UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Uma Extensão de Rede de Petri para Modelagem de Processos e Controle de Projetos

TESE DE DOUTORADO

Cecir Barbosa de Almeida Farias

Ulrich Schiel

Orientador

Campina Grande, Paraíba, outubro de 2008.

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Uma Extensão de Rede de Petri para Modelagem de Processos e Controle de Projetos

TESE DE DOUTORADO

Proposta de Tese de Doutorado submetida à coordenação dos Cursos de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de doutor em Ciências no Domínio da Engenharia Elétrica.

Cecir Barbosa de Almeida Farias

Ulrich Schiel

Orientador

Campina Grande, Paraíba, outubro de 2008.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

F224e

2008 Farias, Cecir Barbosa de Almeida

Uma extensão de rede de Petri para modelagem de processos e controle de projetos / Cecir Barbosa de Almeida Farias.─ Campina Grande, 2008.

282 f.

Tese (Doutorado em Engenharia Elétrica) - Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

Referências.

Orientador: Prof. Dr. Ulrich Schiel.

1. Engenharia de Software 2. Gerência de Projetos 3. Redes de Petri 4. Rede de Projeto 5. Sistema de Suporte à Decisão 6. Ambiente de Gerenciamento de Projetos 7. Processos de Desenvolvimento 8. Redes de Planejamento I. Título.

CDU – 004.5(043)

UMA EXTENSÃO DE REDES DE PETRI PARA MODELAGEM DE PROCESSOS E CONTROLE DE PROJETOS

CECIR BARBOSA DE ALMEIDA FARIAS

Tese Aprovada em 02.10.2008

ULRICH SCHIEL, Dr.rer.nat., UFCG
Orientador

NILSON SANT'ANNA, Dr., INPE
Componente da Banca

HERMANO PERRELLI DE MOURA, Dr., UFPE
Componente da Banca

JORGE CÉSAR ABRANTES DE FIGUEIREDO, D.Sc., UFCG
Componente da Banca

JOSÉ ANTÃO BELTRÃO MOURA, Ph.D., UFCG
Componente da Banca

CAMPINA GRANDE – PB
OUTUBRO - 2008

Dedicatória

Dedico esta tese ao meu pai Ruy (*in memorian*), minha mãe Adacy,
meu marido Jalf e meu filho Leonardo.

Dedico esta tese ao meu orientador Ulrich Schiel.

Agradecimentos

A Deus por ter me dado determinação, coragem e serenidade durante todos os dias que estive entre salas de aulas e laboratórios em busca desta conquista profissional e pessoal.

A minha mãe Adacy pelo amor, amizade inestimável e pela compreensão nos períodos que estive ausente. A meu pai Ruy que faleceu durante o período do doutorado e não viu a conclusão deste trabalho, mas que sempre havia dado conselhos para eu seguir em frente, estudando e lendo muito.

Ao meu esposo Jalf que tanto me apoiou durante os anos de realização do doutorado e a meu querido filho Leonardo.

Agradeço a minha tia Alba Lenita pelo apoio e amizade constantes, e aos meus familiares Sandra, Andréa, Sylvia, Débora, Nano, Ana Alice, Téa, Susi e Susie.

A minha sogra Antonieta e meu sogro Zezito que me acolheram com carinho e sua família Zita, Gerard, Geórgya e Beatriz.

Especialmente agradeço ao meu orientador Ulrich Schiel pela orientação ao longo de todo o período de doutorado, pelos conhecimentos transmitidos, pelo incentivo, dedicação, zelo com o nosso trabalho e amizade.

Aos Professores Hermano Perreli, Nilson Sant'anna, Antão Moura e Jorge Abrantes pela participação na banca de doutorado, pela atenção e pelas sugestões dadas à tese.

Ao Prof. Guilherme Travassos pelas sugestões e material enviados na época do exame de qualificação, e pelo tempo de auxílio na COPPE com o planejamento do estudo avaliatório.

Aos professores Peter e Marcus Sampaio por proporcionarem o uso prático deste trabalho com o projeto SAD – Sistema de Apoio à Decisão da CHESF – Companhia Hidro-Elétrica do São Francisco, à Milena e à equipe de desenvolvimento.

Aos professores Juarez e Eustáquio Queiroz pelo auxílio e orientação no ensaio avaliatório.

A Rodrigo, Júlio e Márcio por ajudarem na implementação do módulo ERA - Editor de RA. E a Rodrigo Pinheiro pelo apoio no final da caminhada.

Aos participantes do ensaio avaliatório por sua preciosa colaboração.

Ao pessoal da COPELE pelo auxílio sempre que necessário, especialmente ao prof. Benemar Souza, Ângela, Pedro e Suênia.

À CODATA – Companhia de Processamento de Dados, em especial ao professor Marcos Brasileiro, Alfredo Lucena e a Maria José Paiva pela liberação para cursar o doutorado o que me possibilitou novos estudos. Ao pessoal da AESA - Agência Estadual de Gestão das Águas: José Ernesto Bezerra, Laudízio Diniz e Patrice Rolando pelo apoio decisivo nos últimos meses, e a Carmem Becker, Alexandre Magno, Isnaldo Cândido, Gutembergue Silvino, Carlos Lamarque, Nilton Melo e Roberto Oliveira pelo companheirismo.

Ao professor Hamurábi Medeiros, coordenador do curso de Sistemas e Computação da FFM-Fundação Francisco Mascarenhas, pela força no início do doutorado.

Aos amigos do DSC e do DEE, Andréa, Ayla, Rodrigo, Expedito, Patrício, Roberto, Hellen, Michael, Cláudia, Rute, Luciana, Luis Júnior, Rex, Edmar e Karina pelo tempo de convívio e amizade.

Ao CNPQ, pelo apoio financeiro ao desenvolvimento deste trabalho.

Às pessoas que não tiveram o nome citado aqui, mas que de alguma forma também contribuíram para esse trabalho, meu muito obrigada !!

RESUMO

Freqüentemente no desenvolvimento de projetos ocorrem desperdícios, defeitos e prazos esgotados. Na área de desenvolvimento de software também percebemos isso com muita ênfase, onde projetos complexos são desenvolvidos com dificuldades, apesar de existirem técnicas, modelos ou ferramentas tradicionais de gestão de projetos. *Deadlines* são estendidos e o custo do software acaba sendo incrementado significativamente. Os modelos de gestão de projetos tradicionais não são muito adequados a projetos de grande escala e com demanda de curto prazo. As ferramentas de gerenciamento de projetos não dão cobertura ao acompanhamento de processos e projetos (tempos, custos, recursos e atividades) de software simultaneamente. As metodologias de desenvolvimento de software existentes representam muitos avanços para auxiliar no desenvolvimento de sistemas, mas falta uma utilização mais adequada delas nos projetos, de acordo com a complexidade do tipo de aplicação desenvolvida e suas características específicas. Este trabalho é centrado na extensão de uma rede para gerenciamento de projetos, chamada Rede de Atividades, desenvolvida para detalhar informações sobre o desenvolvimento de processos e projetos, calcular e atualizar custos, tempos, especificar recursos humanos e o desenvolvimento de atividades. Esta rede é o componente principal de um ambiente integrado de gerência de desenvolvimento projetos, denominado GDP, considerando o desenvolvimento de projetos em quatro camadas: (1) determinação da metodologia adequada; (2) modelagem da metodologia; (3) modelagem do projeto específico; (4) execução do projeto. Para determinar a qualidade estrutural de uma Rede de Atividades e permitir simulações e cálculos corretos dos custos e prazos de um projeto, foi desenvolvido um tipo especial de rede de Petri, denominado Rede de Projeto, e determinada a conversão de uma Rede de Atividades em uma Rede de Projeto.

ABSTRACT

Frequently in the development of projects occurs waste of resources, defects and missing deadlines. In the software development area this issue becomes more significant, where complex projects are developed with difficulty, although techniques, models and traditional tools for project management are popular. Deadlines always are extended and the cost of software being developed gowns significantly. The traditional models of project management don't fit very well to large projects with a short schedule. The actual project management tools do not cover the track of both processes (times, costs, resources and activities) and software projects. The actual software development methodologies represent many advances to assist a manager into a software development process, but it lacks a more adequate use of them, in accordance with the specific complexity of the type of application being developed and its characteristics. This work is centered in the extension of a net for project management, called Activities Net, designed to track detailed information about the development of processes and projects, calculate and update costs, allocate human resources and schedule activities. This net is the main component of an integrated environment for project development management, called GDP, based on a four-layer approach: (1) determine de best methodology; (2) model the metodology; (3) model the project; (4) execute the project. In order to determine the structural quality of an Activities Net and allow simulations and correct cost a time calculations, a special class of Petri Nets has been developed, called Project Net (NP). Also the mapping of an Activities Net to a Project Net has been defined.

Sumário

Lista de Figuras

Lista de Tabelas

Glossário

Abreviatura	Descrição
PERT	*Program Evaluation and Review Technique*
GERT	*Graphical Evaluation and Review Tecnhique*
VERT	*Venture Evaluation and Review Tecnhique*
CPM	*Critical Path Method*
PDM	*Precedence Diagram Method*
XP	eXtreme Programming
PCDB	Processo de desenvolvimento baseado em componentes
RUP	*Rational Unified Process*
YP	*easY process*
FDD	*Feature Driven Development*
MSF	*Microsof Solutions Framework*
GAIA	*Generic Arquitecture for Informaton Availability*
MDRAP	*Multi-agente Dinamic Resource Allocation Planning*
AUML	*Agente Unified Modelling Language*
CommonKADS	*Conceptual Modelling of knowlege systems*
CoMoMAS	*Conceptual Modelling of MultiAgent Systems*
MAS-CommonKADS	*Multi Agent System*
GENESIS	*Generalised eNviroment for process management in cooperative software enginnering*
ODE	*Ontology- based software Devlopment Environment*
VSTS	*Visual Studio Team System*
TABA	Meta Ambiente para instanciação de ambientes de desenvolvimento de software convencionais e orientados a domínios
PMBOK	*Project Management Body of Knowledge*
CMMI	*Capability Maturity Model Integration*
MPS.BR	Melhoria de Processos de Software Brasileiro
PRINCE 2	*PRoject IN Controlled Enviroments*
SPICE	*Software Process IMprovement and CApability dEtermination*
TSP	*Team Software Process*
PSP	*Personal Software Process*
PMI-OPM3	*Project Management Institute*
RA	Rede de Atividades
RP	Rede de Projeto
SAD	Sistema de Apoio à Decisão
CHESF	Companhia Hidro –Elétrica do São Francisco
ERA	Editor de Rede de Atividades
SISEM	Sistema para Seleção de Metodologias de Desenvolvimento de Software

Capítulo 1

Introdução

Um projeto consiste em uma proposição de realização de certo objetivo pré-definido. A realização do objetivo consiste na execução de uma série de atividades com a finalidade de se conseguir este objetivo, que pode ser chamado de produto do projeto. Portanto, a execução tem um início e um fim, tem um orçamento a ser executado, uma equipe formada por um coordenador e desenvolvedores da equipe, e tem recursos necessários para construir o produto. A forma de executar o projeto segue um processo bem definido, chamado de metodologia de desenvolvimento. Assim, projetos semelhantes poderão seguir a mesma metodologia, ou seja, dependendo do tipo de projeto, deverá ser adotada a metodologia adequada.

Podemos distinguir três atores principais envolvidos na realização de projetos. O gerente de processos, que conhece bem as diversas metodologias de desenvolvimento e deve enquadrar um novo projeto na metodologia adequada. O gerente de projeto, responsável pela execução adequada de um projeto específico e os desenvolvedores, que irão executar as tarefas que compõe o projeto.

Freqüentemente, no desenvolvimento de projetos, ocorrem desperdícios, defeitos e prazos esgotados. Na área de software também ocorre isso com muita ênfase, onde projetos complexos são desenvolvidos com dificuldade, e falta um acompanhamento detalhado das fases de desenvolvimento.

Entre os principais problemas que um gerente de projetos enfrenta, estão os relacionados à estimativa dos custos e prazos do projeto. “Influenciando estes prazos e suas estimativas, tem-se ainda a composição da equipe de projeto, o desenvolvimento e o acompanhamento das etapas do projeto” (Campos, 2004).

A presente tese tem como contexto o acompanhamento do desenvolvimento de

projetos, em particular, projetos de software. Pretende-se contribuir para o gerenciamento de processos e de projetos, em particular, no controle de custos e prazos. Foi extendido uma estrutura de modelagem das atividades de um projeto, denominada Rede de Atividade – RA. Além disso, foi proposto um ambiente de desenvolvimento de projetos denominado GDP - Gerência do Desenvolvimento de Projetos.

Com a Rede de Atividades é possível realizar a modelagem das etapas de metodologias de desenvolvimento de software e definir atividades. Ela foi estendida para permitir modelar, além do acompanhamento detalhado do processo de desenvolvimento, a interdependência de todas as atividades com os respectivos custos, prazos e recursos alocados, calcular tempos globais e um confronto contínuo dos dados reais com os dados previstos.

Para garantir a composição precisa das atividades para calcular custos e prazos globais, foi criado um tipo especial de rede de Petri, denominada Rede de Projetos, que permite analisar as características de uma Rede de Atividades.

1.1. Contexto e Motivação

1.1.1. Gerenciamento de Projetos

A importância da gestão de projetos cresce exponencialmente face a necessidade sobrevivência das organizações e, desta forma, precisam ser cada vez mais bem sucedidos. Projetos tendem a crescer em complexidade, por envolverem a utilização de tecnologias de ponta, possuírem abrangência multifuncional e serem realizados em ambiente de alta competitividade com disponibilidade decrescente de recursos. Diante disto, pode-se considerar que diversos aspectos de Gerenciamento de Projetos podem ser decisivos para a obtenção do produto final de um projeto.

"O ambiente onde os projetos se desenvolvem está evoluindo em complexidade. Os projetos estão se inserindo em contextos onde a tecnologia avança mais velozmente, os cenários políticos mais diversificados e a estabilidade econômica mais frágil." (Possi, 2006).

Entender como desenvolver e concluir projetos neste contexto é o grande desafio de hoje. Como captar o que foi aprendido no passado, adaptá-lo e aplicá-lo no contexto atual, considerando a pressão constante de geração rápida de resultados?

Basicamente, um projeto é um esforço para criar um produto ou um serviço, e tem um objetivo específico. "Projetos são temporários - eles têm um começo e um fim definidos. Eles são originais, ou seja, o produto ou o serviço que criam são diferentes em

alguma maneira de outros produtos ou serviços similares" (PMBOK, 2004).

Projetos típicos podem incluir a análise, projeto, codificação, testes e documentação de um programa do software, ou a engenharia e a construção de um edifício, ou o desenvolvimento e teste clínico de uma nova droga. A duração de um projeto é o tempo de sua concepção até sua conclusão que pode durar dias, semanas, meses ou anos.

O controle do projeto é o processo de ajustar, durante a execução das fases, o realizado com o planejado, e consiste de: acompanhar ou coletar os dados ou informações sobre o andamento das etapas do projeto; comparar a situação real com a planejada, identificando os desvios e alternativas de correções; selecionar as alternativas de correções e determinar as providências decorrentes e retro-alimentar o projeto se for necessário.

A gerência de projeto tenta garantir o controle do projeto sobre cinco variáveis: a) tempo - requerido para terminar os componentes do projeto, b) custo - o custo para desenvolver um projeto, depende de diversas condições iniciais para o desenvolvimento, tais como: taxas materiais, gerência de risco, equipamentos e lucro, c) qualidade - característica, propriedade ou atributo essencial ou distintivo do produto, d) escopo – envolve as exigências especificadas para o resultado fim, ou seja, o que se pretende realizar, e e) risco – possibilidade de um acontecimento inesperado e externo, causador de danos materiais ou pessoais e de prejuízo ou necessidade econômica (Valeriano, 2005).

Algumas destas variáveis podem ser informadas por clientes externos ou internos. Os valores das variáveis são atualizados então pela gerência de projeto, que se baseia idealmente em métricas e técnicas contínuas de estimativas. Os valores finais têm que ser concordados em um processo de negociação entre a gerência de projeto e o cliente. Geralmente, os valores do tempo, custo, qualidade, risco e escopo são alterados durante o desenvolvimento do produto.

"O custo de desenvolvimento de software corresponde a uma porcentagem cada vez maior no custo total de um sistema informatizado. A principal razão para isto é que a tecnologia de desenvolvimento de software implica em grande carga de trabalho, e os projetos de sistemas de software envolvem em geral um grande número de pessoas num prazo relativamente longo de desenvolvimento" (Standish Group, 2004).

Há muito tempo fala-se nos processos ou metodologias de desenvolvimento de software. A pesquisa em processo de software trata dos métodos e tecnologias utilizados para avaliar, apoiar e melhorar as atividades de desenvolvimento e manutenção de software (Fuggetta, 2000). Uma contribuição importante da pesquisa na área de processo

de software é o convencimento de que desenvolver software é fruto de um esforço coletivo, complexo e criativo e de que a qualidade do software depende das pessoas, da organização, das ferramentas e dos procedimentos usados em seu desenvolvimento.

Assim, diversas ferramentas, conhecidas como ferramentas *Computer-Aided Software Engineering* - CASE, têm sido amplamente utilizadas no desenvolvimento de software. Entretanto, elas geralmente são ferramentas isoladas, não são capazes de compartilhar serviços, ou sequer informações. Muito embora o uso de ferramentas individuais possa trazer benefícios em atividades separadas, o real poder dessas ferramentas de apoio somente pode ser alcançado através da integração (Pressman, 2006).

Desta forma, Ambientes de Desenvolvimento de Software (ADS) vêm ganhando cada vez mais importância. Tais ambientes buscam combinar técnicas, métodos e ferramentas para apoiar o engenheiro de software na construção de produtos de software, abrangendo todas as atividades inerentes ao processo, tais como a gerência, desenvolvimento e controle de qualidade, custos, prazos e recursos.

Muito já vem sendo realizado na área de Engenharia de Software para manter o controle sobre o projeto. Um gerente pode usar técnicas, modelos ou ferramentas tradicionais de gestão de projetos, tais como: *Program Evaluation and Review Technique* - PERT e o *Critical Path Method* - CPM (Moder, Phillips e Davis, 1983). As técnicasPERT/CPM foram estabelecidas desde 1958, mas continuam sendo usadas gestão de projetos, apesar das suas limitações.

Hoje existe um grande número de guias e técnicas que têm sido desenvolvidas e que podem ser usadas para formalizar especificamente como o projeto será gerenciado: o *Project Management Body of Knowledge* - PMBOK (2004a), algumas idéias do *Personal Software Process* – PSP (2005), *Team Software Process* - TSP (2006) e PRINCE2 (2005). Estas técnicas vêm sendo utilizadas por equipes de desenvolvimento para ajudar a prever e gerenciar o projeto. Outros estudos recentes mostram como promover o conceito do controle do projeto através da incorporação da gerência de processo. Esta área foi dirigida pelo uso de modelos de maturidade tais como o CMMi - *Capability Maturity Model Integration* (Pinto, 2006) e ISO/IEC 15504 SPICE - *Software Process Improvement and Capability Determination* (SPICE, 2005). Estes modelos estão sendo adotados por organizações em um esforço de se ter um melhor controle sobre projetos e processos com o objetivo de melhorar a exatidão da estimativa, reduzir custos e impedir defeitos.

Especialistas em gerência de projetos vêm tentando identificar metodologias ágeis "de pouco peso" para o desenvolvimento de projetos de software, tais como: *eXtreme Programming* – XP (2003) e *Microsoft Solutions Framework* - MSF (2005),

entre outros, além de ferramentas de software para automatizar estes processos (XPlanner, 2003; RUP Builder (2004) e Visual Studio Team System (2008).

Porém, alocar tempo de projeto e acompanhar o desenvolvimento de um software, os custos e a data de entrega do mesmo para o cliente, é um processo delicado que envolve muita sensibilidade do gerente de projetos de software e, principalmente, uma excelente capacidade de estimativa e controle, o que não é fácil.

1.1.2. O Sucesso de Projetos Desenvolvidos

Não existe outro setor que tenha se desenvolvido e evoluído tanto e em um ritmo tão devastador quanto o de tecnologia (IEEE, 2001). E, particularmente, quando nos referimos ao desenvolvimento de software, no que diz respeito à taxa de sucesso de projetos.

Segundo estudos do *Standish Group International Inc.* (Standish Group, 2004), uma organização que coleta dados relativos a projetos de software, os projetos podem ser classificados em três categorias:

- Projeto de sucesso: quando é concluído dentro do prazo, com os custos previstos e de acordo com os requisitos dos usuários;
- Projeto com problemas: quando é concluído, mas com atraso, ou acima do custo esperado, ou incompleto em suas funcionalidades requeridas;
- Projeto fracassado ou cancelado: quando nunca é implementado ou quando é cancelado.

Desde 1994, o *Standish Group* faz pesquisas sobre projetos de Tecnologia da Informação. No primeiro dos relatórios, publicado em 1995, identificou-se que bilhões de dólares estavam sendo desperdiçados em projetos que eram cancelados ou concluídos sem todas as funções e características requeridas pelos usuários, daí o nome CHAOS. Nos anos posteriores, resultados melhores foram detectados, mas ainda são preocupantes, conforme pode ser visto na Tabela 1.1.

O relatório de 2004 relata a análise feita em 13.522 projetos e estabelece uma comparação com os números obtidos em 1994. Em pesquisas ao longo de 2004, foi verificado que apenas 29% dos projetos obtiveram sucesso, enquanto que 53% tiveram alguns problemas como atraso, alta nos custos e diminuição das funcionalidades, porém, entregues e 18% falharam por completo ou os clientes desistiram do projeto. É interessante notar que a quantidade de projetos

com problemas permaneceu basicamente estável no período avaliado, em torno de 50%. Ainda nesta pesquisa, foi observado que 36% utilizaram linguagem padrão, 19%

utilizaram as metodologias tradicionais, enquanto que 16% compraram ferramentas e tentaram usá-las sem sucesso.

De acordo com o *Standish Group*, a taxa de sucesso em projetos de tecnologia da informação ainda é baixa. Houve uma melhoria em relação ao mesmo relatório na versão de 1994, conforme pode ser observado na Tabela 1.1., mas ainda considerada insuficiente em função do total de projetos da amostragem. Na mesma pesquisa, ainda mais da metade dos projetos estavam de alguma forma apresentando problemas ligados a prazo, escopo ou orçamento.

Tabela 1.1 Comparação de percentuais de sucesso entre 1994 e 2004.

Fonte: Standish Group (2004)

	1994	**1998**	**2000**	**2004**
Projeto de Sucesso	16%	26%	28%	29%
Projeto com Problemas	53%	46%	49%	53%
Projeto Fracassado	31%	28%	23%	18%

Um estudo do *Standish Group*, divulgado em abril de 2006, revelou que são gastos mais de U$ 250 bilhões de dólares anualmente no desenvolvimento de aplicações na área de Tecnologia da Informação e desperdiçados, porque as empresas falham na utilização de efetivas práticas de gerenciamento de projetos, e apresentou também que: "31% de todos os projetos são cancelados antes de seu término; 88% dos projetos ultrapassam seu prazo, orçamento ou ambos; os projetos ultrapassam, em média, 189% dos custos originalmente estimados; e os projetos ultrapassam, em média 222% do prazo originalmente estimado." (Standish Group, 2006)

Os resultados dos estudos descritos e expostos na Tabela 1.1 foram obtidos a partir de organizações localizadas em países da América do Norte (58%), Europa (27%) e resto do mundo (15%) no estudo de 2004 (Standish Group, 2004).

Foi realizada uma *survey* (pesquisa), entre 2005 e 2006 através de questionários (Farias, 2007a), para encontrar os maiores problemas das instituições que desenvolvem software no Brasil. O questionário criado (Anexo I) foi enviado para 23 (vinte e três) núcleos do SOFTEX – Sociedade Brasileira para Promoção da Exportação de Software, incluindo o núcleo de Campina Grande - PAQTC PB - Fundação Parque Tecnológico da Paraíba.

Solicitamos que os representantes dos núcleos enviassem os questionários para os gerentes das empresas de software associadas. De uma maneira geral, os questionários deveriam ser enviados para 200 (duzentas) empresas. O questionário foi respondido por

20 (vinte) gerentes de projetos de TI das empresas associadas ao SOFTEX – Sociedade Brasileira para Promoção da Exportação de Software.

O questionário possui 8 (oito) questões. As sete primeiras questões são objetivas, e tratam de metodologias, padrões, ferramentas e técnicas utilizadas. A última questão é composta ainda de 10 (dez) perguntas objetivas para identificar quais os problemas que elas enfrentam. Possui ainda 2 (duas) questões subjetivas, e a partir delas nós coletamos comentários sobre o desenvolvimento de *software*.

Os 10 (dez) problemas listados são relacionados às etapas do ciclo de vida do *software*. Analisando os resultados da pesquisa, identificamos os maiores problemas encontrados durante o desenvolvimento de software e encontram-se listados na Tabela 1.2. Mais informações e comentários desta pesquisa encontram-se no Anexo I e em (Farias, 2007).

O primeiro lugar nas respostas foi o problema da falta de um controle efetivo de custos nos projetos e a falta de ferramentas para auxiliar no gerenciamento dos projetos e no direcionamento do processo adotado, em todas as etapas do desenvolvimento.

Tabela 1.2. Ranking de problemas Fonte: (Farias, 2006).

Ranking	Percentual	Problemas possíveis
1	20%	Falta de um controle efetivo de custos.
	20%	Ferramentas de gerenciamento de projetos não possuem todas as funcionalidades necessárias.
2	15%	Falta de um controle de acompanhamento de múltiplas atividades paralelas.
	15%	Falta de um controle efetivo de cronogramas.
3	11%	Metodologia inadequada ao tipo de aplicação/ ambiente de desenvolvimento.
4	6%	Conflitos entre analistas e programadores.
5	5%	Padrão de gerenciamento de projeto inadequado.
6	3%	Falta de um processo de desenvolvimento claramente definido e efetivo.
	3%	Conflitos entre analistas e clientes.
7	2%	Clientes insatisfeitos, má usabilidade.

Apesar da reduzida população de estudo (20 respondentes), uma pequena amostra não probabilística e para inferir e generalizar os dados, pois foram obtidas respostas de menos de 10% da amostra, a pesquisa serviu para obtermos um *ranking* de problemas reais existentes nas empresas de desenvolvimento de software brasileiras.

A análise dos resultados permitiu identificar dificuldades reais, e desta forma, o

estudo mostrou indícios de que novas técnicas para um controle mais efetivo de custos, cronogramas e tarefas, além de ferramentas para auxílio durante o processo de desenvolvimento podem ser viáveis, assim como apontou algumas direções em que a nossa pesquisa de tese deveria avançar.

1.1.3. Declaração dos Problemas

Durante o planejamento e realização da pesquisa mencionada na seção 1.1.2, observamos alguns problemas essenciais que atingem o processo de desenvolvimento de software e sobre os quais vamos trabalhar:

Problema 1 - Freqüentemente no desenvolvimento de projetos ocorrem desperdícios, defeitos e prazos esgotados. Projetos complexos são desenvolvidos com dificuldade, apesar de já existirem modelos de gestão para gerenciamento de projetos (Meredith e Mantel, 2000; Jeetendra, Chetty e Reddy, 2000; Chen, Hsu e Chang, 2008) e modelos formais (Jeetendra, Chetty, e Reddy, 2000; Zhu e Schnieder, 2000).

Problema 2 - Ferramentas de gerenciamento de projetos são geralmente pontuais, com foco limitado e priorizam uma variável específica (tempo ou custo ou recurso). Mesmo as mais abrangentes não são genéricas o suficiente para serem usadas para qualquer tipo de projeto. No cotidiano das organizações, não são encontradas ferramentas que levem em consideração aspectos como: **modelar etapas** de metodologias de desenvolvimento de software (ODE, 2005; Aversano, 2004) **e gerenciar projeto** (tempos, custos e recursos) (MS Project, 2003; Analyzer, 2003; MinuteMan, 2003) simultaneamente, e baseadas em métodos formais, tais como redes de Petri.

Problema 3 - Escolha de metodologia inadequada para um projeto de desenvolvimento. Muitas vezes o gerente de projeto utiliza uma metodologia para desenvolvimento de seu sistema, sem considerar as características específicas deste sistema e do ambiente onde ele será desenvolvido (complexidade, orçamento, tamanho da equipe, agilidade, reusabilidade e ambiente, entre outros) (Farias, 2006; Farias e Schiel, 2008).

Estes três problemas são levados em consideração nesta tese e as soluções prováveis são apresentadas na seção 1.1.7.

1.1.4. Trabalhos Relacionados

Neste item são apresentados os trabalhos relacionados com os problemas anteriormente descritos.

Em estudos sobre gerência de projetos (Meredith e Mantel, 2000; Jeetendra, Chetty e Reddy, 2000), notou-se que a maioria dos modelos baseados em PERT/CPM não é muito adequada a projetos de grande escala e com demanda de prazo pequena. "Estes modelos tradicionais de gestão de projetos têm foco na programação (*scheduling*) do projeto sem considerar se os recursos estão disponíveis e não permitem modelar situações dinâmicas" (Chen, Hsu e Chang, 2008).

Redes de Petri, por outro lado, são conhecidas pela sua capacidade em modelar atividades concorrentes, simular a evolução dos processos e compartilhamento dos recursos.

Existem alguns esforços (Jianxin e Juanwei, 2005; Jeetendra, Chetty e Reddy, 2000) para uso de redes de Petri para planejamento e alocação de recursos no gerenciamento de projeto e modelagem de sistemas complexos, estes trabalhos são citados a seguir e detalhados no Capítulo 3.

Os trabalhos de (Chen, Hsu e Chang, 2008), (Kim, Desrochers e Sanderson, 1995) e (Meredith e Mantel, 2000) apresentam soluções para modelar o controle de recursos e a divisão de tarefas. Jeetendra, Chetty, e Reddy (2000) elaboraram uma sintonia através de *P-matrix* para permitir determinar caminhos críticos e *deadlock*. Chang e Christensen (1999) introduziram uma técnica chamada *Net Practice for Software Project Management Net* para gerar alocação de recursos e *schedules* baseada em algoritmos genéticos. O trabalho de Zhu e Schnieder (2000) apresenta uma abordagem para modelar técnicas de desenvolvimento de software orientadas a objetos (classes e objetos) e diagrama de fluxo de dados (DFD). O trabalho de Wang, Xuw e Yang (2007) apresenta um modelo de rede de Petri como modelo de *workflow* de gerenciamento direcionado apenas à pesquisa científica.

Observa-se que, estas extensões de redes de Petri tratam de características específicas do gerenciamento de projeto (tempo e recurso, ou só tempo) atuando apenas com um foco isolado e não foram desenvolvidas para gerenciamento de processos de desenvolvimento de software (XP, RUP e SCRUM, dentre outros).

Existem ferramentas que auxiliam o processo de desenvolvimento de software tais como: Rational Unified Process Builder (2004), XPlanner (2005) e Microsoft Visual Studio Team System (2008). Elas possuem funcionalidades para acompanhamento dos processos RUP, XP e MSF respectivamente. Mas não permitem o uso para vários tipos diferentes de processos e não controlam todas as variáveis de projeto (custo, tempo e

recursos) (Farias e Schiel, 2007; Farias e Schiel, 2008).

Outras ferramentas, tais como: GENESIS (Aversano, 2004), ODE (2005), APSEE (2004), TABA (2004) e dotProject (2007) permitem acompanhamento de processos distintos. Em TABA e dotProject não há nenhum recurso visual para a modelagem de processo, cabendo ao gerente mapeá-lo usando uma estrutura de formulários e não um diagrama visual. As redes que são usadas nas ferramentas GENESIS, ODE e APSEE não seguem um modelo formal de redes de Petri, elas foram criadas especificamente para as ferramentas e não provêem recursos que permitam o acompanhamento do processo, confrontando o planejado com o realizado, necessário para a criação de uma ferramenta real de controle de projetos.

As ferramentas comerciais, por exemplo: Microsoft Project (2005), ManagePro Advisor (1999); Analyzer (2003), MinuteMan (2003) e PlanView Web Software (2005), dentre outras, permitem descrever tarefas, custos, geram relatórios de cronogramas, gráficos de barras, e permitem o compartilhamento do trabalho entre gerente e equipes. Elas apenas coletam, agrupam e organizam dados do projeto, mas não permitem modelagem e acompanhamento de metodologias de desenvolvimento de software.

O que se observa é que muitas vezes são necessárias duas ou três ferramentas dentre as citadas, em um mesmo projeto para gerenciar todas as etapas do **processo** de desenvolvimento **e** todos os dados do **projeto** (atividades, tempos, custos e recursos) realizando totalizações e simulações, simultaneamente.

1.1.5. Declaração dos Pressupostos da Pesquisa

Os pressupostos que motivaram o desenvolvimento desta pesquisa estão apresentados a seguir:

Pressuposto 1

O desenvolvimento de um projeto é composto essencialmente por uma seqüência de fases que, por sua vez, são compostas por atividades. As atividades do projeto podem ser independentes ou dependentes entre si. Existem formalismos, como redes de Petri, que permitem descrever com precisão estas dependências. Assim, quantificar um projeto como um todo pode ser reduzido a quantificar cada atividade individualmente e analisar as dependências entre elas.

Pressuposto 2

Como qualquer tipo de projeto possui muitas características comuns é possível abstrair os detalhes dos tipos de projeto e criar um metamodelo genérico de um projeto que será usado como guia para cada tipo de projeto.

Pressuposto 3

Há diversos sistemas sendo criados por empresas e instituições utilizando diferentes paradigmas e tecnologias que exigem metodologias de desenvolvimento de software mais adequadas para seu sucesso.

1.1.6. Hipótese

Considerando os pressupostos, observamos que é viável uma nova abordagem de gerência de projetos e processos, baseada em redes de Petri, que permita a criação de um modelo para mapear um processo de desenvolvimento de software, fornecendo informações precisas sobre o andamento do projeto.

1.1.7. Soluções Propostas

A partir dos problemas e pressupostos apresentados anteriormente, concluímos que, para se conduzir um projeto complexo de forma bem-sucedida, deve-se conhecer: o escopo do trabalho a ser realizado, os padrões de referência, os métodos, os riscos, os recursos exigidos, as metodologias de desenvolvimento de sistemas, as tarefas a serem executadas, a programação a ser seguida durante um período de tempo, e o custo despendido.

Uma solução para o problema 1 – *Projetos complexos são desenvolvidos com dificuldade, apesar de já existirem modelos de gestão para gerenciamento de projetos e trabalhos de gerenciamento com métodos formais,* é obtida com a **Rede de Atividades** estendida nesta tese, e a **Rede de Projetos** aqui proposta, e permitem modelar metodologias de desenvolvimento, as atividades e suas interdependências funcionais e acompanhar todas as etapas do processo de desenvolvimento. A Rede de Atividades está de acordo com o pressuposto 1, segundo o qual é possível modelar **qualquer** processo de desenvolvimento de projetos com um único modelo e, portanto, criar uma ferramenta genérica para o gerenciamento adequado de qualquer tipo de projeto.

Uma solução para o problema 2 - *Ferramentas de gerenciamento de projetos são geralmente pontuais, com foco limitado e priorizam uma variável específica,* é obtida através da anális e implementação ambiente de **Gerência de Desenvolvimento de Projetos – GDP** que permite acompanhar processos de desenvolvimento e controlar custos, tempos e recursos do projeto. O GDP está de acordo com o pressuposto 2.

