC Card.*

Responsabilidades	Colaborações
• Manter as informações cadastrais básicas sobre os produtos gerados no projeto; • Ser desenvolvido durante a execução de atividades de um determinado projeto; • Ser de responsabilidade de um papel	• *Project* • *Activity* • *Role*

7.3 INTERFACES DO SISTEMA

Esta seção apresenta as principais interfaces gráficas do protótipo desenvolvido nesta pesquisa, explicando suas respectivas funcionalidades. O conjunto restante das interfaces do sistema é apresentado no Apêndice B deste volume.

7.3.1 MÓDULO *ADD-IN*

7.3.1.1 VALIDAÇÃO DAS REGRAS DO SPIM

Conforme mencionado anteriormente, o protótipo SPIT foi desenvolvido para demonstrar os conceitos e regras propostos pelo modelo integrado SPIM. Estas regras foram implementadas em um módulo do protótipo que atua como um *Add-in* para um software de gerenciamento de projetos já existente.

A Figura 21 ilustra a integração do módulo *Add-in* do SPIT com o software comercial de gestão e projetos

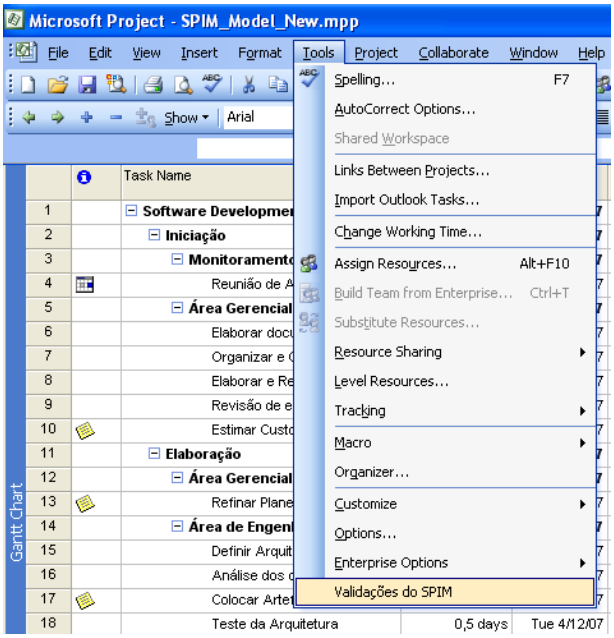


Figura 21. Integração do *Add-in* do SPIT com o software comercial de gestão e projetos

Ao selecionar a opção "Validações do SPIM", o software comercial inicializa o módulo *Add-in* do SPIT. Dessa forma, a Figura 22 apresenta a interface para a validação das regras do modelo integrado SPIM, a qual permite a validação do cronograma de um projeto de software de acordo com os conceitos propostos nesta pesquisa. Esta interface apresenta campos para definir o conjunto de regras que devem ser validadas.

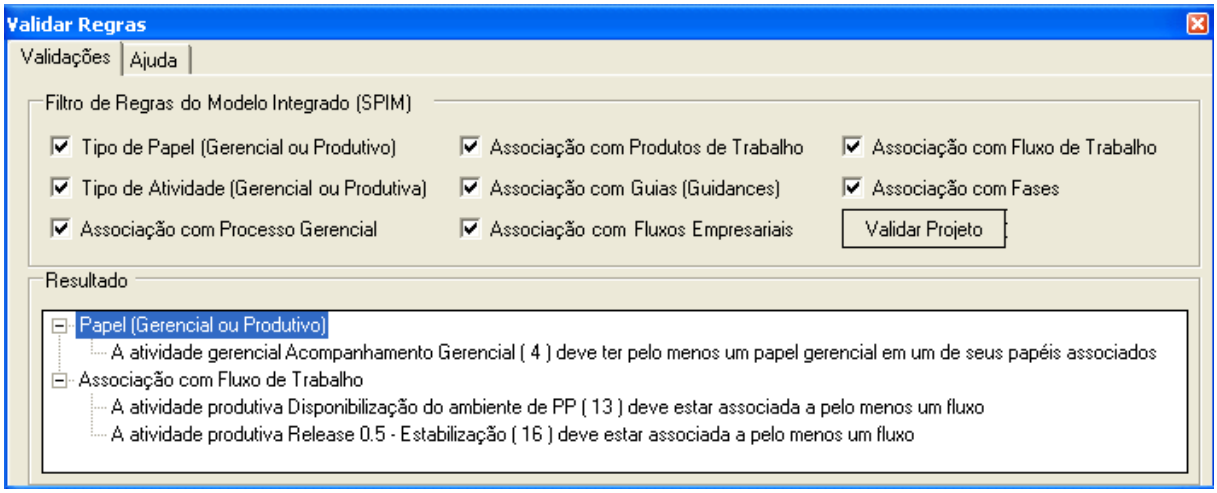


Figura 22. Tela para validação das regras do SPIM

Para realizar a operação de validação de uma restrição através do *Add-in*, deve-se primeiramente abrir um projeto cujos campos customizados (*custom fields*) já tenham sido previamente configurados. Após, deve-se selecionar a regra que se deseja avaliar e clicar no botão "Validar Projeto". O sistema irá realizar o processamento e retornará os resultados em uma lista.

Este procedimento é válido para quase todas as regras implementadas. A única exceção a este comportamento é a restrição "Associação com Fluxos Empresariais", a qual deverá re-configurar os prazos das atividades do projeto de maneira a auxiliar o gerente de projetos a antecipar as necessidades advindas das áreas de apoio durante o planejamento do projeto.

7.3.2 MÓDULO *BACKOFFICE*

7.3.2.1 MANUTENÇÃO DE ATIVIDADES

O módulo de *BackOffice* do SPIT permite o cadastro das atividades que estarão disponíveis para a geração de *templates* para a ferramenta comercial de gerenciamento de projetos. A Figura 23 apresenta a interface de manutenção de atividades, a qual permite a criação, modificação, exclusão e visualização das informações correspondentes às atividades que servirão para a geração de *templates* de projetos de software.

A definição dos atributos para estes *templates* devem estar de acordo com o processo de desenvolvimento de software adotado pela organização. Dessa forma, esta interface apresenta campos para definir as informações de detalhes das atividades, a ordenação entre as atividades, além das associações com papéis, produtos de trabalho e *guidances* do projeto.

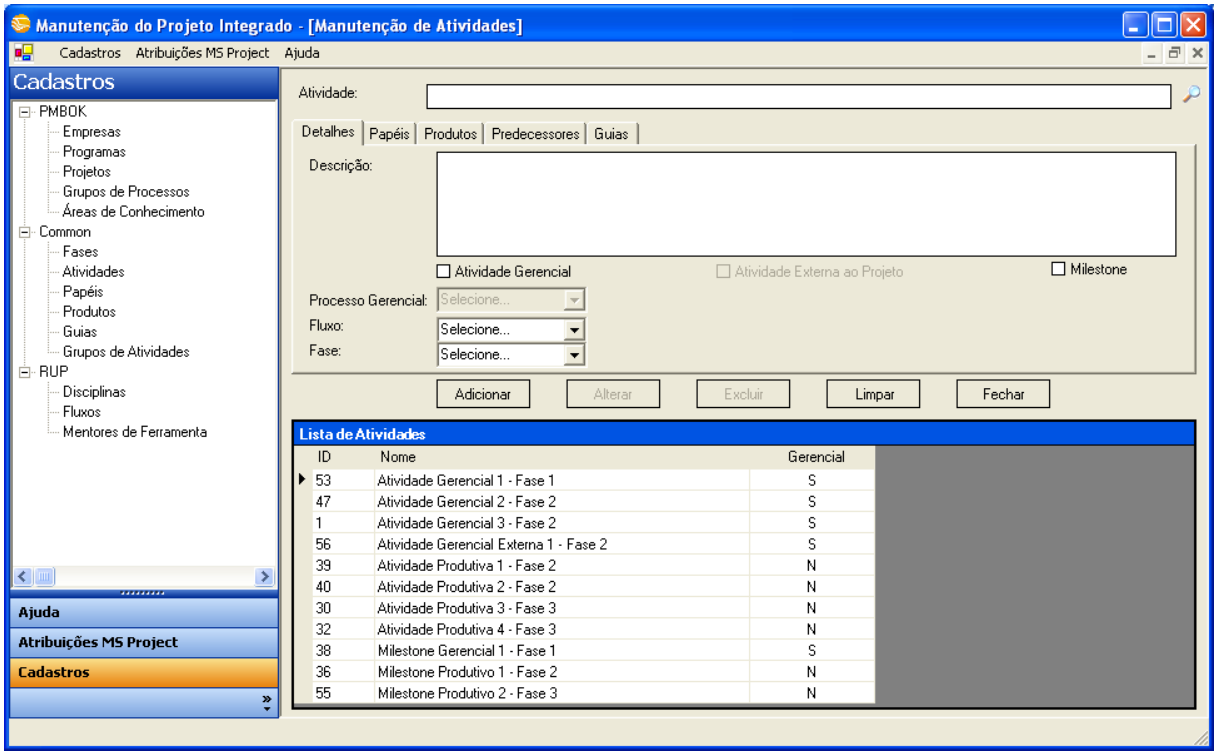


Figura 23. Tela para manutenção de atividades

A guia "Detalhes" apresenta campos para definir o nome e a descrição da atividade, se é um *milestone*, qual a fase que esta atividade pertence, além de um campo para definir se é uma atividade gerencial de apoio, gerencial ou produtiva. Além disso, também contém campos para definir se esta atividade está associada a um processo gerencial do PMBOK ou a um fluxo de trabalho definido pelo RUP.

Inicialmente, uma lista das atividades já cadastradas no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar. Para realizar a operação de inclusão de uma atividade no sistema, deve-se obrigatoriamente preencher o campo referente ao seu nome, associar a atividade a uma fase e selecionar um processo gerencial do PMBOK ou fluxo de trabalho do RUP (para atividades gerenciais ou produtivas, respectivamente), além de clicar no botão "Adicionar". Caso o nome desta atividade já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação.

Conforme ilustrado na Figura 24, a guia "Papéis" apresenta campos para definir o papel responsável por uma determinada atividade no projeto. Esta interface também permite definir os papéis participantes pela execução desta atividade.

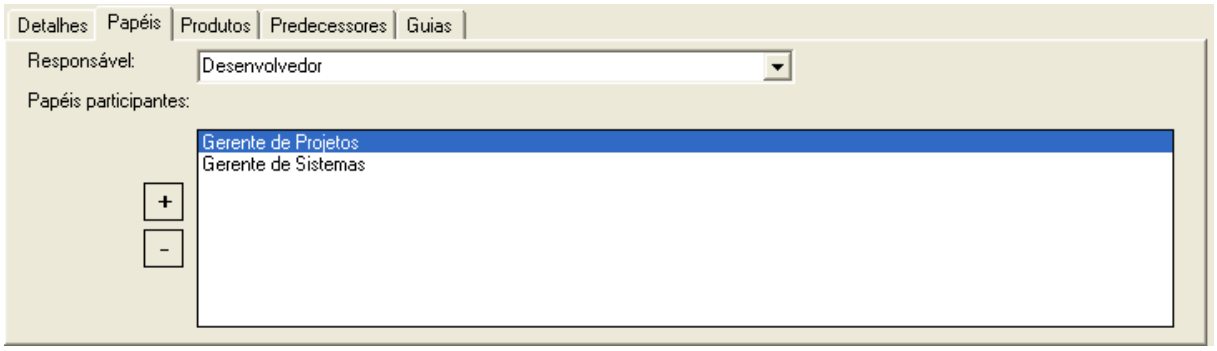


Figura 24. Guia "Papéis" da tela para manutenção de atividades

A guia "Produtos" (Figura 25) apresenta campos para definir as possíveis relações entre uma atividade e um produto de trabalho (criar/atualizar/consultar). Enquanto que a lista de entrada define os produtos de trabalho que são consultados por esta atividade, a lista de saída define os produtos de trabalho que são criados pela atividade. Além disso, para que se possa definir que um produto é atualizado por uma atividade, este o produto de trabalho deve pertencer tanto à lista de entrada quanto a lista de saída.

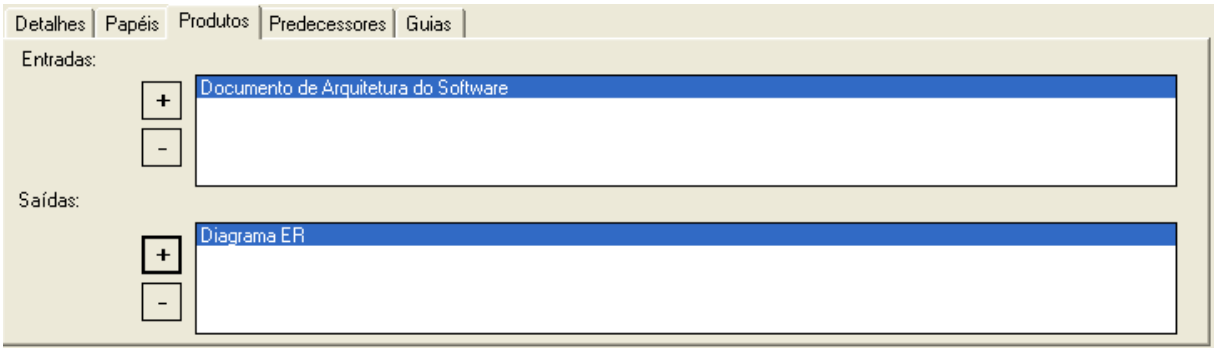


Figura 25. Guia "Produtos" da tela para manutenção de atividades

Uma atividade pode depender de zero ou mais atividades para poder iniciar. Dessa forma, a guia "Predecessores" (Figura 26) apresenta campos para definir os relacionamentos e dependências entre as atividades do cronograma de um projeto.

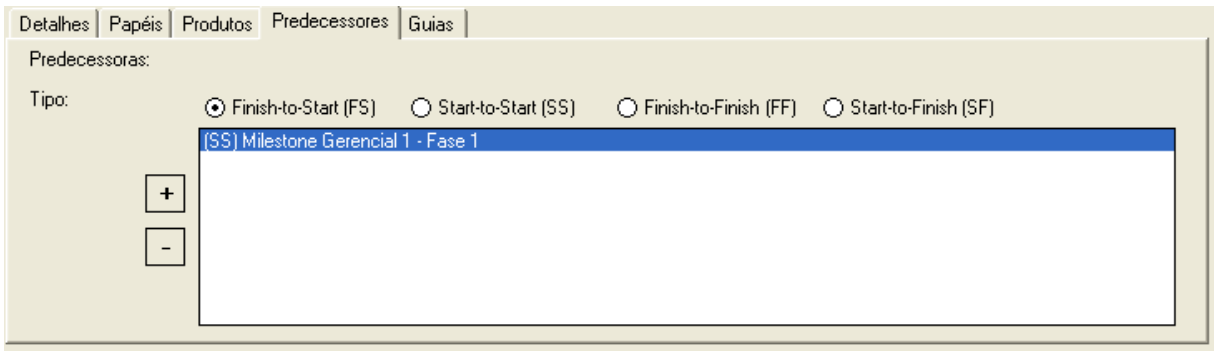


Figura 26. Guia "Predecessores" da tela para manutenção de atividades

A guia “Guias” apresenta campos para definir o uso de ferramentas no contexto de algumas atividades (Figura 27). Os tipos de orientação podem ser gerenciais ou produtivas, por exemplo: guias, técnicas, métricas, exemplos, UML *Profiles*, *checklists* e modelos.

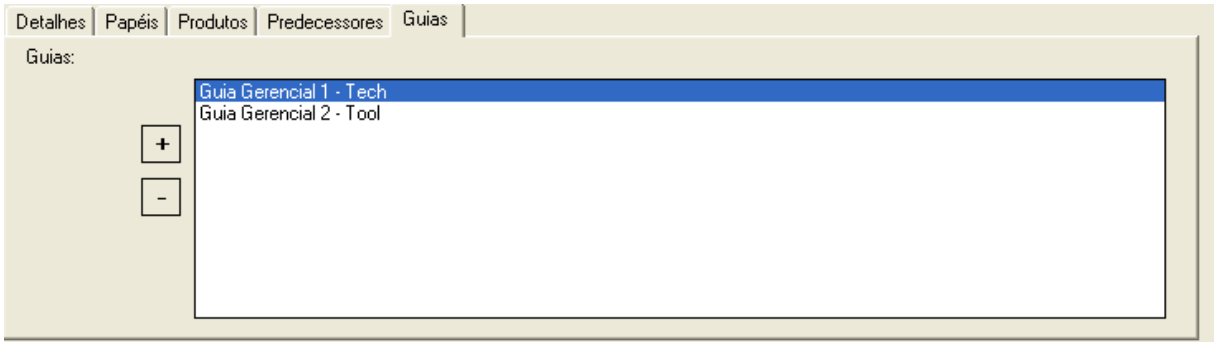


Figura 27. Guia “Guias” da tela para manutenção de atividades

Para as operações de alteração ou exclusão deve-se primeiramente selecionar a atividade a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar uma atividade específica cadastrada na base de dados do sistema. Para realizar esta operação é necessário digitar o nome da atividade e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

7.3.2.2 MANUTENÇÃO DE GRUPOS DE ATIVIDADES

De acordo com [SCH02], deve-se considerar o fato de que projetos sempre são esforços únicos. Dessa forma, o sistema permite definir diferentes grupos de atividades para a criação de *templates* de projetos de software. A Figura 28 apresenta a interface de manutenção de grupos de atividades, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos grupos de atividades que formam os *templates* de projetos de software. A interface apresenta campos para definir o nome do grupo e a fase a qual pertence. Além disso, é possível adicionar a uma lista as atividades pertencentes a este grupo de processo. O *checkbox* “Atividades Externas ao Projeto” permite que ao adicionar atividades a grupo específico seja possível fazer a distinção entre as atividades gerenciais de apoio (externas ao fluxo de atividades do projeto de software) das atividades gerenciais e produtivas do projeto.