Uma solução para o problema 3 - *Escolha inadequada de metodologia ao projeto*, citado anteriormente, é um sistema de suporte a decisão, denominado **SiSeM - Sistema para Seleção de Metodologias de Desenvolvimento** que faz parte do ambiente proposto - GDP. Este sistema auxilia na escolha da metodologia mais adequada a cada tipo de aplicação para que os engenheiros de software possam identificar qual das metodologias é melhor para desenvolver o seu projeto, levando em consideração as características particulares do projeto a ser desenvolvido, bem como o ambiente onde será empregada essa metodologia. O SiSeM está de acordo com o pressuposto 3.

O ambiente proposto, que integra Rede de Atividades e SiSeM é denominado GDP - Gerência do Desenvolvimento de Projetos. A Rede de Atividades é fundamentada em um tipo de rede de Petri aqui desenvolvida, denominada Rede de Projeto.

1.2. Objetivos do Trabalho

1.2.1. Objetivo Geral

Criar uma abordagem de gerência de projetos com um acompanhamento detalhado de seus custos e prazos baseado em técnicas formais de Redes de Petri e um ambiente de desenvolvimento de projetos independente do tipo de projeto a ser desenvolvido e da metodologia de desenvolvimento adotada.

1.2.2. Objetivos Específicos

Os objetivos específicos deste trabalho são:

1. Extensão de uma estrutura denominada **Rede de Atividades** para acompanhamento de variáveis de um projeto. Estas variáveis devem incluir recursos, prazos e custos alocados a cada atividade do projeto;
2. Definição formal da Rede de Atividades, que permite descrever as principais interdependências entre as atividades de um projeto e determinar as características de um projeto bem estruturado;

3. Para garantir a composição precisa das atividades para calcular custos e prazos globais, deve ser criado um tipo especial de rede de Petri, denominada **Rede de Projetos**;
4. Criação de uma ambiente de suporte ao processo de gerência do desenvolvimento de aplicações em quatro níveis: i) escolha da metodologia adequada ao projeto, através de um sistema de suporte à decisão; ii) modelagem da metodologia; iii) adaptação e instanciação da metodologia para o projeto a ser desenvolvido; e iv) execução e simulação da rede de acordo com o desenvolvimento do projeto;
5. Validação do ambiente proposto num estudo de caso real.

1.3. Metodologia da Pesquisa

Os passos realizados para alcançar a solução dos problemas expostos são apresentados a seguir:

1.3.1. Revisão bibliográfica e pesquisa de campo

Foi realizada uma revisão bibliográfica sobre gerenciamento de projetos, tipos de redes de planejamento, modelos de gestão de projetos, métodos, metodologias de desenvolvimento de software, padrões para gerenciamento de projetos e trabalhos relacionados, com o objetivo de estudar os principais conceitos das áreas relacionadas à tese e também para investigar o estado da arte dos temas envolvidos. Foi realizada também, uma pesquisa através de questionários com instituições da área de desenvolvimento de software brasileiras para investigar a existência de dificuldades no desenvolvimento de software.

1.3.2. Definição dos critérios de avaliação

Foram definidos, com base na pesquisa e na literatura especializada, os critérios a serem utilizados para avaliar trabalhos correlatos e a solução proposta nesta tese. As principais ferramentas e trabalhos acadêmicos relacionados a gerenciamento de projetos, de processos e de modelagem foram avaliados, com base nos critérios pré-definidos na

análise requisitos e em critérios selecionados a partir do SEI (*Software Engineering Institute*).

1.3.3. Escolha do tipo de rede, realização de novas definições para gerenciamento de projetos

Foram pesquisados tipos de redes para auxiliar no controle dos projetos. Foi escolhida a Rede de Atividades, que foi expandida com novos elementos de modelagem, novos conceitos, novos componentes e definições formais para a rede. Foi definida também um tipo específico de rede de Petri, denominada Rede de Projeto, foram criados algoritmos de cálculos de tempos e custos, e a conversão de uma Rede de Atividades para uma Rede de Projeto.

1.3.4. Análise e desenvolvimento de um ambiente de gerência de desenvolvimento de projetos

Foi realizada a análise e o desenvolvimento de dois módulos do ambiente GDP - Gerência de Projetos de Software (SiSeM - Sistema para Seleção de Metodologias de Desenvolvimento de Software e ERA - Editor de Redes de Atividades) para auxiliar no controle e acompanhamento informatizado do desenvolvimento de projetos.

1.3.5. Avaliação do ambiente proposto

O ambiente proposto foi utilizado com um projeto real, pelo gerente e equipe de desenvolvimento do Projeto SAD – Sistema de Apoio à Decisão da CHESF para verificar se a hipótese e os objetivos pré-definidos nesta tese são alcançados com base no projeto. Desta forma, a Rede de Atividades e o módulo ERA foram avaliados com base nos critérios definidos.

1.3.6. Estudo avaliatório

Foi realizado um estudo avaliatório com o projeto SAD que está sendo desenvolvido na UFCG - Universidade Federal de Campina Grande, no Departamento de Sistemas e Computação, laboratório SAD, para verificar qual avaliação os usuários fazem das redes e do ambiente proposto. O estudo de caso previu uma descrição de como criar e usar a

Rede de Atividades, como adaptar uma RA para um projeto específico, instanciar, executar e realizar simulações da rede no ambiente proposto nesta tese em um contexto real.

1.4. Contribuições da Pesquisa

As principais contribuições pretendidas com esta pesquisa são:

- Extensão e especificação formal de estrutura denominada Rede de Atividades para descrever os detalhes das atividades de desenvolvimento de projetos, e sua conversão para Redes de Projeto;
- Proposição de um tipo especial de rede de Petri, denominada Rede de Projeto;
- Proposta de um ambiente de suporte ao desenvolvimento de qualquer tipo de projeto, denominado GDP, considerando quatro niveis desde a caracterização do tipo do projeto até sua execução;
- Desenvolvimento de um Editor de Redes de Atividades do ambiente GDP a fim de modelar as etapas do processo de desenvolvimento de atividades de um projeto, alocar atividades e recursos humanos, auxiliar no controle de custos, prazos, totalizar tempos e custos previstos, realizados e remanescentes;
- Criação de um sistema de suporte a decisão, denominado SiSeM - Sistema para Seleção de Metodologias para escolha da metodologia mais adequada ao desenvolvimento de sistemas de software.

1.5. Limitações da Tese

Embora o ambiente proposto para dar suporte ao gerenciamento de projetos permita acompanhar e controlar o processo de desenvolvimento de um projeto, ele não oferece suporte ao acompanhamento do código fonte que está sendo desenvolvido pelo programador e o módulo de Edição de RA possui algumas dificuldades de usabilidade verificadas no estudos de caso, precisando de alterações.

Apesar da proposta da tese ser de servir para a gerência de qualquer tipo de projeto, não foram estudados a fundo possíveis características específicas de projetos que não sejam da engenharia de software.

1.6. Estrutura do Documento

Este trabalho possui, além deste, mais seis capítulos, brevemente apresentados a seguir.

O Capítulo 2 - *Métodos e Metodologias* - apresenta a fundamentação teórica, resultado da revisão bibliográfica dos itens envolvidos neste trabalho: os tipos de Redes de Planejamento, que são formas de representação de projetos através de grafos e metodologias de desenvolvimento de software que servem de base conceitual para esta pesquisa.

O Capítulo 3 - *Gerenciamento de Projetos* – apresenta conceituações de projeto, gerenciamento de projetos, ciclo de vida de projetos, parâmetro de custo, tempo e recursos e padrões de Gerenciamento de Projetos. É descrita uma pesquisa mais criteriosa do estado da arte sobre as soluções que a comunidade vem desenvolvendo para contribuir com o gerenciamento de projetos e processos de software. Neste capítulo são discutidas também ferramentas de gerenciamento de forma a expor as principais diferenças, deficiências e qualidades que serviram de base para esta proposta.

O Capítulo 4 - *Rede de Atividades e Rede de Projeto* - descreve um dos principais focos deste trabalho. Neste capítulo, são descritos o conceito da Rede de Atividades, Rede de Projeto, conversão de Rede de Atividade para Rede de Projeto, definição formal da RA, elementos, e algoritmos para auxílio no controle de custos e tempos, uma visualização tabelar das Redes de Atividades, conceitos da Rede Expandida e um exemplo de aplicação de passos para criação de Rede de Atividades.

O Capítulo 5 - *Gerência de Desenvolvimento de Projetos* - apresenta o Ambiente GDP, a visão geral sobre o sistema, arquitetura, módulos do sistema, perspectivas, limitações, atores do sistema, descrição dos componentes com classes e atributos, requisitos funcionais, requisitos não funcionais, descrição da Interface com o usuário e o estudo de caso com o projeto SAD - Sistema de Apoio à Decisão da CHESF (SAD, 2007). É apresenta uma aplicação da Rede de Atividades na criação de um modelo para a Metodologia XP1 – eXtreme Programming1 utilizada no projeto, adaptação e instanciação da rede com dados reais e totalização de valores.

O Capítulo 6 - *Estudo Avaliatório e Discussão de Resultados* - descreve a metodologia utilizada, analisa e discute os dados coletados com a utilização da Rede de Atividades e ambiente GDP descritos nos capítulos 4 e 5, no ensaio avaliatório do projeto SAD e mostra os resultados obtidos.

O Capítulo 7 - *Conclusões* - apresenta as conclusões advindas dos resultados obtidos com a aplicação do ambiente proposto, avaliação, limitações, principais contribuições desta pesquisa e trabalhos futuros.

No Anexo I encontra-se o questionário e a descrição da *survey* realizada sobre Desenvolvimento de Software (Farias, 2006), o formulário eletrônico e resultados da pesquisa.

No Anexo II encontra-se a documentação do módulo SiSeM.

No Anexo III encontra-se a documentação do módulo Editor de RA - ERA.

No Anexo IV estão os questionários criados e utilizados no estudo avaliatório.

CAPÍTULO 2

Métodos e Metodologias

Neste capítulo, são apresentados conceitos básicos referentes aos métodos de diagramação utilizados para visualização gráfica e acompanhamento de projetos e metodologias de desenvolvimento de software a fim de permitir uma fundamentação teórica antes de se partir para uma avaliação mais criteriosa do estado da arte dos problemas envolvidos, das soluções propostas e dos resultados obtidos.

Métodos de diagramação mostram uma visualização gráfica da seqüência de atividades e o tipo de dependência entre suas atividades através dos eventos de início e término dos vários elementos. Aqui são descritos PERT – *Program Evaluation and Review Technique*, CPM – *Critical Path Method,* PDM – *Precedence Diagram Method*, GERT - *Graphical Evaluation e Review Technique* e VERT - *Venture Evaluation e Review Technique,* conhecidos como modelos de gerenciamento de projetos tradicionais, e Redes de Petri. Antes de descrevermos estes tipos de redes, apresentamos dois conceitos: o de rede de planejamento e o de caminho crítico, pois as redes são baseadas nestes conceitos e, nesta tese, os dois conceitos serão utilizados para o cálculo dos tempos totais das Redes de Atividades.

São descritas metodologias de desenvolvimento de software utilizadas para gerenciamento de processos de software, suas etapas e uma análise entre as metodologias orientadas a objetos, baseadas em agentes e conhecimento, apontando as características de cada uma e os pontos positivos e negativos.

A Rede de Atividades é utilizada nesta tese para modelar metodologias de desenvolvimento de software e para um projeto concreto, indicar aos gerentes de projeto e componentes da equipe quais os passos a seguir.

2.1. Definições

Esta seção tem o objetivo de mostrar a forma como pode ser estruturado o seqüenciamento das atividades de um projeto, levando à geração de um diagrama de rede.

Por meio do processo de seqüenciamento de atividades passa a ser viável a identificação dos diversos relacionamentos lógicos entre atividades, em função das relações de precedência. É preciso não só representar, mas documentar todos os tipos de relacionamentos, suas eventuais exigências de antecipações ou atrasos, e tudo que se faça necessário para que o desenvolvimento do cronograma do projeto seja feito da maneira mais realista possível (Barcaui et al., 2006).

O objetivo final do seqüenciamento das atividades é a obtenção de um diagrama de rede completo, compreendendo todas as atividades do projeto e suas diversas inter-relações. Para tanto, "com o intuito de tentar minimizar problemas futuros indesejáveis e ao mesmo tempo, maximizar as chances de sucesso do seqüenciamento, é importante que o gerente do projeto e sua equipe estejam preparados com todos os documentos, processo adotado e ferramentas necessárias para a geração do diagrama de rede" (Cleland, 1999). Quanto maior o tempo gasto com o planejamento em todas as fases do ciclo de vida do projeto, maiores as chances de sucesso do mesmo.

2.1.1 Rede de Planejamento

Uma rede de planejamento é a representação gráfica de um conjunto de tarefas, tendo por fim atingir um determinado objetivo. Para que se estabeleça uma rede é necessário conhecer (Paula, 2001):

a) Atividades: são tarefas que serão executadas, ou a execução efetiva de uma operação, consumindo tempo e/ou recursos.

b) Seqüência das atividades: determina a ordem de antecedência e subseqüência entre as diversas atividades que serão executadas.

c) Duração de cada atividade: tempo de execução das tarefas, em uma única unidade (horas, semanas, meses, etc.).

d) Evento (ou Acontecimento): é constituído de marcos que caracterizam determinados instantes de um planejamento.

Existem alguns tipos de atividades, a saber:

a) Atividades paralelas: podem ser executadas paralelamente;

b) Atividades dependentes: atividades que só podem ser executadas depois da realização da atividade que a precede;
c) Atividades independentes: atividades que para serem realizadas não dependem de atividades anteriores ou podem estar em um caminho alternativo;
d) Atividades condicionais: atividades que só podem ser realizadas sob certas condições ou datas.

Com o intuito de facilitar a construção de uma rede é preciso observar alguns princípios fundamentais:

1. Relatar as atividades com suas durações, através de experiência e informações de todas as fontes disponíveis;
2. Economizar tempo, verificando se há atividades que podem ser paralelas;
3. Observar que atividade consome tempo e/ou recursos financeiros; eventos não consomem nem tempo nem recursos financeiros;
4. Evento atingido é o que tem concluído todas as atividades que a ele chegam;
5. Uma atividade somente pode ser executada desde que o evento inicial tenha sido atingido;
6. Entre dois eventos sucessivos existe uma e somente uma atividade;
7. Não existe ciclo em uma rede, pois se existisse, teríamos o fato de que uma atividade poderia dar origem a si mesma.

Para a construção de redes de planejamento podem-se utilizar dois métodos:

- Método de Regressão: este método parte do evento final e caminha para o evento inicial. Segue-se da atividade final à atividade inicial do processo;
- Método da Progressão: este método é mais fácil. Parte-se da "esquerda" para a "direita", ou seja, do "início" para o "fim", seguindo a seqüência natural das atividades. É de mais fácil compreensão e, portanto, o mais utilizado.

2.1.2 Caminho Crítico

O caminho crítico corresponde à soma mais desfavorável, ou seja, "a seqüência de atividades em que a soma dos tempos é maior. Qualquer atraso, em qualquer atividade do caminho crítico, determina um atraso correspondente no projeto" (Paula, 2001). Caminho crítico "é todo caminho de maior duração em um projeto, compondo-se, embora não necessariamente, de uma seqüência de atividades críticas". Onde a atividade crítica "é a atividade compreendida entre eventos críticos e correspondendo à maior duração, entre os mesmos, limitada pelos valores de datas mais cedo de início e fim" (Paula, 2001).

Através das ferramentas de software atuais tais como Microsoft Project, descrito no Capítulo 3, podem-se criar redes de planejamento tal como na Figura 2.1, onde são incorporados pelos softwares todos os cálculos de datas, folgas e determinação do caminho crítico.

Na rede da Figura 2.1, podemos concluir que as atividades A, B, D e F pertencem ao caminho crítico do projeto. Podemos concluir também que a soma das durações das atividades A e F é igual à duração prevista para o projeto (14), o mesmo acontecendo com a soma das durações das atividades B, D e F. O projeto neste caso possui dois caminhos críticos: A-F e B-D-F.

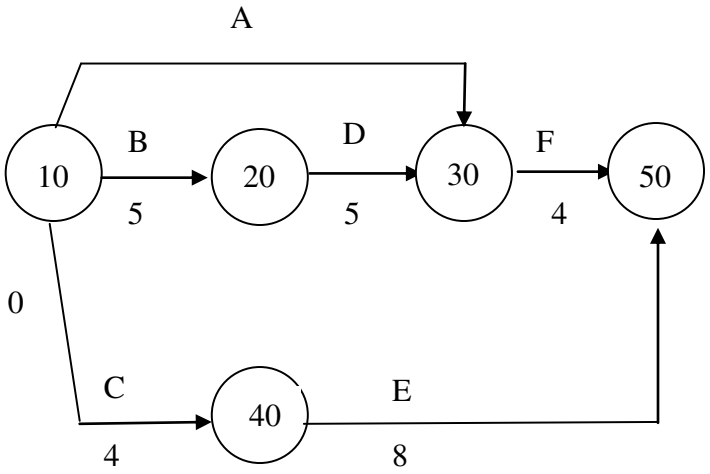


Figura 2.1: Representação de uma Rede PERT com determinação do caminho crítico

Existem diversas métodos de representação de projetos através de gráficos, baseados nos *modelos de gestão de projetos* tais como: PERT – *Program Evaluation and Review Technique* (Cuckierman, 1993 *apud* Paula, 2001), CPM – *Critical Path Method* (Cuckierman, 1993 *apud* Paula, 2001) e PDM – *Precedence Diagram Method* (Barcaui et al., 2006), GERT *Graphical Evaluation and Review Technique* (Pritsker e Happ, 1966) e VERT - *Venture Evaluation and Review Technique* (Moeller, 1972). Elas podem ser aplicadas a diversas áreas na busca de soluções para problemas distintos, por exemplo:

1. melhor distribuição de recursos humanos e equipamentos
2. minimização de custos totais do empreendimento
3. localização de gargalos em processos

Planejamento através de PERT-CPM "consiste em modelar o projeto numa rede ou grafo onde se representam as ações de acordo com as respectivas relações de correspondência, de modo que o conjunto mostre a seqüência em que todas as atividades do empreendimento devam ser executadas" (Cukierman, 1993).

Os métodos PERT, CPM e PDM são chamados de Métodos do Caminho Crítico, porque através do cálculo de suas variáveis, é possível diagnosticar o caminho mais longo do grafo, denominado **Caminho Crítico**, formado pelas denominadas Atividades Críticas (Ichihara, 1999 *apud* Paula, 2001).

2.2. Tipos de Redes

2.2.1. CPM - *Critical Path Method*

CPM foi o termo inicialmente empregado por caracterizar o "tempo determinístico" como atributo de cálculo (Cukierman, 1993).

O método teve origem em 1956, na Companhia *E.I. Du Pont de Nemours,* com o objetivo de reduzir as deficiências de planejamento e controle de projetos e construções (Quezado, 1999 *apud* Silva, 2001). Ele representa o fluxo de um projeto empregando diagrama de flechas, onde as atividades são exibidas através de flechas ou arcos, sendo eles delimitados por nós (eventos). Em geral, a flecha é identificada por uma letra e os nós por números, empregando-se também a identificação da atividade pelo conjunto da numeração dos nós do início e término da mesma.

O nó que origina uma atividade é denominado de evento de início, para essa mesma atividade, e o nó que a finaliza é o evento de término. Denomina-se atividade predecessora aquela que ao ser finalizada desencadeia uma ou mais atividades posteriores. As atividades desencadeadas por uma predecessora são denominadas de sucessoras.

Quando existem atividades que podem ser realizadas paralelamente, precisa-se observar a integridade da numeração dos nós para identificação das atividades evitando duplicidade, como é visto a seguir na Figura 2.2.

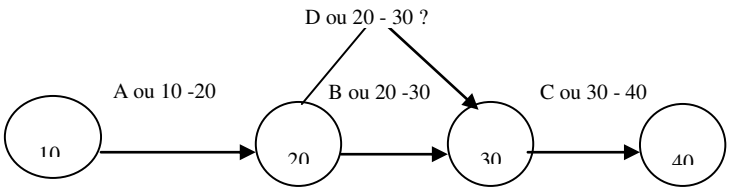


Ilustração 2.2: Diagrama de flechas para três atividades linearmente seqüenciais com inserção errada de uma atividade paralela

Para evitar a duplicidade, torna-se necessária a inclusão de uma nova atividade sem duração e recursos apenas, criada para corrigir o erro da ambigüidade, sendo chamada de "atividade fantasma". Um exemplo dela pode ser visto na Figura 2.3.

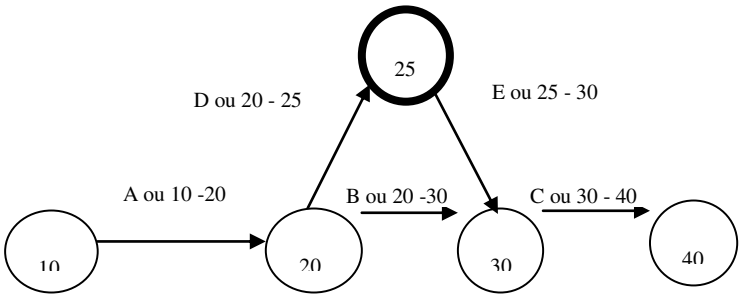


Figura 2.3: Diagrama de flechas para três atividades linearmente seqüenciais com inserção correta de uma atividade paralela

A simbologia do diagrama de flechas cria um esforço adicional que não agrega valor ao processo de planejamento e controle dos empreendimentos em função da necessidade da criação de novas atividades denominadas "fantasmas". Segundo Prado (1998), elas representam em geral de 20% a 40% do número total de atividades de um projeto.

2.2.2. PERT - *Program Evaluation e Review Technique*

Um grafo PERT (Cuckierman, 1993 *apud* Paula, 2001) é uma representação gráfica do relacionamento entre várias atividades, onde as mesmas são operações que consomem tempo e recursos.

O PERT é basicamente um método para analisar as tarefas envolvidas em um projeto, relacionado ao tempo necessário para terminar cada tarefa, e identificar o tempo mínimo para terminar o projeto total.

O método teve origem a partir da necessidade de planejamento e controle do sistema de defesa da Marinha Americana (Quezado, 1999). Sua apresentação seguiu a mesma do CPM.

Os grafos PERT possuem um único nó inicial e um único nó final, o que indica que qualquer caminho que sigamos partindo do evento inicial, sempre nos conduzirá ao evento final.

O uso principal das redes PERT é a determinação do tempo de duração de um projeto pelo método do caminho crítico. A diferença fundamental entre PERT e CPM reside em que o primeiro estima a duração de cada tarefa ou operação dos projetos, baseando-se simplesmente num nível de custo; enquanto que o segundo relaciona duração com custo, do que se deriva uma diversidade de durações para cada atividade e elegendo a duração adequada que fará com que o custo total do projeto seja mínimo.

Os métodos PERT e CPM foram unidos para melhor planejar, seqüenciar e acompanhar atividades em processos por projeto. Na década de 90, PERT/CPM passou a ser considerada uma das sete ferramentas da qualidade e produtividade, ou ainda, uma das sete ferramentas para o planejamento e administração, visando o aprimoramento contínuo da qualidade e produtividade, competências essenciais à competitividade exigida das empresas pelo mundo atual (Cuckierman, 1993 *apud* Paula, 2001).

2.2.3. PDM – *Precedence Diagram Method*

O método do diagrama de precedências teve origem em 1964, na Universidade de Stanford, desenvolvido pelo professor Dr. John Fondahl (Barcaui et al., 2006). Seu desenvolvimento teve como objetivo a simplificação da representação e da metodologia de cálculo do CPM (Caldas, 1998; Limmer, 1997).

O PDM utiliza a representação das atividades através de nós ou blocos, sendo conhecido também por Diagrama de Blocos.

O fluxo do projeto é exibido pelas flechas que conectam as atividades. Uma atividade é denominada de predecessora, e a atividade onde é finalizada uma dependência é a sucessora.

Para o mesmo trecho de um projeto exibido com Diagrama de Flechas da Figura 2.3., podemos observá-lo na Figura 2.4, através do Diagrama de Blocos, percebendo que se torna desnecessária a introdução de atividades fantasmas, o que reduz significativamente o esforço envolvido nos cálculos e no desenho.

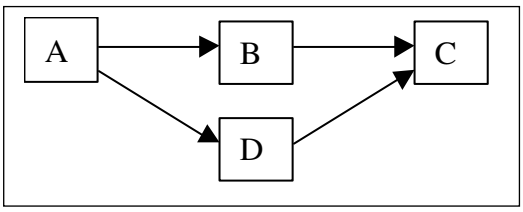


Figura 2.4: Diagrama de flechas para três atividades linearmente seqüenciais com inserção correta de uma atividade paralela

Uma vantagem em relação ao método de diagrama de setas é a representação das atividades nos nós e as relações nas setas. Isto facilita muito a visualização do diagrama e suas interdependências, tornando o processo de planejamento mais intuitivo e prático.

Uma vantagem adicional do Diagrama de Blocos é a facilidade de detalhamento do tipo de dependência entre duas atividades pela combinação dos eventos de início e término de ambas. “Além disso, é possível fazer outros tipos de relações de precedência que não somente início-fim. Tal característica permite uma maior flexibilização do

diagrama de rede, com relações demonstradas de várias formas em função da necessidade de cada projeto e situação". (Barcaui et al., 2006). Por meio de softwares de planejamento, é possível indicar o tipo de relacionamento desejado para cada atividade: fim-início, início-fim, fim-fim, início-fim.

2.2.4. GERT - *Graphical Evaluation e Review Technique*

O Método de Diagrama GERT foi apresentado em meados dos anos 60 como um procedimento para análise de redes com atividades que necessitam de diferentes distribuições probabilísticas (Pritsker e Happ, 1966). GERT é uma técnica gráfica de avaliação e revisão para analisar redes estocásticas (Pritsker e Happ., 1966, *apud* Cates, 2004) na qual é possível a inclusão de derivações probabilísticas, múltiplas saídas, desvios condicionais, efeitos recursivos e nós com repetição de eventos (*loops*). Acompanhanahando Pritsker no desenvolvimento de GERT estavam Happ e Whitehouse (Happ e Whitehouse, 1966 *apud* Cates, 2004)

As características desse tipo de diagramação a tornam bastante flexível e adequada a projetos de pesquisa e desenvolvimento. Mas, no âmbito comercial, é muito difícil observar aplicações que usem este conceito. É possível também que o gerente possa fazer uso de diagramas de rede montados anteriormente para projetos semelhantes ou até mesmo provenientes de seus fornecedores (Barcaui et al., 2006) *apud* Cates, 2004).

No PERT e no CPM todas as mudanças da rede são ajustadas durante o projeto. Entretanto, em GERT probabilidades poderiam ser atribuídas às várias atividades a fim de indicar quais as que podem ou não realmente ser executadas durante o projeto.

Como apresentado originalmente, o procedimento de GERT podia ser usado para determinar o meio e a variação do tempo de terminar uma rede. Observou-se que um trabalho adicional é requerido para poder determinar intervalos de confiança.

GERT foi realçado subseqüentemente com um módulo de processamento de custo (Arisawa e Elmaghraby, 1972) e a habilidade para modelar recursos (Hebert, 1975 *apud* Cates, 2004). Em 1977 o livro de Pritsker intitulado: *Modeling e* Analysis *Using* Q-GERT *Networks* apresentou a versão atualizada de GERT que continha esta e outras melhorias. "Após seu desenvolvimento e com os anos 80, GERT não foi usado extensamente" (Morris, 1997; Pritsker e Whitehorse, 1968 *apud* Cates, 2004). Isto foi devido às complexidades computacionais, aos custos elevados, e ao desempenho limitado dos computadores.

Nos anos 90, com os avanços nas estatísticas e avanços dos computadores, houve o aumento do interesse por análise de risco que é permitida por programação probabilística de redes tais como GERT.

2.2.5. VERT - *Venture Evaluation e Review Technique*

VERT foi desenvolvida para avaliar os riscos envolvidos com novos projetos. Moeller (1972) introduziu VERT que foi atualizado mais tarde por Moeller e Digman (Moeller e Digman, 1982 *apud* Cates, 2004).

VERT é similar a PERT/CPM visto que também está estruturado como uma rede. Entretanto, cada atividade é caracterizada pelos custos e pelo desempenho gerado em adição ao tempo consumido.VERT foi baseado em idéias de que há um relacionamento entre tempo, custo e desempenho. Com VERT, um gerente poderia obter uma análise mais integrada de riscos de um projeto.

Treze tipos de estatísticas, incluindo uniforme, triangular, normal, e *lognormal* estão disponíveis para o uso direto em VERT. Há também uma potencialidade da entrada do histograma para outras estatísticas.

A análise de uma rede VERT é feita através de simulação. Informações do tempo completo, do custo, e do desempenho podem assim ser obtidas. São produzidas distribuições para o tempo e custo da conclusão.

O método de VERT foi desenvolvido para fornecer estimativas exatas do tempo requerido para concluir projetos (Kidd, 1987 *apud* Cates, 2004). Kidd pesquisou o uso de VERT por gerentes de projeto. Ele observou que criar o modelo do projeto baseado em VERT poderia consumir bastante tempo, mas que a habilidade resultante com o processo de simulação do processo com questões "e se" seriam benéficas.

VERT não se transformou em uma ferramenta extensamente usada como PERT/CPM. Ele é usado menos freqüentemente do que GERT (Morris, 1997 *apud* Cates, 2004).

2.2.6. Redes de Petri

Redes de Petri (*PN - Petri Nets*) são redes que permitem descrever sistemas. Elas são indicadas para a modelagem, especificação e análise de sistemas concorrentes. Elas constituem um formalismo extremamente atrativo para a modelagem e análise de sistemas de software distribuídos e concorrentes.

Estas redes (Murata, 1989) são utilizadas para modelar situações que dificilmente são representadas por outros modelos como concorrência, paralelismo, sincronização entre vários processos e competição por um recurso. Por isto, elas são consideradas importantes para descrever o comportamento de sistemas complexos.

Definição

Formalmente, uma rede de Petri é definida como uma tripla (Reisig, 1985):

$N = (P, T, F)$

$P = \{P_1, ..., P_m\}$ é um conjunto finito de lugares (*places*).

$T = \{t_1, ..., t_n\}$ é um conjunto finito de transições.

$F \subseteq (PxT) \cup (TxP)$ é um conjunto de arcos.

Para uma rede de Petri pode ser definida uma **marcação** como uma função M: P -> {0,1).

Um *sistema de Petri* é um par <N, M_0>, onde N é uma rede, M_0 é uma marcação chamada inicial, e existe uma regra de transição associada a rede marcada.

A esta estrutura são associadas regras de comportamento. Em uma rede de Petri, os estados estão associados aos lugares e suas marcações, e os eventos às transições. Uma transição *t* está **habilitada** se cada um de seus lugares de entrada $P_i \in \bullet t$ [1] (possuir uma marca e cada lugar de saiida não tem marca.

Estando habilitada, uma transição pode ser disparada. O **disparo** de *t* remove *1 tokens* de cada um de seus lugares de entrada *Pi* e adiciona *w(t, Po) tokens* a cada lugar de saída $P_o \in t\bullet$ [1].

Dado um Sistema <N, M_0>, uma seqüência de marcações $M_1,..,M_k$ do fecho de M_1 é dita um **caminho**, se existem $t_1,..,t_{k-1}$ tal que $t_i(Mi) = M_{i+1}$para i=1,..k-1. Esta seqüência $[t_1,..,t_{k-1}]$ é chamada de um **caminho de transição** de t_1 para $t_{k-1.}$

Para um sistema <N,M_0> a **fecho** de M_0 é o conjunto $/M_0 = \{M \mid$ existe um caminho de M_0 para M$\}$. O fecho junto com as transições correspondentes é chamado de **grafo de cobertura** de $M_{0.}$ Um sistema <N,M_0> é chamado de **vivo** se para todo lugar $p \in P$ existe um $M \in /M_0$ que permite que seja marcado a cada disparo.

Uma notação gráfica de PNs é também muito usada. Nesta notação, os lugares são representados por círculos, as transições por barras ou retângulos, os *tokens* por pontos, e os arcos por setas com os pesos escritos em cima (por definição, um arco não marcado tem peso 1).

---

[1] $\bullet t$ é o conjunto de lugares de entrada da transição t e $t\bullet$ o conjunto de lugares de saída de t. Similarmente, $\bullet P$ e $P\bullet$ são os conjuntos de transições de entrada e saída, respectivamente, do lugar P.

Além do modelo básico, várias variantes e extensões de PN existem na literatura (Murata, 1989): redes temporais, redes de predicado/transição, redes coloridas e redes hierárquicas. Dentre as extensões mais utilizadas destacam-se as redes de Predicado/Transição (Genrich, 1981), e as redes Coloridas (Jensen, 1992), como representantes da classe de redes denominada redes de alto nível – de especial interesse para especificar sistemas de informação.

O modelo básico de PN não faz nenhum tipo de consideração quanto ao tempo de disparo das transições, ou seja, a partir do momento em que estão habilitadas, as transições podem ser disparadas. Uma maneira de incluir a noção de tempo em uma PN é estabelecer um tempo de espera para o *token* em um lugar, antes de ele habilitar as transições de saída (Ramamoorthy, 1980). Também é possível estabelecer funções de probabilidade para o tempo de disparo de uma transição (Bause, 1996). Neste tipo de PN (chamada PN estocástica), a transição dispara algum tempo depois de habilitada, tempo este determinado pela função de

probabilidade associada à transição. O tempo também pode estar associado à "execução" do disparo das transições. Neste caso, os *tokens* ficam nos lugares de entrada esperando o disparo da transição, são retirados deles e algum tempo depois (tempo de disparo) são entregues aos lugares de saída. Este tipo de disparo não-instantâneo é também chamado de disparo com reserva de *tokens*.

Além das interessantes características de modelagem, tais como simplicidade da notação gráfica, formalidade da notação matemática e modelo de descrição hierárquico (encapsulamento de detalhes), as redes de Petri também oferecem importantes ferramentas de análise do sistema modelado. Há três tipos possíveis de análise: verificação, validação e desempenho (Van der Aalst, 1998). As análises de verificação são realizadas para garantir que a rede esteja corretamente definida e corresponda com exatidão ao sistema modelado. Neste tipo de análise é verificado se a rede apresenta *deadlocks*, se atinge algum estado não permitido, etc. A análise de validação testa se a rede funciona como esperado. Os testes são feitos por meio de simulação interativa de situações fictícias para verificar se a rede as trata corretamente. A análise de desempenho avalia a capacidade de o sistema atingir certos requisitos, tais como tempo médio de espera, número médio de casos pendentes, uso de recursos, *throughput times*, etc. Análises de desempenho podem ser feitas por meio de simulação, cadeias de Markov, e outras técnicas (Magalhães, Raposo e Ricarte, 1998), (Marsan, 1984), (Molloy, 1982).

Redes de Petri são utilizadas para a análise e simulação de sistemas, o que, juntamente com as características de modelagem, as tornam ferramentas adequadas para o planejamento e acompanhamento de projetos (Jianxin e Jaunwei, 2005; Jeetendra, Chety e Reddy, 2000).

A próxima tabela mostra as principais características das redes descritas nesta seção.

Tabela 2.3: Resumo das características dos tipos de redes descritas

	CPM	PERT	PDM	GERT	VERT	PETRI
Custo		√			√	√ (redes coloridas)
Tempo de duração do projeto	√	√	√	√	√	√ (redes coloridas)
Derivações probabilísticas				√		
Análise de Risco				√	√	√
Estatísticas					√	
Simulação					√	√
Representação de unidades de recursos						√
Processos concorrentes e conflitos		√				√

A seguir são descritas metodologias de desenvolvimento de software pois a Rede de Atividades (um dos focos deste trabalho) é utilizada para modelar metodologias de desenvolvimento de software e para um projeto concreto, indicar aos gerentes de projeto e componentes da equipe quais os passos a seguir.

2.3. Metodologias de Desenvolvimento de Software

Uma metodologia ou processo de desenvolvimento de software é um conjunto de atividades, parcialmente ordenadas, com a finalidade de obter um produto de software. É um dos principais mecanismos para se obter software de qualidade e cumprir corretamente os contratos de desenvolvimento (Somerville, 2003). Uma metodologia visa atender necessidades de construção, implantação e manutenção de software, de acordo com o tipo de software (comercial, industrial, técnico-científico, "batch", "on-line", monitoração, simulação de processos, aplicativo, básico, utilitário, etc.).

A seguir são descritas as principais metodologias de desenvolvimento de software, pois elas são modeladas através das redes de Atividades para guiar o processo de desenvolvimento dos proejtos de software. Aqui é dada maior com ênfase para XP, pois o estudo de caso baseia-se nela.

2.3.1. EXtreme Programming - XP

Extreme Programming é uma metodologia para desenvolvimento de software que

melhora um projeto de software através de quatro valores: comunicação, simplicidade, *feedback* e coragem (Garcia, 2004).

XP é formado por 4 fases: Planejamento, Modelagem, Codificação e Testes. Em cada uma destas fases encontram-se regras e práticas que são descritas a seguir:

User Stories (estórias do usuário) são escritas pelos clientes com tarefas que o sistema necessita fazer para servir a eles. *User stories* servem ao mesmo propósito como *use cases* em UML, mas não da mesma forma. Eles são utilizados para criar *release planning meeting* (reunião de planejamento). Eles são usados ao invés de um grande documento de requisitos.

Release Planning (reunião de planejamento) é utilizada para criar um *release plan* (plano de liberação) que prepara o projeto global. O plano de liberação é então utilizado para criar *iteration planning* (planos de iteração) para cada interação individual.

Spike Solution: deve-se criar soluções para encontrar respostas para problemas técnicos ou de projeto. Uma *spike solution* é um programa simples para explorar soluções potenciais. Constrói-se um sistema que endereça unicamente o problema sob exame e ignora todos os outros assuntos. A meta é reduzir o risco de um problema técnico ou aumentar a confiança da estimativa de uma *user story*. Após isto, há uma nova reunião de planejamento e a *spike solution* é integrada à *user story*.

CRC Cards: utilizam-se CRC-Cards (*Class, Responsibilities e Collaboration*) para modelar o sistema. O valor maior de cartões CRC está em permitir que pessoas saiam do modo procedural de pensamento e que apreciem a tecnologia de objetos. *CRC Cards* permitem que equipes de projeto contribuam com a modelagem.

Iteration Planning (reunião de planejamento) é realizada no começo de cada iteração para produzir o plano daquela iteração das tarefas de programação. *User stories* são escolhidas para esta iteração pelo cliente.

Acceptance Tests (testes de aceitação): ensaios são criados a partir de *user stories*. Durante uma iteração as user stories selecionadas durante reuniões de planejamento são traduzidas em *acceptance tests*. O cliente especifica cenários para testar quando uma *user story* está corretamente implementada.

Small Releases **(Pequenos Lançamentos):** o time de desenvolvimento necessita liberar versões iterativas do sistema aos clientes freqüentemente.

Collective Code Ownership (Propriedade de Código coletiva) encoraja todos do grupo de desenvolvimento a contribuir com novas idéias para todos os segmentos do projeto. Qualquer desenvolvedor pode mudar qualquer linha de código para somar funcionalidade ou fazer refatoramento.

Unit Tests: primeiro deve-se criar *unit tests* (testes de unidade) para testar todas classes do sistema antes de escrever o código. *Unit tests* são liberados dentro do

repositório de código junto com o código testado. Código sem teste não pode ser liberado.

Alguns dos melhores fatores de XP são suas regras simples e práticas e a forte ênfase nos testes. XP é adequada a projetos onde os requisitos mudem constantemente, mas ela requer um time altamente qualificado, pois não existe a divisão clara de papéis (exemplo: analista ou programador). Ela produz retornos muito rápidos para o cliente, mas requer muita disciplina para o seu sucesso.

XP tem obtido muita atenção por parte dos desenvolvedores de software. Parcialmente por conta do processo rápido. Algumas vezes, entretanto, a utilização de XP torna-se um problema principalmente quando é utilizado em projetos complexos, porque a metodologia não gera muita documentação, requisito importante para este tipo de projeto.

2.3.2. O Processo de Desenvolvimento Baseado em Componentes

Um componente é definido como uma unidade de software independente, que encapsula dentro de si seu projeto e implementação, e oferece interfaces bem definidas para o meio externo. A técnica de desenvolvimento baseada em componentes visa fornecer um conjunto de procedimentos, ferramentas e notações que possibilitem que, ao longo do processo de desenvolvimento software, ocorra tanto a produção de novos componentes quanto a reutilização de componentes existentes.

Aqui são apresentados aspectos relevantes a serem considerados no Processo de Desenvolvimento de Software Baseado em Componentes (*Process Component Based Development* - PCBD) (Dogru et al, 2003). Particular atenção tem sido dada a abordagens como o (D'Souza, 1999; Wills, 1999). Essas abordagens têm a vantagem de utilizar UML - *Unified Modelling Laguage* (Booch, Rumbaugh e Jacobson, 2004).

Catalysis enfatiza os estágios técnicos de especificação de requisitos, especificação do sistema, projeto da arquitetura e projeto interno dos componentes. Ciclos de vida incremental, iterações e *feedbacks*, têm uma gree ênfase em um PCBD.

Os benefícios da reutilização de componentes são a redução dos custos e tempo de produção de software e o aumento da confiabilidade dos produtos de software, uma vez que esses produtos podem ser construídos a partir de partes bem especificadas e testadas.

Mas a obtenção desses benefícios depende ainda do tratamento de questões pendentes como: a formação de políticas gerenciais e organizacionais que incentivem e permitam a reutilização de componentes; a disseminação de arquiteturas de software que facilitem o mercado e o intercâmbio de componentes e técnicas que facilitem a identificação dos componentes adequados às necessidades das aplicações (exemplo:

biblioteca de componentes e boa documentação).

2.3.3. *Rational Unified Process* (RUP)

O Processo Unificado de Desenvolvimento de Software – RUP (Jacobson, Booch e Rumbaugh, 2004) é um conjunto de atividades necessárias para transformar requisitos do usuário em um sistema de software. O processo unificado utiliza UML no preparo da maioria dos artefatos do sistema. Cada ciclo de desenvolvimento de RUP consiste de quatro fases descritas a seguir (Booch, Rumbaugh e Jacobson, 2004).

Iniciação: fase de compreensão do problema e da tecnologia através da definição dos *use cases* (casos de uso) mais críticos. No final desta fase deve-se ter definido o escopo do produto, os riscos e ter demonstrado que o projeto é viável do ponto de vista do negócio da organização.

Elaboração*:* fase de descrição da arquitetura do software na qual os requisitos que mais impactam na arquitetura são capturados em forma de *use cases*. No final da fase de elaboração deve ser possível determinar o cronograma e o plano de construção.

Construção*:* fase na qual o software é construído e preparado para a transição para os usuários. Além do código, são produzidos os casos de teste e a documentação.

Transição*:* fase de treinamento dos usuários e transição do produto para utilização.

Cada fase consiste de uma ou mais iterações. O produto final inclui artefatos, tais como manuais, diagramas e códigos fonte incorporados em componentes que podem ser compilados e executados, e outros artefatos como os requisitos, os casos de uso, as especificações não funcionais e os casos de teste. Inclui também modelos da arquitetura e todos os elementos já mencionados, os quais permitem especificar, projetar, implementar, testar e utilizar o sistema.

Para executar uma nova iteração eficientemente, os desenvolvedores precisam construir os modelos desta iteração: Modelo de Requisitos, de Análise, de Projeto, de Implementação, de Distribuição e de Teste.

Dentre os benefícios podemos destacar a redução dos riscos envolvendo custos a um único incremento, o reconhecimento das necessidades dos usuários e os requisitos correspondentes.

O RUP é personalizável através da criação de instâncias de processos complexos e normalmente requer uma equipe de processo de software na organização. RUP é mais adequado a projetos de maior escala (10-120 pessoas na equipe de desenvolvimento).

RUP apresenta alguns problemas de diversas naturezas, por exemplo, falta uma

integração mais consistente com aspectos de gerência do desenvolvimento de software. E, para certas aplicações, RUP, pela sua estrutura essencialmente tradicional, baseada no modelo cascata, se torna pesado e de retorno lento, o que pode comprometer a qualidade do projeto.

2.3.4. EasY Process (YP)

EasY Process (Garcia, 2005) é um processo simplificado, apoiado em práticas do XP, RUP e *Agile Modeling*.

As etapas do easY Process são descritas a seguir.

Papéis no Processo: um papel constitui um conjunto de responsabilidades que determina qual será o comportamento de uma pessoa durante o processo.

Reunião com o Cliente: a partir da primeira conversa com o cliente, após uma identificação do escopo do problema, um artefato contendo a visão sobre os processos de negócios do cliente deve ser gerado.

Inicialização: logo após a equipe de desenvolvimento adquirir uma idéia geral sobre problema a ser resolvido, devem ser iniciadas algumas atividades de **análise** do sistema.

Planejamento: o planejamento consiste de dois elementos, o de ***release*** e o da **iteração**. Estes descrevem de forma clara um planejamento das atividades que serão desenvolvidas.

Padrões de codificação: antes de começar a codificar, a equipe de desenvolvimento deve definir um padrão e a maneira como devem ser nomeados os métodos e variáveis.

Testes: testar o código significa verificar se a funcionalidade atende ao que foi especificado.

Os **testes de unidade** validam as menores partes do sistema, ou seja, os métodos, classes ou até trechos de códigos confusos.

Os **testes de aceitação** contemplam um conjunto de situações definidas pelo cliente sobre como medir o sucesso do projeto.

Reunião de Acompanhamento: a reunião de acompanhamento deve ocorrer semanalmente, visando avaliar sistematicamente os resultados obtidos no projeto até o momento.

É importante destacar a forte ênfase do YP na interação com o usuário final e na construção de diagramas com modelos que já são de uso da equipe de desenvolvimento. Como destacamos ao longo do texto, os testes também têm uma grande ênfase no YP.

YP propõe que o planejamento seja focado em pequenos *releases,* garantindo

uma maior interação com o cliente. As *User Stories* alocadas em cada *release* só podem ser consideradas como finalizadas após a realização dos testes de aceitação. Um benefício desta prática é a redução do impacto das mudanças de requisitos

Este processo, assim como XP, também não requer um time altamente qualificado, pois pode ser utilizado por estudantes e estagiários. Ela produz retornos muito rápidos para o cliente, mas requer disciplina para o seu sucesso e revezamento da equipe de desenvolvimento nas funções de desenvolvedor e gerente.

2.3.5. Metodologias Crystal

Crystal (2004) inclui um grande número de diferentes métodos que são selecionados de acordo com as características do projeto a ser desenvolvido.

Existem três métodos principais: *Crystal Clear*, *Crystal Orange* e *Crystal Orange Web* (*Crystal Orange* acrescido de práticas específicas para *web*). O *Crystal Clear* foi desenvolvido para projetos muito pequenos, de até 6 desenvolvedores, e de curta duração. Já o *Crystal Orange* é desenhado para projetos de tamanho médio, com um total de 10 a 40 desenvolvedores, e com uma duração de 1 a 2 anos.

O ciclo de vida desta "família de metodologias" é baseado nas seguintes práticas: progresso monitorado por marcos baseados nas decisões das reuniões, ao invés de documentos escritos; envolvimento direto do usuário; testes automáticos de funcionalidades e workshops refletivos.

O ciclo de vida desta família de metodologia é baseado nas seguintes práticas:

Staging: planejamento do próximo incremento do sistema. A equipe seleciona os requisitos que serão implementados na iteração e o prazo para sua entrega.

Edição e revisão: construção, demonstração e revisão dos objetivos do incremento;

Monitoramento: o processo é monitorado com relação ao progresso e estabilidade da equipe e é medido em marcos e em estágios de estabilidade.

Paralelismo e fluxo: em *Crystal Orange* as diferentes equipes podem operar com máximo paralelismo. Isto é permitido através do monitoramento da estabilidade e da sincronização entre as equipes.

Inspeções de usuários: são sugeridas duas a três inspeções feitas por usuários a cada incremento.

***Workshops* refletivos**: são reuniões que ocorrem antes e depois de cada iteração com objetivo de analisar o progresso do projeto.

***Local matters**:* são os procedimentos a serem aplicados, que variam de acordo com o tipo de projeto.

Work Products: seqüência de lançamento, modelos de objetos comuns, manual do usuário, casos de teste e migração de código.

Standards: padrões de notação, convenções de produto, formatação e qualidade usadas no projeto.

Tools: ferramentas mínimas utilizadas. Para *Crystal Clear*: compiladores, gerenciadores de versão e configuração. Para *Crystal Orange*: ferramentas de versão, programação, teste, comunicação, monitoramento de projeto, desenho e medição de performance.

Um dos benefícios da utilização de *Crystal* é que pode-se discutir o projeto e sua criticalidade e prioridade. Pode ser aplicada a equipes pequenas de seis pessoas ou grandes equipes de 10-40 pessoas.

Mas para a boa utilização da metodologia existe uma necessidade de comunicação entre o atual e o futuro *staff* e manter um baixo custo e tempo.

2.3.6. *Feature Driven Development* (FDD)

Desenvolvimento Voltado a Funcionalidades (FDD, 1999) é um método ágil de abordagem adaptativa. Ele não atende todo o processo de desenvolvimento, mas foi elaborado para ser trabalhado junto com outros métodos de desenvolvimento. FDD é um método iterativo que enfatiza tópicos de qualidade e inclui entregas freqüentes de artefatos para monitorar o progresso do projeto.

O desenvolvimento é voltado a *feature,* ou funcionalidade, que representa um requisito funcional do sistema. É definido como sendo mais apropriado a projetos iniciais, à atualização de código existente, à criação de uma segunda versão, ou ainda à substituição de um sistema inteiro em partes.

Diferentemente de outros métodos ágeis, o FDD possui características específicas para desenvolver sistemas críticos. FDD descreve uma série de passos, desde a concepção até a implementação. O ciclo de Vida de FDD consiste em quatro processos seqüenciais:

Desenvolver um modelo geral: definição do domínio do sistema, contexto e requisitos para a construção, assim como uma documentação em forma de casos de uso ou uma especificação das funcionalidades.

Construir uma lista de funcionalidades: construção de uma lista de funcionalidades. Cada funcionalidade é uma função para uma área de domínio estudado.

Planejar por funcionalidade: criação de um plano de alto nível, no qual as funções são seqüenciadas de acordo com a prioridade e a dependência de cada uma, e então designadas ao líder de programadores.

Projetar por funcionalidade e Construir por funcionalidade: São a parte iterativa do método. Um pequeno grupo de funcionalidade é selecionado, assim como as funções necessárias para o seu desenvolvimento. Esses processos podem levar alguns dias, no máximo duas semanas para serem realizados.

Na iteração ocorre: *Design* e *Design Inspection*, que são o projeto e a inspeção de cada tarefa do grupo de funções; *Coding*, criação de código fonte; *Unit testing*, testes das unidades, que são formadas pelas funções; *Integration*, integração das funções; e *Code Inspection*, a inspeção de código. Depois de concluída a iteração com sucesso as funções são levadas à construção da *main build*, enquanto outro grupo de funcionalidade entrará no processo iterativo.

Os benefícios de FDD são:

a) Clientes têm resultados rápidos e relatório do *status* numa linguagem que eles entendem
b) Gerentes de projeto têm uma visão completa e exata do status do projeto
c) Desenvolvedores conseguem trabalhar em novas atividades em poucos dias e ficam mais envolvidos em análise, projeto e codificação.

"FDD é uma metodologia mais indicada para projetos críticos, não se adequando para projetos pequenos e simples" (FDD, 1999).

2.3.7. *Microsoft Solution Framework* (MSF)

Microsoft Solution Framework (MSF, 2005) foi definida para padronização da linguagem dos documentos utilizando UML. MSF originou-se da análise de times de projetos e grupo de produtos. As fases de MSF estão descritas a seguir.

Fase de Visão: os produtos gerados na fase de visão são documentos e diagramas que conseguem ilustrar o escopo do projeto. Esta fase é documentada com os seguintes artefatos: definição dos papéis, Levantamento Técnico Inicial (LTI), diretrizes de Arquitetura e métricas do sistema.

Fase de Especificação: conhecendo o escopo do projeto, detalha-se todo o sistema e utilizam-se as técnicas da UML para documentar os comportamentos, estruturas e a arquitetura do sistema. Os artefatos gerados são: glossário do sistema e protótipo de navegação.

Fase de Desenvolvimento: nesta fase de desenvolvimento, a especificação do sistema é submetida ao time de desenvolvedores que irão construí-lo.

Fase de Estabilização: é a fase em que são feitas entregas (parciais) do projeto ao cliente em três releases.

Planilha de Inconsistências: neste documento são descritas todas as

inconsistências encontradas no sistema, e a planilha é enviada aos desenvolvedores para que eles corrijam os problemas.

Recibos de Release: o principal objetivo é documentar a entrega dos módulos do sistema. Este documento é elaborado e em seguida protocolado pelo cliente.

Fase de Implantação: o sistema é implantado no ambiente do cliente. Nesta fase é fornecido o manual de implantação do projeto, onde estarão documentados todos os pré-requisitos para a instalação do sistema, o processo de instalação e as configurações necessárias.

Fase de Controle: esta fase está presente em todas as fases do projeto. Os documentos gerados são: Cronograma: com base no planejamento criado, é gerado o cronograma detalhado do projeto e o *Task Plan*: gerenciamento e acompanhamento semanal de cada membro do projeto.

Os focos da metodologia são o negócio, a comunicação aberta entre a equipe, clientes e outros componentes do time; a visão de projeto compartilhado onde o processo de compartilhamento de visão de projeto é especificado no início do projeto e o time se comunica para identificar e resolver conflitos; esclarecer as responsabilidades compartilhadas; delegar mais poderes aos membros do time; investimento em qualidade: todo o time é responsável por balancear os custos, e funcionalidades para preservar a solução em qualidade e assegurar a qualidade.

A metodologia auxilia no gerenciamento de projetos, mas para usá-la, a equipe deve ser experiente, integrada e coesa.

Existem ainda outras metodologias baseadas no conceito de agentes. Um agente pode ser definido como uma entidade de software que exibe um comportamento autônomo, que está situado em um ambiente sobre o qual é capaz de realizar ações para alcançar seus próprios objetivos e a partir do qual percebe alterações.

Em Sistemas Multiagentes (SMA), investiga-se o comportamento de um conjunto de agentes autônomos, possivelmente preexistentes, que interagem entre si objetivando a resolução de um problema que está além das capacidades de um único indivíduo (Odell e Parunak, 2000).

A abordagem multiagentes para o desenvolvimento de software tem sido aplicada a muitos domínios, tais como Sistemas Tutores Inteligentes, Gestão de Conhecimento, Recuperação de Informação, Interfaces Adaptativas, Robótica, Simulação, *Web Semântica*, *E-business*, Planejamento de Viagens, Comércio Eletrônico e Redes de Sensores Distribuídos dentre muitos outros. A seguir, encontram-se algumas metodologias baseadas em agentes:

2.3.8. Metodologia Gaia

A metodologia *Generic Arquitecture for Information Availability* (GAIA) proposta por Wooldridge (Wooldridge, Jennings e Kinny, 2004), foi desenvolvida para análise e projeto de sistemas baseados em agentes flexíveis, autônomos e interativos e vem sendo estendida para diversas aplicações (Huang et al., 2007).

Esta metodologia é aplicada a Sistemas Multiagentes e acompanha as fases de análise e projeto. **Na fase de análise**, o objetivo é decompor o sistema em papéis que serão desempenhados pelos agentes na organização (objetivo do modelo de papéis) e definir a maneira como eles interagem de acordo com protocolos específicos (objetivo do modelo de interação). A metodologia apresenta dois modelos para a fase de análise: Modelo de Papéis e Modelo de Interação.

Modelo de Papéis - identifica os papéis existentes no sistema, sendo estas descrições abstratas de uma função pretendida por um agente.

Modelo de Interação - é constituído de um conjunto de definições de protocolo, um para cada tipo de interação entre papéis.

Ao final deste processo, o documento esperado é um modelo de papéis bem elaborado, descrevendo cada papel em termos de responsabilidades, permissões, protocolo de interação e atividades, e um modelo de interação, descrevendo cada protocolo em termos de troca de informações e padrões envolvidos

A **fase de Projeto** utiliza os modelos definidos na fase de análise para gerar novos modelos com níveis de abstração mais baixos, obtendo uma definição do sistema multi-agentes mais próxima da implementação.

Na pesquisa de Wooldridge, são definidos três modelos: Modelo de Agente, Modelo de Serviços e Modelo de Afinidades.

Gaia é uma das metodologias mais completas para o desenvolvimento de sistemas multiagentes, mas existe uma grande distância entre os modelos definidos em suas diretrizes e a implementação do software. As diretrizes não guiam o desenvolvedor para produzir um software e atributos essenciais de soluções de software como flexibilidade e reusabilidade não são considerados.

2.3.9. M-DRAP

Esta metodologia para sistemas multi-agentes foi baseada no processo de modelagem chamado M-DRAP- *Multi-agent Dinamic Resource Allocation Planning* (Choren et al, 2004; Bastos e Oliveira, 2000; Bastos, 1998). M-DRAP define uma abordagem baseada

em agentes para planejamento de alocação dinâmica de recursos em um ambiente de produção de uma organização.

A proposta de Bastos considera a definição de um conjunto de modelos de representação e atividades para o desenvolvimento de SMA:

Definição de Requisitos - a primeira atividade é a definição dos requisitos do sistema considerando o domínio de aplicação.

Modelagem da Solução usando Multiagentes - em uma segunda etapa é realizada a definição de uma solução através da utilização do paradigma OA. Isto é feito através de um processo de identificação e derivação dos agentes que compõe a sociedade (Choren et al, 2004).

Modelagem da Sociedade de Agentes - a terceira etapa é a modelagem da sociedade de agentes, cujo objetivo é representar a perspectiva da sociedade, na qual seriam identificadas as relações de interação existentes entre os agentes. O diagrama utilizado para este propósito é uma extensão do diagrama de objetos proposto pela UML (Medeiros, 2004). No Diagrama da Sociedade, cada retângulo representa uma classe de agentes e os arcos representam os canais de comunicação entre elas.

Modelagem dos Agentes - A última atividade deste processo é a modelagem de cada agente da sociedade e a definição de sua respectiva arquitetura.

Esta metodologia define um conjunto de modelos que permite representar um comportamento complexo. Nesta metodologia, um aspecto importante é a especificação de uma estratégia formal para alocação de recursos apoiada através de um sistema multiagente que permite reduzir a complexidade do modelo considerando eventos em tempo real que afetam a alocação.

2.3.10. *Agent Unified Modelling Language* (AUML)

AUML é uma técnica para modelagem de sistemas multiagentes baseada em UML. Em geral, os diagramas são muito similares; por exemplo, diagramas de estado em UML têm uma visão centrada nos estados, já em AUML, os diagramas estão centrados no agente ou processo.

AUML foi criada a partir de um projeto desenvolvido por (Odell e Parunak, 2000), visando adaptar a linguagem UML às características inerentes ao paradigma de Orientação a Agentes (Wooldridge e Jennings, 1997).

A proposta inicial de AUML é representar Protocolos de Interação de Agentes (PIA) que descrevam um padrão de comunicação, como sendo uma seqüência de mensagens permitida entre agentes e as restrições sobre o conteúdo destas mensagens.

Quando invocado, um agente denominado *Iniciador* emite um pedido de proposta

a um agente que esteja disposto a participar (*Participante*) e fornecer uma proposta. O agente *Participante* pode então responder ao *Iniciador* recusando proposta, aceitando uma proposta, ou não compreendendo a proposta. Se uma proposta for oferecida, o *Iniciador* tem uma escolha de aceitar ou de rejeitar a proposta. Quando o *Participante* recebe uma aceitação da proposta, informará ao *Iniciador* sobre a execução da proposta.

Segundo Bastos (2000), o protocolo como um todo é tratado como sendo uma entidade, sendo colocado em um diagrama que represente o **fluxo de mensagens** entre os agentes (diagramas de seqüência, de colaboração, de atividade e de estados) dentro de um pacote, constituindo uma agregação conceitual de seqüências de interação.

A disposição dos elementos gráficos no d**iagrama de seqüência** enfatiza a ordem cronológica de comunicações.

A sintaxe Agente-Nome/Papéis: Classe já faz parte de UML (em AUML, indica-se o nome do agente ao invés do nome do objeto).

Já no **diagrama de colaboração** são enfatizadas as associações entre agentes, no qual a seqüência das interações é representada através da numeração das mensagens.

Odell examinou quais os diagramas de UML que poderiam estender-se para AUML, como por exemplo, de atividade, seqüência ou colaboração, a fim de abranger todas as características dos agentes, mas alguns diagramas ainda estão em estudo para tentar suprir as características dos agentes.

2.3.11. COMPOR-M

As diretrizes metodológicas do COMPOR-M têm como objetivo reduzir o espaço entre análise, projeto e a implementação do software multiagentes (Almeida, 2004).

COMPOR-M contempla quatro fases: análise, projeto, implementação e testes. Todo o desenvolvimento é iterativo-incremental.

Fase de análise: As habilidades, suas dependências e os contextos a que pertencem são descritas através de um Diagrama de Habilidades.

Fase de projeto: uma vez identificadas as habilidades necessárias em cada contexto para a resolução do problema, pode-se então definir o projeto arquitetural da sociedade de agentes, assim como a arquitetura interna de cada agente.

Fase de Implementação: assim como acontece no Desenvolvimento Baseado em Componentes (Cronkovic, 2001 *apud* Almeida, 2004), os componentes identificados na fase anterior, caso não existam, devem ser implementados. Este é o primeiro passo da fase de implementação. A implementação dos novos componentes deve ser feita com

base em algum modelo de componentes, que ditará a forma como os componentes são construídos e depois conectados entre si.

Fase de Testes: as funcionalidades dos agentes são providas pelos componentes e, sendo assim, testes em cada um dos componentes tornam-se independentes dos testes dos agentes e da sociedade como um todo.

A metodologia é voltada para a especificação de serviços através da abstração de habilidades não perdendo o vínculo da engenharia do conhecimento através das ontologias (Almeida, 2004).

As diretrizes metodológicas representam uma contribuição e um diferencial em relação às demais metodologias baseadas em agentes, pois contempla todas as fases clássicas de desenvolvimento de software, enquanto que as outras não abordam todas as fases.

O principal objetivo das diretrizes de COMPOR-M é a flexibilidade do software para sistemas multiagentes. O conceito de habilidade, por exemplo, só tem sentido de análise se houver uma infra-estrutura de software suficientemente flexível para que mudanças nas habilidades sejam refletidas em implementação, sem grandes esforços.

Foram pesquisadas também metodologias da área de Engenharia do Conhecimento – EC. Engenharia de conhecimento é um termo usado para descrever o processo global de desenvolvimento de um Sistema Especialista – SE. Nos últimos anos vários Sistemas Especialistas Baseados em Conhecimento dirigidos a várias áreas vêm sendo usados: bancos, seguradoras, controle de recursos humanos, diagnóstico de doenças e prescrição de tratamento, suporte à decisão e gerenciamento de redes de computadores

2.3.12. CommonKADS

CommonKADS (Schreiber e Van de Velde, 1994 *apud* Zhou et al, 2007) é uma metodologia projetada para o desenvolvimento de sistemas baseados em conhecimento. Ela é a metodologia principal para apoiar a engenharia de conhecimento estruturada. CommonKADS usa notações de UML, em particular diagramas de classe, de atividades e diagramas de estado.

As fases de CommonKADS são: análise de conhecimento e desenvolvimento de sistemas de conhecimento.

A **análise de conhecimento** é voltada para estudar tarefas de conhecimento intensivas (*knowledge intensive task*) em um nível conceitual. A análise resulta em uma descrição da informação, na estrutura de conhecimento e funções envolvidas na tarefa. Os

resultados da análise de conhecimento são documentados no modelo de conhecimento. Ele contém uma especificação da informação e estruturas de conhecimentos envolvidos em uma tarefa conhecimento intensiva.

Desenvolvimento de sistemas de conhecimento: resultados de análise de conhecimento podem ser usados como especificações para o desenvolvimento de um sistema que implementa esta tarefa.

Durante estas fases de análise e desenvolvimento de conhecimento, CommonKADS usa notações de UML, em particular diagramas de classe, de atividades e diagramas de estado.

Há seis modelos definidos: Modelo da Organização, de Tarefa, de Agente, de Comunicação, de Especialista/Conhecimento, e Modelo de Projeto.

CommonKADS não é uma metodologia completa, mas está sendo utilizada na prática como uma ferramenta poderosa para auxiliar o gerenciamento de conhecimento. O *framework* de análise fornece um método extensivo para descrever processos de negócio em que tarefas de conhecimento-intensivo são necessárias.

2.3.13. *Multi Agent System-CommonKADS* (MAS-CommonKADS)

Iglesias, Garijo e González (1998, *apud* Medina, Sanchez e Castellanos, 2004) propuseram uma extensão de CommonKADS para adequar as características de uma abordagem de agente no ciclo de vida de desenvolvimento de software e definiu um modelo novo, o Modelo de Coordenação (MC), por descrever os protocolos de coordenação entre agentes.

A metodologia segue estas fases: Concepção, Análise, Projeto, Codificação e teste de cada agente, Operação e manutenção.

Concepção: durante esta fase obtém-se uma descrição preliminar do problema.

Análise: determinam-se as necessidades do sistema a partir da declaração do problema. Os resultados desta fase serão a especificação de necessidades de um Sistema Multiagente, através do desenvolvimento dos modelos: Modelo da Organização, Modelo de Tarefa, Modelo de Agente, Modelo de Comunicação e Modelo de conhecimento.

Projeto: como resultado da fase de análise, um conjunto inicial de agentes foi determinado. Durante a fase de projeto o modelo de projeto é desenvolvido. O modelo de projeto consiste dos sub-modelos: Projeto da Rede, Facilidades de Rede, Facilidades de Conhecimento, Facilidades de Coordenação, Projeto do

Agente e Projeto da Plataforma.

Codificação e teste de cada agente: esta é uma questão em aberto, pois depende se é utilizada uma linguagem de propósito geral ou uma linguagem formal de agentes que pode ser diretamente executável ou traduzida para uma forma executável.

Operação e manutenção - o comportamento correto de um sistema global pode ser testado usando cenários típicos que tratam dos possíveis conflitos e dos métodos para resolução de conflito. Desde que o comportamento global do sistema não possa ser determinado durante a análise ou o projeto, porque depende das obrigações particulares e acordos entre os agentes, a simulação do comportamento é necessária.

Esta metodologia se restringe à definição dos modelos e não representa a característica de dinamicidade. Nenhuma consideração é feita sobre a iteração no desenvolvimento com conseqüência na dinâmica no sistema multiagentes.

2.3.14. *Conceptual Modelling of MultiAgent Systems* (CoMoMAS)

Glaser (1996, *apud* Glaser, 2002) também propôs uma extensão à metodologia CommonKADS para Sistemas Multiagentes. CommonKADS foi estendida com a inclusão de características de conhecimento reativo, cognitivo, cooperativo e competências sociais de um agente autônomo.

CoMoMAS tem foco na fase de análise, na construção de modelos para sistemas multiagentes. Ela se baseia em CommonKADS para prover um conjunto de diretrizes que auxiliam na construção destes modelos.

A construção de um modelo de agentes é vista como um processo composto de cinco passos de análise:

Análise funcional - identificação das tarefas que deveriam ser resolvidas pelo sistema multiagente, definindo uma hierarquia de tarefas. Dependências entre as tarefas também devem ser identificadas. O resultado da análise é descrito em um Modelo de Tarefas.

Análise de requisitos - identificação dos requisitos do sistema e suas interdependências. Aspectos não funcionais devem ser levados em conta. Os resultados obtidos são descritos em um Modelo de Projeto.

Análise de competência - identificação as competências cognitivas e reativas que o sistema deve ter para resolver as tarefas descritas anteriormente. Estas competências são assumidas como sendo métodos para a resolução de problemas. O resultado desta análise é descrito em um Modelo Especialista.

Análise cooperativa - identificação dos protocolos e métodos de cooperação. Os resultados são descritos em um Modelo de Cooperação.

Análise social - identificação da organização e da arquitetura do sistema multiagente. Os resultados são descritos em um Modelo de Sistema.

O desenvolvimento de um sistema necessita primeiro de uma descrição do comportamento funcional e uma identificação dos requisitos funcionais e não-funcionais para agente e para o projeto do sistema. Isto é seguido por uma descrição das competências necessárias para resolução das tarefas.

Assim como MAS-CommonKADS, a metodologia é voltada para a construção de modelos, aspectos de implementação não são considerados e não há referência a base de conhecimento.

2.3.15. *Petri Nets Oriented Knowledge Engineering Research* (POKER)

O método POKER (Schiel e Mistrik, 1997) é o resultado de um esforço de se criar uma Metodologia para Desenvolvimento de Sistemas de Informação em que as modelagens do comportamento e da estrutura da aplicação se mantêm em equilíbrio nas fases de análise e projeto.

POKER é uma ISDM (*Information Systems Design Methodology*) que permite a modelagem de funções complexas, na análise e no projeto de sistemas. Este modelo de processos conceituais permite atualizações mais fáceis no modelo da aplicação tanto na estrutura dos dados quanto na estrutura dos processos. Outra característica da metodologia é a integração da modelagem de diversos tipos de regras: regras de produção, regras de integridade e regras de dedução.

POKER é composta de uma seqüência de oito etapas, que podem ser agrupadas em dois estágios: o primeiro (etapas 1-4) é a análise dos requisitos da aplicação e o desenvolvimento do modelo inicial do sistema de informação (*Requirements Model*). No segundo estágio (etapas 5-8) o modelo de requisitos é usado para o *design* do sistema de informação (com a conversão dos diagramas da análise para um esquema conceitual de classes de objetos e uma rede de Petri de alto nível) e a descrição na Linguagem Conceitual do TOM (desenho do esquema TOM -*Temporal Object Model* (Schiel e Carvalho, 1993).