Inicialmente, uma lista dos grupos de atividades já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar. Para realizar a operação de inclusão de um grupo de atividade no sistema, deve-se obrigatoriamente preencher o campo referente ao nome do grupo de atividade, selecionar uma fase, adicionar pelo menos uma atividade à lista e clicar no botão “Adicionar”. Caso o nome

deste grupo de atividades já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação.

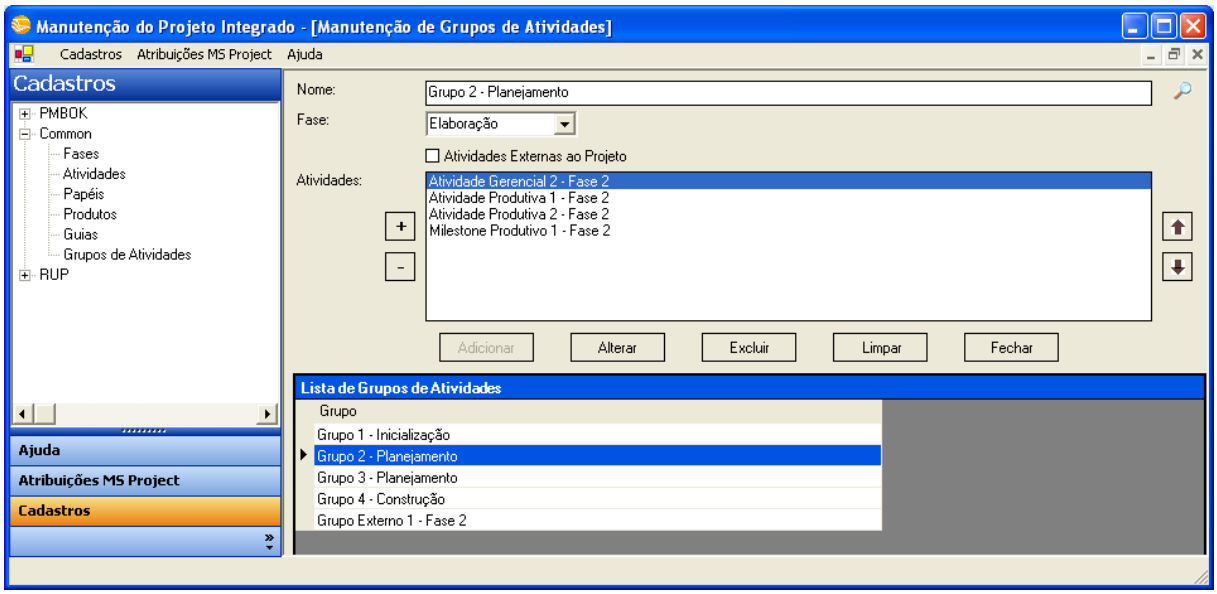


Figura 28. Tela para manutenção de grupos de atividades

Para as operações de alteração ou exclusão deve-se primeiramente selecionar o grupo de atividades a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar um grupo de atividades específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do grupo de atividades e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

7.3.2.3 EXPORTAÇÃO PARA A FERRAMENTA COMERCIAL DE GESTÃO DE PROJETOS

Conforme mencionado anteriormente, o módulo de *BackOffice* do SPIT é responsável por cadastrar as informações definidas pelo modelo de integração (papéis, tipos de atividades, *guidances* e produtos de trabalho associados, por exemplo). Estas informações podem ser exportadas para uma ferramenta de gerenciamento de projetos comercial através de campos customizados (*custom fields*).

7.3.2.3.1 DEFINIÇÃO DE TEMPLATES

Uma maneira de realizar a exportação das informações cadastradas no SPIT para a ferramenta comercial de gestão de projetos é feita através de um projeto *template*. As atividades e associações do projeto *template* serão exportadas para o software comercial. Além disso, as informações necessárias para realizar a validação do projeto de acordo com as

restrições definidas pelo modelo integrado (tais como a diferenciação de atividades produtivas, gerenciais e gerenciais de apoio) serão exportadas para campos customizados.

A Figura 29 apresenta a interface de exportação de grupos de atividades (definidas no SPIT) para a geração de *templates* de projetos para a ferramenta comercial de gerência de projetos utilizada neste trabalho. A interface apresenta campos selecionar o conjunto de atividades que serão exportadas para um determinado projeto. A lista que contém os grupos de atividades é agrupada por fases de projeto. Para realizar a operação de exportação de um grupo de atividades para a ferramenta comercial de gestão de projetos, deve-se obrigatoriamente especificar o nome do projeto, selecionar pelo menos um grupo de atividade e clicar no botão "Atualizar".

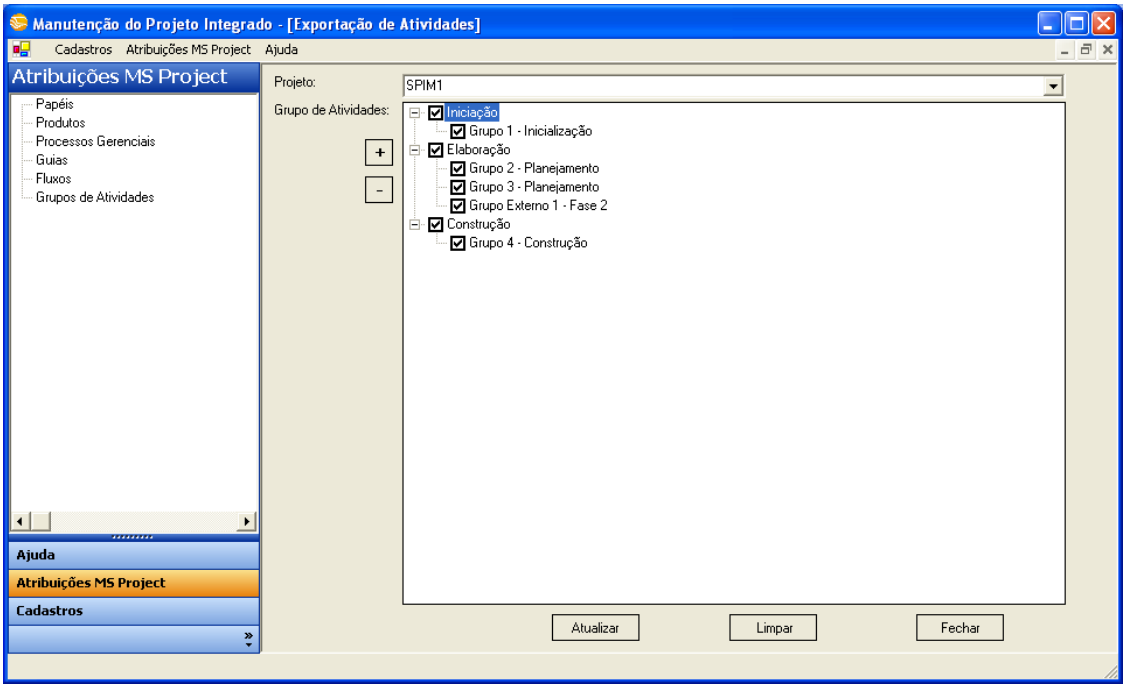


Figura 29. Tela para exportação de atividades

7.3.2.3.2 EXPORTAÇÃO DE CAMPOS CUSTOMIZADOS

Outra forma de realizar a exportação das informações cadastradas no SPIT para o software comercial é selecionar especificamente quais as informações que serão exportadas para campos customizados de um projeto específico (Figura 30).

Dessa forma, o módulo *BackOffice* do SPIT permite que sejam exportadas informações sobre papéis, produtos de trabalho, processos gerenciais do PMBOK, *guidances* e fluxos de trabalho do RUP.

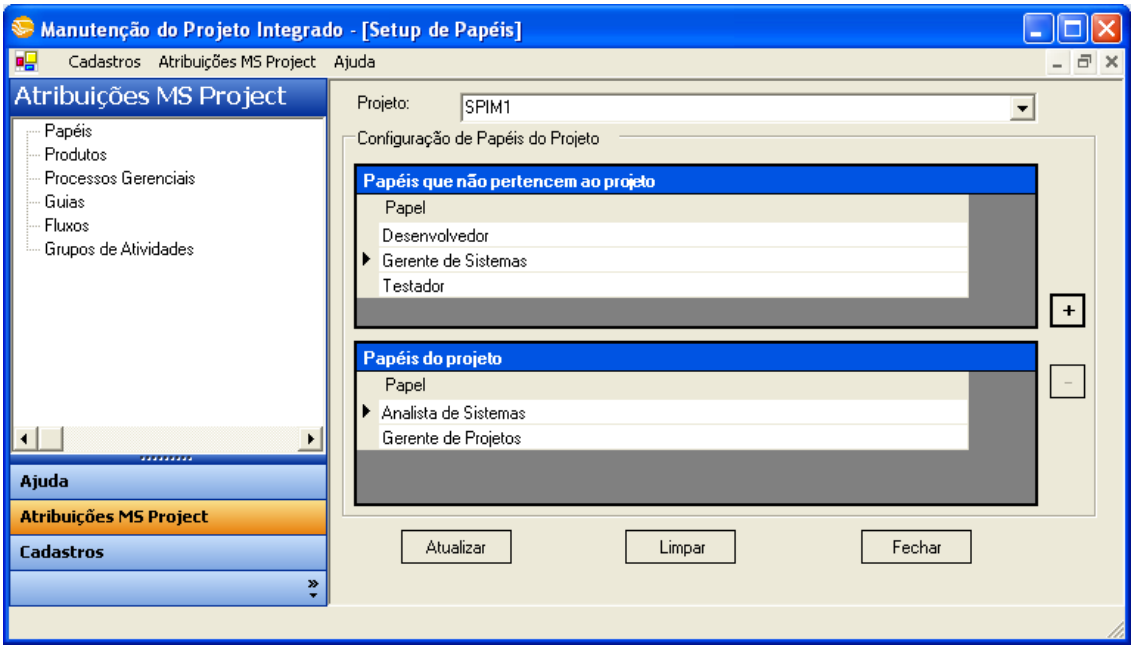


Figura 30. Tela para exportação de papéis

7.4 CONSIDERAÇÕES SOBRE O CAPÍTULO

Este capítulo destacou aspectos referentes à implementação do protótipo desenvolvido para demonstrar os conceitos propostos pelo modelo integrado SPIM, descrevendo as ferramentas utilizadas no seu desenvolvimento bem como a arquitetura da solução proposta. Foram apresentadas as interfaces do protótipo, com suas características e funcionalidades. Também, foram ressaltados os aspectos referentes à modelagem da solução.

No próximo capítulo serão apresentados os resultados de uma avaliação realizada com gerentes de projetos de software utilizando o protótipo SPIT.

8 AVALIAÇÃO DO MODELO

Neste capítulo apresenta-se a metodologia de pesquisa adotada neste trabalho. Posteriormente, discorre-se sobre a pesquisa qualitativa realizada sobre a utilização do modelo SPIM em empresas de desenvolvimento de software. Além disso, apresentam-se os resultados obtidos e as conclusões acerca destas informações.

8.1 METODOLOGIA DE PESQUISA

O conhecimento científico exige formulações exatas e claras, pois requer que estas sejam verificadas antes de serem aceitas como verdadeiras [CER02]. A ciência é o conhecimento racional, sistemático, exato e verificável da realidade. Assim, o conhecimento científico depende de investigação metódica da realidade, por isso, emprega procedimentos e técnicas para alcançar resultados.

De acordo com [WOH02], o método científico normalmente é definido como quantitativo ou qualitativo em função do tipo de dados coletados. Segundo [WIL93], a pesquisa quantitativa é associada a estudos positivistas confirmatórios enquanto que a pesquisa qualitativa é geralmente associada à pesquisa exploratória interpretativa.

O estudo positivista segue o paradigma hipotético-dedutivo, onde se assume que a realidade é objetiva, transcendendo a perspectiva individual, e é expressa por regularidades estatísticas observáveis. A pesquisa quantitativa é apropriada quando existe a possibilidade de medidas quantificáveis de variáveis e inferências a partir de amostras de uma população. Esta estratégia é particularmente adequada para o teste e validação de teorias baseada na manipulação de variáveis. Além disso, os dados quantitativos permitem comparações e análises estatísticas. Normalmente os estudos quantitativos são conduzidos na forma de experimentos controlados, de laboratório, uso em larga escala de simulação ou por meio da coleta de dados em estudos de caso.

A pesquisa interpretativa, por sua vez, segue o paradigma holístico-indutivo, pois busca compreender o fenômeno a partir dos próprios dados, das referências fornecidas pela população estudada e dos significados atribuídos ao fenômeno pela população [WIL93]. Assim, a pesquisa qualitativa se caracteriza, principalmente, pela ausência de medidas

numéricas e análises estatísticas, examinando aspectos subjetivos do tema em estudo. Esta estratégia é particularmente adequada para o desenvolvimento de novas teorias em áreas ou abordagens ainda não consolidadas. Ainda, segundo [WOH02], a pesquisa qualitativa estuda o objeto de análise em seu ambiente natural.

Os autores [KAP88] destacam que a ênfase exclusiva em aspectos quantitativos restringe a análise do objeto estudado estritamente a uma esfera técnica. Dessa forma, esta estratégia normalmente não é adequada quando objetiva-se explicar os resultados da aplicação de teorias e ferramentas que envolvem aspectos comportamentais nas organizações.

A definição do tipo de pesquisa é um fator determinante para a continuidade do processo de investigação, representado no desenho de pesquisa. Segundo [BAB05], os objetivos gerais que permeiam os propósitos para a realização de uma pesquisa são os seguintes:

- **Descritivos:** quando o objetivo é descobrir a distribuição de certos traços e atributos; de modo que o pesquisador não se preocupa com o porquê da distribuição observada existir, mas com o que ela é.
- **Explicativos:** quando o objetivo é de fazer asserções explicativas sobre porque ocorre um fenômeno e em que condições este ocorre. Centra-se em responder as causas dos fenômenos nos eventos físicos ou sociais do objeto de estudo.
- **Exploratórios:** fornece um "mecanismo de busca" para o pesquisador que está começando a investigação de algum tema. Isto ocorre quando a revisão da literatura mostra que ainda não existem teorias consistentes e na área estudada, somente existindo fragmentos teóricos ou idéias vagamente relacionadas com o problema de estudo. Dessa forma, um estudo exploratório suscita novas possibilidades, que podem ser mais detalhadas em outros tipos de pesquisas.

De acordo com [PAT86], dependendo da finalidade da avaliação e das condições para a investigação empírica pode-se adotar diferentes métodos para a condução da pesquisa, dentre elas a abordagem experimental e a não experimental. Os estudos experimentais se referem a uma pesquisa onde variáveis independentes (supostas causas) são manipuladas objetivando-se analisar as conseqüências obtidas sobre variáveis dependentes (supostos efeitos). Em contrapartida, os estudos não experimentais são classificados de acordo sua dimensão temporal: transversais ou longitudinais. Segundo [BAB05], opta-se pelo estudo transversal quando a investigação se compromete a identificar e explicar uma ou mais variáveis no limite de um determinado espaço de tempo. O estudo longitudinal, em contraste,

se compromete a investigar a evolução ou as transformações, ou ainda as mudanças ocorridas em determinadas variáveis no curso de diferentes espaços de tempo.

8.2 APLICAÇÃO DO MODELO

Frente às observações aqui apresentadas, a fim de avaliar os conceitos advindos do modelo integrado SPIM no que diz respeito à sua aceitação e aplicabilidade foi realizado um Survey qualitativo que envolveu doze gerentes de projetos que trabalham para um total de nove empresas de desenvolvimento de software no Brasil. Este estudo visou levantar dados iniciais que orientassem desenvolvimentos futuros. Além disso, a amostra foi escolhida de acordo com a acessibilidade e conveniência, portanto caracteriza-se como sendo uma amostra não-probabilística. Assim, os gerentes de projetos foram selecionados a participar desta pesquisa através de convites enviados por email.

Nesta pesquisa, a aquisição dos dados foi realizada por meio de um questionário (Apêndice C) elaborado em conformidade com [REA05]. Assim, a elaboração do questionário foi realizada de acordo com os seguintes passos:

- coleta de dados preliminares a respeito do tema e da população alvo da pesquisa;
- discussão em grupo sobre as questões;
- elaboração do rascunho do questionário;
- realização de um pré-teste;
- revisão do instrumento baseado nas observações obtidas no pré-teste; e
- delineamento do questionário final.

Na primeira etapa realizou-se o levantamento bibliográfico e o estudo do referencial teórico que permitiu aprofundar os conhecimentos sobre os processos de desenvolvimento de software e os modelos de gerência de projetos. Além disso, foi definida a população alvo desta pesquisa (gerentes de projetos de software, neste caso). Em seguida, foram realizadas reuniões entre o pesquisador e o professor orientador para o levantamento das questões e a estruturação do roteiro de entrevistas. Estas reuniões resultaram na elaboração de um rascunho do questionário necessário para a aquisição dos dados desta pesquisa. Posteriormente, foi realizada a validação de face e conteúdo do protocolo de avaliação da pesquisa por um pesquisador sênior. Com base nesta validação o protocolo sofreu pequenas correções. A seguir foi realizado o pré-teste com um gerente de projetos, onde foi possível realizar os últimos ajustes no roteiro de entrevistas. A versão final do protocolo de pesquisa

contendo o roteiro de entrevistas e descrevendo os procedimentos seguidos encontra-se no Apêndice C deste volume.

O questionário desenvolvido nesta pesquisa é constituído por um conjunto de 33 questões, as 8 primeiras capturam o perfil das empresas enquanto que outras 10 perguntas foram dirigidas especificamente aos gerentes de projetos participantes. As questões restantes abordaram a percepção das contribuições do modelo de SPIM do ponto de vista dos gerentes de projetos.

A aquisição dos dados deste questionário foi realizada durante o mês de novembro de 2007. De maneira a respeitar a disponibilidade de horários dos entrevistados, cada indivíduo participante da pesquisa respondeu separadamente o questionário em seu ambiente de trabalho. Dessa forma, todos os participantes receberam um breve treinamento no modelo SPIM e, posteriormente, foi dada a oportunidade para fazer as primeiras perguntas sobre o trabalho proposto. Em seguida, eles foram apresentados a uma mesma descrição sobre um projeto (Apêndice D) e foram convidados a realizar o planejamento do projeto correspondente usando o módulo *Add-in* do SPIT. Depois disso eles responderam ao questionário, cujos resultados foram utilizados para refinar o modelo. Objetivando-se evitar possíveis distorções nos resultados obtidos, tanto na fase de experimentação do SPIT quanto na fase de resolução do questionário não houve qualquer interação com o entrevistador. Estes resultados são apresentados na próxima seção.

8.3 RESULTADOS DA PESQUISA

As empresas envolvidas estão distribuídas em diferentes segmentos da tecnologia da informação (três fábricas de software, duas companhias de desenvolvimento de tecnologia móvel, duas companhias de desenvolvimento web, uma instituição governamental e um centro de pesquisa e desenvolvimento). Juntas estas empresas somam mais de 1620 profissionais relacionados a área de tecnologia da informação (TI), os quais 90 são gerentes de projetos e 823 são desenvolvedores de software.

Uma análise inicial das perguntas relacionadas ao perfil das empresas revela que elas apresentam uma média de experiência em TI superior a 13 anos. A descrição detalhada do perfil de cada empresa é apresentada no Apêndice E deste volume. Assim, como podemos ver na Tabela 28, existe uma grande disparidade no tamanho e na finalidade entre as empresas, resultando em um valor de desvio padrão de 155,34, consistindo em uma amostra adequada para nossa análise. Além disso, quase 67% destas empresas adotam algum tipo de processo de

desenvolvimento de software baseado no RUP. Também, todas as empresas adotam (total ou parcialmente) os conceitos encontrados no PMBOK Guide nos seus projetos. Isto confirma outro aspecto do questionário que revela que 100% dos indivíduos receberam treinamento de gestão de projetos e, assim, indica que todos os gestores entrevistados trabalham em ambientes que permitem a utilização das idéias propostas pelo modelo SPIM.

Tabela 28: Alguns critérios relativos ao perfil das nove empresas.

Critérios para as empresas	Média	Desvio Padrão
Tempo de mercado na área de TI (anos)	13,33	8,25
Número de profissionais relacionados à área de TI	140,22	155,34
Número de gerentes de projetos	10,00	8,06
Número de desenvolvedores	91,44	125,85
--	--	--
Processo de software é baseado no RUP	67%	-
Gerenciamento de projetos é baseado no PMBOK	100%	-

Sobre a análise dos resultados obtidos a partir das perguntas relacionadas ao perfil dos indivíduos entrevistados, pode-se observar que estes obtiveram uma média de 12,25 anos de experiência profissional (mín.=7; máx=20). Além da referida experiência profissional dos entrevistados, a média de experiência em gestão de projetos foi de 5,04 anos, variando de 1 a 12 anos (3,43 foi o desvio padrão e a média da amostra foi superior a 7 anos). Estes dados traduzem a vasta gama de experiência dos entrevistados relativo à gestão de projetos. Junto com a informação relativa ao treinamento de gestão de projetos, estes dados parecem explicar o fato que 58% da amostra declararam a sua experiência em gerenciamento de projetos como "avançado" enquanto que os 42% restantes declararam-na como sendo "moderado." Além disso, 83% dos indivíduos classificaram os seus conhecimentos sobre o RUP como "moderado" ou "avançado".

A análise das questões do modelo SPIM será iniciada com uma avaliação sobre os benefícios diretos identificados pelos entrevistados em realizar o planejamento integrado de atividades gerenciais e produtivas em um projeto de software. Os resultados são apresentados na Tabela 29. De acordo com a segunda e a última linha desta tabela, todos os gestores acharam que o planejamento integrado permite a identificação das dependências (que não são facilmente percebidas) entre as atividades gerenciais de apoio e as atividades de produção, evitando as freqüentes distorções no planejamento dos projetos devido a desconsideração de que as atividades gerenciais de apoio utilizam recursos não alocados diretamente ao projeto de software. Além disso, alguns entrevistados mencionaram a possibilidade de manter a conformidade com os processos organizacionais e de desenvolvimento, bem como o auxílio

na identificação prévia dos tempos das demandas de algumas atividades (evitando impactos nas atividades produtivas dependentes).

Tabela 29: Benefícios percebidos em fazer o planejamento integrado de atividades gerenciais e produtivas.

Questão	Média
Redução do tempo durante o processo elaboração do projeto.	58%
Identificação das dependências entre as atividades gerenciais de apoio e as atividades do projeto de software.	100%
Identificação e mensuração dos custos indiretos do projeto, devido as atividades gerenciais de apoio.	67%
Capacidade de ter acesso a informações dos fluxos empresariais	40%
A capacidade de evitar distorções ao planejar o projeto quando atividades gerenciais de apoio são envolvidas.	100%

Um subconjunto de oito de todas as regras também foi avaliado pelos entrevistados (Tabela 30). Os gestores avaliaram cada regra de acordo de acordo com a seguinte pontuação: 1-nenhum, 2-baixo, 3-moderado, 4-alto e 5-muito alto.

Tabela 30: Conjunto de regras de validação do modelo SPIM e a sua avaliação pelos gestores.

	Restrições Verificadas pelo Modelo SPIM	Média	Desvio Padrão
1	Uma atividade não pode criar, modificar ou consultar um mesmo artefato num dado momento. Estas operações devem ser efetuadas por atividades distintas.	3,75	0,75
2	As atividades gerencial e gerencial de apoio não podem produzir ou modificar produtos de trabalho produtivos, mas somente produtos de trabalho gerenciais. Entretanto, elas ainda podem consultar produtos de trabalho produtivos.	4,42	0,79
3	A atividade produtiva não pode produzir ou modificar produtos de trabalho gerenciais, somente produtos de trabalho produtivos. Entretanto, ela ainda pode consultar produtos de trabalho gerenciais.	4,42	0,79
4	Uma atividade somente pode atualizar ou consultar um produto de trabalho que já tenha sido criado por uma atividade antecessora.	4,08	0,90
5	O papel do *stakeholder* envolvido deve ser compatível com o tipo de atividade (gerencial ou produtivo).	4,25	0,87
6	O valor de informar o tipo de guidance relacionado, se produtivo ou administrativo, para cada atividade.	3,83	1,03
7	As atividades gerencial e gerencial de apoio devem ter pelo menos um papel gerencial como um de seus papéis.	4,50	0,67
8	A atividade produtiva deve ter pelo menos um papel produtivo como um de seus papéis.	4,42	0,67

A média geral para as regras selecionadas foi de 4,20, o que indica um ótimo nível de aceitação das restrições propostas no modelo SPIM. Além disso, este número aumenta 4,47 se consideramos apenas os gerentes que possuem mais de 7 anos de experiência de gestão de projetos.

Apesar da alta pontuação, as regras #1 e #6 foram consideradas como as que menos trouxeram benefícios entre todas as regras (respectivamente, 3,75 e 3,83, valores próximos da pontuação considerada como sendo alto grau de benefício). Além disso, esta média aumenta para 4,25 considerando apenas os entrevistados que possuem experiência entre oito e doze anos em gestão de projetos. Em contrapartida, as regras de número #2, #3, #7 e #8 obtiveram a maior média dentre as restrições analisadas, próximo de 4,5 pontos, sendo consideradas como as regras que mais favoritas entre os entrevistados.

As restrições que apresentaram menor disparidade entre as respostas foram as regras #7 e #8, resultando em um valor de desvio padrão de 0,67. A restrição que apresentou maior divergência entre as opiniões foi a regra #6, que está relacionada ao tipo de *guidance* da atividade. Estes números revelam uma relativa importância de todos os assuntos que devem ser considerados ao se realizar a verificação do plano do projeto de acordo com o modelo SPIM. Também, estas informações indicam quais itens exigem preocupação mais específica.

A visibilidade das atividades gerenciais de apoio em conjunto com as atividades do projeto do software (tanto produtivas quanto gerenciais), também foi identificada como um forte benefício do SPIM por 58,33% dos entrevistados. Quando questionados se eles concordavam sobre a natureza distinta dos três tipos de atividades, todos os respondentes responderam estar de acordo. Além disso, todos os entrevistados também concordaram que a dificuldade de identificar as atividades gerenciais de apoio durante o planejamento do projeto pode afetar negativamente o projeto.

Posteriormente, os entrevistados foram questionados sobre os pontos fortes observados no modelo integrado SPIM. Desta forma, todos os gerentes entrevistados apontaram pelo menos um ponto forte adicionado pela solução proposta, sendo que alguns gerentes mencionaram a facilidade de validar algumas das regras diretamente pela ferramenta de planejamento de projeto como um fator importante no SPIM. A visibilidade das atividades gerenciais de apoio em conjunto com as atividades do projeto de software também foi identificada como sendo um ponto forte do modelo SPIM. Dentre os pontos fracos observados no modelo integrado SPIM, alguns entrevistados mencionaram que a quantidade de

informações necessárias para cadastrar no cronograma poderia retardar o processo de planejamento de projetos.

Questionados se concordavam ser adequada a distinção proposta nesta pesquisa entre as atividades produtivas e gerenciais de um projeto de software com as atividades gerenciais de apoio dos demais departamentos da organização, todos os entrevistados responderam positivamente.

Relativo à questão 6 do questionário, cerca de 60% dos entrevistados salientaram que muitas vezes o gerente de projetos somente percebe a necessidade de ter solicitado anteriormente uma informação de outro departamento da empresa no momento de executar uma determinada atividade do projeto que depende deste outro departamento (por exemplo: a aquisição de equipamento ou a contratação de um novo desenvolvedor). Estas observações corroboram com a intuição desta pesquisa ao desenvolver este modelo integrado.

Questionados se concordavam que as atividades gerenciais de apoio não são exclusivas de um projeto em especial, mas de um fluxo comum da empresa, compartilhado pelos projetos em andamento e que utiliza recursos não alocados diretamente ao projeto de software, cerca de 80% concordaram totalmente com o fato de que esta lógica é benéfica na definição do planejamento do projeto. Deste percentual participam todos os entrevistados que possuem entre oito a doze anos de experiência profissional em gestão de projetos. Em contrapartida, os outros 20% concordaram parcialmente com esta afirmação, salientando que esta lógica depende da estrutura organizacional da empresa (funcional, matricial, projetizada, etc.). Os gerentes de projetos que fizeram esta observação qualificaram seu conhecimento relacionado aos conceitos advindos do PMBOK Guide como sendo avançado. Isto indica que estas observações possuem uma boa fundamentação teórica por parte dos entrevistados. Além disso, estes entrevistados mencionaram que algumas vezes certos recursos de outros setores são alocados exclusivamente para um determinado projeto.

Como último ponto a ser analisado, todos os gerentes de projetos constataram que o modelo SPIM colaborou na identificação das dependências entre as atividades pertencentes ao fluxo de um projeto de software e aquelas pertencentes aos fluxos empresariais. Dessa forma, há evidências que o SPIM permite antecipar as necessidades advindas das áreas de apoio durante o planejamento do projeto de software.

Os resultados observados neste questionário reafirmam os benefícios que o modelo SPIM contribui para solucionar os problemas relacionados à definição inadequada de tarefas

(por exemplo, o aumento dos custos e atrasos nos prazos do projeto) ocasionadas pela a dificuldade de visualizar a interdependência entre os fluxos de trabalho da empresa e de um projeto de software específico.

8.4 CONSIDERAÇÕES SOBRE O CAPÍTULO

Neste capítulo foram apresentadas as análises realizadas sobre os resultados obtidos um Survey de caráter qualitativo com um grupo de gerentes de projetos que atuam em empresas de desenvolvimento de software. Dessa forma, objetivando-se avaliar os conceitos propostos pelo modelo integrado SPIM foi desenvolvido um questionário que é constituído por um conjunto de 18 perguntas relacionadas ao perfil das empresas e dos gerentes de projetos participantes. Além disso, oito perguntas foram formuladas objetivando coletar informações preferenciais dos entrevistados em relação ao modelo integrado SPIM.

Através da análise dos resultados obtidos para as perguntas relacionadas ao perfil das empresas que fazem parte desta amostra foi possível observar, entre outros pontos relevantes, que os gerentes de projetos participantes desta pesquisa atuam em um ambiente que permite a utilização dos conceitos propostos pelo modelo SPIM. Ainda, quando questionados se concordavam ser adequada a distinção proposta nesta pesquisa (entre as atividades produtivas e gerenciais de um projeto de software com as atividades gerenciais de apoio dos demais departamentos da organização) todos os entrevistados responderam positivamente.

Os resultados observados neste questionário constataram que o modelo SPIM contribui para solucionar os problemas relacionados a definição inadequada de tarefas (por exemplo, o aumento dos custos e atrasos nos prazos do projeto) ocasionados pela a dificuldade de identificar a interdependência dos fluxos de trabalho da empresa e do projeto de software. No próximo capítulo serão apresentadas as considerações finais deste trabalho e os trabalhos futuros nesta área de pesquisa.

9 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as considerações finais deste trabalho e ainda destaca os rumos para futuras pesquisas na área.

Este trabalho apresentou uma proposta para a integração dos principais conceitos sobre gerência de projetos e os processos de desenvolvimento de software. Inicialmente, identificou-se a importância das atividades de gerência de projetos durante um projeto de desenvolvimento de software. Em seguida, observou-se a carência existente no quesito de gestão de projetos nos processos de desenvolvimento de software atuais. Após uma análise individual de cada metamodelo base, foi proposto um modelo que cobre ambas as perspectivas em um único modelo integrado. Depois, foram analisados os resultados de uma série de entrevistas com gerentes de projeto experientes, baseado em um protótipo desenvolvido.

A análise de como o conhecimento sobre gerência de projetos permite aperfeiçoar os processos de desenvolvimento de software atuais torna possível o desenvolvimento de novas ferramentas capazes de suportar diferentes níveis de automatização no planejamento e na execução de atividades num projeto de software. Dessa forma, este trabalho contribui com algumas conclusões interessantes que reafirmam o objetivo de definir um modelo de apoio que auxilie os gerentes de projetos de software.

O protótipo desenvolvido implementa um conjunto de regras objetivando fornecer suporte ao gerente de projetos para lidar com diferentes fluxos de trabalho, variáveis de projeto, responsabilidades e tarefas. Mais uma vez, o objetivo é facilitar o trabalho do gerente de projetos, que tem de levar em conta cada vez mais elementos em suas decisões e com cada vez menos tempo.

Acredita-se ser possível estender estes metamodelos de integração com outros processos de desenvolvimento de software, porque, de acordo com [SOM95], os diferentes modelos de desenvolvimento de software compartilham atividades fundamentais, tais como a especificação, projeto, implementação, validação e evolução do software. Na seção seguinte são apresentados os trabalhos futuros identificados para esta pesquisa.

9.1 CONTRIBUIÇÕES

O presente trabalho apresenta uma proposta de modelo de integração entre a gerência de projetos e os processos de desenvolvimento de software para o planejamento de projetos de software, denominado SPIM. Em função da integração proposta, foram desenvolvidos dois metamodelos base de integração durante esta pesquisa: PMBOK+RUP e PMBOK+OPEN. O estudo da integração do PMBOK com estes dois processos distintos de desenvolvimento de software (RUP e OPEN) permitiu elaborar uma visão mais ampla de como a gerência de projetos pode positivamente contribuir no desenvolvimento de um produto de software. O modelo SPIM foi concebido considerando a complexidade de identificar as relações de dependência entre as atividades dos fluxos de trabalho empresariais e do fluxo de trabalho de um projeto de software particular. Mais do que isso, o modelo SPIM tem como objetivo apoiar o processo decisório do gerente de projetos permitindo o planejamento integrado de atividades gerenciais e técnicas. Ainda, a análise das classes e relacionamentos do SPIM originou dois conjuntos de restrições, implícitas e explícitas, para a avaliação de seus elementos.

Um protótipo foi desenvolvido para demonstrar os conceitos propostos pelo modelo integrado SPIM e pelo seu conjunto de restrições, fornecendo, desta forma, o suporte necessário para trabalhar com diferentes fluxos de trabalho, variáveis de projeto, responsabilidades e tarefas.

Objetivando-se avaliar os conceitos propostos pelo modelo integrado SPIM, foi realizado um Survey de caráter qualitativo com um grupo de doze gerentes de projetos que atuam setores de pesquisa ou desenvolvimento de software. Este estudo visou levantar dados iniciais que orientassem desenvolvimentos futuros.

9.1.1 PUBLICAÇÕES

O desenvolvimento desta pesquisa resultou na publicação e submissão de artigos em congressos nacionais e internacionais de sistemas de informação. Segue abaixo a listagem destes artigos:

- Um artigo foi publicado no SBSI 2006 - Simpósio Brasileiro de Sistemas de Informação. Em [ROS06], foi apresentado um estudo comparativo dos processos de desenvolvimento de software RUP, *Microsoft Solutions Framework* (MSF) [HUN05] e OPEN em relação ao *Software Process Engineering Metamodel*

Specification (SPEM) [OMG06], que permitiu identificar as classes e relacionamentos comuns entre estes metamodelos.