POKER apresenta uma forma de modelagem orientada a objetos e modelagem de conhecimento baseado em regras sem perder características de processos complexos que são muito difíceis de serem modelados em um sistema orientado a objetos puro. Mas, apesar de dar a devida atenção à modelagem de processos complexos integrada a modelagem de objetos, não considera aspectos gerenciais de acompanhamento do desenvolvimento, não inclui artefatos arquiteturais nem dá ênfase ao processo iterativo de desenvolvimento.

2.4. Classificação das Metodologias de Engenharia de Software

Realizamos (Farias, 2004) uma análise entre as metodologias pesquisadas a fim de obter as principais características de cada uma. Esta análise foi baseada em critérios que selecionamos do *Software Engineering Institute* (Firth, 1988 *apud* Scott e Scott, 2001). As métricas e procedimentos para avaliar e classificar metodologias de software incluem: critérios técnicos e gerenciais.

a) Critérios Gerenciais

Os itens de critérios gerenciais abordam o **custo** e a **complexidade** envolvendo o uso de uma metodologia, bem como todo o material necessário, como livros, **ferramentas** case, etc.

O **custo da adoção** avalia a quantidade de recursos (pessoas, software, hardware) necessários a um projeto.

A **disponibilidade de ferramentas de apoio** mostra a disponibilidade atual de ferramentas para uso das metodologias para facilitar o acompanhamento do uso da metodologia.

b) Critérios Técnicos

Os itens utilizados são: Agilidade, Robustez, Quantidade de Artefatos, Uso de padrões, Reusabilidade, Escalabilidade e Ambiente.

A **agilidade** avalia a rapidez que a metodologia evolui e produz resultados.

A **robustez** avalia a qualidade do resultado dos diversos momentos de iteração.

A **quantidade de artefatos** avalia a quantidade de artefatos necessários à metodologia.

O **uso de padrões** avalia o comprometimento da metodologia com a adoção de normas e procedimentos padronizados pela indústria.

A **reusabilidade** avalia a o uso ou produção de componentes reutilizáveis.

A **escalabilidade** aborda questões sobre o tamanho do projeto adequado à metodologia e por fim, o **ambiente** avalia o ambiente típico para o uso efetivo da metodologia.

Análise das metodologias e dificuldades

Foi observado na pesquisa bibliográfica que:

- Muitas metodologias são influenciadas ou particularmente concebidas para serem utilizadas com linguagens de programação específicas.
- Muitas metodologias assumem um contexto de aplicação onde não existem os problemas que no mundo real têm que enfrentar.

Elaboramos a Tabela 2.2, e com base nela, gerentes de projetos de software têm disponíveis os critérios mais relevantes das metodologias. Baseado nestes critérios e no ambiente onde será desenvolvido o projeto, o gerente de projeto terá informações necessárias para decidir qual a melhor metodologia para o seu projeto, dentre as inúmeras existentes. A Tabela 2.2. também foi utilizada para implementação do módulo SiSem a ser descrito no Capítulo 5.

A Tabela 2.2. auxilia na seleção da metodologia adequada fornecendo itens e pontos observados para a escolha de uma metodologia de acordo com os critérios que envolvem: o tipo do software, tamanho do projeto, disponibilidade financeira para arcar com os gastos, ferramentas de apoio de software e outras questões tais como uso de padrões, complexidade, reusabilidade, qualidade, documentação e tempo requerido para o projeto.

Tabela 2.2: Tabela Comparativa das Metodologias (Farias, 2004)

Nome	***Custo***	***Ferramentas de apoio***	***Agilidade***	***Robustez***	***Quantidade de Artefatos***	***Uso de padrões***	***Reusa-bilidade***	***Escala-bilidade***	***Ambiente***
XP	médio	XPlanner	Ciclo de vida pequeno. Resultados diários.	Pouco documentado	10	Componentes	permite	Diferentes tipos de projetos	Equipes pequenas e integradas
RUP	alto	Rational Rose, MS Visio, Jude, Poseidon.	Ciclo de vida completo. Iterativo e incremental	Bem documentado	25	UML	permite	Sistemas comple-xos	Equipes grees, não especialistas
YP	baixo	Xplanner, Cactus, CVS, WinCVS	Ciclo de vida pequeno. Resultados semanais.	Documentado	11	Componentes	permite	Pequenos e médios	Equipes pequenas, Não requer equipe experiente
Catalysis	alto	Ferramentas para OO	Ciclo rápido, iterativo e incremental	Bem documentado	9	UML e componentes	permite	Diferentes tipos de projetos	Equipes qualificadas
Crystal	médio	Ferramentas para OO	Ciclo iterativo e incremental	Desenvolvimento incremental	7	UML	permite	Diferentes tipos de projetos	Equipes pequenas ou grees
FDD	médio	Não foram encontradas	Ciclo iterativo e incremental	Documentado	6	UML	permite	Sistemas críticos	Equipes de especialistas
MSF	alto	Visual Studio Team System	Ciclo iterativo e incremental	Bem documen-tado	14	UML	permite	Diferentes tipos de projetos	Equipes de especialistas
GAIA	alto	Não foram encontradas	Ciclo rápido	Análise incremental	5	Não especificado	Não permite	Problemas complexos	Equipe de especialistas em

									agentes
M-DRAP	alto	Não foram encontradas	Ciclo rápido	Análise completa	6	UML	Não permite	Pequenos projetos	Equipe de especialistas em agentes
AUML	médio	DIA, Rose, Poseidon, Rational , ArgoUML, MagicDraw	Ciclo rápido	Bem documentado	6	UML	permite	Problemas complexos	Equipe de especialistas em agentes
COMPOR-M	médio	COMPOR-E	Ciclo rápido	Bem documentado	4	UML, Componentes	permite	Diferentes tipos de projetos	Equipe com conhecimento de agentes
Common KADS	alto	Modeldraw, KADS22, PC-PACK	Ciclo rápido	Bem documentado na análise	6	UML	permite	Diferentes tipos de projetos	Equipe de especialistas em EC
CoMoMAS	alto	Modeldraw, KADS22, PC-PACK	Ciclo rápido	Bem documentado na análise	5	UML	permite	Diferentes tipos de projetos	Equipe de especialistas em EC e OA
MAS-Comom KADS	alto	Modeldraw, KADS22, PC-PACK	Ciclo rápido	Bem documentado na análise	7	UML	permite	Diferentes tipos de projetos	Equipe de especialistas em EC e OA

Legenda: OO – orientação a objetos

OA -orientada a agentes

EC- engenharia de conhecimento

2.5. Comentários

Neste capítulo, foram mostrados os **métodos de diagramação**, baseados nos modelos de gestão de projetos: PERT, CPM, VERT, GERT, PDM e de redes de Petri. Os grafos PERT e outros métodos gráficos tais como CPM e PDM foram usados por muitos anos e ainda são utilizados hoje em dia, pois mostram uma visualização gráfica da seqüência de atividades e o tipo de dependência entre duas atividades através dos eventos de início e término de ambas.

CPM, GERT, e o PERT são citados como as "técnicas matemáticas mais usadas para auxiliar o processo do desenvolvimento" (Cates, 2004), mas o guia PMBOK cita que estes métodos não consideram recursos no modelo (no caso das redes PERT é requerido um cronograma integrado (como diagrama de Gantt). As redes de Petri devido a sua versatilidade tornam-se apropriadas para representar unidades de recursos físicos e diferentes fluxos de atividades.

Às vezes pode se tornar muito difícil a preparação correta e a visualização de toda a rede de precedências do projeto, devido à complexidade e o tamanho do mesmo. Nesses casos, é muito comum a preparação de um diagrama de rede macro do projeto, e à medida que as fases vão evoluindo, a montagem do diagrama detalhado fase a fase (Barcaui, 2006).

Foram apresentadas também as principais características de várias **metodologias** de desenvolvimento de software com paradigmas de orientação a objetos, agentes e engenharia do conhecimento, resumidas neste capítulo e apresentadas em detalhes em (Farias, 2004) e citadas em (Farias e Schiel, 2006). Foram descritos aspectos positivos e negativos de cada uma delas e foi gerada uma tabela com as principais características de cada uma, de acordo com critérios técnicos e gerenciais escolhidos a partir do *Software Engineering Institute*.

No próximo capítulo, são descritos conceitos relacionados com gerenciamento de projetos. São relatados pesquisas e estudos desenvolvidos na comunidade nacional e internacional de software. É mostrado como estão sendo buscadas soluções acadêmicas ou comerciais que contribuem direta e/ou indiretamente para a melhoria da qualidade do gerenciamento de projetos e processos de software.

Capítulo 3
Gerenciamento de Projetos

Este capítulo tem o objetivo de apresentar uma revisão bibliográfica sobre o estado da arte em gerenciamento de projetos. Particularmente, projetos de desenvolvimento de software.

São apresentados os conceitos de projeto e gerenciamento de projetos. Em seguida, o ciclo de vida de um projeto, as variáveis envolvidas (tempo, custo, recursos e atividades), os trabalhos e ferramentas relacionados com a abordagem proposta nesta tese e padrões nacionais e internacionais para gerenciamento de projetos.

3.1. Conceituação: Projetos e Gerenciamento de Projetos

O conceito de projeto está principalmente vinculado à inovação e à melhoria de desempenho. Segundo Cleland (2002) "Um projeto consiste em uma combinação de recursos organizacionais para se criar algo que não existia anteriormente e que irá fornecer uma melhora na capacidade de desempenho, tanto do desenho quanto na execução de estratégias organizacionais."

Kerzner (2002) apresenta um conceito de projeto definindo-o da seguinte forma "Trata-se de um empreendimento com objetivo identificável, que consome recursos e opera sob pressões de prazos, custos e qualidade."

O PMI (2004) sintetiza projeto como "Um esforço para criar um produto ou um serviço com um objetivo específico. Projetos são temporários - eles têm um começo e um fim definidos. Eles são originais, ou seja, o produto ou o serviço que criam são diferentes em alguma maneira dos produtos ou serviços similares" (PMBOK, 2004).

Consolidando os conceitos apresentados, pode-se afirmar que o grande diferencial entre as atividades comuns e os projetos é o fato de os projetos serem temporários e exclusivos, enquanto os demais serviços são contínuos e repetitivos. Vale esclarecer que, independente do tempo necessário para se realizar o objetivo do projeto, sempre existe um prazo previamente estabelecido, ou seja,

início e fim previamente definidos.

Há muitos anos já existe a percepção de que a gestão dos projetos é importante para que se obtenham os resultados desejados. Entretanto, o conceito de gestão de projetos precisou se adequar ao longo dos anos, em função do contexto sócio econômico.

A gestão de projetos surgiu na indústria americana da construção civil nos anos 50, ainda de maneira modesta, segundo Cleland (2002). Apesar disso, já se pôde verificar uma preocupação maior com as responsabilidades da equipe de projetos e com os requisitos que se pretendem cumprir. Desde então, já se pode observar a importância dos parâmetros escopo, tempo, custo e qualidade.

Nos anos seguintes, a gestão de projetos começou a despertar interesse e, em 1969, foi fundado o PMI - *Project Management Institute*, organização sem fins lucrativos voltada para o estudo e o desenvolvimento do gerenciamento de projetos. Em 1983, foi lançada a primeira base de conhecimentos do PMI, contendo seu código de ética, uma linha de base de normas constituída por seis principais áreas de conhecimento (gerenciamento do escopo, gerenciamento de custos, gerenciamento de tempo, gerenciamento da qualidade, gerenciamento de recursos humanos e gerenciamento das comunicações).

Ao longo dos anos, a evolução desses documentos deu origem ao PMBOK – *The Project Management Body of Knowledge*, ou seja, um guia contendo as melhores práticas e conhecimentos adquiridos sobre gestão de projetos pelo PMI (2004). Observa-se que, a partir do reconhecimento do competente e continuado trabalho do PMI, o PMBOK tornou-se a principal referência em gestão de projetos da atualidade.

No cenário atual, a gestão de projetos ganhou ainda maior relevância, segundo Valeriano (2005) "o grau de complexidade dos projetos aumentou significativamente, em função do alto volume de informações, equipe multidisciplinar de profissionais, surgimento de novas tecnologias, necessidade de atualização continuada, clientes e equipes de projetos distribuídas geograficamente, concorrência global, maior abrangência do projeto, margens de lucro pequenas, exigência constante de personalização, enfim uma gama de requisitos e variáveis enormes a serem geridos fazendo com que a gestão de projetos necessitasse se adequar ao contexto contemporâneo, tornando seu conceito mais amplo".

Apesar das diferentes visões, algumas complementares e outras antagônicas, existe um consenso de que os projetos precisam ser bem gerenciados a fim de alcançar os objetivos planejados. Dessa forma, a gestão adequada dos projetos é fundamental para as organizações que desejam atender plenamente seus clientes, ou melhor, para as empresas que desejam sobreviver no mercado ágil e competitivo.

3.2. Ciclo de Vida de Projetos

Em função da complexidade da gestão de projetos no ambiente organizacional, os projetos podem ser divididos em fases, visando facilitar o seu acompanhamento.

Segundo Valeriano (2005) um projeto possui fases que constituem o ciclo de vida deste projeto: fase de concepção, planejamento, implementação e encerramento. De acordo com Prado (2002), "o entendimento dessas fases permite ao time do projeto um melhor controle do total de recursos gastos para atingir as metas estabelecidas". O conjunto de fases do projeto é conhecido como *Ciclo de Vida do Projeto*.

O ordenamento das principais fases do projeto permite a sua visualização macro, do início ao fim. Em geral, as fases de um projeto são apresentadas de forma seqüencial, sendo que, após conclusão de uma fase, deve ser feita uma verificação e, posteriormente, iniciada a fase seguinte. Entretanto, observa-se que o início de uma fase não está condicionado ao termino da anterior, podendo haver fases realizadas concomitantemente.

Alguns autores apresentam um conjunto genérico de fases, comuns a grande parte de projetos. Xavier (2005) concorda com Valeriano (2005), segundo ele, um ciclo de vida genérico também possui as seguintes fases: iniciação, planejamento, execução e encerramento.

- A iniciação é a fase de identificação das expectativas, análise da viabilidade e de definição dos objetivos e metas do projeto.
- O planejamento é a fase de detalhamento de todas as ações a serem realizadas no âmbito do projeto, como cronograma, orçamento, relação entre as atividades, escopo, equipe, responsabilidades, dentre outras.
- A execução é a fase de realização, quando as ações previstas no planejamento são concretizadas.
- A fase de encerramento é caracterizada pelo arquivamento, documentação e comunicação dos envolvidos.

O PMI (2004) cita que o final de cada fase deve estar vinculado à conclusão de um produto ou resultado mensurável, associando inclusive, o nome das fases aos produtos a serem concluídos, ou seja, "as fases normalmente recebem os nomes de acordo com esses seus produtos: requisitos, projeto, construção, teste, inicialização, entrega e outros".

3.3. Parâmetros de custo, prazo e recursos humanos

Em geral, todos os gestores de projetos almejam o sucesso nos projetos sob sua liderança, seja por profissionalismo, por orgulho pessoal, por recompensas associadas, pela manutenção do seu *status*, pelo desafio de cumprir o objetivo pactuado, pela sensação de dever cumprido, ou por mais de um

desses fatores relacionados.

Diante de uma visão mais genérica e com uma definição objetiva, Vargas (2002) defende que "Um projeto bem sucedido é aquele que é realizado conforme planejado".

Essa visão não é ratificada por Meredith e Mantel (2003), pois, segundo eles, o atendimento aos requisitos de custos (orçamento), prazo (programação) e qualidade (desempenho) não asseguram o sucesso de um projeto. De acordo com os autores, os critérios de uso e satisfação do cliente também devem ser atendidos.

Conforme Kerzner (2002), o conceito de sucesso em gestão de projetos sofreu modificações ao longo do tempo. Inicialmente, o projeto era visto apenas de forma tecnicista, com foco apenas no escopo, ou seja, produto final conforme especificado garantia o sucesso. Em outro momento, a questão empresarial foi evidenciada, passando a definição de sucesso no projeto a ser "a conclusão da programação no prazo, no custo e no nível de qualidade pré-estabelecidos, sendo a qualidade definida pelo cliente, não mais pelo fornecedor." E, finalmente, segundo o autor, a melhor e mais moderna definição de sucesso de projeto, que vincula os fatores primários e secundários na mensuração dos resultados, sendo:

- Fatores primários: prazo, custo orçado, qualidade prevista;
- Fatores secundários: satisfação explicita do cliente quanto ao resultado.

Entretanto, apesar das inúmeras definições e estudos sobre gestão de projetos não há como assegurar que o sucesso alcançado por uma ou mais empresas possa ser replicado para outra, ou seja, mesmo que seja seguida toda a metodologia aplicada por uma determinada empresa em um projeto de sucesso, qualquer que seja o conceito, não garante a realização de um outro projeto bem sucedido por outra organização. Da mesma forma, uma **metodologia de desenvolvimento de software** aplicada em projetos distintos de uma mesma organização, não pode garantir que se consiga replicar plenamente os resultados alcançados em um projeto anteriormente executado.

Portanto, cada empresa, ou melhor, cada equipe de projeto, precisa avaliar criteriosamente todas as características de cada projeto específico e fazer as adequações metodológicas necessárias.

Além desses dos parâmetros de custo, prazo e qualidade, a gestão do escopo também é considerada fundamental, pois, juntamente com os fatores tempo e custo, impacta diretamente a qualidade dos projetos. Portanto, esses quatro fatores estão intimamente relacionados.

A qualidade do projeto é afetada pelo balanceamento desses três fatores, projetos de alta qualidade entregam o produto, serviço ou resultado solicitado dentro do escopo, no prazo e dentro do orçamento. A relação entre esses fatores ocorre de tal forma que se algum dos três fatores mudar, pelo menos outro fator provavelmente será afetado.

Nesta tese, além dos **parâmetros de custo e prazo, são levados em consideração os recursos humanos, as ferramentas de gerenciamento de projetos** (descritas no próximo item deste capítulo) e as metodologias de desenvolvimento de software (descritas no Capítulo 2), pois juntamente com os fatores tempo e custo, eles impactam diretamente a qualidade dos projetos.

Portanto, esses fatores estão intimamente relacionados. A seguir eles são descritos e utilizados posteriormente para fundamentar as áreas de atuação e comparar os trabalhos relacionados.

3.3.1. Custo

O custo para desenvolver um projeto de software, depende de diversas condições iniciais para o desenvolvimento de cada projeto tais como:

- custos de hardware e software, inclusive manutenção;
- custos de viajens e treinamento, e
- custos relativos ao esforço empregado (custo de pagamento dos engenheiros)

Para a maioria dos projetos, o custo dominante é o que se refere ao esforço empregado. Os computadores que são suficientemente poderosos para o desenvolvimento do software são relativamente baratos. Embora os custos com viagens possam ser significativos quando um projeto é desenvolvido em diferentes locais, eles são relativamente baixos, para a maior parte dos projetos. Além disso, o uso do correio eletrônico, fax e teleconferências podem reduzir as viagens necessárias. Em projetos de outras categorias o primeiro item sería substituído por `custos materiais` e, dependendo do produto a ser desenvolvido, podem ser bem mais significativos do que no caso de projetos de software.

As organizações calculam os custos de esforço em termos de custos gerais, considerando o custo total de operação da organização, dividindo-o pelo número de pessoas produtivas. Portanto, os custos que se seguem fazem parte do custo total de esforço:

- Custos relativos ao fornecimento de energia;
- Custos com o pessoal de apoio (contadores, secretárias, técnicos e pessoal de limpeza);
- Custos com operações em rede e comunicações;
- Custos com instalações centrais, como biblioteca, instalações recreativas, etc. e
- Custos com previdência social.

O cálculo dos custos de software deve ser realizado objetivamente, com a intenção de prever com precisão qual será o custo de desenvolvimento de software para o fornecedor.

3.3.2. Prazo

A programação de tempo de um projeto é uma tarefa particularmente necessária para os engenheiros de projeto. Os gerentes estimam o tempo e os recursos exigidos para completar as atividades e as organizam em uma seqüência coerente. A menos que o projeto em programação seja similar a um projeto anterior, estimativas precedentes são uma base incerta para uma nova programação de projeto.

A programação de tempo de um projeto segundo Somerville (2003) envolve dividir o trabalho total de um projeto em atividades distintas e avaliar o tempo necessário para completar estas atividades. De modo geral, algumas atividades são realizadas em paralelo. Os programadores de projeto devem coordenar estas atividades paralelas e organizar o trabalho, de modo que a força de trabalho seja otimizada. Eles devem evitar uma situação em que todo o projeto é atrasado porque uma tarefa importante não foi incluída.

Inevitavelmente, as programações iniciais de projeto podem estar incorretas. À medida que um projeto se desenvolve, as estimativas devem ser comparadas com o tempo real empregado. Essa comparação pode ser utilizada com base para revisar a programação das partes posteriores do projeto. Quando os números reais são conhecidos, o diagrama de atividades deve ser reexaminado. As atividades de projeto seguintes podem ser então reorganizadas, a fim de reduzir a extensão do caminho principal.

3.3.3. Recursos Humanos

O gerenciamento eficaz diz respeito, portanto ao gerenciamento de pessoas em uma organização. Os gerentes de projetos têm de solucionar problemas técnicos e não técnicos, utilizando a capacidade das pessoas de sua equipe da maneira mais eficaz possível. Eles precisam motivar o pessoal, planejar e organizar seu trabalho e assegurar que o trabalho seja feito adequadamente. O gerenciamento inadequado de pessoas é uma das mais significativas contribuições para o fracasso dos projetos (Standish Group, 2004).

"As pessoas que trabalham em uma organização de software são seu maior patrimônio. Elas representam o capital intelectual, e é responsabilidade dos gerentes de software garantir que a organização obtenha o melhor retorno de seu investimento em pessoas" (Sommerville, 2003).

3.4. Ferramentas de Gestão

Na literatura, existem vários trabalhos relacionados com gestão de projetos. No entanto, a maioria deles aborda a gestão parcialmente, sem conseguir englobar todos os aspectos necessários para realizar um gerenciamento completo.

Os modelos tradicionais de gestão de projetos, tais como o método do caminho crítico, o método de Gantt, e os métodos de avaliação gráfica e revisão técnica são utilizados para gerenciar projetos sem considerar se os recursos estão disponíveis e não permitem modelar situações dinâmicas (Elmaghraby, Baxter e Vouk, 1995; Liu e Horowitz, 1989; Meredith e Mantel, 2000).

Redes de Petri, por outro lado, são conhecidas pela sua capacidade em modelar atividades concorrentes e simular a evolução dos processos (Van der Aalst, 1998; Alpan e Jafari, 1997; Hasegawa et al. 1999). As vantagens das Redes de Petri incluem a semântica formal com interfaces

gráficas, e o estado de evolução do modelo pode ser expresso através da marcação de mudanças, com um conjunto de técnicas de análise, como medidas de desempenho (tempo em espera, decorrer tempo, etc.) e para gestão de projeto: repartição dos recursos, compartilhamento de recursos, etc.

A gestão de projetos tem sido identificada por alguns pesquisadores como uma área próspera onde Redes de Petri podem ser usadas. Liu e Horowitz (1989) sugeriram um modelo utilizando gráficos e/ou e notações de rede de Petri para fornecer a descrição de um projeto e as especificações das relações entre diferentes projetos de desenvolvimento de software. Magott (1989) sugeriu a avaliação de desempenho de processos concorrentes combinando Redes e Petri estocásticas concomitante com redes PERT. Mas estas obras não tratam da alocação de recursos em projetos.

3.4.1. Controlando Recursos e tempos

A definição de recursos tem sido há muito tempo uma questão crítica nas práticas de gestão de projetos. A questão está relacionada com a gestão de recursos limitados para atividades e a programação de atividades, como resultado da atribuição (Meredith e S. J. Mantel, 2000; PMI, 2004). Isto é, para organizar a execução das atividades em projetos deve-se considerar a disponibilidade dos recursos necessários. Se os recursos necessários não podem ser garantidos com o tempo, os horários das atividades afetadas têm de ser reajustados.

Algumas pesquisas estendem Redes Petri para o controle de recursos. Uma delas tem modelado e simulado a partilha dos recursos assim como a dependência de atividades em projetos (Jongwook, Desrochers e Sanderson, 1995). Jongwook et al. introduziram uma abordagem de gerenciamento interativo do projeto para dinamicamente acompanhar e controlar o andamento do projeto e para modelar recursos (***Task planning and project management using Petri nets***). O trabalho mostra uma abordagem interativa de gestão de projetos baseado em rpara superar as limitações dos diagramas de rede (PERT, CPM, etc.). Nesta abordagem o gerente de projetos tem de afinar o planejamento e programação de tempo com o recurso embutido no plano da representação. A rede de Petri controladora mantém um plano de três representações: referência, monitor e preditor de redes de Petri. Estes três redes são configuradas com os recursos de modo a formar um feedback assim que o controlador pode monitorar e controlar o projeto *on-line*.

O trabalho apresenta três conceitos importantes:

1) Um mapeamento formal entre o tradicional diagrama de rede de planejamento e uma rede de Petri. Este mapeamento permite ao gerente de projeto de rede de Petri interagir diretamente com um método CPM, ou outro método de interface de rede baseada em planejamento.

2) A definição do monitor rede de Petri que cria uma linha de referência entre o plano de referência e o estado atual do sistema durante a execução. O monitor dá suporte a uma rede interativa que permite ao usuário visualizar o estado atual do projeto e os recursos.

3) Um mecanismo de apoio à decisão baseado na comparação entre "on-line preditor Petri net" e "reference Petri net". Resultando em decisões de controle que podem modificar evento ou seqüências de tempo, ou podem dar início a um re-planejamento e re-programação.

A rede de Petri interativa apresentada incorpora parâmetros de recursos para o planejamento e programação, e também introduz o controle de evento para dinamicamente acompanhar, controlar e re-planejar o projeto. O modelo de controlador é formal e geral para que possa ser utilizado para o controle de sistemas dinâmicos de eventos.

Kumar e Ganesh (1998) propuseram um modelo de rede de Petri para descrever atividades em tempo real com operações Fuzzy. O objetivo do trabalho é demonstrar a relevância e a utilização de redes de Petri para a atribuição dos recursos em projetos. Kumar e Ganesh desenvolveram um modelo que mantém as vantagens dos modelos tradicionais de gestão de projeto (CPM, PERT e Gantt charts) e modelos (VERT, GERT, etc.) e pode ser utilizado para lidar com algumas dificuldades de gestão de recursos encontradas na prática.

O modelo recebe recursos a partir ambiente, e utiliza-os para uma unidade tempo (um dia por exemplo) e depois libera-os. Isto é repetido em cada tempo unidade até a conclusão do projeto.

O modelo pode ser utilizado para análise de diferentes aspectos de alocação de recursos em projetos:

1)avaliação da importância dos diferentes recursos em diferentes pontos do tempo durante a execução do projeto, ou seja, análise de sensibilidade dos recursos do projeto;
2)alocação on-line e programação de atividades;
3)cálculo do requisito de tempos ótimos em cada dia, isto é útil para recalcular os recursos necessários dos dias subseqüentes, quando os recursos não podem ser implantados em determinados dias devido a constrangimentos;
4)da substituição de recursos (por exemplo, da substituição de um recurso por outro de eficiência diferente);
5)determinação da utilização e não utilização dos recursos;
6)análise das interdependências de recursos e prioridades de recursos conflitantes;
7)estudo do efeito das limitações à atribuição parcial e exclusividade mútua de recursos.

Apesar das interdependências entre recursos no modelo, isso não limita a sua aplicabilidade, porém técnicas e modelos mais atuais ainda são necessárias e têm sido desenvolvidas.

Ainda visando resolver o problema das ferramentas convencionais de gestão (CPM, PERT, etc.) que são limitadas na sua aplicação, o trabalho de Jeetendra, Chetty e Reddy (2000) entitulado ***Petri Nets for Project Management e Resource Levelling*** detalha o uso de redes de Petri no

gerenciamento de projeto. Neste contexto, extensões das redes de Petri são propostas, para se adequar a esta área de aplicação. É proposta a utilização de uma P-matriz para movimento de marcas e um algoritmo.

Um *PN-aided software* - "*PLS_net*" foi desenvolvido para lidar com gestão de projeto, quer para uso com tempos de atividades determinísticos ou estocásticos. Desenvolvido em Borland C, o software é constituído por dois módulos: um módulo *Project_net* e um *RESL_net module*, O primeiro trata da modelagem e análise e o segundo é utilizado para nivelamento de recursos. Informações relativas a precedência de relações, e os recursos necessários por hora para cada atividade são introduzidos.

A P-matrix é inicialmente formulada para representar a precedência de relacionamentos de atividades na rede. Um elemento "1" na posição (i, j) indica que a atividade na coluna j é um predecessor para a atividade na linha i. Transição(ões) para que a soma da linha seja zero é(são) ativada(s). A P-matriz tem o auxílio em cálculos tradicionais, tais como: data mais cedo de início, data mais cedo de término, data mais tarde de início e data mais tarde de término. Ela também ajuda a determinar o caminho crítico. A P-matriz pode ser usada para estudar propriedades de *deadlocks*.

Outro trabalho, de Chen, Hsu e Chang (2008) propõe uma nova rede de Petri extendida que descreve como os recursos são distribuídos entre atividades concorrentes de múltiplos projetos. O modelo proposto é denominado ***A Petri Net Approach to Suporte Resource Assignment Petri Net (RAPN***) que estende rede de Petri com novos lugares, transições, atributos e regras para modelar partilha de recursos e estratégias de concessão de recursos. RAPN modela o consumo dos recursos dos projetos e os comportamentos, além de computar o tempo total necessário para realizar, acompanhar os projetos envolvidos. Possui uma interface que permite ao usuário visualizar o estado atual do projeto e de recursos.

O trabalho propõe modelar o controle de recursos e a divisão de tarefas através de um modelo formal. O modelo RAPN é utilizado para avaliar o impacto de atribuições de recursos no agendamento do projeto através de alguns elementos definidos tais como: atividades, recursos, dependência temporal entre as atividades, recursos compartilhados por atividades, atribuição dos recursos empregados e tempo de execução dos projetos.

O modelo RAPN é usado para formalmente representar recursos e cronogramas de atividades para simular tempo de execução de projetos.

Na RAPN, os recursos são divididos em duas categorias: consumíveis (por exemplo, materiais) e reutilizáveis (por exemplo, recursos humanos e dispositivos). Depois de ser utilizado, um recurso consumível é consumido e esgotado, mas os recursos podem ser reutilizáveis. Em outras palavras, recursos consumíveis só podem ser usados uma vez, mas recursos reutilizáveis podem ser usados repetidamente.

Embora RAPN possa descrever recursos básicos e designar seqüenciamento de uso, é

necessária mais investigação na modelagem de consumo de recursos no projeto. Por exemplo, o modelo não pode descrever cenários onde as atividades consomem recursos parciais, em vez de todas as unidades, ou utilizar atividades do mesmo recurso intermitentemente.

3.4.2. Modelagem de sistemas complexos

Redes de Petri também estão sendo empregadas na modelagem e investigação de diferentes tipos de sistemas, e até mesmo para sistemas grandes e complexos.

A maior parte das pesquisas concentra-se em análise, design, implementação de tecnologia, *framework* e assim por diante (Jianyi, 2003), mas existem poucos trabalhos sobre gestão de fluxo de trabalho (*workflow*) e modelagem de processos de desenvolvimento de software.

O trabalho de **Jianxin e Juanwei** (2005) descreve um modelo de *workflow* baseado em rede de Petri e discute como converter o modelo de *workflow* para uma rede de Petri, e proporcionar uma extensão da teoria de rede de Petri como uma ferramenta de modelagem.

Da análise da literatura, sobre gestão baseada na teoria de *workflow*, o trabalho de Wang, Xuw e Yang (2007) apresenta um modelo de rede de Petri para estabelecer um modelo de *workflow* de gerenciamento de pesquisa científica, e apoia na criação de um sistema de informação de gestão que pode proporcionar uma reorganização dinâmica.

O modelo de *workflow* é diferente de outras ferramentas de modelagem. Ele não só contém relação de conjuntos de atividades e definição de atividades, como também contém as informações necessárias para executar outras atividades. O conteúdo principal do modelo de workflow de Wang et al. oferece uma linguagem de modelagem integrada que pode ser utilizada para definir um modelo de processo. O modelo é simples e rigoroso, e foi utilizado no gerenciamento de uma pesquisa científica na *School of Management, Xi'an University of Architecture & Technology* da China.

O trabalho de Wang et al. (2007) está sendo ampliado e deve ser pesquisado mais profundamente após estabelecer o modelo de workflow e realizar análise de desempenho, controle quantitativo, regras de concepção e implementação de um sistema de informação.

O trabalho de **Zhu e Schnieder** (2000) também aborda a modelagem de sistemas complexos com redes de Petri. O trabalho discute uma abordagem para integrar técnicas de desenvolvimento de software orientadas a objetos (OO) para o processo de modelagem de sistemas complexos com redes de Petri. No início, a estrutura global do sistema é modelada por meio de um Diagrama de Estrutura de Componente (*Component Structure Diagram* - CSD) que é uma rede de Petri. Então a modelagem de componentes do sistema é enfatizada por suas funcionalidades ou dados (objeto) que correspondam às suas propriedades.

Os modelos de diferentes pontos de vista (dinâmica, funções e dados) de um componente são integrados na rede de Petri. Os escopos e as fronteiras dos diferentes pontos de vista dos modelos são

claramente definidos. O modelo de sistema global é criado através de uma integração dos modelos de cada um dos pontos de vista do sistema de componentes, bem como a integração de modelos de componente com o modelo de estrutura do sistema.

Para sistemas complexos, o modelo CSD pode dispor de uma estrutura hierárquica. Os componentes do CSD e do modelo de estrutura deveriam ser organizados de acordo com as funções do componente e as relações entre eles. Existem três padrões para a estrutura dos componentes: decomposição vertical, em camadas e de construção de classes (Rumbaugh, Blaha, Premerlani e Eddy, 1991):

1) Visão dinâmica - o ponto de vista da dinâmica é modelado por **diagrama de transição de estado** ou *statecharts* (Harel, 1987). Tal como muitos métodos orientados a objetos, cada modelo dinâmico na modelagem é também atribuído a certo componente (funcional), de modo que a complexidade de todos os modelos dinâmicos pode ser limitada ao âmbito de um único componente e uma transformação a partir do diagrama de rede de Petri pode ser realizada.
2) Visão de funções - para especificar as ações e atividades em um modelo dinâmico ou operações de um componente de dados, uma modelagem funcional é necessária e pode ser descrita pelo tradicional **diagrama de fluxo de dados** (DFD).
3) Visão de dados - no desenvolvimento orientado a objeto, o ponto de vista dos dados é modelado por **diagramas de classes/objetos** que dizem respeito a todos dados observados no sistema.

Depois que a estrutura global do sistema e os diferentes pontos de vista sobre os componentes são modelados com rede de Petri, como discutido anteriormente, uma base comum para a integração destes modelos parciais é criada. Depois é possível combinar os diferentes modelos parciais para um modelo global do sistema observado.

O modelo CSD e o modelo de visão de componente são definidos, e se complementam, mas sem sobreposição, a sua integração pode ser realizada sem sobressaltos.

Com esta abordagem de modelagem é possível tratar sistemas complexos, onde o sistema envolve múltiplos subsistemas bem como estruturas dinâmicas e distribuídas. Mas a abordagem de Zhu e Schnieder não envolve modelagem de processos de desenvolvimento de software (RUP, XP, etc.), documentação de artefatos gerados, nem controle de variáveis de projetos (custo, tempo e recursos).