- Um artigo foi aceito para publicação no SBSI 2008 - Simpósio Brasileiro de Sistemas de Informação. Em [ROS08], foi apresentado o modelo SPIM, que integra os conceitos de gerência de projetos e dos processos de desenvolvimento de software, auxiliando no planejamento das atividades pertencentes a estas duas áreas de conhecimento. Também foi apresentado um conjunto de restrições para o modelo SPIM e um protótipo, denominado SPIT, que foi desenvolvido para demonstrar os conceitos propostos neste modelo de integração.
- Um artigo foi aceito para publicação no ICEIS 2008 – *International Conference on Enterprise Information Systems*. Em [CAL08], foi apresentada a análise dos resultados obtidos de uma pesquisa qualitativa sobre o modelo SPIM, realizada com nove empresas de desenvolvimento de software.

9.2 LIMITAÇÕES

A principal limitação deste estudo está associada à estratégia adotada para a avaliação dos conceitos advindos do modelo integrado SPIM. Este estudo visou levantar dados iniciais que orientassem desenvolvimentos futuros, dessa forma a amostra foi escolhida de acordo com a acessibilidade e conveniência. Esta abordagem resultou em um pequeno número de empresas e respondentes envolvidos na pesquisa realizada. Além disso, seria interessante que a pesquisa envolvesse projetos reais de desenvolvimento de software. Entretanto, infelizmente não foi possível realizar tal pesquisa dada à limitação imposta pelo cronograma deste trabalho.

Outra limitação importante deve-se ao fato que o modelo proposto SPIM implementa exclusivamente os conceitos advindos da integração do PMBOK com o RUP, restringindo sua aplicação em empresas que adotam processos baseados nestas propostas em seus projetos.

9.3 TRABALHOS FUTUROS

Durante o desenvolvimento desta pesquisa foram identificadas algumas questões que necessitam de maior desenvolvimento:

- formalização das restrições do metamodelo através da linguagem OCL;
- ampliação do estudo sobre a integração do processo de gerência com outros processos de desenvolvimento de software, tais como XP e MSF; e

- desenvolvimento de um protocolo para a avaliação do modelo proposto com empresas de software, utilizando o protótipo em projetos reais.

Atualmente, existem diversos artigos na literatura sobre diferentes integrações da gerência de projetos com os processos de desenvolvimento de software, tais como [CAL07] e [ALH07]. Entretanto, parece não haver estudos suficientes para suprir a carência no quesito de gerência de projetos destes processos. Dessa forma, a continuidade desta pesquisa indica novas contribuições para engenharia de software, melhorando nossa compreensão dos relacionamentos da gerência de projetos com os processos de desenvolvimento de software. Em um trabalho paralelo, um modelo multi-critérios para alocação de recursos já está sendo integrado ao protótipo.

REFERÊNCIAS BIBLIOGRÁFICAS

[ALH07] S. S. ALHIR. “Integrating the Project Management Body of Knowledge (PMBOK) Guide and the Unified Process (UP)”. Capturado em: http://home.comcast.net/~salhir/IntegratingThePMBOKGuideTheUP.PDF, Junho 2007.

[BAB05] E. BABBIE. “Métodos de pesquisas de survey”. Belo Horizonte: Ed. UFMG, 2005, 519p.

[BEC04] K. BECK. “Extreme Programming Explained: Embrace Change”. Boston: Addison-Wesley, 2ª edição, 2004, 224p.

[BEL96] W. BELASSI, O.I. TUKEL. “A new framework for determining critical success/failure factors in project”. International Journal of Project Management. Oxford: Elsevier Pergamon, vol. 14, June 1996, pp. 141–51.

[BEN06] A. BENCOMO. “Extending the RUP”. Capturado em: http://www.ibm.com/developerworks/rational/library/05/323_extrup1/index.html, Novembro 2006.

[BOO00] G. BOOCH, J. RUMBAUGH, I. JACOBSON. “UML, guia do usuário”. Rio de Janeiro: Elsevier, 2000, 472 p.

[BUR01] R. BURKE. “Project Management: Planning and Control Techniques”. Chichester: John Wiley e Sons, 3ª edição, 2001, 356p.

[CAL07] D. CALLEGARI, R. BASTOS. “Project Management and Software Development Processes: Integrating RUP and PMBOK”. In: ICSEM - International Conference on Systems Engineering and Modeling, Israel, Haifa, March 2007. Piscataway, NJ, USA: IEEE, 2007. p. 1-8.

[CAL08] D. CALLEGARI, M. ROSITO, M. BLOIS, R. BASTOS. “An Integrated Model for Managerial and Productive Activities in Software Development”. In: ICEIS - 10th International Conference on Enterprise Information Systems, Spain, Barcelona, June 2008, 8p. "Aceito para publicação".

[CER02] A. L. CERVO, P. A. BERVIAN. “Metodologia científica”. São Paulo: Prentice Hall, 2002, 5ª edição, 242p.

[CHA06] S. CHARBONNEAU. “Software Project Management - A Mapping between RUP and the PMBOK”. Capturado em: http://www.ibm.com/developerworks/rational/library/4721.html, Novembro 2006.

[CHA03] C. CHATFIELD, T. JOHNSON. “Microsoft Office Project 2003 Step by Step”. Redmond: Microsoft Press, 2003, 560p.

[COO02] T. COOKE-DAVIES. “The real success factors on projects”. International Journal of Project Management. Oxford : Elsevier, vol. 20, April 2002, pp. 185–90.

[DER99] J. C. DERNIAME, B. A. KABA, B. WARBOYS. “Software Process: Principles, Methodology, and Technology”. Berlin: Springer-Verlag, 1999, 307 p.

[GOB87] D.H. GOBELI, E.W. LARSON. “Relative effectiveness of different project structures”. Project Management Journal. New York: Wiley & Sons, vol. 18, June 1987, pp. 81–5.

[GRA97] I. GRAHAM, B. HENDERSON-SELLERS, H. YOUNESSI. “The OPEN Process Specification”. New York: ACM Press, 1997, 314 p.

[HEN00] B. HENDERSON-SELLERS, R. DUÉ, I. GRAHAM, G. COLLINS. “Third generation OO processes: a critique of RUP and OPEN from a project management perspective”, In: Seventh Asia-Pacific Software Engineering Conference, Singapore, December 2000. Washington, USA: IEEE Computer Society, 2000, pp. 428-435.

[HEN01] B. HENDERSON-SELLERS, G. COLLINS, R. DUÉ, I.M. GRAHAM. “A Qualitative Comparison of Two Processes for Object-Oriented Software Development”. Information and Software Technology. United Kingdom: Elsevier Science, vol. 43, November 2001, pp. 705-724.

[HUN05] R. HUNDHAUSEN. “Working with Microsoft Visual Studio 2005 Team System”. Redmond: Microsoft Press, 2005, 336 p.

[JAC99] I. JACOBSON, G. BOOCH, J. RUMBAUGH. “The Unified Software Development Process”. Reading, MA : Addison Wesley, 1999, 463 p.

[KAP88] B. KAPLAN, D. DUCHON. “Combining qualitative and quantitative methods in information systems research: a case study”. Management Information Systems Quarterly. USA: University of Minnesota, vol. 12, December 1988, pp. 571-586.

[KER00] H. KERZNER. “Applied project management: best practices on implementation”. New York: Wiley e Sons, 2000, 534p.

[KRU00] P. KRUCHTEN. “The Rational Unified Process: An Introduction”. Boston: Addison-Wesley, 2000, 298 p.

[MIC07] MICROSOFT CORPORATION. “SQL Server 2005 Express Edition”. Capturado em: http://msdn2.microsoft.com/en-us/sqlserver/aa336346.aspx, Junho 2007.

[OGC07] OFFICE OF GOVERNMENT COMMERCE (OGC). “Introduction to PRINCE2”. Capturado em: http://www.ogc.gov.uk, Março 2007.

[OMG06] OBJECT MANAGEMENT GROUP (OMG). “Software Process Engineering Metamodel (SPEM)”. Capturado em: http://www.omg.org/technology/documents/formal/spem.htm, Outubro 2006.

[OMG07] OBJECT MANAGEMENT GROUP (OMG). “Unified Modeling Language Specification”. Capturado em: http://www.uml.org/, Janeiro 2007.

[OPE02] OPEN CONSORTIUM. “OPEN Home Page”. Capturado em: http://www.open.org.au, Fevereiro, 2007.

[OPF07] OPEN PROCESS FRAMEWORK (OPFRO). “OPFRO Home Page”. Capturado em: http://www.opfro.org, Fevereiro 2007.

[PAT86] M. Q. PATTON. “Qualitative evaluation methods”. Beverly Hills: Sage Publications, 1986, 381 p.

[PEP06] PROCESS ENGINEERING PROCESS (PEP). “A RUP Adoption Process”. Capturado em: http://www-128.ibm.com/developerworks/rational/library/6001.html, Agosto 2006.

[PIN87] J.K. PINTO, D.P. SLEVIN. “Critical factors in successful project implementation”. IEEE Transactions on Engineering Management. New York: IEEE Engineering Management Society, vol. 34, February 1987, pp. 22–7.

[PMI00] PROJECT MANAGEMENT INSTITUTE (PMI). “PMBOK Guide: A Guide to the Project Management Body of Knowledge”. Newtown Square: Project Management Institute, 3ª edição, 2000, 216 p.

[PMI06] PROJECT MANAGEMENT INSTITUTE (PMI). “PMI Home Site”. Capturado em: http://www.pmi.org/info/default.asp, Novembro 2006.

[PRE01] R. PRESSMAN. “Software Engineering: a practitioner's approach”. Boston: McGraw-Hill, 5ª edição, 2001, 888 p.

[RAT06] RATIONAL SOFTWARE CORPORATION. “Rational Unified Process: Best Practices for Software Development Teams”. Capturado em: http://www.rational.com/products/rup/whitepapers.jsp, Novembro 2006.

[REA05] L. M. REA, R. PARKER. “Designing and Conducting Survey Research: A Comprehensive Guide”. San Francisco: Jossey-Bass, 2005, 304 p.

[REH07] A. REHMAN, R. HUSSAIN. “Software Project Management Methodologies/Frameworks Dynamics ‘A Comparative Approach’ ”. In: 7th International Conference on Information and Emerging Technologies, Karachi, Pakistan, July 2007, pp. 1-5.

[ROS06] M. ROSITO, D. CALLEGARI, R. BASTOS. “Metamodelos de processos de desenvolvimento de software: Um estudo comparativo”. In: III Simpósio Brasileiro de Sistemas de Informação. Curitiba : UnicenP, Novembro 2006, 8p.

[ROS08] M. ROSITO, D. CALLEGARI, R. BASTOS. “Gerência de Projetos e Processos de Desenvolvimento de Software: uma proposta de integração”. In: IV Simpósio Brasileiro de Sistemas de Informação. Rio de Janeiro: Unirio, Abril 2008, 12p. "Aceito para publicação".

[SCH02] K. SCHWALBE. “Information Technology Project Management”. Austrália: Course Technology, 2ª edição, 2002, 561 p.

[SOM95] I. SOMMERVILLE. “Software engineering”. Harlow: Addison-Wesley, 5ª edição, 1995, 742 p.

[WAR99] J. WARMER, A. KLEPPE. “The Object Constraint Language - Precise Modeling with UML”. Massachusetts: Addison Wesley, 1999, 112 p.

[WIL93] B. M. WILDEMUTH. “Post-positivist research: two examples of methodological pluralism”. Chicago: The Library Quarterly, vol. 63, October 1993, pp. 450-468.

[WIL02] M. WILLIAMS. “Microsoft Visual C# .NET (Core Reference)”. Redmond: Microsoft Press, 2002, 784 p

[WOH02] C. WOHLIN, P. RUNESON, M. HÖST, B. REGNELL, A. WESSLÉN. “Experimentation in software engineering: an introduction”. Boston: Kluwer Academic Publishers, 2002, 204 p.

APÊNDICE A – Modelos de casos de uso do módulo *add-in* do SPIT

A.1 MODELOS DE CASO DE USO DO MÓDULO *ADD-IN* DO SPIT

Este apêndice contém a descrição detalhada dos modelos de caso de uso do módulo Add-in do protótipo SPIT, apresentados no capítulo 7.

A.1.1 CASO DE USO: VALIDAR PROJETO

A definição do modelo SPIM resultou em um conjunto de restrições identificadas para apoiar a consistência do modelo proposto. A validação destas restrições é possível devido à adição de campos customizados a uma ferramenta comercial de gestão de projetos. Dessa forma, na Tabela 31 é apresentada a especificação deste caso de uso e, posteriormente, é descrito o cenário de sucesso e extensões para este caso de uso:

Tabela 31: Detalhes caso de uso Validar Projeto.

Caso de Uso Validar Projeto	
Identificação: UCS1.	
Finalidade: Validar o projeto de software de acordo com os conceitos propostos pelo modelo de integração SPIM.	
Ator: Gerente de Projetos.	
Pré-Condições: O Gerente de Projetos deve ter gerado um *template* que contenha todos os campos customizados necessários para a validação do projeto.	
Pós-Condições: Mensagem contendo uma lista não conformidades com modelo SPIM.	
Cenário de Sucesso Principal	
1.	O Gerente de Projetos cadastra as atividades e recursos para um projeto.
2.	O Gerente de Projetos seleciona, para cada atividade ou recurso, as informações necessárias para os campos customizados que são necessários para a validação de acordo com o modelo SPIM.
3.	O gerente de projetos preenche os atributos acima e seleciona a opção "Validar Projeto" do *Add-in*.
4.	O sistema realiza a validação dos dados informados no passo 2.
5.	O sistema exibe uma mensagem informando as não conformidades com o modelo SPIM encontradas no projeto.
Extensões	
3a.	Não foi possível carregar as informações de validação do SPIM
3a1.	O sistema informa sobre o erro do *Add-in*.
3a2.	O sistema cancela a operação.

A.1.2 CASO DE USO: RE-CONFIGURAR CRONOGRAMA

O modelo SPIM pretende colaborar na identificação das dependências entre as atividades pertencentes ao fluxo de um projeto de software e aquelas pertencentes aos fluxos empresariais. Assim, o modelo proposto pretende auxiliar o gerente de projetos a antecipar as necessidades advindas das áreas de apoio durante o planejamento do projeto. Na Tabela 32 é apresentada a especificação deste caso de uso e, posteriormente, é descrito o cenário de sucesso e extensões para este caso de uso.

Tabela 32: Detalhes do caso de uso Re-Configurar Cronograma

Caso de Uso Re-Configurar Cronograma	
Identificação: UCS2.	
Finalidade: Re-configurar os prazos das atividades do projeto de maneira a auxiliar o gerente de projetos a antecipar as necessidades advindas das áreas de apoio durante o planejamento do projeto.	
Ator: Gerente de Projetos.	
Pré-Condições: • O Gerente de Projetos deve ter gerado um *template* que contenha todos os campos customizados necessários para a validação do projeto. • O Gerente de Projetos deve ter selecionado a opção "Associações com os Fluxos Empresariais" no SPIT.	
Pós-Condições: • Antecipação das datas de início das atividades gerenciais de apoio. • Re-configuração dos prazos das atividades do projeto de software que possuem relações de dependência com alguma atividade gerencial de apoio..	
Cenário de Sucesso Principal	
1.	O Gerente de Projetos cadastra as atividades e recursos para um determinado projeto, fazendo a distinção entre as atividades do projeto de software e as atividades gerenciais de apoio.
2.	O Gerente de Projetos seleciona a opção "Associações com os Fluxos Empresariais" no SPIT e clica no botão "Validar Projeto".
3.	O sistema realiza a validação dos dados informados pelo gerente de projetos nos passos 1 e 2.
4.	O sistema atualiza os prazos do projeto de acordo a antecipação da data de início das atividades gerenciais de apoio deste projeto.
Extensões	
3a.	Não foi possível carregar as informações de validação do SPIM
3a1.	O sistema informa sobre o erro do *Add-in*.
3a2.	O sistema cancela a operação.

A.1.3 CASO DE USO: ATUALIZAR ATIVIDADES

O cadastro de atividades ao cronograma, suas associações a recursos, fases, produtos e *stakeholders* do projeto, são descritas no caso de uso Atualizar Atividades. A seguir é apresentada a especificação deste caso de uso (Tabela 33) e, posteriormente, cada um dos cenários identificados é descrito em subseções.

Tabela 33: Detalhes do caso de uso Atualizar Atividades.

Caso de Uso Atualizar Atividades
Identificação: UCS3
Finalidade: Manutenção das informações correspondentes as atividades do cronograma, associando a recursos, fases, produtos e *stakeholders* do projeto
Ator: Gerente de Projetos
Pré-Condições: O Gerente de Projetos deve ter cadastrado no sistema um projeto
Pós-Condições: Definição das atividades que compõem o cronograma
Cenários: • Criar Atividade • Modificar Atividade • Apagar Atividade

A.1.3.1 CENÁRIO CRIAR ATIVIDADE

Este cenário inicia quando o gerente de projetos solicita a inclusão de uma atividade no sistema. Na Tabela 34 é descrito o cenário de sucesso e extensões para esta situação.

Tabela 34: Cenário de sucesso do cenário Criar Atividade.

Cenário de Sucesso Principal
1. O sistema solicita ao gerente de projetos o preenchimento dos seguintes atributos: • Informações gerais da atividade (nome, data de início, data de término, duração,definir se é um *milestone*,etc.); • Especificar Atividades Predecessoras (nome da atividade predecessora e tipo de dependência); • Especificar Recursos Relacionados (*stakeholder* ou recurso físico); • Especificar informações customizadas ▪ Identificador da atividade ▪ Tipo de atividade (produtiva, gerencial ou gerencial de apoio); ▪ Associar a um ou mais workflows do RUP; ▪ Associar a um processo gerencial do PMBOK; ▪ Associar a um tipo de guidance; ▪ Selecionar quais produtos de trabalho são afetados (criar, consultar, modificar) pela execução desta atividade;

2.	O gerente de projetos preenche os atributos acima e confirma a inclusão.
3.	O sistema realiza a inclusão dos dados informados pelo gerente de projetos no passo 2.
4.	O sistema exibe uma mensagem de confirmação informando que a inclusão da atividade foi efetivada com sucesso.
	Extensões
3a.	A ordem das atividades formou um ciclo.
3a1.	O sistema informa sobre o erro de ordenamento das atividades.
3a2.	O sistema cancela a operação.