3.4.3. Modelagem de processos

A ferramenta **XPlanner** (2003) é uma ferramenta de planejamento de projeto utilizado na metodologia eXtreme Programming (XP). Para resumir o processo do planejamento de XP, os

clientes escolhem as características a serem adicionadas (*user stories*) a cada iteração do desenvolvimento. Os desenvolvedores estimam o esforço para completar as estórias ou decompor a estória em tarefas e estimando-as. A informação sobre a velocidade do desenvolvimento da equipe da iteração precedente é usada para estimar se a equipe pode terminar as estórias propostas pelo cliente. Se a equipe parecer estar saindo do caminho planejado, as estórias são revistas com o cliente. A ferramenta XPlanner foi criada para dar suporte a este processo.

Alguns dos objetivos de Xplanner são fornecer: uma maneira para que os desenvolvedores acompanhem as *user stories* e as tarefas a serem realizadas; um *feedback* contínuo do *status* da equipe aos clientes, no local ou remotamente; aumentar a comunicação da equipe com uma visão compartilhada das atividades e do *status* delas e prover meios automatizados de planejamento de métricas para cada iteração.

Xplanner não controla custos do projeto e não possui forma de instanciação para outras metodologias, apenas para XP. Ela não possui visualização gráfica do *workflow* do projeto e as atividades são acompanhadas através de barras.

Outra ferramenta - ***Visual Studio Team System*** (VSTS) (2008) tem foco no ciclo de vida no desenvolvimento de software gerenciando etapas durante o processo de desenvolvimento. Esta ferramenta é apenas utilizada com a metodologia MSF ou padrão CMM, com definição e distribuição de tarefas e papeis no projeto, alem do acompanhamento e testes.

Vianna (2004) cita que o MSF provê um *framework* flexível e escalável, que pode ser adaptado para atender as necessidades de qualquer tipo de projeto de desenvolvimento de software e infra-estrutura de TI, independente do tamanho e da complexidade.

A ferramenta não controla custos e não possui algumas funcionalidades úteis ao acompanhamento do projeto como, por exemplo: simulação, análise probabilística, comparação de caminhos, geração automática de gráficos (de barra, pizza, etc.) e visualização gráfica do *workflow* do projeto, ela mostra apenas a seqüência de atividades em uma lista e seu o valor de aquisição é alto, tendo em vista que ao adquirir o produto a empresa tem de adquirir também o pacote Microsoft Office.

O ***RUP Builder*** (2004) inclui conteúdos de processos que são combinados com ferramentas que permitem a customização e instalação da metodologia RUP para um projeto particular de uma equipe. O *RUP Builder* permite que os gestores configurem os seus processos para o seu próprio uso, utilizando um conjunto predefinido de *plug-ins*. Os gestores de processos, utilizando o *RUP Builder*, podem criar um projeto para uma equipe, melhorando a compreensão e o uso desse processo. Quando um projeto é planejado, freqüentemente deseja-se que a instância de um processo siga o processo que especificamente vai ao encontro das necessidades do projeto. O *RUP Builder* permite definir o RUP de forma a dar conta de necessidades específicas da organização, através da criação de um *plug-in* do processo que muda o modelo do processo em estudo.

As três últimas ferramentas citadas (RUP *Builder*, XPlanner e VSTS) não permitem gerenciamento de diversos tipos de processos de desenvolvimento. Estas ferramentas não permitem adaptar (customizar) uma metodologia e utilizar um processo da própria instituição, não permitem o uso para vários tipos diferentes de processos e não controlam todas as variáveis de projeto (custo, tempo e recursos).

Aversano et al. (2004) desenvolveram um ambiente para auxiliar processos da engenharia de software em ambientes distribuídos – **GENESIS** - *GEneralised eNviroment for procESs management In cooperative Software engineering*. O sistema auxilia a cooperação e coordenação em processos de software e os processos complexos são divididos em sub-processos que podem ser distribuídos e executados em sites de diferentes organizações.

Em GENESIS, tecnologias de gerenciamento de *workflow* são integradas com gerenciamento de artefatos e serviços de comunicação para encontrar os requisitos necessários de gerenciamento e serviços de comunicação entre times de desenvolvimento distribuídos geograficamente.

GENESIS possui gerenciamento de artefatos, ferramenta de visualização de *workflow* de processos, gerenciamento de recursos humanos e sites representando instituições que participam de um projeto. O ambiente coleta dados sobre o projeto: atividades, artefatos gerados, esforço e duração e apresenta relatórios sintetizados sobre o *status* dele. As medidas são armazenadas em um banco de dados para que o gerente de projeto possa ter uma visão da evolução do projeto.

O ambiente possui boas funcionalidades para gerenciamento de projetos, mas não possui todas as funcionalidades necessárias ao controle de um projeto, por exemplo, ele não controla custos de atividades, recursos operacionais, não proporciona simulação e análise probabilística, importação/exportação de dados, calendário, geração de gráficos e comparação de caminhos.

Outra ferramenta que permite a modelagem de processos é **ODE (*Ontology-based software Development Environment*)** (2005) é um ADS – Ambiente de Desenvolvimento de Software centrado em processo, fundamentado em ontologias. A ferramenta permite entre outras coisas, a definição de processos. ODE utiliza um suporte do conhecimento para apoiar o gerente na definição de um modelo de ciclo de vida para o processo, de suas atividades, dos artefatos a serem produzidos e consumidos, dos recursos humanos, de software e de hardware necessários à execução, e dos métodos, técnicas, roteiros e normas utilizados.

O acompanhamento de Projetos também é uma das características de ODE. Assim que um projeto é iniciado, é possível acompanhá-lo. O processo do projeto é apresentado na forma de um grafo que exibe as atividades, destacando sua hierarquia, ordem de execução e estados atuais. Para cada uma das atividades, são mostradas informações relevantes como seus artefatos, procedimentos,

recursos alocados e datas.

O ambiente ODE possui módulos úteis a várias tarefas do desenvolvimento de software, como pôde ser visto. Contudo, ainda não existe armazenamento em um repositório único; a organização não pode definir o seu processo, instanciar os processos de projetos específicos e também não possui ainda manipulação de pessoal, competências e atividades da organização. ODE não realiza um controle rigoroso de prazos, não possui controle de custos, calendário de tarefas x pessoas, comparação de caminhos (previstos x alcançados), simulação e análise probabilística.

O ambiente **TABA - Meta-Ambiente para instanciação de Ambientes de Desenvolvimento de Software convencionais e Orientados a Domínios** (2004) é um ambiente capaz de gerar, através de instanciação, ambientes de desenvolvimento de software adequados às particularidades de processos de desenvolvimento e projetos específicos. Desta forma, TABA possui funcionalidades para a definição de processos, métodos e ferramentas CASE a serem utilizadas no processo de desenvolvimento e manutenção de software.

Ela apóia o planejamento, execução, gerência e controle do projeto de desenvolvimento de software durante o seu ciclo de vida apoiado por diversos módulos.

TABA é um software bastante completo, dotado de recursos de controle de custos, recursos, e diversos outros aspectos do desenvolvimento de um projeto. Também é possível configurá-lo para trabalhar seguindo uma metodologia específica (em especial, o MPS-BR) através de parâmetros definidos pelo próprio ambiente.

Como as demais ferramentas citadas (Xplanner, Rup Builder e VSTS) não é possível modelar um processo baseado em uma outra metodologia específica, cabendo ao gerente realizar esse mapeamento através de sua própria experiência. Também não há nenhum recurso visual para a modelagem desse processo, cabendo ao gerente mapeá-lo usando uma estrutura de formulários e não um diagrama visual.

Primavera (2002) é uma suíte de ferramentas que vão desde o gerenciamento de contratos até o controle de processos de desenvolvimento. Existem ferramentas para o gerenciamento de recursos, projetos, portfólios de projetos e organização de atividades. Todas as ferramentas são integradas entre si e acessíveis pela web. Analisando especialmente a ferramenta de gerência de projetos, ela oferece recursos bastante comuns em ferramentas do tipo, tais como a inicialização rápida de um novo projeto, definição de caminhos críticos, geração de gráficos de Gant, dentre outros. A utilização é semelhante a de outras ferramentas de gerência de projeto, onde o gerente aloca recursos ao longo de uma linha definida de tempo. Não há nenhuma associação do projeto com alguma metodologia em si, cabendo ao gerente realizar essa atividade em uma ferramenta externa de terceiros. Nota-se uma grande eficiência no acompanhamento dos custos projetados e realizados, com relatórios de diversos níveis sobre vários aspectos do projeto.

dotProject (2006) é um projeto de software livre que visa desenvolver um aplicativo de gerenciamento de projetos baseado na Web, que fornece funções de desenho e controle de projeto de qualquer tipo. dotProject visa fornecer ao gerente de projeto uma ferramenta para gerenciar tarefas, agendas, comunicação e compartilhamento, mas ela é uma ferramenta bastante limitada em diversos aspectos, o que obriga o gerente a ter em mãos ferramentas de terceiros para complementar as falhas do produto. Não há um controle de custos efetivos, nem gráficos de caminho-crítico ou de qualquer tipo. O forte da ferramenta é a colaboração (com a criação de fóruns de discussão para os projetos) e a centralização das informações do projeto disponibilizadas por uma interface web.

Outro trabalho encontrado é **APSEE** (2004), ele fornece uma infra-estrutura que integra diferentes serviços voltados à modelagem, simulação, execução, reutilização e visualização de processos.

A linguagem APSEE-PML auxilia a descrição de processos, sendo também útil para habilitar políticas em modelos de processos de software. As atividades normais são aquelas desempenhadas por pessoas, enquanto que as automáticas são realizadas sem intervenção humana. As conexões simples e múltiplas são responsáveis por descrever o encadeamento temporal das atividades: o tipo *end-start* restringe o início da execução da atividade destino para apenas depois do término da atividade origem; o tipo *end-end* habilita o término do destino somente após o término da atividade origem; finalmente, a conexão do tipo *startstart* habilita o início da atividade destino para ocorrer somente após o início da origem.

APSEE não é uma ferramenta de acompanhamento de projetos por si, mas uma eficiente ferramenta para modelagem (de forma visual) e simulação de projetos, de forma a se obter uma mensuração prévia de um projeto mapeado, em um processo específico.

A rede que é usada na ferramenta não segue um modelo formal de redes de Petri e foi criada especificamente para a ferramenta e não provê recursos que permitam o acompanhamento do processo, confrontando o planejado com o realizado, necessário para a criação de uma ferramenta real de controle de projetos.

3.4.5. Gerenciamento de Projetos

Existem várias ferramentas que controlam aspectos gerais do gerenciamento de projetos e variáveis de tempo, custo e recursos de atividades de qualquer área de atuação. Elas realizam análise probabilística, comparação de caminhos, geração automática de gráficos (de barra, pizza, etc.,) geram de impressão de relatórios e simulação. Estas ferramentas são descritas a seguir.

COSMOS (1998) combina os conhecimentos dos modelos de pontos de função, Putnam e COCOMO. os usuários podem ganhar uma compreensão das mudanças de requisitos do projeto e recursos que causam impacto no tamanho, esforço e cronograma do projeto.

Pode-se combinar vários projetos para formar um grande sistema, ou adicionar projetos existentes dentro do novo sistema. Isto permitirá que se calculem os pontos de função, total de linhas de código, total de esforço e o tempo de desenvolvimento acumulativo. Após ter incorporado um sistema novo, pode-se realizar algumas mudanças nos dados. Pode-se modificar a descrição do sistema, adicionar projetos ao sistema e configurar os atributos do sistema.

COSMOS permite que se ajuste o modelo para permitir incertezas nas entradas do modelo (as contagens do ponto da função ou as linhas de fonte do código). Pode-se incorporar uma porcentagem para indicar o nível da incerteza que se tem para uma entrada no modelo. Esta porcentagem do ajuste é usada então por COSMOS para calcular uma escala ao produzir os resultados para seu projeto ou modelo.

Pode-se usar a engenharia reversa do COSMOS para fazer análise do modelo. Começando com os resultados de ponto da função, COCOMO, ou modelo de Putnam, a engenharia reversa permite que se verifique o impacto do cronograma programando ou outros parâmetros do modelo.

Pertmaster (2005) permite realizar análise de risco e verificar incertezas de **tempo e custo** de análise em projetos. Usando a análise do risco pode-se encontrar a probabilidade de terminar um projeto no tempo e no orçamento. Quando um projeto é inscrito em Pertmaster o usuário insere a duração de cada tarefa baseado em sua experiência e a duração mais provável para essa tarefa. Há naturalmente uma possibilidade (ou risco) que a tarefa poderia ser mais longa ou mais curta do que esta duração. Se a análise do risco não for usada, então durações mínimas e máximas não estão sendo usadas. Usando a análise do risco muitos resultados diferentes para o projeto são analisados. Uma análise do risco permite que o usuário incorpore a duração mais provável e as durações mínimas e máximas. Um plano sem análise do risco mostra que há somente uma maneira possível para o projeto funcionar. Usando a análise de risco muitos resultados diferentes para o projeto são analisados.

ManagePro (1999) ajuda a definir recursos humanos de um projeto alocados a atividades. A seqüência de tarefas do projeto indica o que deve ser feito primeiro e a seqüência das tarefas. A seqüência representa suposições e planejamento sobre o que se vai fazer para alcançar os objetivos e resolver exigências ou obstáculos. Objetivos secundários podem ser inseridos em uma lista ou *checklist*. *Checklists* são usados para detalhar melhor as tarefas simples ou repetitivas, ou para conter todos os detalhes ou índices a serem dirigidos. A estrutura dos objetivos principais e secundários fornece uma boa opção para reuniões, com facilidade para acompanhar o progresso e adicionar objetivos novos. Os usuários desta ferramenta podem também sincronizar seus dados de ManagePro

com o *Outlook* ou *Palm Pilot*.

Analyzer (2005) permite realizar simulações e análises de atividades de um projeto. Ele também possui funções do *Flowchart* - diagrama que usa símbolos gráficos para mostrar a natureza e o fluxo das etapas em um processo. Os fluxogramas permitem a visualização de cada etapa e permite melhorar um processo. A ferramenta permite atribuição de dados aos campos para identificar e ajustar os valores além de definir as probabilidades nos pontos de decisão para proporcionar análise ou simulação. Todas as estatísticas do projeto são mostradas em tabelas e os caminhos das atividades são destacados em gráficos. A opção de simulação fornece uma possibilidade de criação de hipóteses: por exemplo: "Se a máquina número 12 quebrar ? E se José não voltar para o trabalho amanhã? E se nós cortássemos o custo disto, como isso nos afetaria ?"

A ferramenta **MinuteMan Project Management Software** (2003) é uma ferramenta para controlar projetos, atividades de planejamento e realizar coleta de dados. MinuteMan é usado vários tipos de projetos, incluindo construção e a engenharia. A ferramenta controla tempo, custo e pessoal.

PlanView (2005) permite gerenciar estratégias e capacidades de recursos (pessoas, processos, materiais); ganhar o controle de portifólios do projeto e gerenciar o custo total de serviços. Proporciona também alinhar estratégias e gerência de portfólio da empresa permitindo analisar e controlar o risco do projeto, usar o planejamento para preparar-se para mudanças e incertezas dando prioridade aos investimentos baseados nos critérios de decisão chaves (benefícios, *timelines*, custos e riscos).

A ferramenta **Microsoft Project** (MSPROJECT, 2003) dá apoio ao gerenciamento de pessoas e atividades. O Project é utilizado pelos gerentes de projetos que precisam de uma ferramenta de área de trabalho para gerenciar seus projetos de maneira independente, mas que não exigem coordenação rigorosa com outros gerentes de projeto nem a capacidade de gerenciar recursos a partir de um repositório central. Ele auxilia a organizar e gerenciar melhor o trabalho para tentar garantir que os projetos sejam entregues na data e dentro do orçamento.

É importante observar que, com a apresentação dessas ferramentas de gestão de projetos é possível observar que existem diversas ações sendo implementadas para auxiliar no gerenciamento de projetos, bem como diferentes variáveis sendo tratadas (tempo, custo, atividades e recursos). A Tabela 3.1. mostra o resumo destas ferramentas e as características de cada uma delas.

Mas estas ferramentas tratam de características específicas do gerenciamento, atuando apenas com um foco isolado, e não permitem o gerenciamento de processos de desenvolvimento de

software (modelagem de metodologias e visualizam do *workflow* do projeto).

A seguir são apresentadas as Tabelas 3.1. e 3.2. e as principais características das ferramentas discutidas.

Tabela 3.1: Tabela comparativa das ferramentas relacionadas com gerenciamento de processos

Característica	ODE	GENESIS	TABA	APSEE	RUP Builder	DotProject	VSTS	XPlanner	Primavera
Definição de Tarefas	√	√	-	-		√	√	√	√
Início, duração e fim de tarefa	√	√	-	-	√	√	√	√	√
Calendário	-	-	-	-	-	√	√	√	√
Importação/Exportação de dados	√	-	-	-	-	-	√	√	√
Help on line	√	√	√	√	√	√	√	√	√
Multiusuário	√	√	-	-	-	√	√	√	√
Banco de Dados	√	√	√		√	√	√	√	√
Domínio	**Sw**	**Sw**	**Sw**	**Sw**	**Sw**	**G**	**Sw**	**Sw**	**G**
Controle de Tempo	√	√	√	-	-	√	√	√	√
Controle de custos	√	-	√	-	-	-	-	√	√
Visualização gráfica do workflow	√	√	-	√	-	-	-	-	√
Instanciar metodologia ou padrão	-	√	√	√	**RUP**	√	**MSF, CMM**	**XP**	√
Análise de Riscos	√	-	-	-	-	-	√		√
Atualização de dados	√	√	√	√	√	√	√	√	√
Visualização de Relatórios	√	√	√	-	√	√	√	√	√
Impressão de Relatórios	√	√	√	-	√	√	√	√	√
Geração de Gráficos	√	-	√	-	-		√	√	√
Comparação de caminhos	-	-	√	-	-	√	-	-	√
Sincronismo com Outlook ou Palm Pilot	-	-	-	-	-	-	√	-	-
Aviso por e-mail	-	√	-	-	-	-	√	√	√
Web	√	√	-	-	-	√	√	√	√
PERT	-	-	-	-	-	-	-	-	-
Simulação	-	-	√	√	-	√	√	-	-
Análise probabilística	-	-	-	-	-	-	√	-	√
Preço	-	-	-	-	-	**free**	**US$ 2.000**	**free**	**NE**

Legenda:

G: uso geral

Sw: uso para desenvolvimento de softwares

P: Uso pessoal

CC: Caminhos críticos

P.S.C.: preço sob consulta

NE: preço não encontrado

Tabela 3.2: Tabela comparativa das ferramentas relacionadas com gerenciamento de projetos

Software Característica	**COSMOS 4.1**	**Pertmaster for MS Project 7.8**	**ManagePro Advisor 6.7**	**AllCLEAR Analyzer 6.1**	**Minute Man**	**PlanView Web Software 7.0**	**MS Project 2003**
Definição de Tarefas	-	√	√	√	√	-	√
Início, duração e fim de tarefa	-	√	√	√	-	-	√
Calendário	-	√	√	√	-	-	-
Importação/ Exportação de dados	√	-	√	√	√	--	√
Help on line	√	√	√	√	√	√	√
Multiusuário	-	-	√	√	√	√	√
Banco de Dados	-	-	√	√			√
Domínio	Sw	G	G	G	G	Sw	G
Controle de Tempo	√	√	√	√	√	√	√
Controle de custos	√	√	√	√	√	√	√
Visualização gráfica do *workflow*	-	-	-	-	-	-	-
Instanciar metodologia	-	-	-	-	-	-	-
Análise de Riscos	-	√	-	-	-	√	√
Atualização de dados	√	-	-	√	√		√
Visualização de Relatórios	√	√	√	√	√	√	√
Impressão de Relatórios	√	√	√	√	√	√	√
Geração de Gráficos	-	√	√	√	-	√	√
Comparação de caminhos	-	√	√	√	CC	-	
Sincronismo com Outlook ou Palm Pilot	-	-	√	-	-	-	√
Aviso por e-mail	-	-	√	-	-	-	√
Web	-	-	√	-	-	-	√
PERT	-	√	-		√	-	
Simulação	√	-	-	√	-	-	√
Análise probabilística	√	√		√			√
Preço	*free*	US$ 3.200	NE	NE	US$ 49,95	P.S.C.	NE

Legenda:

G: uso geral

Sw: uso para desenvolvimento de softwares

P: Uso pessoal

CC: Caminhos críticos

P.S.C.: preço sob consulta

NE: preço não encontrado

3.5. Padrões de gerenciamento de projeto

Nesta seção são apresentados padrões para gerenciamento de projetos e que influem no gerenciamento de projetos de software ou de outras áreas de atuação. Estes padrões podem ser modelados por Redes de Atividades e assim indicar à gerência os passos a seguir em um projeto.

3.5.1. PMBOK – Guia de Boas Práticas em Gestão de Projetos

O PMBOK (*Project Management Body of Knowledge*) é um guia desenvolvido pelo PMI que agrega os conhecimentos da área de Gerenciamento de projetos, baseado em práticas geralmente aceitas e aplicáveis à maioria dos projetos e tem servido como referência para o desenvolvimento das atividades de gerenciamento na busca pelo sucesso dos projetos (Ferraz, 2004).

A primeira versão do PMBOK fornece uma referência básica em nível de conhecimentos e práticas do gerenciamento de projetos, constituindo-se em um padrão mundial, aceito pela ANSI (*American National Steard Institute*). Além da ANSI, o IEEE (*Institute of Eletrical e Eletronic Engineers*) também o reconheceu como padrão de gerenciamento de projetos e está sendo utilizado como referência pela ISO (*International Steards Organization*) e por empresas que desenvolvem sua própria metodologia de gerenciamento de projetos.

O estudo do PMBOK é fundamental para que os gerentes de projetos possam compreender os ensinamentos e relacionamentos que, através das áreas de conhecimento e de processos preconizados pela metodologia, traduzem os conceitos mais atuais da prática de Gerenciamento de Projetos.

O Guia PMBOK 2004 abrange 44 processos em 5 grandes grupos de processos (iniciação, planejamento, execução, monitoramento, controle e finalização) e em 9 áreas de conhecimento (integração, escopo, tempo, custos, comunicações, recursos humanos, qualidade, riscos e aquisições).

A estrutura do PMBOK

O PMBOK fornece uma estrutura básica para o entendimento do gerenciamento de projetos. A estrutura é dividida conforme é descrito a seguir de acordo com o Guia do PMBOK (PMBOK, 2004a; PMBOK, 2004b).

Ciclo de vida e organização do projeto

Descreve o ambiente no qual os projetos operam. A transição de uma fase para a outra dentro do ciclo de vida de um projeto em geral envolve alguma forma de transferência técnica ou entrega. As entregas de uma fase geralmente são revisadas, para garantir que estejam completas e que sejam aprovadas antes que a próxima fase seja iniciada.

A maioria dos ciclos de vida de projetos compartilha diversas características:

As fases: Inicial, Intermediária e Final geralmente são seqüenciais e normalmente são definidas por algum formulário de transferência de informações técnicas ou de entrega de componentes técnicos.

Os níveis de custos e de pessoal são baixos no início, atingem o valor máximo durante as fases intermediárias e caem rapidamente conforme o projeto é finalizado.

As normas de gerenciamento de projetos

As normas de gerenciamento de projetos geralmente se enquadram em uma das duas categorias principais:

a) Os processos de gerenciamento comuns à maioria dos projetos na maior parte do tempo são associados entre si por seu desempenho visando um objetivo integrado. O objetivo é iniciar, planejar, executar, monitorar, controlar e encerrar um projeto. Esses processos interagem entre si de formas complexas, que não podem ser totalmente explicadas em um documento ou por meio de gráficos.

b) Os processos orientados ao produto especificam e criam o produto do projeto. Os processos orientados ao produto são normalmente definidos pelo ciclo de vida do projeto e variam por área de aplicação.

Processos de gerenciamento de projetos

Existem cinco grupos de processos de gerenciamento:

- Grupo de processos de iniciação - Define e autoriza o projeto ou uma fase do projeto.
- Grupo de processos de planejamento - Define e refina os objetivos e planeja a ação necessária para alcançar os objetivos e o escopo para os quais o projeto foi realizado.
- Grupo de processos de execução - Integra pessoas e outros recursos para

realizar o plano de gerenciamento do projeto para o projeto.

- Grupo de processos de monitoramento e controle - Mede e monitora

regularmente o progresso para identificar variações em relação ao plano de gerenciamento do projeto, de forma que possam ser tomadas ações corretivas quando necessário para atender aos objetivos do projeto.

- Grupo de processos de encerramento - Formaliza a aceitação do produto, serviço ou resultado e conduz o projeto ou uma fase do projeto.

Áreas de Conhecimento

a) Gerenciamento de integração do projeto

Descreve os processos e as atividades que integram os diversos elementos do gerenciamento de projetos, que são identificados, definidos, combinados, unificados e coordenados dentro dos grupos de processos de gerenciamento.

b) Gerenciamento do escopo do projeto

Descreve os processos envolvidos na verificação de que o projeto inclui todo o trabalho necessário, e apenas o trabalho necessário, para que seja concluído com sucesso. Ele consiste nos processos de gerenciamento de projetos.

c) Gerenciamento de tempo do projeto

Descreve os processos relativos ao término do projeto no prazo correto. Ele consiste nos processos de gerenciamento de projetos: Definição da atividade, Seqüenciamento de atividades, Estimativa de recursos da atividade, Estimativa de duração da atividade, Desenvolvimento do cronograma, Controle do cronograma – controle das mudanças no cronograma do projeto.

d) Gerenciamento de custos do projeto

Descreve os processos envolvidos em planejamento, estimativa, orçamento e controle de custos, de modo que o projeto termine dentro do orçamento aprovado.

e) Gerenciamento da qualidade do projeto

Descreve os processos envolvidos na garantia de que o projeto irá satisfazer os objetivos para os quais foi realizado. Ele consiste nos processos de gerenciamento: Planejamento da qualidade, garantia da qualidade, controle da qualidade**.**

f) Gerenciamento de recursos humanos do projeto

Descreve os processos que organizam e gerenciam a equipe do projeto. Ele consiste nos processos de gerenciamento de projetos: Planejamento de recursos humanos,

Contratar ou mobilizar a equipe do projeto, Desenvolver a equipe do projeto, Gerenciar a equipe do projeto.

g) Gerenciamento das comunicações do projeto

Descreve os processos relativos à geração, coleta, disseminação, armazenamento e destinação final das informações do projeto de forma oportuna e adequada. Ele consiste nos processos de gerenciamento de projetos: Planejamento das comunicações, Distribuição das informações, Relatório de desempenho, Gerenciar as partes interessadas

h) Gerenciamento de riscos do projeto

Descreve os processos relativos à realização do gerenciamento de riscos em um projeto. Ele consiste nos processos de gerenciamento de projetos:

Planejamento do gerenciamento de riscos, Identificação de riscos, Análise qualitativa de riscos, Análise quantitativa de riscos, Planejamento de respostas a riscos, Monitoramento e controle de riscos.

i) Gerenciamento de aquisições do projeto

Descreve os processos que compram ou adquirem produtos, serviços ou resultados, além dos processos de gerenciamento de contratos. Ele consiste nos processos de gerenciamento de projetos fornecedores, Administração de contrato e Encerramento do contrato, Planejar compras e aquisições, Planejar contratações, Solicitar respostas de fornecedores, Selecionar fornecedores.

O PMBOK foca em processos, dando ênfase na **pessoa como gerente**, e visa definir o processo de gerenciar e estabelecer o programa de gerência de projetos.

PMBOK visa à melhoria no desempenho e qualidade dos projetos desenvolvidos, e pode perfeitamente ser utilizado para melhorar a capacitação profissional dos gerentes de projetos que se baseiam no PMBOK.

3.5.2. *Capability Maturity Model Integration* (CMMI)

O CMMI - *Capability Maturity Model Integration* (Pinto, 2006) criado pelo SEI como

uma integração e evolução de diversos outros modelos é um modelo alinhado com a Norma ISO/IEC 15504 e apresentado em duas representações: uma por estágio (como o CMM, que define um conjunto de áreas de processo para definir um caminho de melhoria para a organização, descrito em níveis de maturidade), e outra contínua (semelhante à ISO/IEC 15504), que usa níveis de capacidade para caracterizar melhoria relacionada a uma área de processo.

Representação Contínua

Possibilita a organização utilizar a ordem de melhoria que melhor atender os objetivos de negócio da empresa. É caracterizado por Níveis de Capacidade (*Capability Levels*):

Nível 0: Incompleto

Nível 1: Executado

Nível 2: Gerenciado

Nível 3: Definido

Nível 4: Quantitativamente gerenciado

Nível 5: Em otimização

Representação Por Estágios

Disponibiliza uma seqüência pré-determinada para melhoria baseada em estágios que não deve ser desconsiderada, pois cada estágio serve de base para o próximo. É caracterizado por Níveis de Maturidade (*Maturity Levels*). Os níveis de maturidade correspondem à capacidade da empresa em realizar projetos grees e complexos. Eles são classificados da seguinte forma:

Nível 1. Inicial – a empresa possui processos ad hoc e caóticos. Neste ponto ela ainda não possui nenhuma das áreas de processo implementadas;

Nível 2. Gerenciado – a empresa possui processos gerenciados e caracterizados por projetos, mas muitas vezes ainda trabalha de forma reativa;

Nível 3. Definido – a empresa possui processos definidos e caracterizados para a organização. Normalmente a empresa trabalha de forma ativa;

Nível 4. Gerenciado Quantitativamente – a empresa mede e controla os seus processos;

Nível 5. Otimização – a empresa tem o foco em descobrir a causa de seus problemas e melhorar continuamente os seus processos.

Os níveis de maturidade de 2 a 5 podem ser caracterizados pelas atividades executadas pela organização para estabelecer ou melhorar o processo de software, pelas

atividades realizadas em cada projeto e pela capacitação do processo resultante nos projetos. A caracterização comportamental do Nível 1 também é descrita para estabelecer uma base de comparação com as melhorias de processos alcançadas nos níveis mais elevados de maturidade.

O CMMI avalia a maturidade de unidades organizacionais específicas de uma empresa e "*estar em um nível de maturidade"* significa que ela *implementa* todas as áreas de processo do respectivo nível. Ele pode ser considerado um guia de boas práticas para o desenvolvimento de projetos, produtos, serviços e integração de processos. Na prática, os

processos, técnicas, ferramentas e métodos utilizados por uma organização serão influenciados pelos conceitos do CMMI.

O CMMI fornece uma estrutura de conceitos para a melhoria da gestão e do desenvolvimento de softwares de forma consistente e disciplinada. Porém ele não garante que os produtos de software serão concretizados com sucesso ou que todos os problemas de desenvolvimento de software serão resolvidos. CMMI identifica práticas para um processo de software maduro e as características de um processo de software efetivo. A organização madura trata de todas as questões essenciais para um projeto bem sucedido, incluindo pessoas e tecnologia, tanto quanto processos.

3.5.3. Melhoria de Processos do Software Brasileiro (MPS.BR)

O **MPS.BR** ou **Melhoria de Processos do Software Brasileiro** (2003) é um movimento para a melhoria e um modelo de qualidade de processo voltada para a realidade do mercado de pequenas e médias empresas de desenvolvimento de software no Brasil. Ele nasceu de um projeto que uniu universidades, grupos de pesquisa e empresas, sob a coordenação da SOFTEX (Associação para Promoção da Excelência do Software Brasileiro) em dezembro de 2003.

O MPS.BR é baseado em modelos e normas internacionais com o objetivo de se tornar aplicável à indústria de software. Ele é baseado no CMMI, nas normas ISO/IEC 12207 e ISO/IEC 15504 e na realidade do mercado brasileiro. O MPS.Br é dividido em 3 partes: MR-MPS, MA-MPS, MN-MPS.

O **MR-MPS** apresenta 7 níveis de maturidade (o que é um diferencial em relação aos outros padrões de processo) que são:

A - Em Otimização;

B - Gerenciado quantitativamente;

C - Definido;

D - Largamente Definido;

E - Parcialmente Definido;

F - Gerenciado;

G - Parcialmente Gerenciado;

Cada nível de maturidade possui suas áreas de processo, onde são analisados os processos fundamentais (aquisição, gerência de requisitos, desenvolvimento de requisitos, solução técnica, integração do produto, instalação do produto, liberação do produto), processos organizacionais (gerência de projeto, adaptação do processo para gerência de projeto, análise de decisão e resolução, gerência de riscos, avaliação e melhoria do processo organizacional, definição do processo organizacional, desempenho do processo organizacional, gerência quantitativa do projeto, análise e resolução de causas, inovação e implantação na organização) e os processos de apoio (garantia de qualidade, gerência de configuração, validação, medição, verificação, treinamento).

Em seguida vem a Capacidade, onde são obtidos os resultados dos processos analisados, onde cada nível de maturação possui um número definido de capacidades:

AP 1.1 - O processo é executado;

AP 1.2 - O processo é gerenciado;

AP 2.2 - Os produtos de trabalho do processo são gerenciados;

AP 3.1 - O processo é definido;

AP 3.2 - O processo está implementado.

MA-MPS – Método de avaliação para melhoria do processo de software

Tem como objetivo orientar a realização de avaliações, em conformidade com a norma ISO/IEC 15504, em empresa e organizações que implementaram o MR-MPS. A avaliação MA-MPS inclui:

- Estruturação da Avaliação: planejar e preparar avaliação: plano de Avaliação / descrição dos indicadores de processo;
- Conduzir Avaliação - resultado da avaliação;
- Relatar resultados - relatório da avaliação;
- Registrar e publicar resultados.

MN-MPS – Modelo de negócio para melhoria do processo de software

Permite que as instituições que se propõem a implantar os processos MPS.Br (Instituições

Implementadoras) possam se credenciar através de um documento onde é apresentada a instituição proponente, seus dados, a estratégia de implementação do modelo, estratégia para seleção e treinamento de consultores para implementação do MR.MPS, etc.

Foi observado que os níveis do MPS.Br permitem que uma empresa implante processos de uma forma gradual. Essa idéia refletida para o mercado brasileiro de software permite que empresas de pequeno e médio porte possam tomar a iniciativa de definir processos. No Brasil, uma das principais vantagens do modelo é seu custo reduzido de certificação em relação as normas estrangeiras.

3.5.4 Modelo PMI-OPM3

O PMI - *Project Management Institute* lançou o seu Modelo OPM3 (OPM3, 2004), acrônimo de *Project Management Maturity Model*, desenvolvido a partir da pesquisa com outros tantos modelos pré-existentes de avaliação de Maturidade Organizacional e do apoio anônimo de aproximadamente 800 voluntários de mais de 35 países, inclusive do Brasil.

O Modelo OPM3 visa oferecer uma estrutura formal capaz de traduzir estratégias em resultados de sucesso, consistentes e previsíveis. Uma estrutura de melhoria contínua do ambiente de gestão de projetos das organizações construída através da recomendação de 'Boas Práticas', geralmente aceitas e previamente experimentadas por seus associados. Desse modo, em princípio, o Modelo OPM3 retrata uma trilha segura e referenciada, capaz de orientar os gestores organizacionais nos seus investimentos em iniciativas de aprimoramento da operação de gestão de projetos.