A.1.3.2 CENÁRIO MODIFICAR ATIVIDADE

Este cenário inicia quando o gerente de projetos solicita a alteração de uma atividade. Na Tabela 35 é descrito o cenário de sucesso e extensões para esta situação.

Tabela 35: Cenário de sucesso do cenário Modificar Atividade.

	Cenário de Sucesso Principal
1.	O gerente de projetos seleciona uma única atividade.
2.	O gerente de projetos altera os dados desejados e confirma a alteração.
3.	O sistema realiza a alteração dos dados informados no passo 2.
4.	O sistema exibe uma mensagem de confirmação informando que a alteração da atividade foi efetivada com sucesso.
	Extensões
3a.	A ordem das atividades formou um ciclo.
3a1.	O sistema informa sobre o erro de ordenamento das atividades.
3a2.	O sistema cancela a operação.

A.1.3.3 CENÁRIO APAGAR ATIVIDADE

Este cenário inicia quando o gerente de projetos solicita a remoção de uma ou mais atividades. Na Tabela 36 é descrito o cenário de sucesso e extensões para esta situação.

Tabela 36: Cenário de sucesso do cenário Apagar Atividade.

	Cenário de Sucesso Principal
1.	O gerente de projetos seleciona quais atividades deseja remover e solicita a remoção.
2.	O sistema solicita a confirmação para a remoção.
3.	O gerente de projetos confirma a remoção.
4.	O sistema remove as atividades selecionadas.
5.	O sistema exibe uma mensagem informando que a remoção das atividades foi efetivada com sucesso.

A.1.4 CASO DE USO: ATUALIZAR RECURSOS

A manutenção das informações relativas aos recursos utilizados em um projeto é descrita no caso de uso Atualizar Recursos. Uma vez que os recursos de um projeto são classificados como pessoas ou recursos físicos, foram definidos os casos de uso Atualizar *Stakeholders* e Atualizar Recursos Físicos.

A seguir é apresentada a especificação deste caso de uso (conforme ilustrado na Tabela 37) e, posteriormente, cada um dos cenários identificados neste caso de uso é descrito em subseções.

Tabela 37: Detalhes do caso de uso Atualizar Recursos.

Caso de Uso Atualizar Recursos	
Identificação: UCS4	
Finalidade: Manutenção das informações correspondentes aos *stakeholders* e recursos físicos do projeto	
Ator: Gerente de Projetos	
Pré-Condições: O Gerente de Projetos deve ter cadastrado no sistema pelo menos uma Empresa	
Pós-Condições: Criação de *stakeholders* e recursos físicos que serão atribuídos as atividades do projeto	
Cenário de Sucesso Principal	
1.	O caso de uso inicia quando o gerente de projetos necessita fazer a manutenção (inclusão, alteração, exclusão ou consulta) de um recurso na empresa.
2.	O gerente de projetos informa o tipo de recurso que deseja fazer manutenção do dados: • **a.** Se for selecionado o tipo de recurso como sendo *Stakeholder*, ver caso de uso Atualizar *Stakeholders*; • **b.** Se for selecionado o tipo de recurso como sendo Recurso Físico, ver caso de uso Atualizar Recursos Físicos;

A.1.5 CASO DE USO: ATUALIZAR STAKEHOLDERS

O cadastro das informações sobre os *stakeholders* dos projetos são descritas no caso de uso Atualizar *Stakeholders*. A seguir é apresentada a especificação deste caso de uso (conforme ilustrado na Tabela 38) e, posteriormente, cada um dos cenários identificados é descrito em subseções.

Tabela 38: Detalhes do caso de uso Atualizar *Stakeholders*.

Caso de Uso Atualizar Stakeholders
Identificação: UCS5.
Finalidade: Manutenção das informações correspondentes aos *stakeholders*.
Ator: Gerente de Projetos.
Pré-Condições: O Gerente de Projetos deve ter cadastrado no sistema pelo menos uma Empresa.
Pós-Condições: Criação de *stakeholders* que serão atribuídos as atividades do projeto.
Cenários: • Criar *Stakeholder* • Modificar *Stakeholder* • Apagar *Stakeholder*

A.1.5.1 CENÁRIO CRIAR STAKEHOLDER

Este cenário inicia quando o gerente de projetos solicita a inclusão de um *stakeholder* no sistema. Na Tabela 39 é descrito o cenário de sucesso e extensões para esta situação.

Tabela 39: Cenário de sucesso do cenário Criar *Stakeholder*

Cenário de Sucesso Principal	
1.	O sistema solicita ao gerente de projetos o preenchimento dos seguintes atributos: • Informações gerais do *stakeholder* (nome, iniciais do recurso, email,etc.); • Especificar informações customizadas (atribuir o *stakeholder* a papéis do projeto). • Preencher a tabela de disponibilidade deste *stakeholder* (data de início,data de término, percentual de disponibilidade para este intervalo de tempo); • Informar quais serão as horas de início, intervalos e fim de expediente do calendário; • Definir tabelas de taxa de custo para este recurso;
2.	O gerente de projetos preenche os atributos acima e confirma a inclusão.
3.	O sistema realiza a inclusão dos dados informados pelo gerente de projetos no passo 2.
4.	O sistema exibe uma mensagem de confirmação informando que a inclusão da atividade foi efetivada com sucesso.
Extensões	
3a.	As datas da tabela de disponibilidade do *stakeholder* se sobrepõem
3a1.	O sistema informa sobre o erro de sobreposição de datas.
3a2.	O sistema cancela a operação.
3b.	O horário de expediente do *stakeholder* ultrapassa 24 horas.
3b1.	O sistema informa sobre o erro de definição de expediente.
3b2.	O sistema cancela a operação.

A.1.5.2 CENÁRIO MODIFICAR STAKEHOLDER

Este cenário inicia quando o gerente de projetos solicita a alteração cadastral de um *stakeholder*. Na Tabela 40 é descrito o cenário de sucesso e extensões para esta situação.

Tabela 40: Cenário de sucesso do cenário Modificar *Stakeholder*

Cenário de Sucesso Principal	
1.	O gerente de projetos seleciona um único *stakeholder*.
2.	O sistema solicita a alteração dos atributos listados no cenário Criar *Stakeholder.*
3.	O gerente de projetos altera os dados desejados e confirma a alteração.
4.	O sistema realiza a alteração dos dados informados no passo 2.
5.	O sistema exibe uma mensagem de confirmação informando que a alteração da empresa foi efetivada com sucesso.
Extensões	
3a.	As datas da tabela de disponibilidade do *stakeholder* se sobrepõem
3a1.	O sistema informa sobre o erro de sobreposição de datas.
3a2.	O sistema cancela a operação.
3b.	O horário de expediente do *stakeholder* ultrapassa 24 horas.
3b1.	O sistema informa sobre o erro de definição de expediente.
3b2.	O sistema cancela a operação.

A.1.5.3 CENÁRIO APAGAR STAKEHOLDER

Este cenário inicia quando o gerente de projetos solicita a remoção de um ou mais *stakeholders*. Na Tabela 41 é descrito o cenário de sucesso e extensões para esta situação:

Tabela 41: Cenário de sucesso do cenário Apagar *Stakeholder*

Cenário de Sucesso Principal	
1.	O gerente de projetos o recurso que deseja remover e solicita a remoção.
2.	O sistema solicita a confirmação para a remoção.
3.	O gerente de projetos confirma a remoção.
4.	O sistema remove o recurso selecionado e todas as entidades que estavam relacionadas a este *stakeholder*, tais como as atividades, perdem suas referência com este recurso.
5.	O sistema exibe uma mensagem informando que a remoção do *stakeholder* foi efetivada com sucesso.

A.1.6 CASO DE USO: ATUALIZAR RECURSOS FÍSICOS

O cadastro das informações sobre os recursos físicos dos projetos são descritas no caso de uso Atualizar Recursos Físico*s*. A seguir é apresentada a especificação deste caso de uso (Tabela 42) e depois cada um dos cenários identificados é descrito em subseções.

Tabela 42: Detalhes do caso de uso Atualizar Recursos Físicos.

Caso de Uso Atualizar Recursos Físicos
Identificação: UCS6.
Finalidade: Manutenção das informações correspondentes aos recursos físicos.
Ator: Gerente de Projetos.
Pré-Condições: O Gerente de Projetos deve ter cadastrado no sistema pelo menos uma Empresa.
Pós-Condições: Criação de recursos físicos que serão atribuídos as atividades do projeto.
Cenários: • Criar Recurso Físico • Atualizar Recurso Físico • Apagar Recurso Físico

A.1.6.1 CENÁRIO CRIAR RECURSO FÍSICO

Este cenário inicia quando o gerente de projetos solicita a inclusão de um recurso físico para uma empresa. Na Tabela 43 é descrito o cenário de sucesso e extensões para esta situação:

Tabela 43: Cenário de sucesso do cenário Criar Recurso Físico.

	Cenário de Sucesso Principal
1.	O sistema solicita ao gerente de projetos o preenchimento dos seguintes atributos: • Informações gerais do recurso físico (nome, iniciais do recurso, tipo de recurso,etc.); • Preencher a tabela de disponibilidade deste recurso físico (data de início,data de término, percentual de disponibilidade para este intervalo de tempo); • Informar quais serão as horas de início, intervalos e fim de expediente do calendário; • Definir tabelas de taxa de custo para este recurso;
2.	O gerente de projetos preenche os atributos acima e confirma a inclusão.
3.	O sistema realiza a inclusão dos dados informados pelo gerente de projetos no passo 2.
4.	O sistema exibe uma mensagem informando que a inclusão do recurso físico foi efetivada com sucesso.
5.	O gerente de projetos preenche os atributos acima e confirma a inclusão.
	Extensões
3a.	As datas da tabela de disponibilidade do recurso físico se sobrepõem
3a1.	O sistema informa sobre o erro de sobreposição de datas.
3a2.	O sistema cancela a operação.

A.1.6.2 CENÁRIO MODIFICAR RECURSO FÍSICO

Este cenário inicia quando o gerente de projetos solicita a alteração cadastral de um recurso físico. Na Tabela 44 é descrito o cenário de sucesso e extensões para esta situação:

Tabela 44: Cenário de sucesso do cenário Modificar Recurso Físico.

	Cenário de Sucesso Principal
1.	O gerente de projetos seleciona um único recurso físico.
2.	O sistema solicita a alteração dos atributos listados no cenário Criar Recurso Físico.
3.	O gerente de projetos altera os dados desejados e confirma a alteração.
4.	O sistema realiza a alteração dos dados informados no passo 3.
5.	O sistema exibe uma mensagem de confirmação informando que a alteração da empresa foi efetivada com sucesso.
	Extensões
3a.	As datas da tabela de disponibilidade do recurso físico se sobrepõem
3a1.	O sistema informa sobre o erro de sobreposição de datas.
3a2.	O sistema cancela a operação.

A.1.6.3 CENÁRIO APAGAR RECURSO FÍSICO

Este cenário inicia quando o gerente de projetos solicita a remoção de um ou mais recursos físico. Na Tabela 45 é descrito o cenário de sucesso e extensões para esta situação:

Tabela 45: Cenário de sucesso do cenário Apagar Recurso Físico.

	Cenário de Sucesso Principal
1.	O gerente de projetos seleciona o recurso físico que deseja remover e solicita a remoção.
2.	O sistema solicita a confirmação para a remoção.
3.	O gerente de projetos confirma a remoção.
4.	O sistema remove o recurso selecionado e todas as entidades que estavam relacionadas a este recurso físico, tais como as atividades, perdem suas referência com este recurso.
5.	O sistema exibe uma mensagem informando que a remoção do recurso físico foi efetivada com sucesso.

APÊNDICE B – Interfaces auxiliares do módulo *BackOffice* do SPIT

B.1 INTERFACES AUXILIARES DO MÓDULO *BACKOFFICE* DO SPIT

Este apêndice contém a descrição detalhada das interfaces auxiliares do módulo BackOffice do protótipo SPIT, apresentadas no capítulo 7.

B.1.1 MANUTENÇÃO DE EMPRESAS

A Figura 31 apresenta a interface de manutenção de empresas, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes às organizações que contém ou participam de projetos de software. A interface apresenta campos para definir o nome e a descrição da empresa, além de um campo para informar se é uma empresa prestadora de serviços. Além disso, uma lista das empresas já cadastradas no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

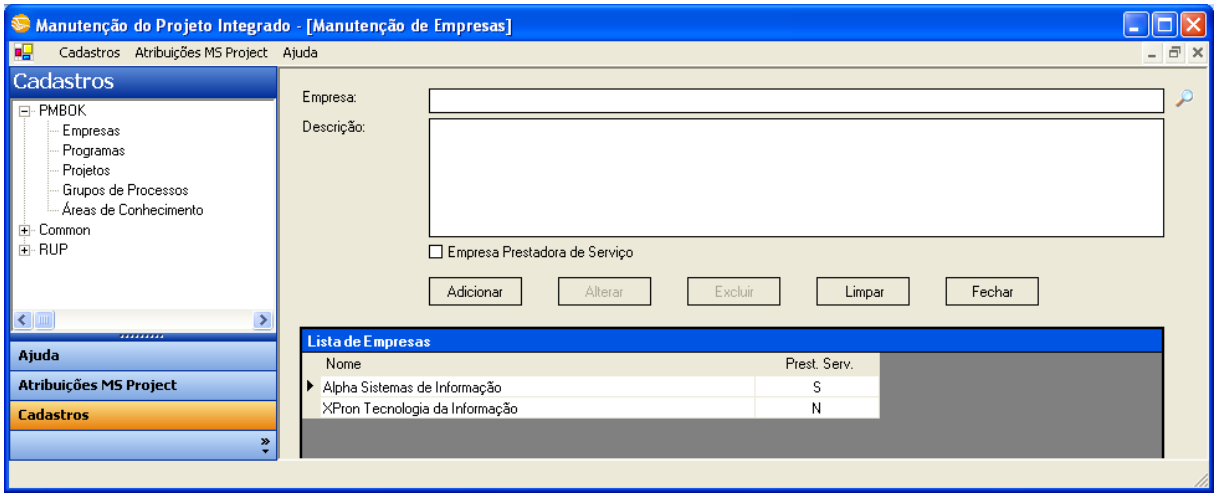


Figura 31. Tela para manutenção de empresas

Para realizar a operação de inclusão de uma empresa no sistema, deve-se obrigatoriamente preencher o campo referente ao nome da organização e clicar no botão "Adicionar". Caso o nome desta empresa já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação. Para as operações de alteração ou exclusão deve-se primeiramente selecionar a empresa a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar uma empresa específica cadastrada na base de dados do sistema. Para realizar esta operação é

necessário digitar o nome da empresa e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.2 MANUTENÇÃO DE PROGRAMAS

A Figura 32 apresenta a interface de manutenção de Programas, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos programas de uma determinada empresa que contém de projetos de software. A interface apresenta campos para selecionar a empresa e definir o nome e a descrição do programa. Além disso, uma lista dos programas (para cada respectiva empresa) já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

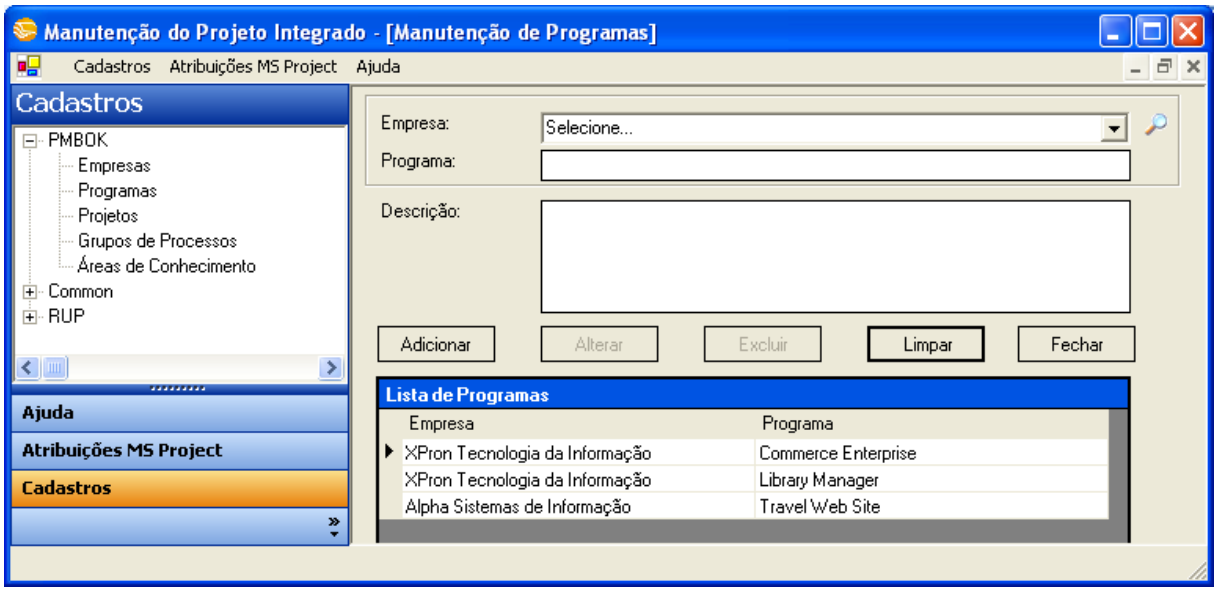


Figura 32. Tela para manutenção de programas

Para realizar a operação de inclusão de um programa no sistema, deve-se primeiramente selecionar um programa previamente cadastrado na base de dados do sistema. Posteriormente, deve-se preencher obrigatoriamente o campo referente ao nome do programa e clicar no botão “Adicionar”. Caso o nome deste programa já exista na base de dados para uma empresa determinada, o sistema informará sobre a duplicação existente e cancelará a operação.