A aplicação do Modelo OPM3 está baseada em três elementos chave, inter-relacionados de modo seqüencial:

1) O **conhecimento** dos componentes do modelo de maturidade;
2) O **questionário de auto-avaliação** do estágio de maturidade da organização;
3) O **processo de melhoria** capaz de orientar gestores a deslocar a organização de um estágio atual para um estágio futuro desejado de maturidade.

O Modelo OPM3 proporciona **benefícios** visíveis às organizações que o adotam:

- Permite habilitar a organização a promover os projetos certos, da maneira certa, alinhados estrategicamente e em uma economia dinâmica e global, (b) Permite a flexibilidade de ser aplicado à diversos tipos de organizações, através de diferentes áreas de atuação, ramos de negócios, tamanhos, localizações

geográficas, etc.,

- Permite promover a conscientização e esclarecer a questão da maturidade organizacional por toda a alta direção e
- Permite associar diretamente o sucesso da organização à gestão eficaz e eficiente de projetos.

Mas algumas dificuldades no uso de OPM3 ainda foram encontradas:

- O fato de não haver ainda uma referência de processos para os domínios de Programa e Portfólio, tal qual existe para o domínio de Projetos (na forma do Guia PMI-PMBOK), tornou a interpretação das Boas Práticas para esses domínios repetitiva e desprovida de sentido aplicado;
- O questionário de auto-avaliação foi considerado repetitivo e burocrático;
- A ausência de um grau mensurável associado à avaliação da maturidade, tal como estabelecem outros modelos de avaliação de maturidade organizacional - que costumam medir maturidade através de uma escala de 1 a 5 - dificulta o entendimento, a comunicação interna e o estabelecimento de metas mensuráveis e compromissos para o aprimoramento da maturidade organizacional através do modelo OPM3.

3.6. Comentários

Neste capítulo foram apresentados os conceitos de projeto, gerenciamento de projetos, ciclo de vida de um projeto, ferramentas e trabalhos relacionados e padrões para gerenciamento de projetos em uso atualmente que são adotados por organizações na busca de se obter um melhor controle sobre projetos de software ou de outras áreas. Em (Farias, 2006) encontram-se estudos e descrições de outros padrões tais como: PSP (2006), TSP (2006), PRINCE2 (2006), Trillium (2005) e SPICE (2005).

Neste trabalho, a intenção é utilizar Rede de Atividades para modelar metodologias e padrões e para um projeto concreto, instanciar cada atividade e informar à equipe de desenvolvimento quais os próximos passos a seguir. Ao tentar desenhar as Redes de Atividades dos padrões foi visto que existe um pouco de dificuldade em transformar alguns deles em RAs, pois alguns não possuem uma seqüência de passos e etapas claras, possuindo apenas dicas e recomendações gerais para os engenheiros e empresas, nestes casos, deve-se consultar um especialista no padrão para ajudar na modelagem da Rede de Atividades.

Neste capítulo também foi observado que já existem na literatura, alguns esforços para uso de redes de Petri para gerenciamento de projeto, mas eles focam em pontos

específicos do projeto (recursos e tempos, ou modelagem de sistemas de documentação científica, ou modelagem de sistemas complexos sem considerar tempos), mas não foram encontrados trabalhos de extensões de redes de Petri para gerenciamento de processos de desenvolvimento de software (XP, RUP e SCRUM, dentre outros).

Existem ferramentas que auxiliam o processo de desenvolvimento de software, e só permitem a modelagem de metodologias específicas (não permitindo que se utilize a metodologia da própria instituição). Ou então permitem a modelagem de vários tipos de processos, mas algumas não possuem interface gráfica (tipo rede), e não seguem um modelo formal (por exemplo, redes de Petri).

As ferramentas comerciais, por exemplo: Microsoft Project 2003, ManagePro Advisor; Pertmaster for MS Project 7.8, dentre outras, permitem descrever tarefas, custos, gerar cronogramas e gráficos de barras. Mas elas apenas coletam, agrupam e organizam dados do projeto, não permitindo modelagem e acompanhamento de processos de desenvolvimento. O que se observa é que muitas vezes são necessárias duas ou três ferramentas dentre as citadas, em um mesmo projeto para modelar e acompanhar as etapas do processo de desenvolvimento e os dados do projeto (atividades, tempos, custos e recursos) e realizar totalizações, simultaneamente.

No próximo capítulo são descritas as Redes de Atividades e as Redes de Projeto, os conceitos relacionados e um exemplo de uso.

Capítulo 4

Rede de Atividades e Rede de Projeto

Neste capítulo, são apresentadas as Redes de Projetos e Redes de Atividades. A Rede de Atividades é uma estrutura que apóia a gerência de projeto de software e é usada para acompanhar o processo de desenvolvimento de um projeto. Ela é uma rede com uma notação gráfica mais expressiva do que as Redes de Petri convencionais, pois possui elementos de representação de atividades compostas, eventos e repositórios de artefatos. É mostrado que a Rede de Atividades é um modelo adequado para modelagem de diversos processos de desenvolvimento de software e controle de custos, tempos e recursos associados a atividades de um projeto.

Para determinar a qualidade estrutural de uma Rede de Atividades e permitir o cálculo correto de custos e prazos, é definido um tipo específico de Rede de Petri, denominado Rede de Projeto. São determinadas as características estruturais de qualquer projeto e definido o que representa um projeto bem estruturado. Para Redes de Projeto bem estruturadas é definido o conceito de bloco e operações de dobramento e desdobramento de blocos, como base para os cálculos de custos e tempos em uma Rede de Atividades. Finalmente é mostrada a conversão da estrutura de uma Rede de Atividades em uma Rede de Projeto.

4.1. Introdução

Um modelo para desenvolvimento de um projeto pode ser visto como uma estrutura gráfica (grafo) composta de um conjunto de atividades com possíveis dependências entre

elas. Duas atividades A1 e A2 podem ser seqüencialmente dependentes, se A2 só pode ser realizada depois de A1 terminar. Elas são independentes ou concorrentes se a execução de A1 não depende da execução de A2 e vice-versa. A terceira relação possível entre duas atividades é que elas formam duas alternativas para realizar a mesma tarefa. Outra característica de um projeto é que ele possui um único começo e um único fim.

Executar um projeto é similar a executar um programa (concorrente) em computação, onde temos declarações seqüenciais, concorrentes e declarações com *if-then-else* ou *case.* Existe uma analogia entre um *loop* em um programa e um *ciclo* no grafo da rede de um projeto, significando uma execução interativa de atividades similares. Desde que em um projeto o conjunto completo de atividades deve ser fixo, cada ciclo deve ter previamente um número de iterações estabelecido (como para uma declaração '*for'* em programação).

Usamos redes de Petri para modelar essa estrutura de projeto, onde as atividades são os lugares da rede, e o movimento de uma atividade para outra é dado por transições. A fim de modelar custos e tempos de um projeto, estendemos esta estrutura anexando valores correspondentes para as atividades e definimos funções para obter custos e tempos globais a partir das atividades individuais.

Como um programa bem estruturado evita declarações *goto* descontroladas, um projeto bem estruturado evita conexões cruzadas entre atividades a partir de pontos distintos da rede. A questão é que dependências entre atividades em um projeto podem ser mais liberais e a estrutura formada pode resultar em restrições indesejáveis.

Baseado nos conceitos de Redes de Petri, definidos no Capítulo 2, introduzimos a seguir um tipo específico de Redes de Petri, chamada Rede de Projeto Bem Estruturada, que servirá como fundamento para outro tipo de rede, a Rede de Atividades que podem ser usadas para desenhar o processo de desenvolvimento de um projeto e controlar custos e prazos.

4.2. Redes de Projeto

A estrutura de desenvolvimento de um projeto pode ser modelada por uma rede de Petri. Definimos, neste parágrafo, uma classe de Redes de Petri que atende exatamente aos requisitos estruturais de um ambiente de desenvolvimento de projetos. As características específicas desta rede servirão de base para a introdução da Rede de Atividades que permite modelar completamente os processos de desenvolvimento de um produto, com suas interdependências funcionais, para especificação de dados de tempo, custo, recursos humanos e materiais, e artefatos produzidos.

4.2.1. Definição da Rede de Projeto

Uma Rede de Projeto é uma rede de Petri N=<P,T,F>, em que

P é um conjunto de lugares, contendo um lugar *fim* ∈ P

T é um conjunto de transições, contendo uma transição *inicio* ∈ T;

F é um conjunto de arestas com as seguintes propriedades:

- Ela é dupla livre-escolha [1];
- O grafo associado a N é um reticulado [2] não-vazio com inf=*inicio* e sup=*fim*;
- O sistema <N, ∅> é vivo[3].
- **inicio*= ∅ e *fim**= ∅

Em uma Rede de Projeto, os lugares P da rede são chamados de atividades. Por ser dupla livre-escolha, evitam-se referências cruzadas indesejáveis na rede. Os limites do reticulado determinam o começo e o fim do projeto. A última propriedade garante que todo projeto chega ao fim.

Os limites inferiores e superiores são denominados *início(inf)* e *fim*(*sup*)do projeto. Deve-se observar que a definição de Rede de Projeto não permite ciclos. Se um projeto real contém execuções repetidas das mesmas atividades (como iterações em um processo incremental ou a construção de um carro com quatro pneus, por exemplo), consideramos, para as definições formais, que o número de repetições é fixo e que, portanto, o ciclo pode ser linearizado.

Relembrando que, dado um Sistema <R, M_0>, uma seqüência de marcações M_0,..,M_k do fecho de M_0 é dita um **caminho** se existem t_1,..,t_k tal que $t_i(M_{i-1}) = M_i$ para i=1,..k. Esta seqüência [t_1,..,t_k] é chamada de um **caminho de transição** de t_1 para t_k.. Deve ser garantido pela rede que todo projeto, em qualquer estado intermediário, chega ao fim. Isto é dado pela seguinte proposição:

Proposição 1*: em uma Rede de Projeto*

- *Todos caminhos maximais são da forma [∅,..,fim], e*
- *Todo caminho da rede contido no fecho de ∅ é um sub-caminho de um caminho maximal.*

---

[1] Uma rede N = <P, T, F> é chamada **livre-escolha** se para cada arco de P para T (p,t) ∈ F, implica que |p*|=1 o |*t| = 1. Chama-se a rede de **dupla livre-escolha** se a mesma propriedade se aplica para todos os arcos (t,p) de T para P, isto é, ., |t*|=1 ou |*p|=1.

[2] Um **reticulado** é um conjunto finito L parcialmente ordenado com a propriedade de que para todo a,b ∈ L existem dois elementos de L, chamados inf(a,b) e sup(a,b), tal que inf(a,b) ≤ a ≤ sup(a,b), inf(a,b) ≤ b ≤ sup(a,b). Desde que é finito, ele contém um menor elemento, chamado **inf**, e um maior elemento, chamado **sup.**

[3] Um sistema <N,M_0> é chamado de **vivo** se para todo lugar p ∈ P existe um M ∈ /M_0 que permite que seja marcado a cada disparo.

Prova: (i) suponhamos que existe um caminho maximal $[M_1,..,M_k]$ que não seja [∅,..,fim]. Analisemos o caso em que M_1 não é vazio, os outros casos são análogos. Seja $p \in M_1$ uma marca. Como uma Rede de Projetos é um reticulado tendo como mínimo a transição *início* e como o sistema <N, ∅> é vivo haverá uma transição t que marca p, ou seja, $p \in t^*$ e, pela vivacidade, deve haver um caminho de transições de *início* até t, e o acionamento de t marca p.

Se M_1 contém várias marcas, o mesmo argumento se aplica a todas as marcas de M_1.Logo temos um caminho $[\emptyset,..,M_1,..,M_k]$ e o caminho original não é mais maximal.

Proposição 2: *O grafo do fecho de ∅ em uma rede de projeto é um reticulado com inf = ∅ e sup = fim.*

Prova: ∅ habilita a transição *início* da rede. Como uma rede de projeto é acíclica, o grafo do fecho também será acíclico. É fácil de ver que o caminho máximo neste grafo também determina um grafo maximal na rede de projeto. Consequentemente, pela Proposição 1, este grafo deve ser da forma [∅,..,*fim*].

Dado um elemento $x \in P \cup T$, distinguimos três casos de evolução do projeto, dependendo da cardinalidade $|x^*|$:

- se, dado um arco (x,y), temos $|x^*| = |^*y| = 1$, nós chamamos o arco (x,y) de **linear**;
- se $x \in T$ e $|x^*| > 1$, temos uma **bifurcação concorrente (*concurrent forking*) de grau |x|**;
- se $x \in P$ e $|x^*| > 1$, temos uma **bifurcação alternativa** (***alternative forking)* de grau |x|.**

Isto significa que múltiplos arcos, saindo da transição, iniciam uma execução concorrente de diversas atividades, enquanto múltiplos arcos saindo de um lugar (uma atividade) permitem a execução de atividades alternativas mutuamente exclusivas para atingir um mesmo objetivo.

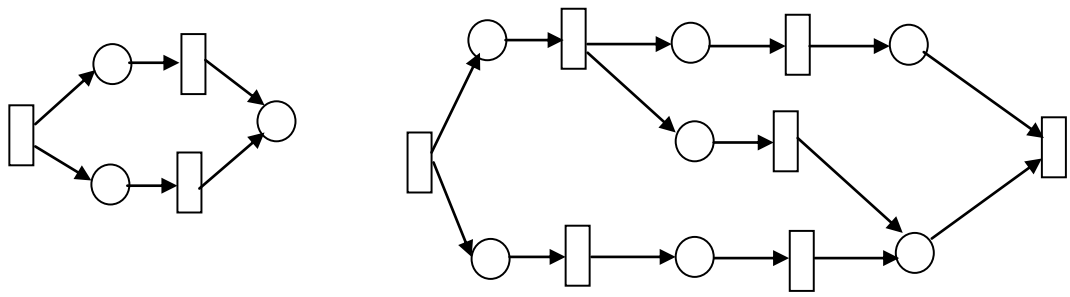
A cada bifurcação em uma rede de projeto deve corresponder, depois na rede, uma ***junção*** correspondente a fim de fechar o caminho múltiplo. Neste sentido definimos uma **junção de concorrência** como uma transição $t \in T$ tal que $|^*t| > 1$. Analogamente uma **junção de alternativas** é uma atividade $a \in P$ tal que $|^*a| > 1$.

Para garantir uma correlação adequada entre bifurcações e junções deve-se evitar dois casos na rede. Primeiro, devemos evitar blocos mistos (Figura 4.1a), isto é, uma bifurcação concorrente se fechar por uma junção alternativa, ou uma bifurcação alternativa se fechar por uma junção concorrente. O tipo da bifurcação deve ser do mesmo tipo da junção. O Segundo é que para uma rede bem estruturada, temos que

evitar caminhos de passagem cruzados de um "braço" da bifurcação para outro (Figura 4.1b).

A fim de evitar estes dois casos, consideramos que uma rede de projeto tem diversos níveis. Na transição *início* os arcos de saída são do nível 1. Se o arco de entrada de uma bifurcação é do nível *n*, a saída será do nível n + 1. Inversamente, se os arcos de entrada de uma junção forem do nível *n*, o arco de saída será do nível n - 1. Chamamos o primeiro caso de uma bifurcação n/n+1 e o retorno de uma junção n+1/n. Para uma junção seguida de uma bifurcação (Figura 4.2), o nível continua o mesmo, uma vez que a junção opera n-1 e a bifurcação compensa isso com n +1. No caso em que duas bifurcações se juntam de uma só vez, consideramos uma transição/atividade fantasma (*dummy)* na rede de tal modo que o nível mais elevado da bifurcação se junta primeiro (Figura 4.3).

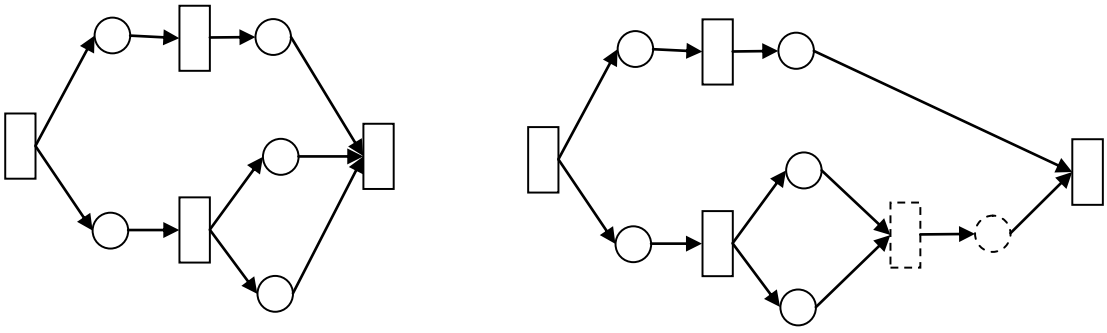
Denotamos por [t]*=n+1 se a transição *t* é uma bifurcação n/n+1 e por *[t]=n+1 se *t* é uma junção n+1/n. Temos então [início]*=*[fim]=1.



(a) bloco misto (b) caminhos de passagem cruzados

Figura 4.1: Estruturas de rede indesejadas

Figura 4.2: Dois casos de *join/fork*



(a) múltiplas transições *join* (b) múltiplos *join* eliminados por transições fantasmas

Figura 4.3: Eliminação de múltiplos joins

Agora, podemos estabelecer a seguinte propriedade:

Propriedade bem-estruturada: o grafo de uma rede de projeto é dito bem-estruturado se:

- para cada junção na rede, depois de eliminar múltiplos *joins*, todos os arcos são do mesmo nível;
- para cada bifurcação concorrente ou alternativa n/n+1 x, existe uma correspondente junção n+1/n y do mesmo tipo tal que x< y. O menor y que satisfaz esta propriedade chama-se o **fecho** (***closure***) de x.

Portanto, definimos uma **Rede de Projeto Bem Estruturada** como uma rede de projeto cujo grafo é bem estruturado.

Nosso objetivo agora é mostrar que a análise de uma rede projeto bem estruturado pode ser reduzida a analisar cada bloco de uma forma *bottom-up*. Assim o cálculo das métricas de um projeto complexo será obtido pela combinação adequada de cálculos em blocos elementares.

Dado um caminho de transição de t_1 para t_n, definimos o **bloco** $[t_1\ t_n]$ como sendo a sub-rede:

$N' = <S', T', F'>$ de N, dada por:

$S' = \{s / s \in t_i^* \text{ para algum } t_i \in \{t_1,..,t_{n-1}\}\}$

$T' = \{t_1,..,t_n\}$

$F' = F|_{<S',T'>}$

Analogamente, seja a_1 uma bifurcação alternativa e a_n, o fecho correspondente. Neste caso, temos que para cada transição $t \in a_1^*$ um caminho

(alternativo) de t para uma transição t' $\in$ *s_n. Se $\{t_1,..,t_k\}$ é o conjunto de todas transições de todos caminhos de a_1 para a_n, definimos o **bloco $[a_1, a_n]$** como a sub-rede:

N' = <S', T', F'> de N, dada por:

S' = $\{s / s \in t_i^*$ para algum $t_i \in \{t_1,..,t_{n-1}\}\}$

T' = $\{t_1,..,t_k\}$

F' = F|<S',T'>

Denotamos um bloco por $[b_1,b_n]$ em que b_1 e b_n são os pontos iniciais e finais de um bloco.

Um **dobramento concorrente** (*concurrent folding*) de um bloco em uma rede N é uma transformação de N pela substituição do bloco $[b_1,b_n]$, em que b_1 é uma bifurcação concorrente e b_n o respectivo fecho, por um único estado $S_{t1n.}$ Isto significa que o bloco N' = <S', T', F'> é reduzido para N" = <S_{t1n}, $\{t_1,t_n\}$, { <t_1, S_{t1n}>, < S_{t1n}, t_n>}>.

Analogamente, um **dobramento de alternativas** (***case folding***) consiste na substituição do bloco dentro de um caminho de lugares $[s_1, s_n]$ por um único caminho T_1-*s*-T_n,, obtendo-se uma sub-rede N" = <$\{s_1,s, s_n\}$, $\{T_1\ T_n\}$,{<s_1, T_1>, < T_1, s>, < s, T_n >, < T_n, s_n>}>.

Se temos um caminho linear T_1-s_1-T_2-s_2-T_3, um **dobramento seqüencial** (***sequential folding***) converte este caminho em um único T_1-s-T_3.

Um bloco é **elementar** se não houver nenhum bloco dentro dele. Isto significa que um bloco [b1, bn] é elementar, se para todo bloco $[c_1, c_n]$ com $b_1 \leq c_1$, e $c_n \leq$ bn temos $b_1 = c_1$ e $c_n = b_{n.}$

Um dobramento de um bloco elementar $[b_1, b_n]$ em uma rede de projeto N, cria uma nova estrutura N' determinada pelas regras de dobramento definidas acima. É importante salientar que qualquer dobramento não afeta a qualidade de uma rede ser uma Rede de Projeto.

Proposição 3: *Para uma rede de projeto N, a rede N' obtida após um dobramento é uma rede de projeto.*

Prova: a rede que substitui o trecho do bloco na rede pelo dobramento é uma sub-rede linear que liga dois pontos x, y em ordem x<y. Portanto, é claro que esta simplificação não introduz arcos que pudessem influir na livre-escolha, não remove a estrutura de reticulado ou evita um andamento até o fim do projeto.

Podemos aplicar dobramentos elementares repetidamente, obtendo uma seqüência N, N^1, .., N^k, onde N^k já não é mais dobrável. Esta seqüência é chamada de um

dobramento completo de N.

Agora, podemos enunciar o principal teorema sobre redes de projeto bem estruturadas:

TEOREMA: *Para uma Rede de Projeto bem estruturada N, todo dobramento total N, N^1, .., N^k, leva ao mesmo N^k que é um bloco seqüencial elementar da forma [início, fim].*

Prova: Dividimos a prova em dois lemas. O primeiro afirma que existe um dobramento completo da rede que é um bloco elementar seqüencial. O segundo garante que todos os dobramentos completos devem sair do mesmo bloco seqüencial [início, fim].

Lema 1: *Em uma Rede de Projeto bem estruturada N, todo dobramento total N, N^1, .., N^k leva ao mesmo N^k.*

Prova: o dobramento ocorre em dois passos:

1. Eliminar todas as seqüências não-elementares para seqüências elementares aplicando dobramento seqüencial repetidamente.
2. Seja n + 1 o nível mais alto que ocorre na rede. Se (n + 1) > 1 existe uma bifurcação n/n + 1 e, uma vez que a rede é bem estruturada, existe um correspondente fecho y de x. Desde que n +1 é o dobramento de mais alto nível, e seqüências não elementares tem sido removidas, o bloco [x, y] deve ser um bloco elementar. Podemos, portanto, dobrar este bloco e eliminar este nível de dobramento n+1. Isso pode ser aplicado a todos os dobramentos atuais x / x +1níveis n+1 da rede.
3. Repetir passos 1 e 2 até que n+1=1

Após n repetições dos passos acima, a rede resultante é um dobramento completo de N. O último bloco N^n não pode ser um bloco concorrente ou alternativo, uma vez que, nestes casos, ainda temos 2 níveis de blocos que podem ser dobrados.

Lema 2: *Todos os dobramentos completos de uma Rede de Projeto levam ao mesmo bloco final da forma [início, fim].*

Prova2: A proposição 1 demonstrou que cada dobramento completo leva a um bloco seqüencial elementar [t_1,t_2]. Mas as transições de *início* e *fim* não podem ser eliminadas, pois uma sempre será o início de um bloco e a outra sempre o final de um bloco. Logo t_1=início e t_2=fim.

A principal conseqüência deste teorema é que a análise das redes de projeto complexas, por meio de custos e tempos pode ser reduzida à análise de blocos elementares e dobramentos correspondentes.

4.3. Rede de Atividades

Uma Rede de Atividades é um tipo de rede com representação gráfica para representar atividades de um projeto e os desenvolvedores alocados. Ela é utilizada para modelar e acompanhar as atividades de um projeto em desenvolvimento e suas interdependências funcionais (Torres, 1996).

A cada atividade podem ser associados atributos de recursos, custo e tempo, possibilitando simulações e correções de custo e prazo globais para um projeto desde que se modele as redes segundo a estrutura da Rede de Projeto.

Uma RA pode ser convertida em uma rede de projeto, podendo-se assim realizar cálculos de custos e tempos totais através de algoritmos citados no item 4.4.1.

Uma forma preliminar destas redes foi criada por Torres (1996) e definida como uma estrutura de apoio à gerência de projeto de software, controlando o processo de desenvolvimento de software de uma metodologia específica denominada FADO (Furtado, 1993). Torres fez uma definição de Rede de Atividades, onde foram definidos elementos de: lugar, transição, evento, atividade simples e composta e definiu regras de ligação entre os elementos. Torres definiu a atividade básica e descreveu um algoritmo simples para conversão de rede de atividade para rede de Petri do tipo Condição-Evento.

Nesta tese, o trabalho de Torres foi continuado e ampliado para abranger novos elementos para o grafo da RA, novos tipos de caminhos para Redes de Atividades (caminhos paralelos, alternativas e *loops*) e regras para eles, blocos mistos (*join/fork*), bloco de concorrência, bloco de conflito e bloco alternativo com caminhos de passagem cruzados. Foi desenvolvido o conceito de Rede de Projeto, uma rede bem estruturada e seu correspondente *dobramento*, a fim de facilitar os cálculos de custos e tempos. Também foram definidos novos algoritmos para os cálculos, fórmulas matemáticas, regras e definições formais que foram criados de acordo com a realidade atual dos projetos, padrões e processos de software.

Redes de Atividades podem ser aplicadas a outros ambientes de desenvolvimento de sistemas complexos, tais como construção civil, construção de máquinas, sistemas hospitalares, etc.

A cada atividade são alocados recursos humanos e operacionais (computadores, impressoras, etc.) e atribuídos custos e prazos. Cada atividade é escalonada e monitorada por gerentes de projeto e executada por desenvolvedores que podem utilizar ambientes de desenvolvimento de software para executá-la. Uma atividade também pode ser executada por ferramentas automatizadas, sem intervenção humana. Toda atividade possui uma descrição, especifica os artefatos necessários, as relações de dependência com outras atividades, as datas de início e fim planejadas, os recursos a serem alocados e os agentes

responsáveis pela mesma.

Quando for aplicada certa metodologia como RUP- *Rational Unified Process* (Jacobson, Booch e Rumbaugh, 1999), eXtreme Programming (2004) ou YP- easY Process (2005), etc., ou outra já existente na instituição, ou ainda um padrão de desenvolvimento de software (Pinto, 2006), PMBOK(2004a), TSP (2006), PSP (2005), etc.), a um projeto, cria-se a RA da metodologia ou do padrão. Esta RA será armazenada no banco de RAs e poderá ser instanciada para o projeto. A cada atividade serão atribuídos recursos, custos e prazos previstos (início e fim estimados) para o projeto, possibilitando simulações e correções e prazos finais para o projeto.

Em analogia às Redes de Petri, podemos distinguir dois tipos de Redes de Atividades: Elementares e Expandidas. Na Rede de Atividade Elementar, a cada atividade pode estar alocado um marcador, indicando que a atividade está em execução ou encerrada. Na Redes de Atividades Expandida pode haver vários marcadores individualizados, cada um representando um desenvolvedor.

4.3.1. Definição Formal da Rede de Atividades (Básica)

Uma **Rede de Atividades Básica** é uma estrutura *ra* = <A, T, F, E, α, β, tp, tr, cp, cr>:

A = AT ∪ AF são atividades (AT) ou artefatos (AF);

AT = AB ∪ AC são atividades básicas (AB) ou composta (AC);

T é um conjunto de transições;

E é um conjunto de eventos;

F ⊆ ATxT ∪ TxAT ∪ ATxAF ∪ AFxAT ∪ ExT ∪ TxE é um conjunto de arestas;

α: (AT∪E) → {d,e,f,s} é a função que determina os estados possíveis das atividades,

podendo estar *desativada*, *em execução, finalizada* ou *executando sub-rede*, respectivamente. O estado *executando sub-rede* só se aplica a elementos de AC (atividades compostas).

β: E → {0,..,n} é a função de marcação de eventos

tp: AT → Tempo, que associa a cada atividade o tempo previsto

tr: AT → Tempo, que associa a cada atividade o tempo real

cp: AT → Custo, que associa a cada atividade o custo previsto

cr: AT → Custo, que associa a cada atividade o custo real

Uma transição está habilitada quando as atividades de entrada estão *finalizadas*

($\alpha(a_t)$=f), os eventos de entrada estão marcados e todas as atividades de saída estão *desativadas* ($\alpha(a_t)$=d).

O *acionamento* de uma transição habilitada consiste em:

- Alterar os estados das atividades de entrada de *finalizada* para *desativada*;
- Desmarcar os eventos de entrada;
- Marcar as atividades de saída como *em execução*;
- Marcar os eventos de saída.

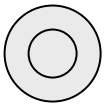
Uma Rede de Atividades permite modelar as principais interdependências que podem existir entre o desenvolvimento de atividades de um projeto. São elas:

- *Execução seqüencial*. Duas atividades a1 e a2 têm uma execução seqüencial se existe uma transição t, tal que < a_1,t> e <t, a2> são arestas em F;
- *Execução paralela*. Duas atividades a1 e a2 são independentes se existe uma transição t tal que, <t, a1> e <t, a2> estão em F;
- *Execução alternativa*. Duas atividades a1 e a2 são mutuamente exclusivas se existe uma atividade a0 e transições t1 e t2, tal que < a_0, t_1>, < a_0,t_2>, < t_1, a_1>e <t_2, a_2> estão em F.
- *Loop*. Uma Rede de Atividades RA tem um loop em uma atividade 'a' se existe um caminho cíclico de 'a' e retorno para 'a'. Neste caso, o grafo RA é cíclico.

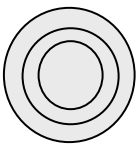
Esta caracterização de paralelismo ou alternativas pode facilmente ser generalizada de atividades individuais para sub-redes.

4.3.2. Visualização gráfica da Rede de Atividades

Uma Rede de Atividades pode ser visualizada com os seguintes elementos gráficos:

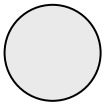


Atividade básica - se estiver com uma marcação no círculo interno, significa que a atividade está *sendo executada*; se estiver com uma marcação no anel externo e sem marcação no anel interno, representa uma atividade *encerrada*, habilitando a transição para outras atividades que dependem desta; se não estiver marcada, significa que a atividade está desativada.



Atividade composta – representa uma atividade realizada por uma sub-rede associada a ela. Uma marcação no anel interno significa que a sub-rede pode ser iniciada, a marca no círculo intermediário significa que a sub-rede esta sendo executada e uma marcação no anel externo significa

que a sub-rede encerrou suas atividades, habilitando a transição para outras atividades.



Artefato – repositório que armazena artefatos produzidos pelas atividades e que poderão ser utilizados pelas atividades posteriores.

Transição - representa a transição entre atividades encerradas e novas atividades, dependendo das anteriores. Se acionadas as atividades de saída se tornam ativas.

Uma aresta dirigida permite estabelecer ligações entre atividades, eventos, artefatos e transições.

Uma marca preta representa uma marcação na rede. Ela determina o estado da atividade (em execução, desativada ou encerrada)

Evento - permite modelar condições ou circunstâncias necessárias para o acionamento de uma transição.

Versões estendidas de Redes de Atividades consideram diversas marcas em uma mesma atividade, significando execuções repetidas de uma mesma atividade. Além disso, se a mesma atividade está sendo executada por diversos indivíduos, eles podem ser identificados por variáveis, tais como ´x´, ´y ´.

O significado das ligações entre elementos é o seguinte:

- Uma ligação de uma atividade básica para uma transição significa que a transição está habilitada para disparar quando todas as atividades de entrada estão marcadas no anel externo e todas as atividades de saída não tem marcação. O disparo da transição remove as marcas de todas as atividades de entrada.
- Uma ligação de um evento para uma transição significa que a transição só pode ser acionada se o evento ocorrer.
- Uma ligação de saída de uma transição para um evento significa que o acionamento da transição torna o evento verdadeiro.
- Uma ligação de uma atividade composta para uma transição significa o mesmo das atividades básicas, exceto para uma transição especial chamada **transição de entrada para sub-rede** ***(snit: sub-net input transition)***. A *snit* tem uma entrada de uma atividade composta e está habilitada quando a entrada da atividade está marcada no anel interno e seu disparo move a marca para o anel

intermediário.

- Uma ligação de uma transição para uma atividade significa que o disparo marca a atividade no anel interno. Uma exceção é uma transição especial para uma atividade composta, chamada **transição de saída da sub-rede** (***snot: sub-net output transition***). Seu disparo move a marca da atividade composta do anel intermediário para o anel externo.
- Uma ligação de entrada de uma atividade para um repositório de artefatos significa que a atividade gerou uma marcação, um artefato, e ele será armazenado no repositório para ser usado por outra atividade;
- Uma ligação de saída de um repositório de artefato para uma atividade significa que quando a atividade estiver em execução irá usar o artefato gerado anteriormente e a remoção da marca ou objeto.

Uma transição está habilitada quando todas as atividades de entrada (que pode ser um conjunto vazio) estiverem encerradas e todas de saída estiverem desativadas. O acionamento da transição torna todas as atividades de entrada desativadas e as de saída em execução.

Nos exemplos a seguir, são consideradas redes com no máximo uma marca, ilustrada pela marca preta. As figuras 4.4, 4.5, 4.6 e 4.7 ilustram a seqüência de passagem de atividades de uma atividade para outra. Estas atividades exemplificam as primeiras etapas do desenvolvimento de um *software* segundo a metodologia eXtreme Programming. A Figura 4.4 mostra a rede com a atividade "Descrever *user stories*" em execução, na Figura 4.5 "Descrever *user stories*" está encerrada e a transição T_2 habilitada. Em 4.6 a transição T_2 já disparou e a marca já caminhou para a atividade seguinte "Realizar reunião de planejamento". A Figura 4.7 mostra que a atividade "Realizar reunião de planejamento" já encerrou e o artefato Plano de liberação foi gerado e armazenado no repositório.

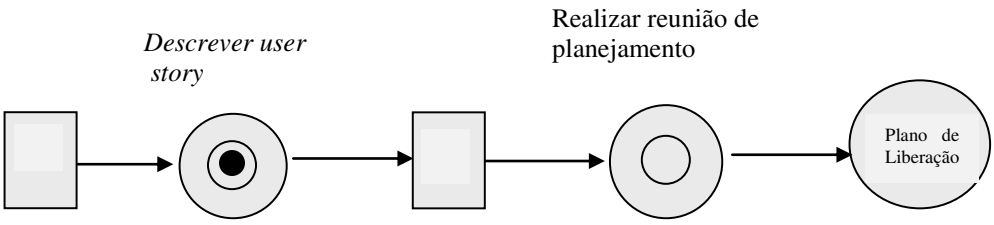


Figura 4.4: Rede de atividades com atividade "Descrever user stories" em execução

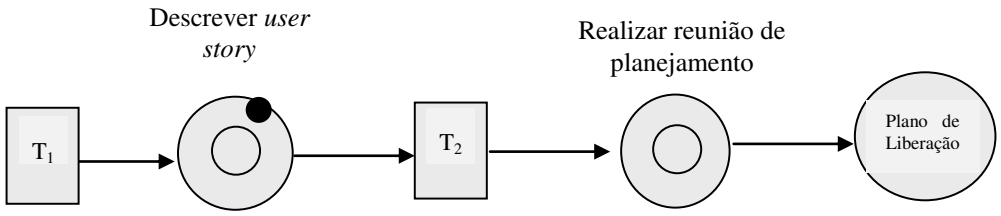


Figura 4.5: Rede de atividades com atividade "Descrever user stories" encerrada e a transição T2 habilitada

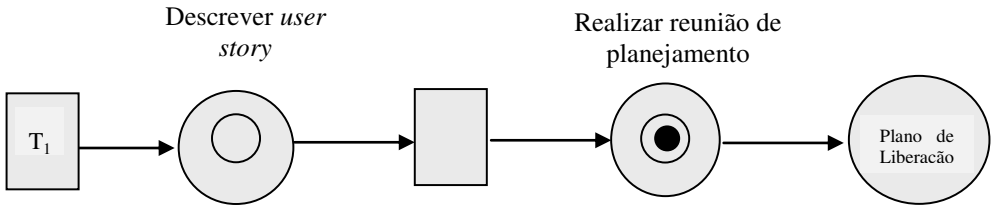


Figura 4.6: Rede de atividades com atividade "Realizar reunião de planejamento" em execução

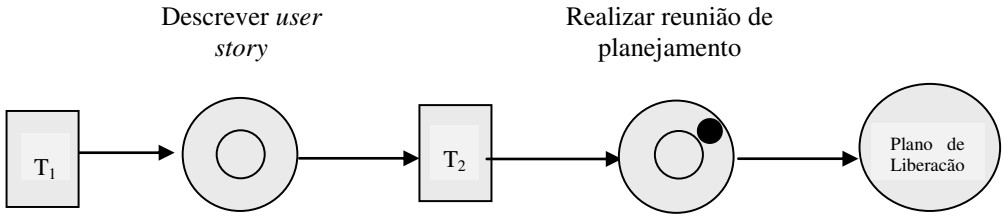


Ilustração 4.7: Rede de atividades com atividade "Realizar reunião de planejamento" finalizada

A execução de uma atividade pode resultar na criação de um artefato, como uma mensagem, um diagrama, código ou documento em geral. Os repositórios de artefatos são visualizados como círculos simples, que serão conectados diretamente com as atividades geradoras e consumidoras destes dados. Isto pode ser visto nas Figuras 4.4 a 4.7, onde se tem o repositório chamado Plano de Liberação que é o artefato gerado na etapa de "Realizar reunião de Planejamento".

O desenvolvimento de projetos complexos não é completamente linear, pois algumas partes do projeto podem ser desenvolvidas em paralelo, e em outros casos, podem ser previstas realizações alternativas de uma mesma atividade.

Uma rede de atividades pode descrever atividades paralelas ou atividades alternativas. Em uma RA podem ser modeladas **atividades paralelas**. Por exemplo, se temos dois desenvolvedores implementando dois módulos 1 e 2 que são independentes

entre si, na figura 4.8, as atividades “Implementar Módulo1” e “Implementar Módulo 2” são paralelas e a atividade “Realizar Teste integrado” depende do encerramento das duas para ser executada.

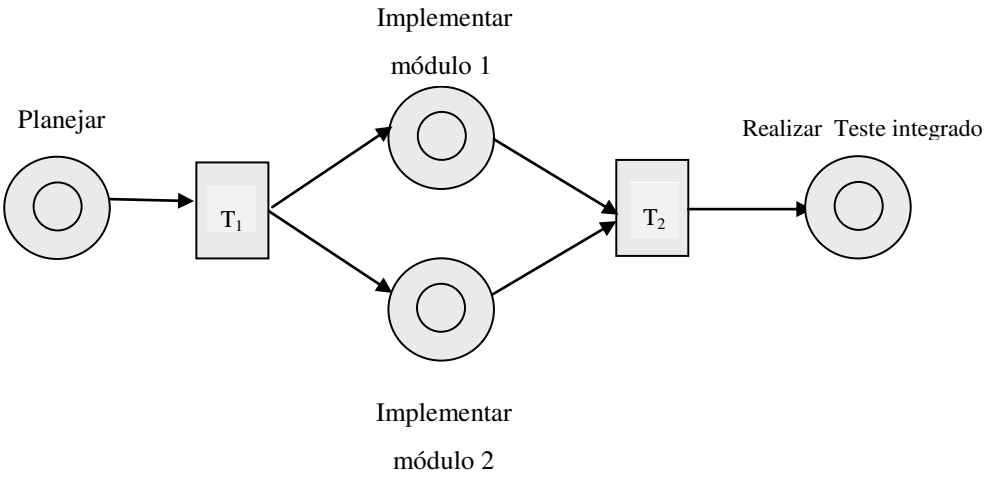


Figura 4.8: Rede de atividades com atividades paralelas

Se uma atividade tem mais de uma transição de saída, apenas uma delas pode ser disparada, configurando **atividades alternativas.** Por exemplo, na Figura 4.9, as atividades “Implementar em Java” e “Implementar em PHP” são duas alternativas diferentes de implementar um módulo, isto é, existem dois caminhos para chegar até a atividade “Realizar Teste integrado”.

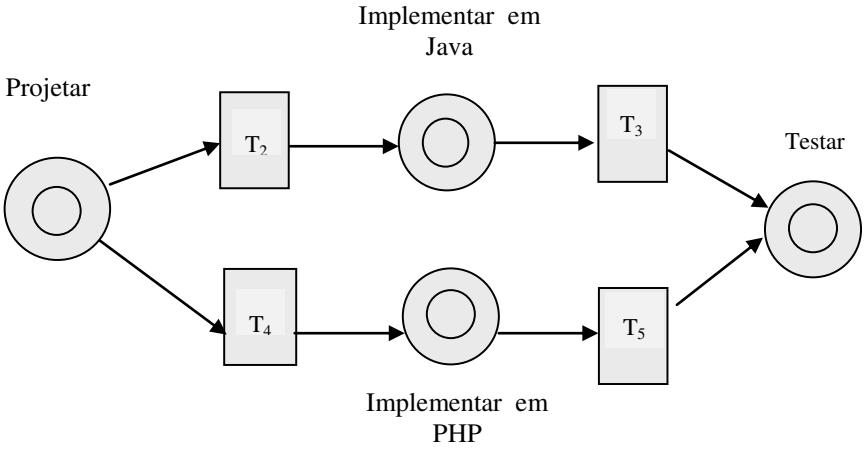


Figura 4.9: Rede de atividades com atividades alternativas Implementar em Java ou Implementar em PHP

4.4. Conversão de Redes de Atividades em Redes de Projeto

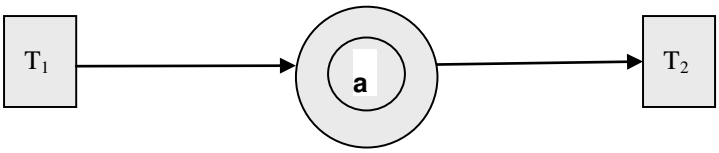
Dada uma Rede de Atividades *ra* = < A, T, F, E, α, β, tp, tr, cp, cr > e subestrutura

<AT, T, F> pode ser convertida em uma **Rede de Projeto** para efetuar cálculos e simulações de tempos e custos. Em particular, queremos obter uma Rede de Projeto para utilizar as características específicas destas redes. Como o principal objetivo desta conversão são os cálculos de custos e prazos, são desconsiderados eventos e artefatos, pois estes não têm influência nestes cálculos.

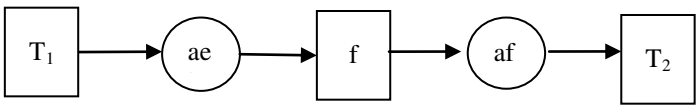
Como uma atividade a ∈ AT em uma RA é um pouco mais complexa do que um lugar de uma rede de Petri, algumas transformações são necessárias. Estas transformações devem converter as regras de disparo das Redes de Atividades para regras de disparo das Redes de Petri.

Dada a subestrutura <AT, T, F> de uma Rede de Atividade, pode ser construída a rede de Projeto correspondente *rp* = <A',T',F' > pelos passos a seguir:

- Para cada atividade básica a ∈ AB, os estados *em execução* e *finalizado* são modelados por dois lugares a_e e a_f. O término da atividade *a* é uma transição f de a_e para a_f. O disparo de T1 em *ra* com marcação α(a) = *em execução* é convertido na marcação de a_e, m(a_e) = TRUE. Quando a atividade é finalizada, *f* é disparada, e move a marcação de a_e para a_f . Este estado habilita a transição T_2 a continuar o projeto. Resumindo, <{*a*}, {T1, T2}, {<T_1,a>, <a,T_2>}> é substituída por <{a_e, a_f}, {T_1, f, T_2}, {<T_1,a_e >, <a_e, f>, <f,a_f>, <a_f,T_2>}>. A Figura 4.10 mostra a transformação de uma atividade básica da RA na correspondente rede de Petri.



(a) Atividade básica a ser convertida

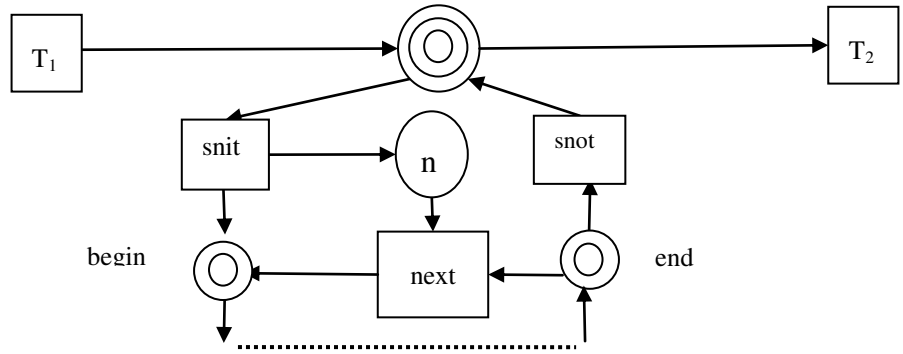


(b) Atividade convertida

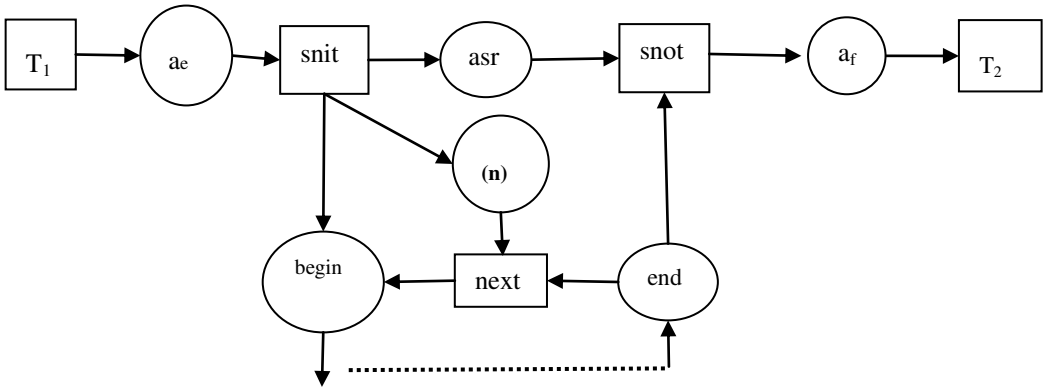
Figura 4.10: Conversão de uma atividade básica

- A conversão de uma atividade composta e os estados: *em execução, executando sub-rede* e *finalizada* são dados por três lugares: a_e, a_{sr} e a_f, respectivamente. A transição *snit* transfere o controle para a sub-rede e marca a_{sr}. Depois de terminar a sub-rede, *snot* move a marca de a_{sr} para a_f (Figura 4.11). A ilustração

da Figura 4.11 mostra o caso quando a sub-rede é um *loop*. Na RA há um artefato denominado *n* que representa um contador do número de repetições da sub-rede. Ele será convertido para um lugar com n marcas a serem consumidas a cada passagem pelo *loop*. A transição *snit* também marca o número de *loops* 'n' e *next* remove uma marca a cada vez. Portanto, se o lugar 'n' está vazio, apenas *snot* está habilitada e o *loop* pode ser terminado. Se a sub-rede não é um *loop* e será executada apenas uma vez, a estrutura é similar a Figura 4.11, sem o lugar 'n'e a transição *next*.



(a) Atividade composta de uma sub-rede com *loop*



(b) Atividade composta convertida

Figura 4.11: Conversão de uma atividade composta

Definição: Uma RA cuja conversão determina uma Rede de Projeto Bem-Estruturada é chamada de **Rede de Atividades Bem-Estruturada**.

4. 4.1. Cálculo de Custos e Tempos das Redes

Um dos maiores propósitos do uso das Redes de Atividades nesta tese é controlar custos e tempos de projetos. Uma condição para obter estes valores é que a cada atividade individual tenham sido atribuídos valores estimados de custos e tempos, e durante a

execução do projeto estes valores são substituídos por custos e tempos reais de desenvolvimento. Uma vez que estas avaliações iniciais não são muito fáceis de definir em projetos reais, pode existir uma ajuda através da abordagem de multi-camadas utilizada no sistema GDP - Gerência de Desenvolvimento de Projetos. Se o gerente de projeto usar uma metodologia conhecida, a RA da metodologia pode ser utilizada para instanciar o projeto concreto e oferecer alguns valores padrão para as atividades típicas da metodologia, obtidos a partir de projetos similares anteriores. Outra condição é que o projeto esteja bem estruturado, tal como definido na Seção 4.2.1. Este requisito é semelhante ao da escrita bem estruturada de programas de computador.

Seja *ra* = <A, T, F, E, α, β, cp, Cr, tp, tr> uma Rede de Atividades de um projeto que será desenvolvido e *rp* = <A', T', F'> a rede de projeto correspondente. Note que para que *rp* seja considerada uma rede de projeto bem-estruturada, todos os *loops* foram linearizados.

A partir do teorema da seção 4.2.1, sabemos que cada rede de projeto *rp* tem um dobramento completo rp_n, rp_{n1}, .., rp_{nk} dado por uma seqüência de blocos elementares b_1, b_2, .., b_k. Este conjunto de blocos elementares pode ser considerado como uma composição hierárquica da rede original para redes de nível mais elevado que representam o mesmo projeto. Portanto, custo e prazo de um projeto como um todo pode ser obtido pelo cálculo de custos e prazos de blocos elementares combinados de acordo com os dobramentos correspondentes.

Para caminhos alternativos de um projeto, pode-se obter o caminho mais rápido ou o menos caro. Por isso, precisa-se de dois algoritmos, um para obter o caminho mais rápido (algoritmo 1) e outro para obter o menos dispendioso (algoritmo 2). Pode-se obter também os valores de custo e tempo referentes a determinados caminhos escolhidos (customizados) (algoritmo 3).

Os algoritmos 1 e 2 mostram que se se o caminho do projeto é seqüencial, o custo do projeto será a soma dos custos de todas as atividades e o tempo total será a soma dos tempos de cada atividade individual. Se o caminho do projeto é concorrente, então o custo total será a soma dos custos individuais das atividades e o tempo será o maior tempo do somatório dos tempos das atividades dos caminhos concorrentes.

Se o bloco tem caminhos alternativos, apenas um caminho será seguido, e pode ser escolhido o menos dispendioso ou o mais rápido.

O usuário pode escolher também um caminho de projeto a percorrer, pelo menor custo ou menor tempo de desenvolvimento, conforme mostra o algoritmo 3.

Vamos converter as funções de custos e tempos da RA para a Rede de Projeto correspondente da seguinte forma:

Dada uma ra = <A, T, F, α , cp, cr, tp, tr> e a Rede de Projeto correspondente rp = <A', T', F'>.

Para uma atividade básica a ∈ A convertida para a_e, a_f ∈ A' considere:

$c(a_e) = 0$ e $c(a_f) = cr(a)$ se $\alpha(a)$=finalizada, e $c(a_f) = cp(a)$ caso contrário;

$t(a_e) = 0$ e $t(a_f) = tr(a)$ se $\alpha(a)$= finalizada, e $t(a_f) = tp(a)$ caso contrário;

Para a atividade composta a ∈ A convertida para a_e, a_{sr}, a_f ∈ A', remova a_{sr} da rede e considere $c(a_e)=c(a_f)=t(a_e)=t(a_f)=0$.

ALGORITIMO 1 – Caminho de projeto **mais rápido**:

Considere a seqüência de blocos elementares b_1, b_2, .., b_k que representa um dobramento completo de *rp*. Usando as funcões c() e t(), o custo e o tempo de *rp* são obtidos por:

c(rp) =0; t(rp)=0;

for i = 1,..k

case

bi é um bloco seqüencial $[t_1,a_1,t_2,a_2,t_3]$:$c(rp) =+ c(a_1)+c(a_2)$; $t(rp) =+ t(a_1)+t(a_2)$;

bi é um bloco concorrente $[t_1,a_1||a_2,t_3]$:$c(rp) =+ c(a_1)+c(a_2)$; $t(rp) =+ \sup\{t(a_1), t(a_2)\}$;

bi é um bloco alternativo $[t_1,a_1<>a_2,t_3]$:

se $t(a_1)<t(a_2)$: $c(rp) =+ c(a_1)$; $t(rp) = + t(a_1)$;

se $t(a_1)>t(a_2)$: $c(rp) =+ c(a_2)$; $t(rp) =+ t(a_2)$;

se $t(a_1)=t(a_2)$: $c(rp) =+ \inf(c(a_1),c(a_2))$; $t(rp) =+ t(\inf(c(a_1),c(a_2))$;

ALGORITIMO 2 – Caminho **menos dispendioso**

c(rp) =0; t(rp)=0;

for i = 1,..k

case

bi é um bloco seqüencial $[t_1,a_1,t_2,a_2,t_3]$:$c(rp) =+ c(a_1)+c(a_2)$; $t(rp) =+ t(a_1)+t(a_2)$;

bi é um bloco concorrente $[t_1,a_1||a_2,t_3]$:$c(rp) =+ c(a_1)+c(a_2)$; $t(rp) =+ \sup\{t(a_1), t(a_2)\}$;

bi é um bloco alternativo [c_1,a_1<>a_2,c_3]:

se $c(a_1)<c(a_2)$: $c(rp) =+ c(a_1)$; $t(rp) = + t(a_1)$;

se $c(a_1)>c(a_2)$: $c(rp) =+ c(a_2)$; $t(rp) =+ t(a_2)$;

se $c(a_1)=c(a_2)$: $c(rp) =+ c(\inf(t(a_1),t(a_2))$; $t(rp) =+ \inf(t(a_1),t(a_2))$;

ALGORITIMO 3 – Caminho **customizado**

c(rp) =0; t(rp)=0;

for i = 1,..k

case

bi é um bloco seqüencial [t_1,a_1,t_2,a_2,t_3]:$c(rp) =+ c(a_1)+c(a_2)$; $t(rp) =+ t(a_1)+t(a_2)$;

bi é um bloco concorrente [t_1,a_1||a_2,t_3]:$c(rp) =+ c(a_1)+c(a_2)$; t(rp=+ sup{$t(a_1)$, $t(a_2)$};

bi é um bloco alternativo [c_1,a_1<>a_2,c3]:

if [$t(a_1)\leq t(a_2)$&$c(a_1) \leq c(a_2)$]: $c(rp) =+ c(a_1)$; $t(rp) = + t(a_1)$;
else if [$t(a_1)>t(a_2)$&$c(a_1)>c(a_2)$]: $c(rp) =+ c(a_2)$; $t(rp) =+ t(a_2)$;
else get(preference);
case preferência:

"menor custo':

if $c(a_1)<c(a_2)$: $c(rp) =+ c(a_1)$; $t(rp) = + t(a_1)$;
if $c(a_1)>c(a_2)$: $c(rp) =+ c(a_2)$; $t(rp) =+ t(a_2)$;
se $c(a_1)=c(a_2)$: $c(rp) =+ c(\inf(t(a_1),t(a_2))$; $t(rp) =+ \inf(t(a_1),t(a_2))$;

"menor prazo":

if $t(a_1)<t(a_2)$: $c(rp) =+ c(a_1)$; $t(rp) = + t(a_1)$;
if $t(a_1)>t(a_2)$: $c(rp) =+ c(a_2)$; $t(rp) =+ t(a_2)$;
se $t(a_1)=t(a_2)$: $c(rp) =+ \inf(c(a_1),c(a_2))$; $t(rp) =+ t(\inf(c(a_1),c(a_2))$;

Ao se executar um projeto, pode-se estar interessado em obter os tempos e custos restantes do projeto. A maneira mais fácil de obter isto é considerar todos os custos e tempos =0 em que $\alpha(a)$=finalizada. Os algoritmos 1, 2 e 3 devem ser executados sem contar com estas atividades.

4.5. Visualização tabelar das Redes de Atividades

A forma diagramática das Redes de Atividades, apresentada na seção anterior, é adequada para a criação de uma rede nova e a partir de uma metodologia determinar os atributos de cada atividade. Para ter uma visão sintética dos custos e prazos do projeto, a visualização gráfica não é adequada. Por isso foi definida uma visão tabelar que permite mostrar todas as atividades com seus atributos, além de mostrar sumarizações por bloco ou para o projeto todo, na forma de uma planilha. Nesta planilha, o gerente poderá efetuar, de forma simples e rápida, simulações de alterações de valores de custos e prazos, para averiguar seus reflexos no projeto como um todo.

Ilustramos a planilha (Tabela 4.1) por um pequeno exemplo descrito a seguir.

Queremos desenvolver um programa indexador de páginas HTML, denominado *PageIndexer*. Há duas possibilidades, podemos comprar um produto pronto ou desenvolver um próprio. Estas opções são mostradas na Figura 4.12.

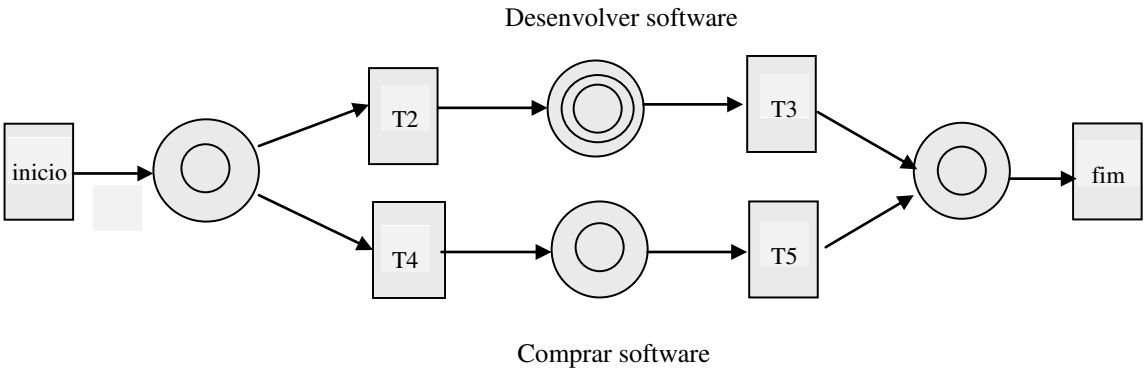


Figura 4.12: Rede de Atividades do projeto

Para decidir qual a melhor opção, temos que completar a RA. A atividade composta "Desenvolver software" é refinada para um *loop* de duas iterações (Figura 4.13). A marca 1 no evento significa que o *loop* será repetido uma vez.

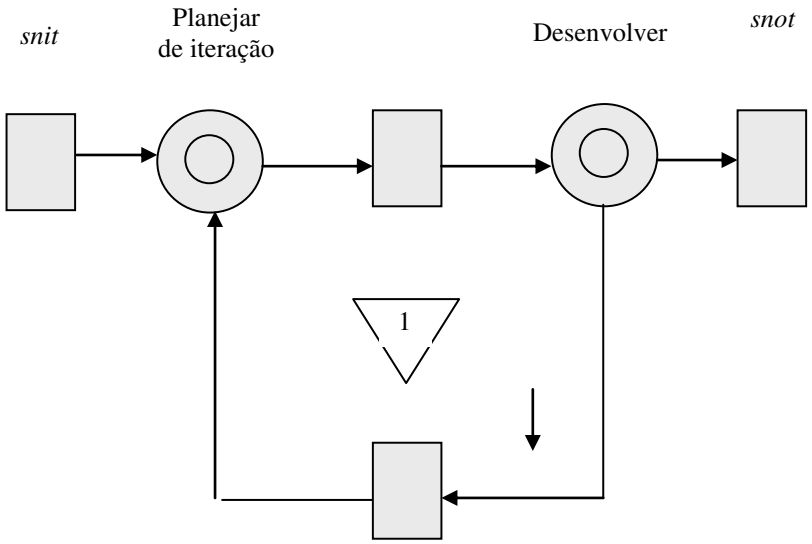


Figura 4.13: Subrede de "Desenvolver software" composta por duas iterações

Agora o *loop* é linearizado, já que cada iteração pode ter custos e prazos distintos. A primeira iteração realiza a remoção de TAGs HTML de um documento e a seguinte indexa o texto restante (Figura 4.14).

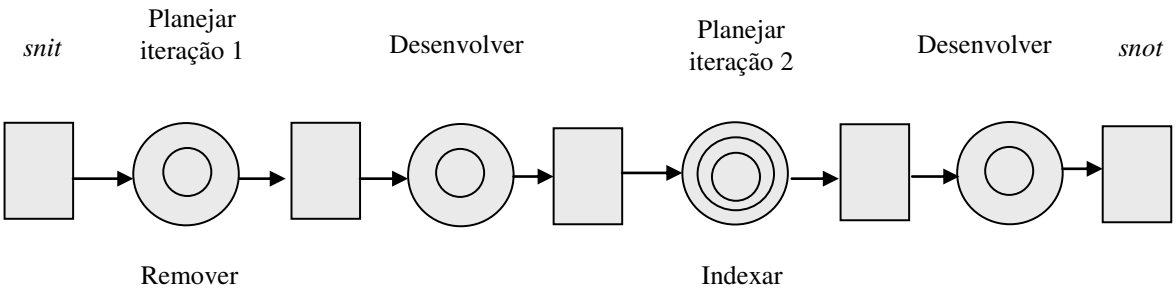


Figura 4.14: Iterações linearizadas

A última atividade de indexação do texto é decomposta uma seqüência de uma atividades chamada "normalizar texto", seguida por duas atividades paralelas "encontrar substantivos" e "encontrar abreviações" (Figura 4.15). Agora todas as atividades podem ser valorizadas.

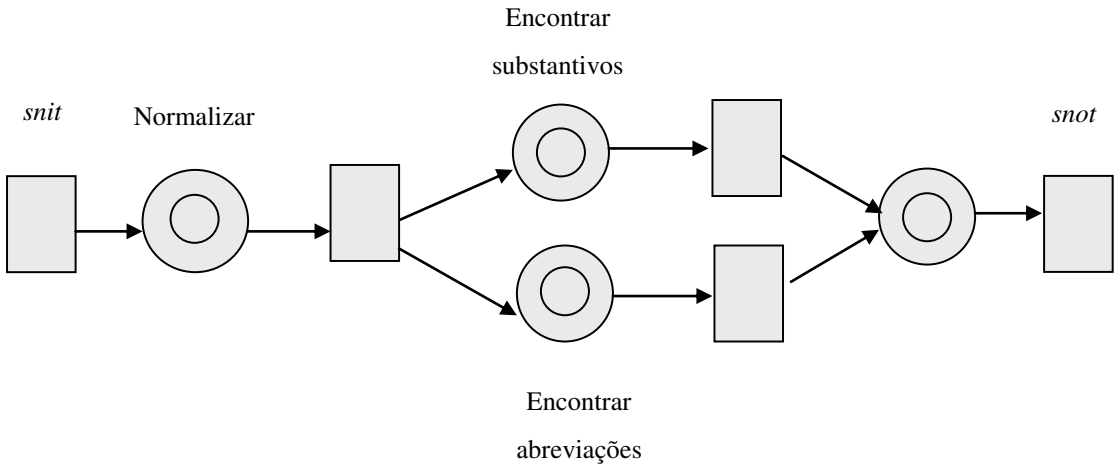


Figura 4.15: Decomposição da atividade "Indexar texto".

No editor de Rede de Atividades (descrito no próximo capítulo), para cada atividade uma tela de formulário pode ser editada para se adicionar detalhes sobre cada atividade. Mostramos na Figura 4.16 os formulários para "normalizar texto", "encontrar substantivos" e "encontrar abreviações".

Atividade: ***normalizar texto*** **Tempo estimado:** 10hs **Custo estimado:** 300 **Tempo real:** 14 **Custo real:** 500 **Recursos humanos:** rec1, rec2.	**Atividade:** ***encontrar abreviações*** **Tempo estimado:** 6hs **Custo estimado:** 150 **Tempo real:** - **Custo real:** - **Recursos humanos:** rec1.	**Atividade:** ***encontrar substantivos*** **Tempo estimado:** 8hs **Custo estimado:** 200 **Tempo real:** - **Custo real:** - **Recursos humanos:** rec2.

Figura 4.16: Detalhes das três atividades da Figura 4.15

Agora, pode-se analisar o projeto: a Tabela 4.1. mostra um conjunto de valores estimados atribuídos a cada uma das atividades. A notação '**V**' significa sub-redes alternativas e '**&**' significa sub-redes concorrentes.

Tabela 4.1: Tabela de custos e tempos estimados e totais

					cp	tp	cr	tr
V	**A1-Comprar**				1200	1		
	A2-Desen-volver	**PIan: Remover TAGS**			50	2	30	4
		Des-Remover TAGS			100	4	250	10
		PIan: Indexação			100	4	150	4
		Des-Indexação	**Normalizar**		300	10	500	12
			&	**Encontrar substan-tivos**	200	8		
				Encontrar abrevia-ções	150	6		
TOTAL A1					1200	1		
TOTAL A2					900	28	930	38
Melhor					(A2)600	(A1)1		

Suponhamos que, após a realização de "Normalizar", os custos e tempos reais são os das colunas 'cr' e 'tr' da tabela 4.1. Agora o custo total de A2 passou de 900 para $1280 (930 realizados + 350 restantes) e o tempo total, 38 horas. Como 'Encontrar substantivos' é a atividade mais longa das restantes, podemos tentar acelerá-la para tentar evitar um atraso do projeto todo.

4.6. Rede de Atividades Expandida

As Redes de Atividades definidas neste trabalho correspondem ao que em Redes de Petri é chamado de Rede Condição/Evento. Em termos de RA, isto significa que uma atividade pode estar desabilitada, em execução ou encerrada. Pelas regras de transição, estas condições podem ser modificadas.

Existem, porém, tipos de projetos para os quais esta estrutura é um pouco simples demais. Uma extensão deverá ser considerada na definição da estrutura e nas regras de

transição.

Assim como atividades distintas podem ser executadas em paralelo, pode-se querer executar uma mesma atividade várias vezes (ao mesmo tempo). Para esses casos, é necessário estender os três possíveis estados de uma atividade (visualizados por uma marca), permitindo várias marcas tanto no *em execução* como no *encerrado*. Ou seja, são necessários lugares que comportam múltiplas marcas. Na figura 4.17, é mostrada a rede de um projeto que prevê o desenvolvimento de seis *user-stories*. Há um artefato denominado `contador` que inicialmente continha seis marcas. Cada acionamento de T1 provoca o consumo de uma marca do artefato. No estado da figura, duas *user-stories* ainda estão por ser desenvolvidas, duas estão em execução, uma está encerrada e a outra já passou para a próxima atividade de `implementar`.

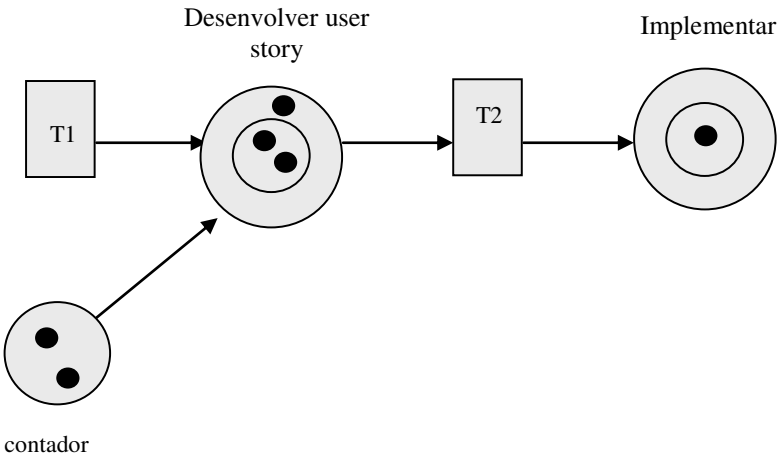


Figura 4.17.: Rede de Atividades Expandida

Denominamos este tipo de rede como **Rede de Atividades Expandida**. Enquanto a RA elementar corresponde às Redes de Petri do tipo Condição/Evento, a RA Expandida corresponde às redes Lugar/Transição. A definição formal das RAs terá que ser adaptada para levar em consideração as múltiplas marcas e as correspondentes regras de transição.

Em projetos complexos, pode ocorrer de uma mesma atividade estar sendo executada várias vezes por membros distintos de uma equipe de desenvolvimento. Esta individualização dos desenvolvedores pode ser importante, pois as funções de custo podem ser distintas de um desenvolvedor para outro e, além disso, pode-se querer fazer um controle de alocação, reduzindo os tempos de inatividade dos membros do projeto. Os membros inativos podem estar custando uma remuneração e, por isso, poderiam ser alocados a uma atividade do tipo “esperando próxima alocação”. Para modelar esta situação, a rede deve permitir possibilitar marcas individualizadas nas atividades. Neste

caso, as marcas pretas são substituídas por identificadores, representados por letras identificando cada pessoa envolvida. Esta mudança caracteriza uma **Rede de Atividades de Alto-Nível:** as atividades recebem marcas individualizadas em seus círculos e as transições movem estas marcas das entradas para as saídas. Marcações no círculo interno representam pessoas ou serviços, executando determinadas atividades e elaborando artefatos (documentos), enquanto as marcações no anel externo representam desenvolvedores que encerraram uma atividade aguardando para serem utilizados nas próximas atividades. A determinação de uma determinada pessoa para iniciar uma atividade pode ser dada por um evento ligado à transição correspondente. Este evento poderá determinar qual é a marca (pessoa) que deve ser escolhida para a próxima atividade.

No próximo exemplo, ao invés de três marcas, existem três identificadores, cada um representando um agente específico que está desenvolvendo a atividade. Na figura 4.18, é apresentada a mesma situação do exemplo anterior da Figura 4.17 com seis agentes individualizados (a,b,c,d,e,f). Para possibilitar a escolha de quem vai ser o próximo desenvolvedor, foi acrescentado um evento que permite determinar o próximo agente a ser considerado. No exemplo, 'a' já está implementando, 'b' terminou o desenvolvimento de sua *user story*, 'c' e 'd' estão desenvolvendo e 'e' e 'f' ainda não começaram, sendo que o próximo deverá ser `e`.