Para as operações de alteração ou exclusão deve-se primeiramente selecionar o programa a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar um programa específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário selecionar a empresa, digitar o nome do

programa e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.3 MANUTENÇÃO DE PROJETOS

A Figura 33 apresenta a interface de manutenção de projetos, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos projetos pertencentes aos programas de uma determinada empresa. A interface apresenta campos para selecionar um programa e definir o nome, escopo, propósito, objetivos e custos indiretos do projeto.

Conforme mencionado anteriormente, o SPIT contém um módulo de validação das restrições definidas pelo modelo integrado, o qual é um *Add-in* para uma ferramenta comercial de gerência de projetos. Dessa forma, a interface de manutenção de projetos do módulo de *BackOffice* do SPIT permite associar um projeto cadastrado na base de dados do protótipo com o arquivo utilizado pela ferramenta comercial. Esta associação permite que as informações adicionais propostas pelo SPIM sejam, futuramente, exportadas para campos customizados da ferramenta comercial de gerência de projetos.

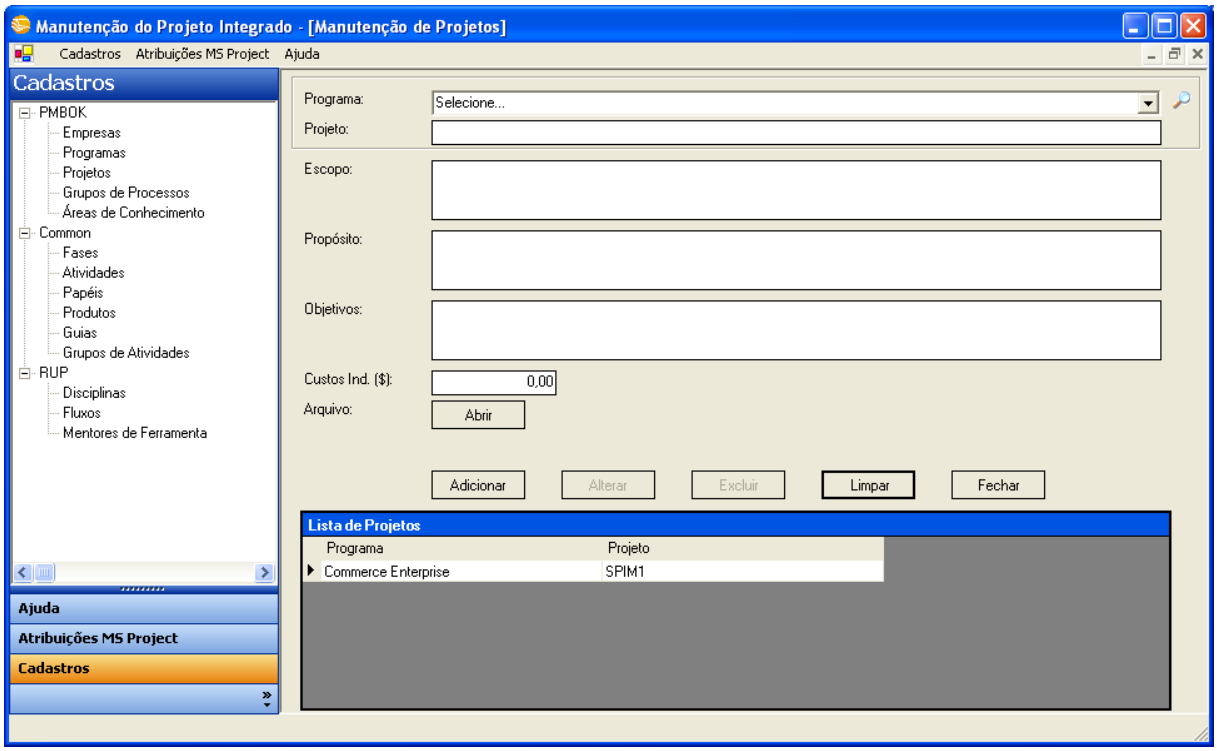


Figura 33. Tela para manutenção de projetos

Inicialmente, uma lista dos projetos (para cada respectivo programa) já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual

operação deseja realizar. Para realizar a operação de inclusão de um projeto sistema, deve-se primeiramente selecionar um programa previamente cadastrado na base de dados do sistema. Posteriormente, deve-se preencher obrigatoriamente o campo referente ao nome do projeto, selecionar o arquivo da ferramenta comercial que receberá as customizações propostas pelo SPIM e clicar no botão "Adicionar". Caso o nome deste projeto já exista na base de dados para um programa determinado, o sistema informará sobre a duplicação existente e cancelará a operação.

Para as operações de alteração ou exclusão deve-se primeiramente selecionar o projeto a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar um projeto específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário selecionar o programa, digitar o nome do projeto e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.4 MANUTENÇÃO DE GRUPOS DE PROCESSOS

A Figura 34 apresenta a interface de manutenção de grupos de processos, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos grupos de processos definidos pelo PMBOK. A interface apresenta campos para definir o nome e a descrição do grupo de processo.

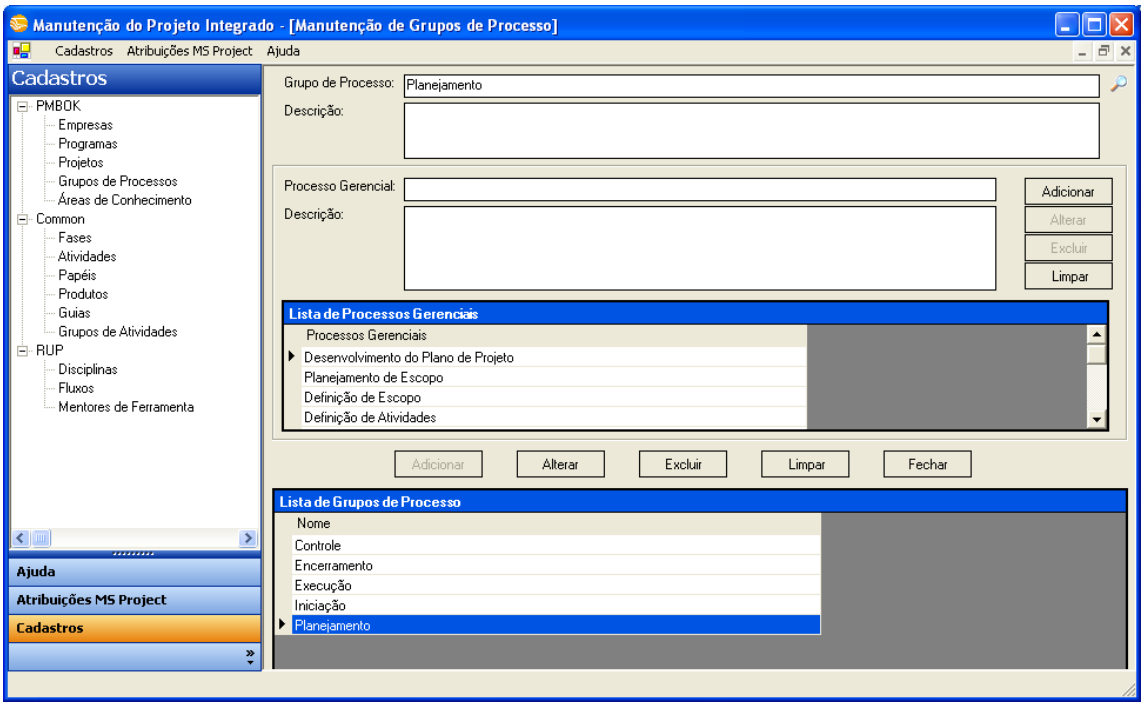


Figura 34. Tela para manutenção de grupos de processos do PMBOK

De acordo com [CHA06], o PMBOK representa as práticas de gerência de projetos em duas dimensões lógicas. Uma dimensão descreve as áreas de conhecimento enquanto que a outra dimensão descreve os processos gerenciais de um projeto, os quais estão contidos em cinco grupos de processo. Dessa forma, esta interface contém campos para definir o nome e a descrição dos processos gerenciais associados aos grupos de processos. Além disso, uma lista dos grupos de processos já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

Para realizar a operação de inclusão de um grupo de processo no sistema, deve-se obrigatoriamente preencher o campo referente ao seu nome, associá-lo a pelo menos um processo gerencial do PMBOK e clicar no botão "Adicionar". Caso o nome deste grupo de processo já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação. Para as operações de alteração ou exclusão deve-se primeiramente selecionar o grupo de processo a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar um grupo de processo específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do grupo de processo e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.5 MANUTENÇÃO DE ÁREAS DE CONHECIMENTO

A Figura 35 apresenta a interface de manutenção de áreas de conhecimento, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes às áreas de conhecimento definidas pelo PMBOK. A interface apresenta campos para definir o nome e a descrição da área de conhecimento.

De acordo com [PMI00], as áreas de conhecimento do PMBOK descrevem as principais competências que os gerentes de projeto devem desenvolver. Desta classe derivam as áreas centrais e as de apoio. Além disso, cada área de conhecimento abrange diversos processos de gerenciamento de projetos. Dessa forma, a interface do sistema apresenta campos para definir o tipo de área de conhecimento (central ou de apoio) e uma lista para selecionar os seus processos gerenciais associados.

Inicialmente, uma lista das áreas de conhecimento já cadastradas no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar. Para realizar a operação de inclusão de uma área de conhecimento no sistema, deve-se obrigatoriamente preencher o campo referente ao seu nome, selecionar pelo menos um

processo gerencial e clicar no botão “Adicionar”. Caso o nome desta área de conhecimento já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação.

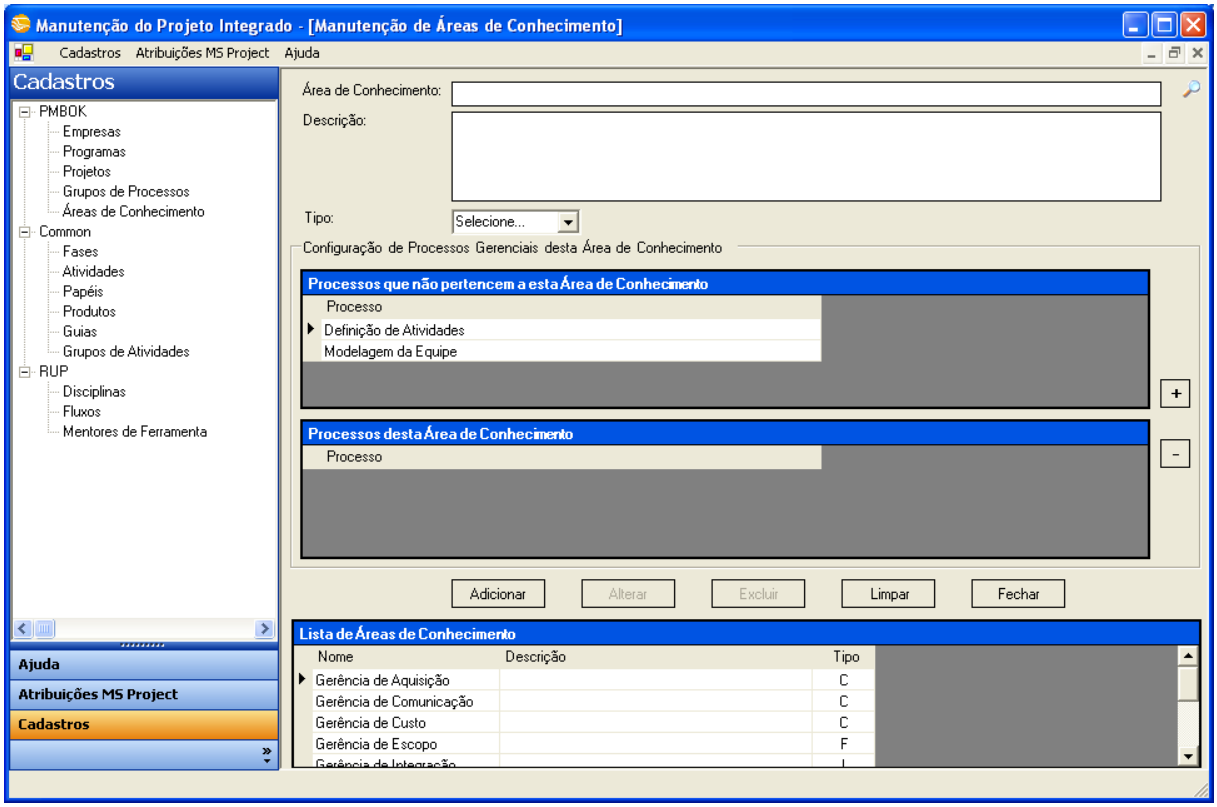


Figura 35. Tela para manutenção de áreas de conhecimento do PMBOK

Para as operações de alteração ou exclusão deve-se primeiramente selecionar a área de conhecimento a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar uma área de conhecimento específica cadastrada na base de dados do sistema. Para realizar esta operação é necessário digitar o nome da área de conhecimento e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.6 MANUTENÇÃO DE DISCIPLINAS

A Figura 36 apresenta a interface de manutenção de disciplinas, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes às disciplinas definidas pelo RUP. A interface apresenta campos para definir o nome e a descrição da disciplina. Além disso, uma lista das disciplinas já cadastradas no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

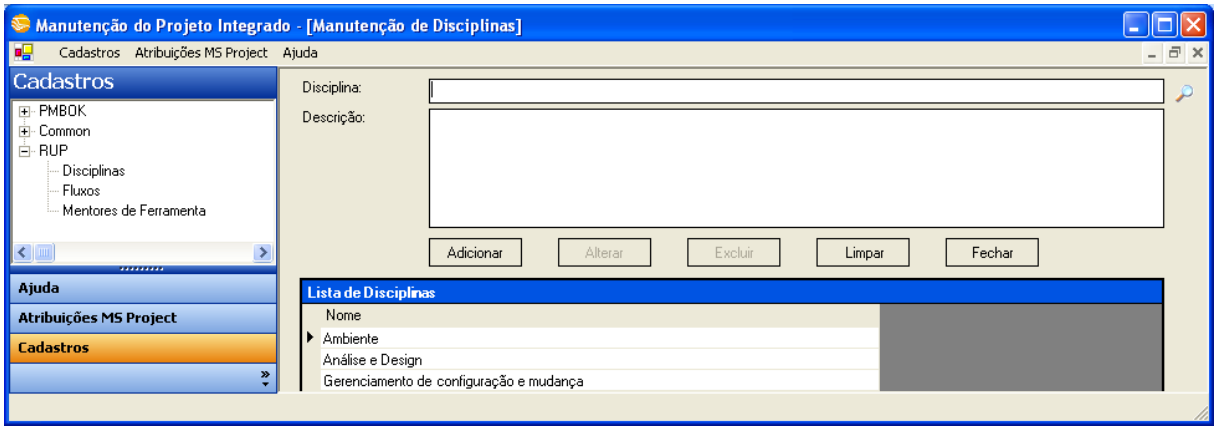


Figura 36. Tela para manutenção de disciplinas do RUP

Para realizar a operação de inclusão de uma disciplina no sistema, deve-se obrigatoriamente preencher o campo referente ao nome da disciplina e clicar no botão "Adicionar". Caso o nome desta disciplina já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação.

Para as operações de alteração ou exclusão deve-se primeiramente selecionar a disciplina a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar uma disciplina específica cadastrada na base de dados do sistema. Para realizar esta operação é necessário digitar o nome da disciplina e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.7 MANUTENÇÃO DE FLUXOS DE TRABALHO

A Figura 37 apresenta a interface de manutenção de fluxos de trabalho, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos fluxos de trabalho definidos pelo RUP. A interface apresenta campos para definir o nome e a descrição do fluxo de trabalho. Além disso, uma lista dos fluxos de trabalho já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

Para realizar a operação de inclusão de um fluxo de trabalho no sistema, deve-se obrigatoriamente preencher o campo referente ao nome do fluxo e clicar no botão "Adicionar". Caso o nome deste fluxo de trabalho já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação.

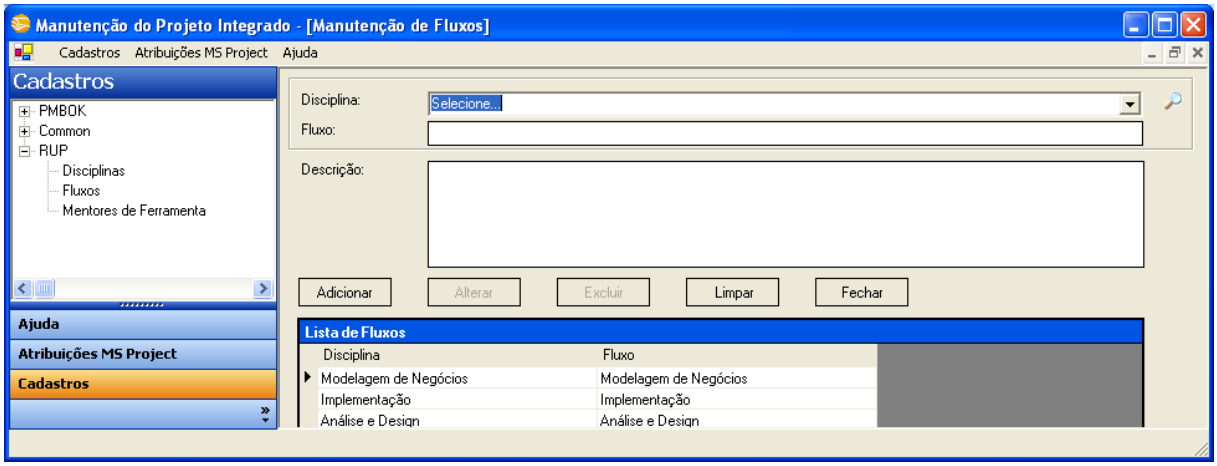


Figura 37. Tela para manutenção de fluxos de trabalho do RUP

Para as operações de alteração ou exclusão deve-se primeiramente selecionar o fluxo de trabalho a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar um fluxo de trabalho específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do fluxo de trabalho e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.8 MANUTENÇÃO DE MENTORES DE FERRAMENTAS

A Figura 38 apresenta a interface de manutenção de mentores de ferramentas, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos mentores de ferramentas definidos pelo RUP.