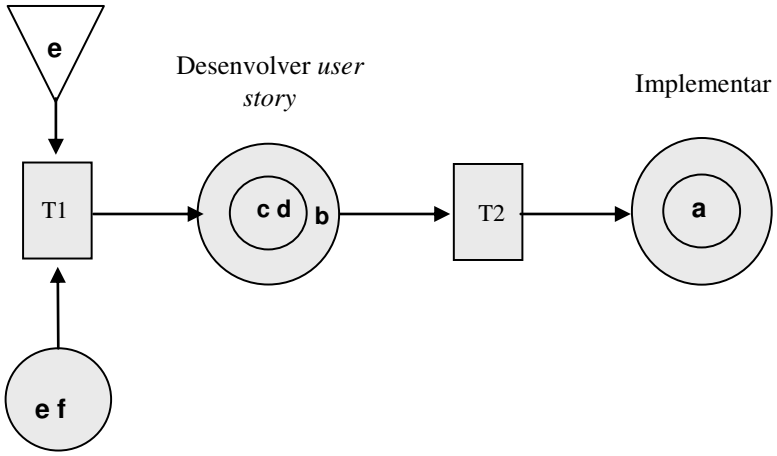


Figura 4.18.: Rede de Atividades de Alto Nível.

Para este tipo de rede, várias características devem ser levadas em consideração. Primeiro, como cada marca representa um desenvolvedor, cada uma pode ter atributos individualizados. Por exemplo, o desenvolvedor 'e' pode ter um salário/hora igual a 30 enquanto 'f' tem um salário/hora de 50. Estes atributos deverão ser levados em conta nos cálculos de custos e tempos.

Outras características também podem ser consideradas. Suponha um atributo que descreve as linguagens de programação que cada um domina. A função lp(x) retorna o conjunto destas linguagens de 'x'. Com isto as transições e os eventos podem ser enriquecidos com regras de transição. Por exemplo, no evento do exemplo acima, pode-se considerar que para escolher entre os desenvolvedores disponíveis para as duas últimas atividades pretende-se colocar a condição de ele dominar a linguagem Java. Neste caso, o evento determinando a escolha de 'e' é substituído pela expressão ('Java' $\in$ lp(x)) em T1 e o 'x' como marca das arestas que entram e saem de T1.

Estas Redes de Atividades de Alto Nível correspondem às Redes de Petri de Alto Nível e deverão ser definidas como tal.

4.7. Rede de Atividades não Estruturada

Outro caso a ser analisado são redes que não são bem estruturadas. A condição da boa estrutura permite um cálculo prático e preciso de custos e prazos, mas, para certos projetos, pode ser restrito demais. Ao contrário de um programa de computador, em que a estrutura só tem que garantir que programa termine, em um projeto queremos prever o menor prazo para encerrá-lo, que poder ter atividades paralelas potencialmente interligadas.

Para ilustrar uma situação destas, utilizamos um exemplo de Boaventura Netto e Oswaldo (2006) de construção de uma estante (Tabela 4.2). Boaventura e Oswaldo modelaram este exemplo com grafos PERT, em que os pré-requisitos cruzados são resolvidos por *atividades-fantasmas*, que não executam nada, mas determinam uma interdependência funcional. O exemplo é o seguinte. Deseja-se construir uma estante e, para tal, tem-se a seguinte tabela de atividades a serem realizadas, com os respectivos predecessores de cada uma:

Tabela 4.2: Atividades de fabricação de uma estante

Atividade	Símbolo	Predecessoras	Duração
Corte de madeira para corpo	a	-	7
Idem, para prateleiras	b	-	5
Montagem do corpo	c	a	9
Preparar ajuste das portas	d	c	11
Fabricação das prateleiras	e	b	6
Ajuste das prateleiras	f	c,e	4
Acabamento do corpo	g	d	3
Acabamento das prateleiras	h	f	8
Lustro das prateleiras	i	g,h	6
Lustro do corpo	j	g,h	4
Entrega e montagem	k	I,j	7

A Rede de Atividades correspondente a este exemplo é mostrada na Figura 4.19.

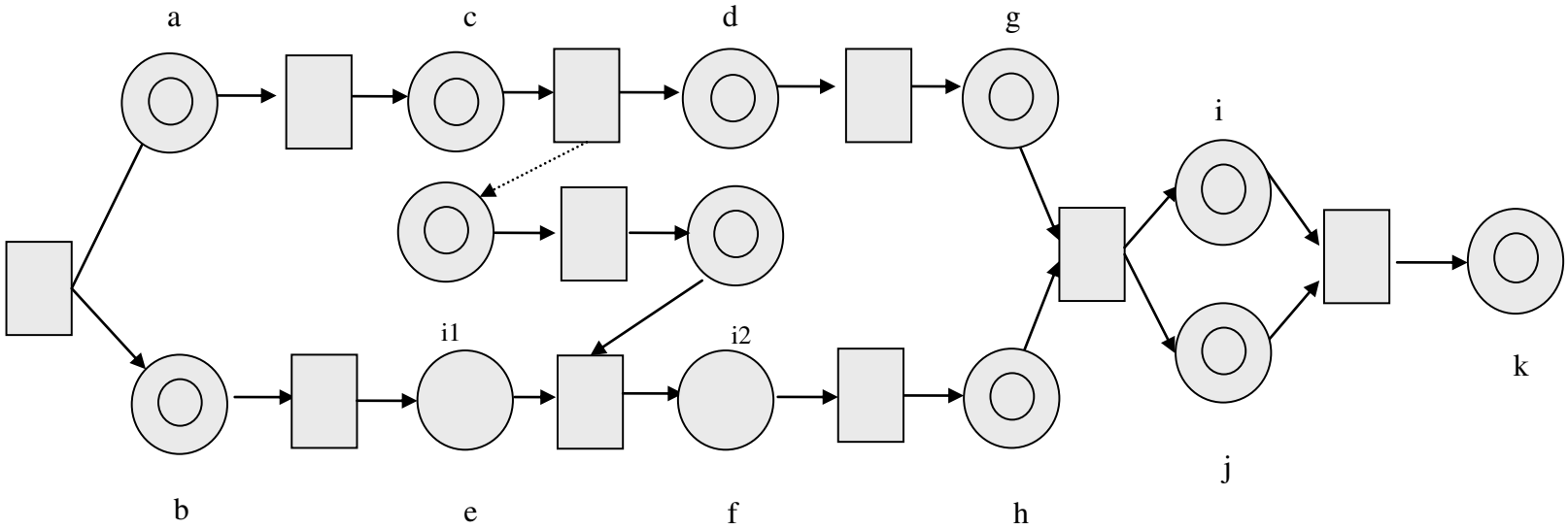


Figura 4.19.: Rede de Atividades de alto nível

Para calcular a duração do projeto, visualizamos a rede de Atividades e a dividimos em blocos de atividades paralelas e seqüenciais.

1) Consideramos o caminho de atividades paralelas: **a,c,f** e **b,e,f** o **total do tempo da parte 1 [a,b,c,e,f] é = 20** que é maior valor ao se comparar

   a + c + f = 20 e

   b + e + f = 15

2) Pegamos o caminho seqüencial a+c+d+g = 20 para chegar a g.
3) O caminho para chegar a h seria [a,c,b,e,f] + h =20+8 = 28
4) Logo, para poder fazer i e j o pior caso é 28.
5) O total será 28 + max(i,j) + k = 28 + 6 + 7 = **41**

O que deve ser distinguido em RPs/RAs genéricas são construções de redes que não fazem sentido em um projeto de construções que podem ocorrer, mas tornam o controle mais complexo. Assim deve-se definir uma Rede de Projeto Bem Formada como uma rede que, apesar de permitir ligações cruzadas como as da figura 4.12, não admite blocos mistos como os da figura 4.1(a), pois não faz sentido, em um projeto, iniciarem-se caminhos concorrentes que se fecham como atividades alternativas**.**

As ligações cruzadas devem ser analisadas com cuidado para ver quais podem ter sentido em um projeto. Basicamente, pode haver ligações `de fora para dentro` de um bloco, `de dentro para fora` de um bloco e `entre` braços de um bloco, como é o caso do exemplo da construção da estante. Por exemplo, uma ligação `de dentro para fora` poderia ocorrer no caso de o resultado de uma atividade provocar um encerramento prematuro do projeto todo, independentemente do estado de outras atividades em execução. A ligação `para fora` iria de dentro de um bloco diretamente ao final do projeto. Isto pode ser visto, por exemplo, na Figura 4.20 onde a atividade ‘e’ tem uma ligação direta para o final do projeto, considerando-se que houve problemas com a atividade ´E´ - ´fabricação das prateleiras´.

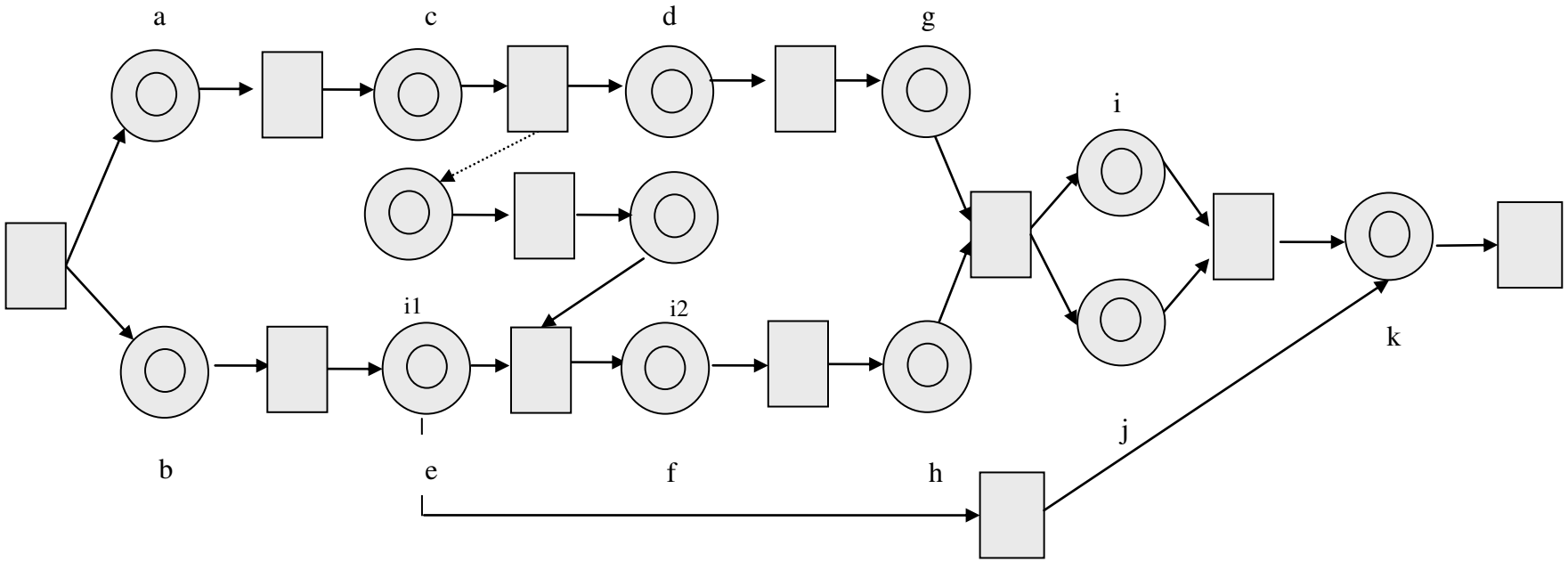


Figura 4.20.: Rede de Atividades de alto nível com encerramento do projeto na atividade ´e´.

4.8. Exemplo de aplicação da Rede de Atividades

Como uma metodologia de desenvolvimento de software é composta essencialmente por um conjunto de atividades que produzem artefatos, ela pode ser modelada por uma RA. Como exemplo, é apresentado aqui, a RA da metodologia XP (XP, 2004) conforme apresentada no Capítulo 2.

Quando se pretende criar a RA de uma metodologia, deve-se:

1. ler a respeito da metodologia
2. entender as suas várias etapas e atividades a serem realizadas
3. identificar e desenhar a atividade de início e se há um evento associado
4. identificar quais são os artefatos gerados na atividade e inserí-los na rede
5. verificar quais as próximas atividades e quais são as atividades que utilizarão os artefatos gerados
6. verificar quais são as dependências existentes entre as diversas atividades realizadas em cada etapa
7. procurar os eventos que poderão vir a acontecer e ligá-los às respectivas transições da rede
8. procurar atividades realizadas em loops e desenhar os arcos com rótulos de testes (se o teste estiver ok segue-se por um caminho) (se não, deve-se percorrer outro caminho na rede)
9. procurar atividades compostas, e se encontrá-las, dividir em sub-redes, respeitando os caminhos de ida e volta, e os passos 3 à 8
10. continuar realizando os passos 4 à 9 e
11. identificar e desenhar a atividade final.

Nas figuras 4.21, a 4.26 seguimos os passos descritos acima para modelar a Rede de Atividades das etapas de XP.

Na Figura 4.21 mostramos as fases da metodologia XP. São descritas as *user stories* e a arquitetura é elaborada. A reunião de planejamento é a próxima atividade a ser realizada e como ela possui outras atividades, ela é representada na figura como uma atividade composta e a sub-rede encontra-se na Figura 4.22. O mesmo acontece com a atividade planejar iteração que possui outras atividades que são mostradas na Figura 4.23. Continuando na Figura 4.24, ainda encontram-se as atividades *realizar teste de aceitação, checar iteração* e *preparar última versão* do software para entregar ao usuário.

A Figura 4.25 mostra a Rede da Atividades da atividade composta *realizar reunião de planejamento*. Na rede encontram-se as atividades: *realizar priorização de*

user stories, definir estimativas de tempo e *analisar estimativas.* Se as estimativas estiverem incertas é necessário *ajustar o tempo* e analisar novamente a estimativa. Se as estimativas estiverem corretas, a próxima atividade é *verificar se acabou* e voltar para a rede de atividades principal. Notamos que se não acabou ainda a estimativa de tempos, retorna-se à atividade *definir estimativas de tempos.*

Na Figura 4.23 está a sub-rede da atividade composta *planejar iteração* com as atividades: *planejar iteração* e *desenvolver* que também são atividades compostas. As atividades delas encontram-se nas sub-redes das Figuras 4.24 e 4.25. A atividade *planejar iteração* produz o artefato plano de iteração que será usado na atividade *desenvolver*. A atividade *desenvolver* armazena os *bugs* que são corrigidos após o desenvolvimento.

Na Figura 4.26 encontram-se as atividades*: traduzir user stories em testes de aceitação.* Esta atividade gera os testes de aceitação e os armazena no repositório que possui este mesmo nome. A atividade *checar teste* realiza os testes para ver se as funções do software estão de acordo com os requisitos dos usuários. Se não, retorna-se ao início do *loop*.

A Figura 4.25 apresenta a atividade *realizar reunião matinal* que utiliza os dados do plano de iteração e a cada nova tarefa ou teste de aceitação que falhou realiza-se a próxima atividade composta *desenvolver código coletivo*.

A Figura 4.26 mostra as atividades de *desenvolvimento* do *software*. São criados e executados os testes de unidade, se eles estiverem ok, a programação é feita, seguindo-se práticas de XP tais como *refatoramento* e *mantendo-se a equipe por perto*, se não, deve-se checá-los novamente. Em seguida, realiza-se a integração, refaz-se os testes de aceitação falhos e entrega-se a última versão ao usuário.

Maiores detalhes sobre as diversas fases e atividades de XP podem ser vistas em (XP, 2004; Farias 2004) e no Capítulo 2 desta tese.

Para modelar um projeto de acordo com a Rede de Atividades de XP deve-se inserir as atividades reais a cada lugar modelado na rede, conforme será visto no Capítulo 5.

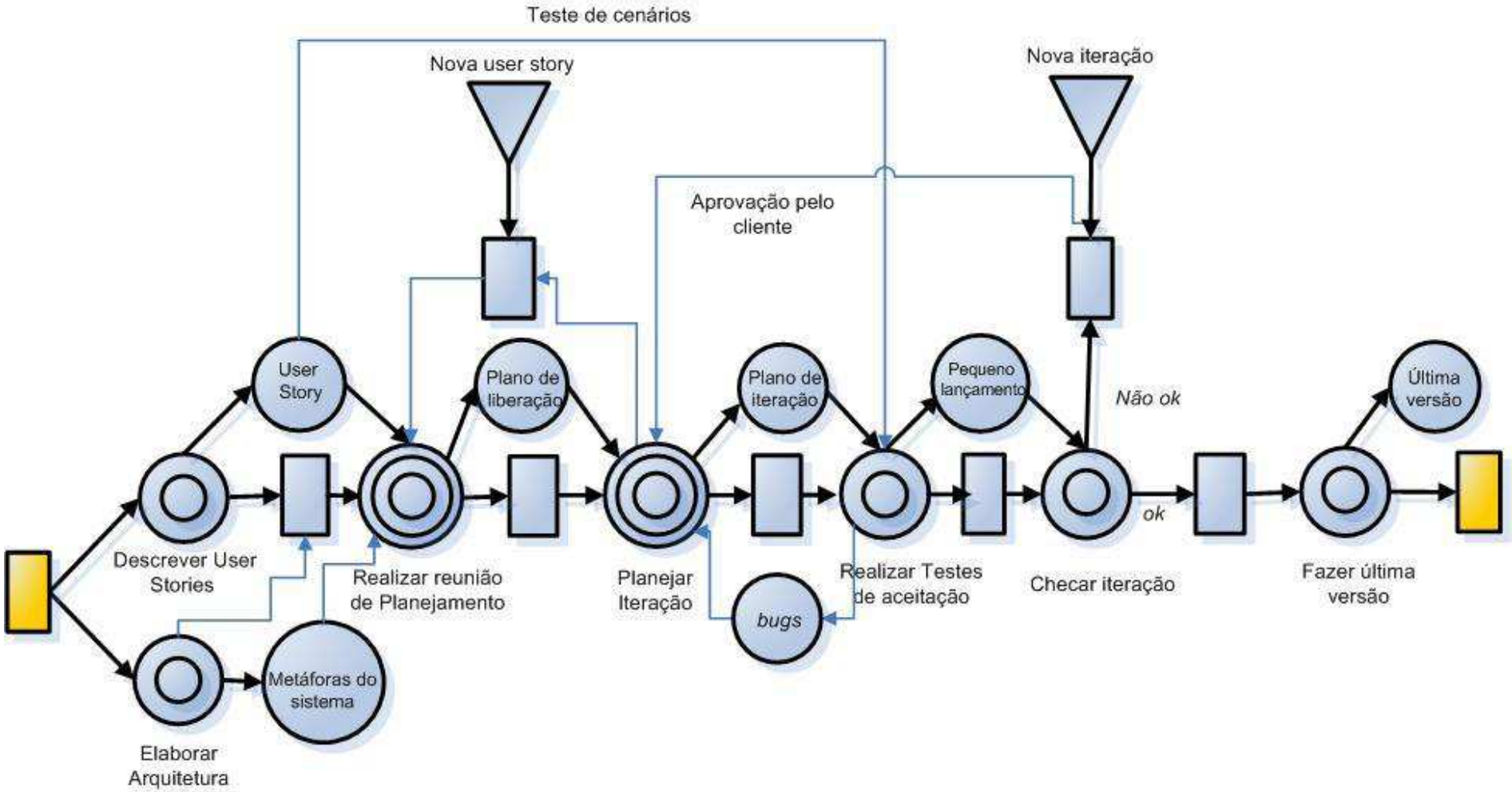


Figura 4.21: Rede de Atividades de XP

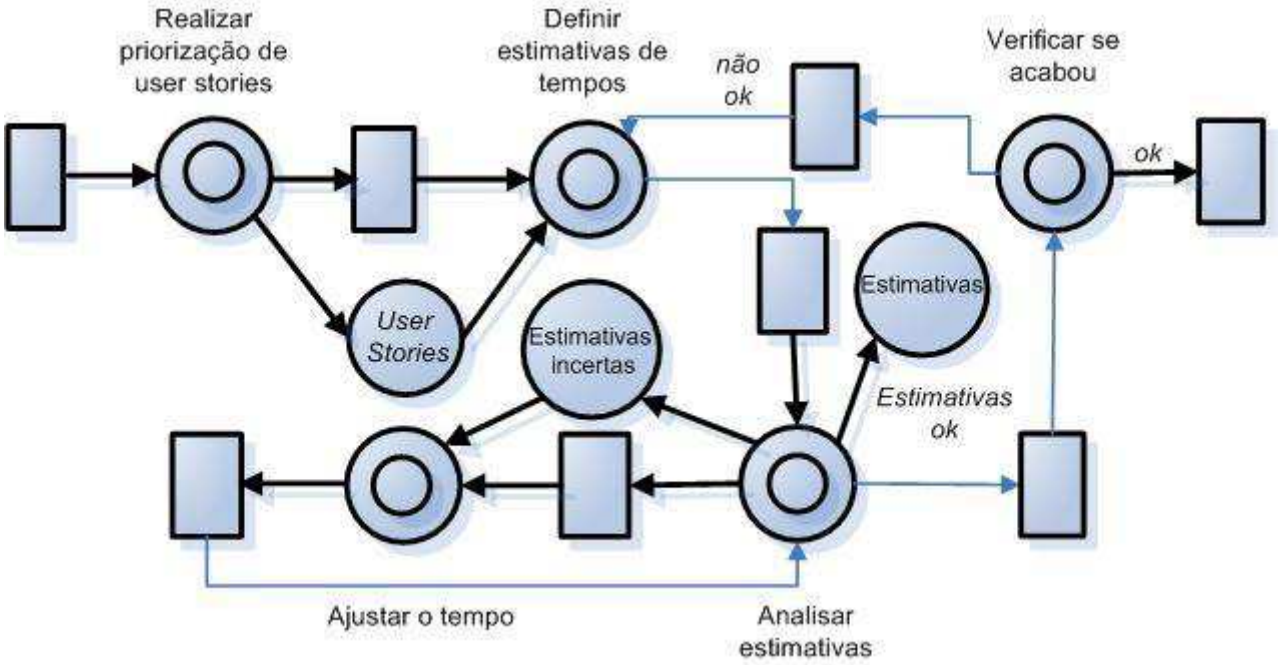


Figura 4.22: Sub-rede da atividade composta "Realizar reunião de planejamento" da figura 4.21.

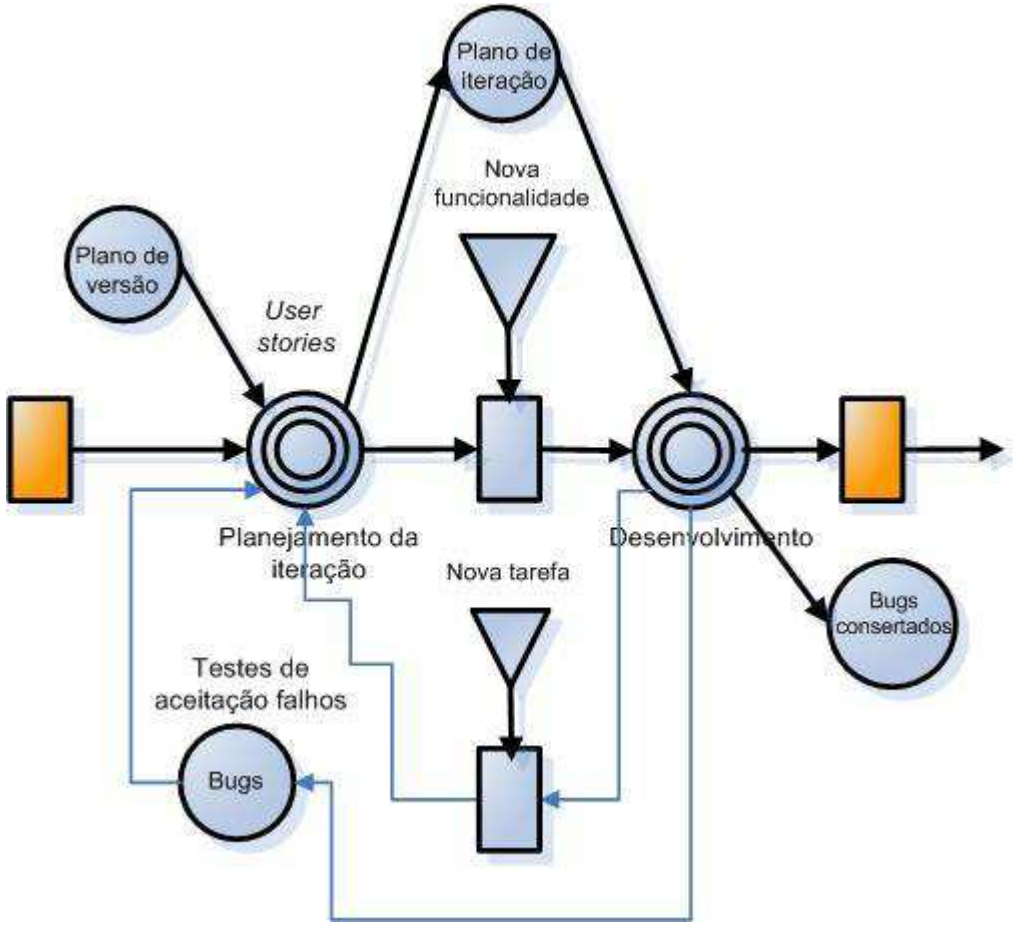


Figura 4.23: Sub-Rede da atividade composta “Planejar iteração” da Figura 4.21.

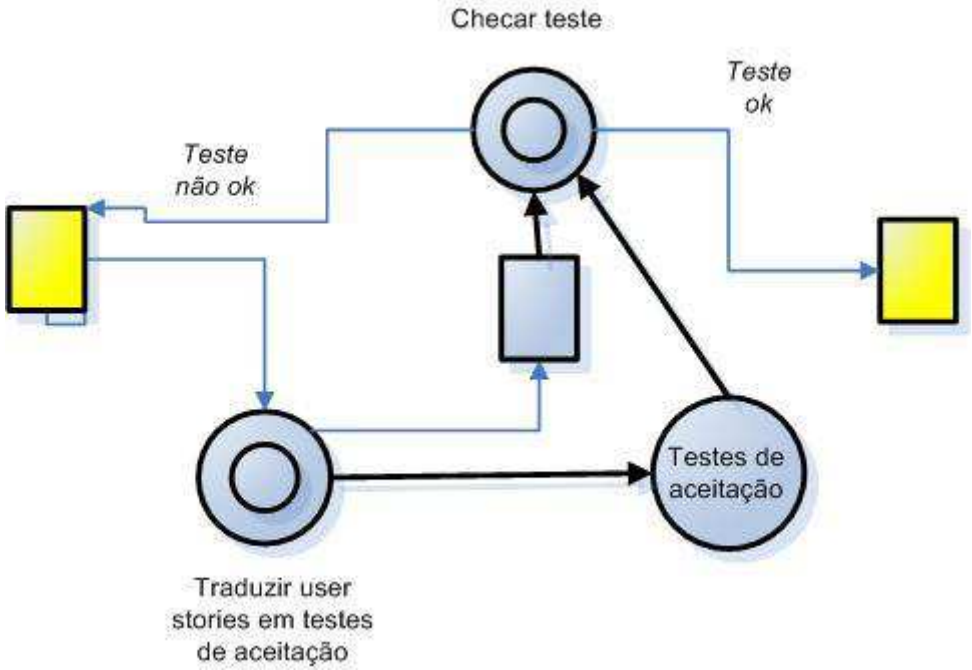


Figura 4.24: Sub-Rede da atividade "Planejar iteração" da Figura 4.23.

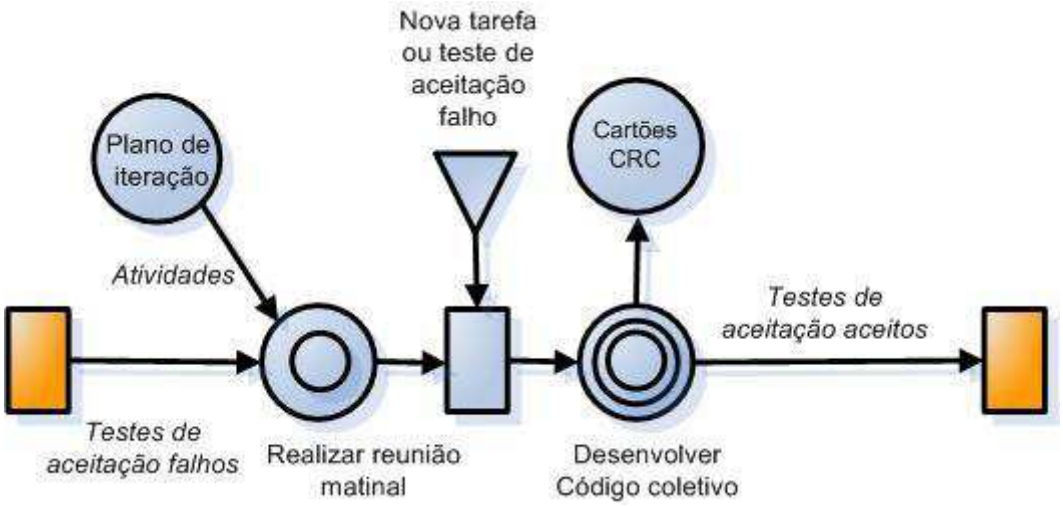


Figura 4.25: Sub-Rede da atividade "Desenvolver" da Figura 4.23.

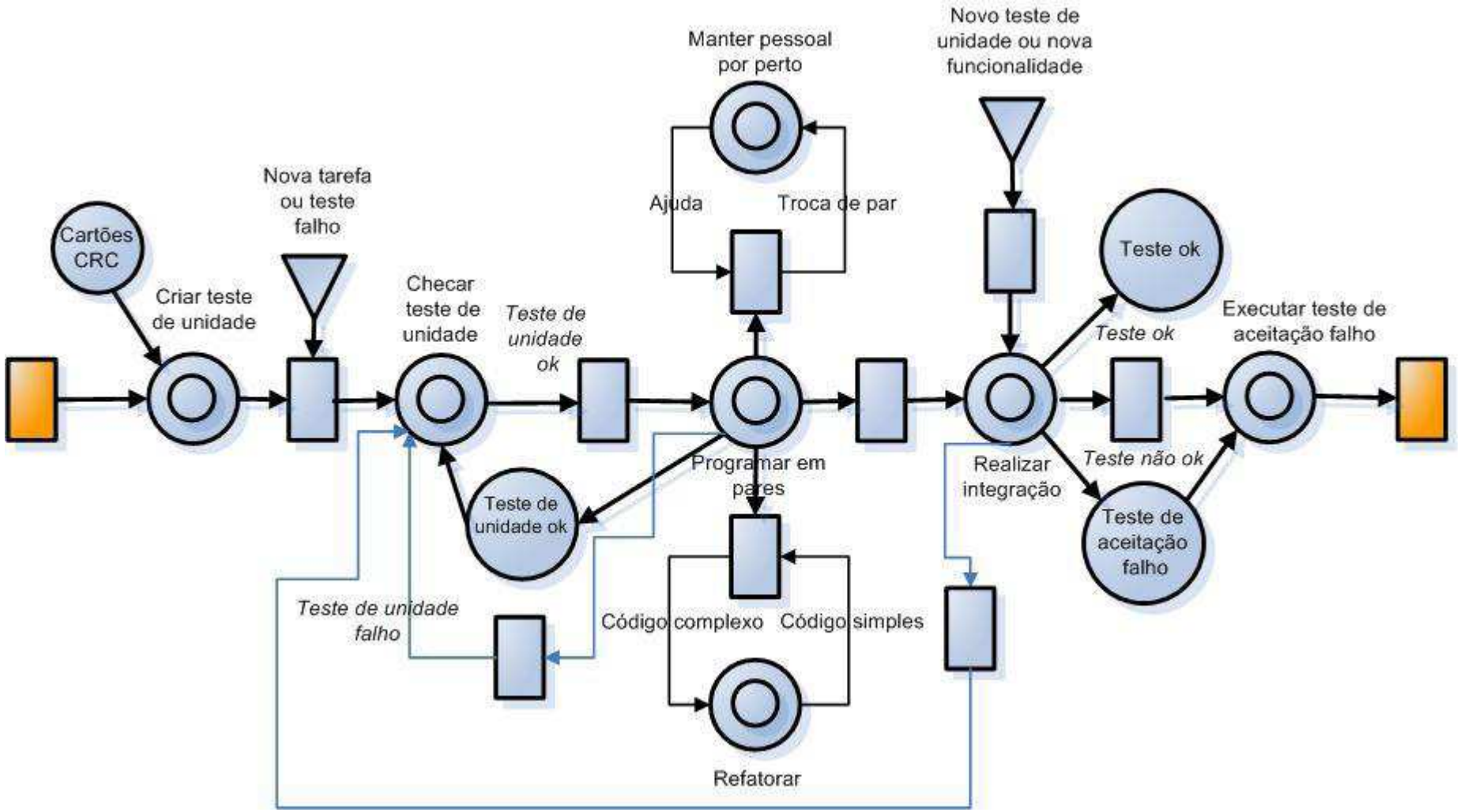


Figura 4.26: Sub-Rede da atividade "Desenvolver código coletivo" da Figura 4.25.

Ao realizar a modelagem da rede de Atividades de XP foram encontradas algumas dificuldades no início, devido à interpretação das atividades de cada etapa de XP descrita na bibliografia pesquisada. Após algumas leituras foram compreendidas as etapas de XP e o que era necessário fazer em cada uma delas. Em seguida foi preciso verificar quais os artefatos gerados em cada atividade, ver as relações existentes entre elas e as dependências interfuncionais. Mas no geral, modelar as atividades na Rede de Atividades não foi uma tarefa difícil. O crucial para modelar uma Rede de Atividades é conhecer, entender as etapas da metodologia a ser modelada, saber os tipos de elementos da rede e aplicá-los.

4.9. Comentários

Neste Capítulo, foram apresentados os conceitos de Rede de Projeto e Rede de Atividades e exemplos de uso com a modelagem da metodologia XP.

Torres (1996) definiu inicialmente a Rede de Atividades. Foram definidos elementos de: lugar, transição, evento, atividade simples e composta e foram definidas regras de ligação entre os elementos. Ele definiu a atividade básica e descreveu um algoritmo para conversão de rede de atividade para rede de Petri do tipo Condição-evento.

Neste trabalho de tese, foram criados novos elementos para as **Redes de Atividades**, tais como, artefatos e, rótulos para eventos, criamos as definições formais para os novos elementos relacionados às redes de atividades elementares e novas marcas para rede expandida. Além da visualização gráfica de uma RA foi criada uma forma de mostrar a rede através de *interface de planilha,* facilitando o acompanhamento da evolução dos custos e prazos de um projeto. Criamos os conceitos de **Rede de Projeto**, descrevemos como devem ser as *Redes de Projeto Bem Estruturadas*, analisamos a situação de conexões transversais em braços de execução paralela de atividades como um suporte para Redes de Atividades. Descrevemos como podem ocorrer casos de redes com blocos seqüenciais, blocos mistos com ligações *join/fork*, bloco de concorrência, bloco de conflito ou bloco alternativo. Definimos como deve ser a transformação da nova Rede de Atividades em sua corresponde Rede de Projeto para obter os custos e tempos totais a partir do dobramento total da rede. Criamos algoritmos para cálculo dos custos e tempos, fórmulas de totalização de custos (previstos, reais e remanescentes) e de tempos (previstos, reais e remanescentes) de acordo com as principais estruturas de uma rede bem definida: blocos paralelos, blocos alternativos e blocos seqüenciais.

Mostramos a necessidade de se estudar com mais detalhes a Rede de Atividades