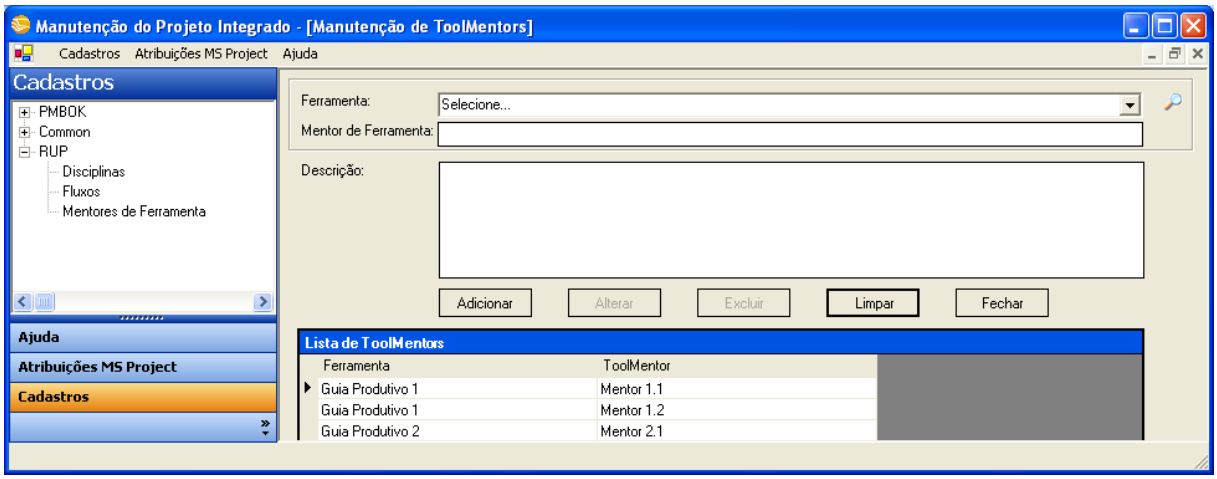


Figura 38. Tela para manutenção de mentores de ferramentas do RUP

De acordo com [JAC99], os mentores de ferramentas descrevem o uso de ferramentas no contexto de algumas atividades. A interface apresenta campos para definir o nome e a descrição do mentor de ferramentas. Além disso, uma lista dos mentores de

ferramentas já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

Para realizar a operação de inclusão de um mentor de ferramentas no sistema, deve-se obrigatoriamente preencher o campo referente ao nome e clicar no botão “Adicionar”. Caso o nome deste mentor de ferramentas já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação. Para as operações de alteração ou exclusão deve-se primeiramente selecionar o mentor de ferramentas a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar um mentor de ferramentas específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do mentor de ferramentas e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.9 MANUTENÇÃO DE FASES

A Figura 39 apresenta a interface de manutenção de fases, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes às fases dos projetos de software. A interface apresenta campos para definir o nome e a descrição da fase, além de um campo para definir a ordenação entre as fases. Além disso, uma lista das fases já cadastradas no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

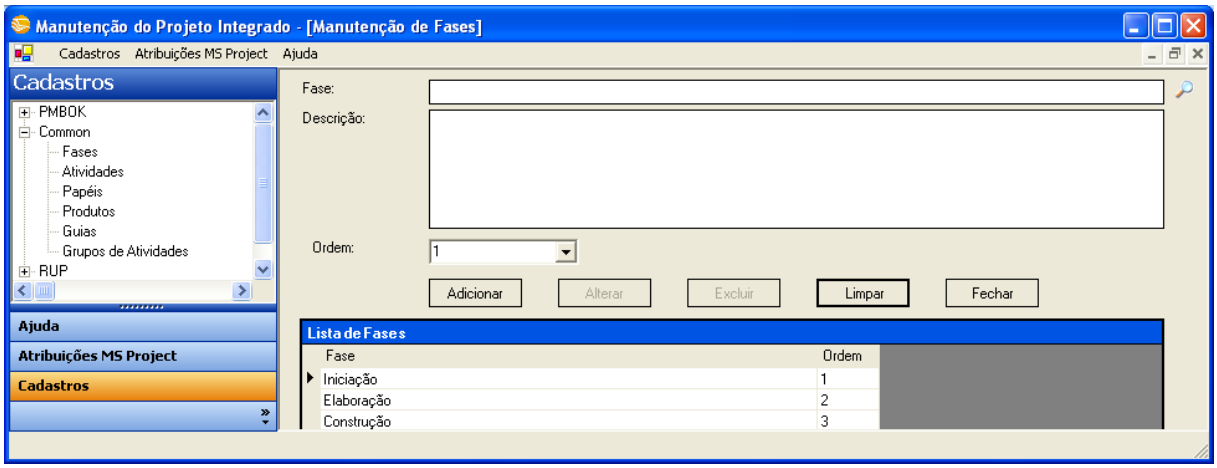


Figura 39. Tela para manutenção de fases

Para realizar a operação de inclusão de uma fase no sistema, deve-se obrigatoriamente preencher o campo referente ao nome da fase e clicar no botão “Adicionar”. Caso o nome desta fase já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação. Para as operações de alteração ou exclusão deve-se

primeiramente selecionar a fase a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar uma fase específica cadastrada na base de dados do sistema. Para realizar esta operação é necessário digitar o nome da fase e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.10 MANUTENÇÃO DE PAPÉIS

A Figura 40 apresenta a interface de manutenção de papéis, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos papéis que participam dos projetos de software. A interface apresenta campos para definir o nome e a descrição do papel, além de um campo para definir se é um papel gerencial ou produtivo (tais como um gerente de projetos ou um desenvolvedor). Além disso, uma lista dos papéis já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

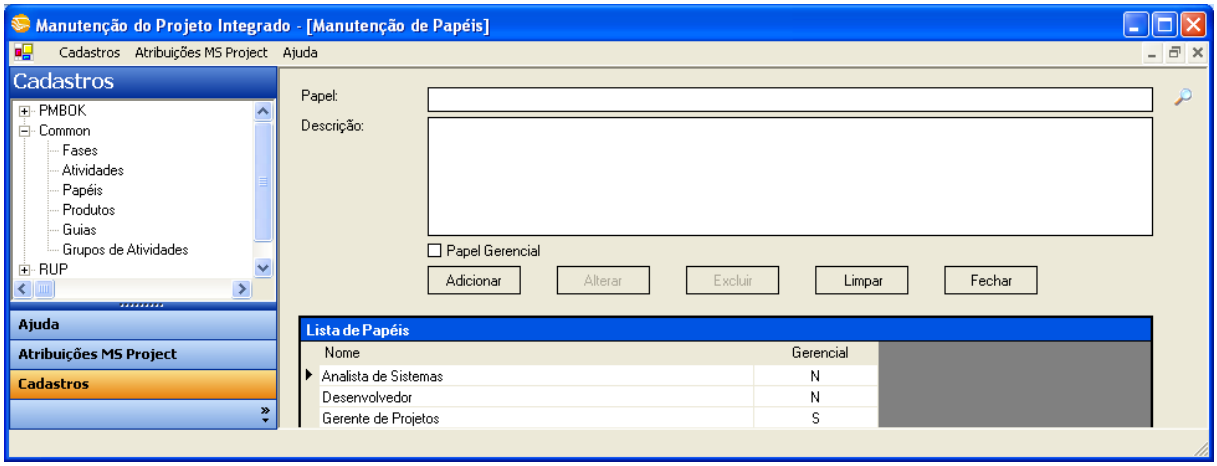


Figura 40. Tela para manutenção de papéis

Para realizar a operação de inclusão de um papel no sistema, deve-se obrigatoriamente preencher o campo referente ao nome do papel e clicar no botão “Adicionar”. Caso o nome deste papel já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação. Para as operações de alteração ou exclusão deve-se primeiramente selecionar o papel a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar um papel específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do papel e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.11 MANUTENÇÃO DE PRODUTOS DE TRABALHO

A Figura 41 apresenta a interface de manutenção de produtos de trabalho, a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos produtos criados/utilizados nos projetos de software. A interface apresenta campos para definir o nome e a descrição do produto de trabalho, além de um campo para definir se é um produto gerencial ou produtivo (tais como um documento de visão do projeto ou um diagrama UML). Além disso, pode-se informar se este produto de trabalho será criado ou não por uma empresa externa ao projeto.

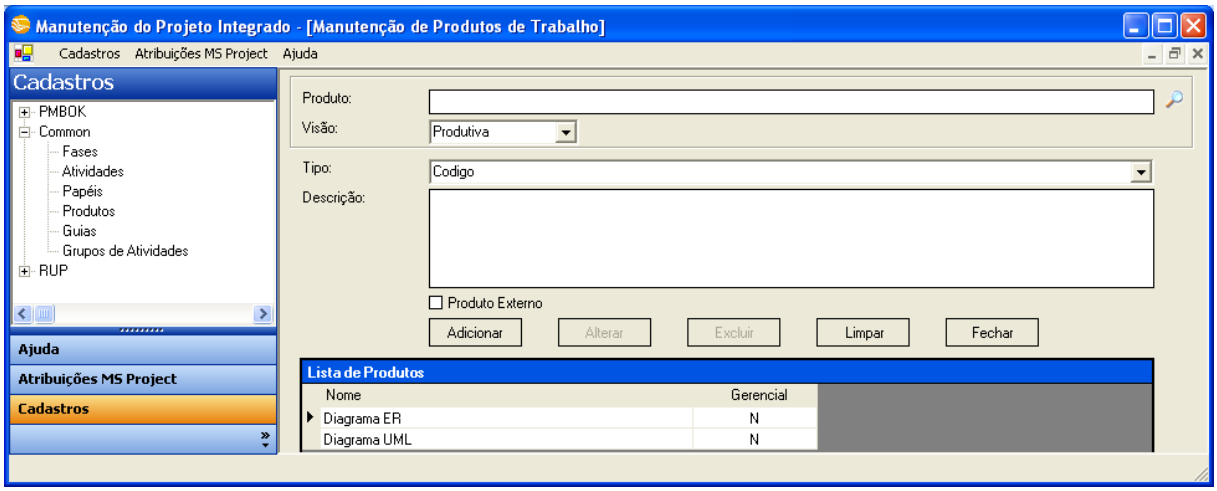


Figura 41. Tela para manutenção de produtos de trabalho

Inicialmente, uma lista dos produtos de trabalho já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar. Para realizar a operação de inclusão de um produto de trabalho no sistema, deve-se obrigatoriamente preencher o campo referente ao seu nome e tipo de visão (gerencial ou produtiva) e clicar no botão “Adicionar”. Caso o nome deste produto de trabalho já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação.

Para as operações de alteração ou exclusão deve-se primeiramente selecionar o produto de trabalho a partir da lista e indicar a opção desejada através dos botões “Alterar” ou “Excluir”. Por fim, é possível pesquisar um produto de trabalho específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do produto de trabalho, seu tipo de visão e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

B.1.12 MANUTENÇÃO DE GUIAS

A Figura 42 apresenta a interface de manutenção de guias (*guidances*), a qual permite à criação, modificação, exclusão e visualização das informações correspondentes aos *guidances* utilizados nos projetos de software. Uma *guidance* define os elementos de orientação, tais como ferramentas e técnicas para as atividades. A interface apresenta campos para definir o nome e a descrição do *guidance*, além de um campo para definir se é um *guidance* gerencial ou produtivo. Além disso, uma lista dos *guidances* já cadastrados no protótipo é apresentada e o usuário pode escolher através dos botões da interface qual operação deseja realizar.

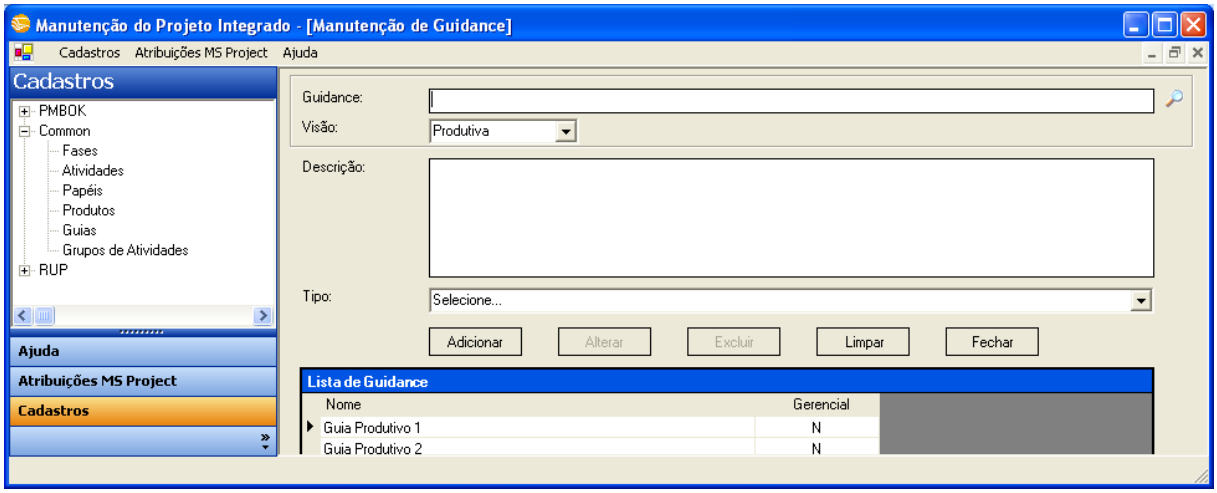


Figura 42. Tela para manutenção de guias

Para realizar a operação de inclusão de um *guidance* no sistema, deve-se obrigatoriamente preencher o campo referente ao seu nome e tipo de visão (gerencial ou produtiva) e clicar no botão "Adicionar". Caso o nome deste *guidance* já exista na base de dados, o sistema informará sobre a duplicação existente e cancelará a operação. Para as operações de alteração ou exclusão deve-se primeiramente selecionar o *guidance* a partir da lista e indicar a opção desejada através dos botões "Alterar" ou "Excluir". Por fim, é possível pesquisar um *guidance* específico cadastrado na base de dados do sistema. Para realizar esta operação é necessário digitar o nome do *guidance*, seu tipo de visão e clicar no botão com formato de lupa. O sistema retornará os resultados encontrados em uma lista.

APÊNDICE C – Protocolo de avaliação do SPIM

C.1 PROTOCOLO DE AVALIAÇÃO DA PESQUISA

Este apêndice contém a descrição detalhada do protocolo de avaliação do SPIM.

C.1.1 OBJETIVO

Avaliar os conceitos advindos do modelo integrado SPIM no que diz respeito à sua aceitação e aplicabilidade, de acordo com o ponto de vista dos gerentes de projetos de software.

C.1.2 CARACTERÍSTICA-CHAVE DO MÉTODO DE PESQUISA

Este é o roteiro para o desenvolvimento e aplicação de um instrumento de pesquisa semi-estruturada com questões abertas que se caracteriza como uma pesquisa do tipo transeccional.

C.1.3 ORGANIZAÇÃO DESSE PROTOCOLO

C.1.3.1 PROCEDIMENTOS

A. Reuniões para levantamento das questões e estruturação do roteiro de entrevista	
Participantes:	Maurício Covolan Rosito Prof. Dr. Ricardo Melo Bastos
Local:	PPGCC (Programa de Pós-Graduação em Ciência da Computação – FACIN – PUCRS)
Datas:	19 e 26 de outubro de 2007

B. Validação de face e conteúdo	
Participante:	Prof. Msc. Eduardo Meira Peres
Local:	DBServer
Data:	05 de novembro de 2007

C. Adequação do roteiro de entrevista com base na validação de face e conteúdo	
Participante:	Maurício Covolan Rosito
Local:	PPGCC (Programa de Pós-Graduação em Ciência da Computação – FACIN – PUCRS)
Data:	06 de novembro de 2007

D. Pré-Teste	
Participantes:	Maurício Covolan Rosito Msc. Ana Karpouzas
Local:	Centro de Pesquisa PUCRS/HP
Data:	09 de novembro de 2007

E. Adequação do roteiro de entrevista com base no pré-teste	
Participante:	Maurício Covolan Rosito
Local:	PPGCC (Programa de Pós-Graduação em Ciência da Computação – FACIN – PUCRS)
Data:	10 de novembro de 2007

F. Aplicação do Instrumento	
Tipo:	Entrevista semi-estruturada com questões abertas
Entrevistador:	Maurício Covolan Rosito
Entrevistados	**Datas**
Gerente de Projetos 1	12/11/2007
Gerente de Projetos 2	12/11/2007
Gerente de Projetos 3	13/11/2007
Gerente de Projetos 4	13/11/2007
Gerente de Projetos 5	14/11/2007
Gerente de Projetos 6	14/11/2007
Gerente de Projetos 7	14/11/2007
Gerente de Projetos 8	15/11/2007
Gerente de Projetos 9	16/11/2007
Gerente de Projetos 10	16/11/2007
Gerente de Projetos 11	19/11/2007
Gerente de Projetos 12	20/11/2007

C.1.3.2 ESCOLHA DAS PESSOAS ENTREVISTADAS

Todos os respondentes atuam como gerentes de projetos em setores de pesquisa ou desenvolvimento de software.

C.1.3.3 OUTROS RECURSOS UTILIZADOS

- **Sistema Computacional e Estatístico:**
  - Weka (Waikato Enviroment for Knowledge Analysis).
- **Recursos Materiais:**
  - Uma sala de reuniões na organização;
  - Um notebook com o software SPIT instalado;

C.1.3.4 ANÁLISE DE DADOS

Esta entrevista insere-se em uma pesquisa qualitativa, cuja aquisição dos dados foi realizada por meio de um questionário elaborado em conformidade com [REA05].

C.1.3.5 ROTEIRO DAS ENTREVISTAS

Questionário de Avaliação do Perfil dos Entrevistados

Perfil da Empresa

1. Razão Social:
2. Ramo de Atividade:
3. Tempo de atuação no mercado de Tecnologia da Informação:
4. Número aproximado de funcionários:
5. Número aproximado de gerentes de projeto:
6. Número aproximado de desenvolvedores de software:
7. A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos?

   ( ) Sim ( ) Não

8. A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos?

   ( ) Sim ( ) Apenas parcialmente ( ) Não

Perfil do Gerente de Projetos

1. Nome:
2. Email:
3. Idade:
4. Experiência profissional no ramo da indústria de software (em anos):
5. Experiência profissional como gerente de projetos (em anos):
6. Número médio de projetos que você costuma gerenciar ao mesmo tempo?
7. Já realizou algum treinamento referente à Gerência de Projetos?

   ( ) Sim Nome do Curso: Duração:

   ( ) Não

8. Como você qualificaria seu conhecimento relacionado à Gerência de Projetos?

   ( ) Nenhum ( ) Pouco ( ) Moderado ( )Avançado

9. Como você qualificaria seu conhecimento relacionado aos conceitos advindos do PMBOK Guide?

   ( ) Nenhum ( ) Pouco ( ) Moderado ( )Avançado

10. Como você qualificaria seu conhecimento relacionado aos conceitos advindos do RUP?

    ( ) Nenhum ( ) Pouco ( ) Moderado ( )Avançado

Questionário de Avaliação do Modelo Integrado SPIM

Glossário

- *Atividade Produtiva:* representa uma unidade de trabalho, desempenhada por um papel produtivo, que produz um resultado significativo no contexto de um projeto de software.
    - *Exemplo: modelagem do banco de dados da aplicação.*
- *Atividade Gerencial:* descreve as atividades gerenciais, desempenhadas por papéis gerenciais, do projeto. Estas atividades gerenciais pertencem exclusivamente ao projeto de desenvolvimento de software.
    - *Exemplo: organizar e conduzir uma reunião de acompanhamento do projeto.*
- *Atividade Gerencial de Apoio:* São atividades gerenciais que pertencem exclusivamente aos demais fluxos de atividades de suporte ao projeto da organização (aqui denominados Fluxos Empresariais).
    - *Exemplo: contratar um administrador de banco de dados (atividade realizada pelo setor de recursos humanos da organização).*

Avaliação do Modelo Integrado SPIM

1. Ao realizar planejamento de um projeto, o gerente de projetos pode necessitar interagir com outros departamentos da organização a fim de obter informações relevantes sobre as atividades gerenciais de apoio necessárias para a realização das atividades produtivas do projeto. Por exemplo, durante o planejamento de atividades de projeto de software, o gerente de projetos informa ao setor de recursos humanos sobre a necessidade de contratação de um administrador de banco de dados. Neste caso, constata-se a existência de uma relação de dependência entre as atividades do projeto de software (tais como, a modelagem do banco de dados) com as atividades pertencentes ao fluxo de trabalho do setor de recursos humanos referentes à contratação do profissional requerido para executar a atividade de produção. Assim, percebe-se que o fluxo de atividades de um projeto de software pode interagir com os demais fluxos de atividades da organização (fluxos empresariais). Dessa forma, marque abaixo os benefícios que você percebe em fazer o planejamento integrado de atividades gerenciais e produtivas para projetos de desenvolvimento de software?

    ☐ Redução do tempo no processo de elaboração do planejamento do projeto;

    ☐ Identificação das dependências entre as atividades gerenciais de apoio e de produção;

    ☐ Identificação e mensuração dos custos indiretos do projeto advindos das atividades gerenciais de apoio;

    ☐ Capacidade de ter acesso a informações dos fluxos empresariais (pertencentes aos outros departamentos da organização);

    ☐ Capacidade de evitar distorções no planejamento de projetos (tais como, o aumento dos custos e atrasos nos prazos do projeto) pela desconsideração de que as atividades gerenciais de apoio utilizam recursos não alocados diretamente ao projeto de software;

    ☐ Outros:
2. Em função da integração dos modelos da gerência de projetos e dos processos de desenvolvimento de software, pode-se identificar um conjunto de regras (ou restrições).

Tais regras visam, em sua maioria, garantir a consistência do modelo de integração SPIM. Dessa forma, para cada questão, marque com um "X" o grau de benefício observado dessas restrições para o planejamento do projeto de acordo com a classificação apresentada abaixo:

1 – Nulo

2 – Baixo

3 – Moderado

4 – Alto

5 – Muito alto

Restrições Verificadas pelo Modelo Integrado SPIM	Resposta				
	1	2	3	4	5
2.1 Uma mesma atividade (produtiva, gerencial ou gerencial de apoio) não pode criar, modificar ou consultar um mesmo artefato. Para realizar tais operações devem ser criadas atividades distintas;					
2.2 As atividades gerencial e gerencial de apoio não podem produzir ou modificar produtos de trabalho produtivos, somente produtos de trabalho gerenciais. Porém, estas atividades podem consultar produtos de trabalho produtivos;					
2.3 A atividade produtiva não pode produzir ou modificar produtos de trabalho gerenciais, somente produtos de trabalho produtivos. Porém, esta atividade pode consultar produtos de trabalho gerenciais;					
2.4 Uma atividade (produtiva, gerencial ou gerencial de apoio) somente pode atualizar ou consultar um produto de trabalho que já tenha sido criado por uma atividade antecessora;					
2.5 O papel do *stakeholder* envolvido deve ser compatível com o tipo de atividade (gerencial ou produtivo);					
2.6 A adição das informações relativas ao relacionamento das atividades com os tipos de guidances (gerencial ou produtivo);					
2.7 As atividades gerencial e gerencial de apoio devem ter pelo menos um papel gerencial como um de seus papéis;					
2.8 A atividade produtiva deve ter pelo menos um papel produtivo como um de seus papéis;					

3 Quais foram os pontos fortes observados no modelo integrado SPIM durante a sua utilização neste projeto?

4 Quais foram os pontos fracos observados no modelo integrado SPIM durante a sua utilização neste projeto?

5 Objetivando realizar o planejamento integrado dos conceitos advindos da gerência de

projetos e dos processos de desenvolvimento de software, o modelo SPIM propõe a distinção explícita entre as atividades produtivas e gerenciais de um projeto de software com as atividades gerenciais de apoio dos demais departamentos da organização. Comente se você achou adequada esta distinção proposta nesta pesquisa.

6 O modelo SPIM foi concebido considerando que a dificuldade para visualizar a interdependência dos fluxos de trabalho da empresa e do projeto de software durante o planejamento do projeto pode resultar, por exemplo, no aumento dos custos e em atrasos nos prazos do projeto. Você concorda que a dificuldade de identificar as atividades gerenciais de apoio, durante o planejamento do projeto, pode afetar negativamente o projeto? Justifique sua resposta.

7 As atividades gerenciais de apoio não são exclusivas de um projeto em especial, mas de um fluxo comum da empresa, compartilhado pelos projetos em andamento e utiliza recursos não alocados diretamente ao projeto de software. Você concorda se esta lógica é benéfica na definição do planejamento do projeto? Discorra sobre os motivos que influenciaram sua resposta.

8 Considerando que o fluxo de atividades de um projeto de software e os demais fluxos de atividades de apoio ao projeto da organização são executados de forma distinta, comente se modelo SPIM colaborou para a identificar a relação de dependência entre as atividades pertencentes a estes dois tipos de fluxos de trabalho, permitindo antecipar as necessidades advindas das áreas de apoio da organização durante o planejamento do projeto.

APÊNDICE D – Projeto exemplo para avaliação do SPIM

		Task Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Software Development Workflow**	**25,5 days**	**Mon 26/11/07**	**Mon 31/12/07**		
2		**Iniciação**	**4 days**	**Mon 26/11/07**	**Thu 29/11/07**		
3		**Monitoramento e Controle**	**0,5 days**	**Mon 26/11/07**	**Mon 26/11/07**		
4		Reunião de Acompanhamento	0,5 days	Mon 26/11/07	Mon 26/11/07		Gerente de Projetos;Sala de Reunião 1[1]
5		**Área Gerencial**	**3,5 days**	**Mon 26/11/07**	**Thu 29/11/07**	**3**	
6		Elaborar documento de Avalia	0,5 days	Mon 26/11/07	Mon 26/11/07	3	Gerente de Projetos
7		Organizar e Conduzir a Reuni	0,5 days	Tue 27/11/07	Tue 27/11/07	6	Gerente de Projetos
8		Elaborar e Revisar Caso de De	1 day	Tue 27/11/07	Wed 28/11/07	7	Gerente de Projetos
9		Revisão de estimativas	0,5 days	Wed 28/11/07	Wed 28/11/07	8	Gerente de Projetos
10		Estimar Custos e Recursos	1 day	Thu 29/11/07	Thu 29/11/07	9	Gerente de Projetos
11		**Elaboração**	**5 days**	**Fri 30/11/07**	**Thu 6/12/07**		
12		**Área Gerencial**	**5 days**	**Fri 30/11/07**	**Thu 6/12/07**		
13		Refinar Planejamento do Projet	5 days	Fri 30/11/07	Thu 6/12/07	2	Gerente de Projetos
14		**Área de Engenharia**	**2 days**	**Fri 30/11/07**	**Mon 3/12/07**		
15		Definir Arquitetura	2 days	Fri 30/11/07	Mon 3/12/07	2	Projetista de Sistemas
16		Análise dos documentos de Ef	1 day	Fri 30/11/07	Fri 30/11/07	2	Analista de Sistemas
17		Colocar Artefatos sob Rastrea	1 day	Mon 3/12/07	Mon 3/12/07	16	Analista de Sistemas
18		Teste da Arquitetura	0,5 days	Fri 30/11/07	Fri 30/11/07	2;47	Projetista de Sistemas
19		**Revisar Projeto e Aprovações**	**0,5 days**	**Tue 4/12/07**	**Tue 4/12/07**		
20		Conduzir Reunião de Avaliaçã	0,5 days	Tue 4/12/07	Tue 4/12/07	14	Gerente de Projetos;Sala de Reunião 2[1]
21		**Construção**	**12,5 days**	**Fri 7/12/07**	**Tue 25/12/07**		
22		**Arquitetura**	**3 days**	**Fri 7/12/07**	**Tue 11/12/07**		
23		Definição do processo de impl	1 day	Fri 7/12/07	Fri 7/12/07	11	Projetista de Sistemas
24		Modelar Banco de Dados	2 days	Mon 10/12/07	Tue 11/12/07	23;50	DBA
25		**Especificação Requisitos - Rel**	**7 days**	**Wed 12/12/07**	**Thu 20/12/07**		
26		RNF6 - Front End em Inglês - D	2 days	Wed 12/12/07	Thu 13/12/07	24	Desenvolvedor 1
27		RNF8 - Construção do Front En	4 days	Wed 12/12/07	Mon 17/12/07	24	Desenvolvedor 2
28		RF227 - Manter Perfis - ERR	3 days	Tue 18/12/07	Thu 20/12/07	27	Desenvolvedor 1

29		**Especificar Teste**	**2 days**	**Mon 10/12/07**	**Tue 11/12/07**		
30		Elaborar Especificação de Plar	2 days	Mon 10/12/07	Tue 11/12/07	23	Analista de Testes 1
31		**Executar Testes**	**2 days**	**Fri 21/12/07**	**Mon 24/12/07**		
32		Executar Plano de Teste	2 days	Fri 21/12/07	Mon 24/12/07	25	Tester 1
33		**Revisar Projeto e Aprovações**	**0,5 days**	**Tue 25/12/07**	**Tue 25/12/07**		
34		Conduzir Reunião de Avaliaçã	0,5 days	Tue 25/12/07	Tue 25/12/07	31	Gerente de Projetos;Sala de Reunião 2[1]
35		**Transição**	**2 days**	**Tue 25/12/07**	**Thu 27/12/07**		
36		**Integrar Sistema**	**2 days**	**Tue 25/12/07**	**Thu 27/12/07**		
37		Preparação e Validação do Ar	1 day	Tue 25/12/07	Wed 26/12/07	21	Projetista de Sistemas
38		Revisar Projeto e Aprovações	1 day	Wed 26/12/07	Thu 27/12/07	37	Projetista de Sistemas
39		**Disponibilização da release 1.0 - F**	**2 days**	**Wed 26/12/07**	**Fri 28/12/07**		
40		Acompanhamento do projeto	2 days	Wed 26/12/07	Fri 28/12/07	37;53	Gerente de Projetos
41		Teste de Aceitação	2 days	Wed 26/12/07	Fri 28/12/07	37;53	Analista de Testes 1
42		**Fechar Projeto**	**1 day**	**Fri 28/12/07**	**Mon 31/12/07**		
43		Encerrar Projeto	1 day	Fri 28/12/07	Mon 31/12/07	39	Gerente de Projetos
44							
45		**Enterprises Workflow**	**21 days**	**Tue 27/11/07**	**Tue 25/12/07**		
46		**Elaboração**	**9 days**	**Tue 27/11/07**	**Fri 7/12/07**		
47		**Administrativo**	**3 days**	**Tue 27/11/07**	**Thu 29/11/07**		
48		Aquisição de Servidor	3 days	Tue 27/11/07	Thu 29/11/07		Administrativo
49		**Recursos Humanos**	**2 days**	**Thu 6/12/07**	**Fri 7/12/07**		
50		Contratar novo DBA	2 days	Thu 6/12/07	Fri 7/12/07		RH
51		**Transição**	**1 day**	**Tue 25/12/07**	**Tue 25/12/07**		
52		**Administrativo**	**1 day**	**Tue 25/12/07**	**Tue 25/12/07**		
53		Comprar passagens Aéreas	1 day	Tue 25/12/07	Tue 25/12/07		Administrativo

APÊNDICE E – Análise do perfil das empresas

E.1 ANÁLISE DO PERFIL DAS EMPRESAS

Este apêndice contém a descrição detalhada do perfil das empresas envolvidas na avaliação do modelo SPIM, apresentadas no capítulo 8.

E.1.1 AVALIAÇÃO DO PERFIL DAS EMPRESAS

Abaixo serão apresentadas em subseções as respostas do questionário de avaliação do perfil das empresas que participaram desta pesquisa. Cabe salientar que por motivo de privacidade o nome destas empresas não foi publicado neste trabalho.

E.1.1.1 EMPRESA 1

- Ramo de Atividade: Fábrica de Software
- Tempo de atuação no mercado de Tecnologia da Informação: 6 anos
- Número aproximado de funcionários: 150
- Número aproximado de gerentes de projeto: 15
- Número aproximado de desenvolvedores de software: 80
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Sim
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Sim

E.1.1.2 EMPRESA 2

- Ramo de Atividade: Fábrica de Software
- Tempo de atuação no mercado de Tecnologia da Informação: 14 anos
- Número aproximado de funcionários: 200
- Número aproximado de gerentes de projeto: 20
- Número aproximado de desenvolvedores de software: 150
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Sim
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Sim

E.1.1.3 EMPRESA 3

- Ramo de Atividade: Soluções em tecnologias móveis
- Tempo de atuação no mercado de Tecnologia da Informação: 15 anos
- Número aproximado de funcionários: 35
- Número aproximado de gerentes de projeto: 4
- Número aproximado de desenvolvedores de software: 10
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Sim
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Apenas parcialmente

E.1.1.4 EMPRESA 4

- Ramo de Atividade: Soluções Web
- Tempo de atuação no mercado de Tecnologia da Informação: 4 anos
- Número aproximado de funcionários: 42
- Número aproximado de gerentes de projeto: 5
- Número aproximado de desenvolvedores de software: 20
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Não
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Apenas parcialmente

E.1.1.5 EMPRESA 5

- Ramo de Atividade: Governamental
- Tempo de atuação no mercado de Tecnologia da Informação: 30 anos
- Número aproximado de funcionários: 50
- Número aproximado de gerentes de projeto: 5
- Número aproximado de desenvolvedores de software: 25
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Sim
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Apenas parcialmente

E.1.1.6 EMPRESA 6

- Ramo de Atividade: Soluções Web
- Tempo de atuação no mercado de Tecnologia da Informação: 15 anos
- Número aproximado de funcionários: 220
- Número aproximado de gerentes de projeto: 18
- Número aproximado de desenvolvedores de software: 105
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Sim
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Sim

E.1.1.7 EMPRESA 7

- Ramo de Atividade: Projetos de Pesquisa e Desenvolvimento
- Tempo de atuação no mercado de Tecnologia da Informação: 5 anos
- Número aproximado de funcionários: 60
- Número aproximado de gerentes de projeto: 1
- Número aproximado de desenvolvedores de software: 30
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Não
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Sim

E.1.1.8 EMPRESA 8

- Ramo de Atividade: Computação móvel
- Tempo de atuação no mercado de Tecnologia da Informação: 11 anos
- Número aproximado de funcionários: 5
- Número aproximado de gerentes de projeto: 2
- Número aproximado de desenvolvedores de software: 3
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Sim
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Sim

E.1.1.9 EMPRESA 9

- Ramo de Atividade: Fábrica de Software
- Tempo de atuação no mercado de Tecnologia da Informação: 20 anos
- Número aproximado de funcionários: 500
- Número aproximado de gerentes de projeto: 20
- Número aproximado de desenvolvedores de software: 400
- A empresa adota algum processo de desenvolvimento de software baseado no RUP para os seus projetos? Não
- A empresa adota os conceitos gerenciais advindos do PMBOK Guide em seus projetos? Sim

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo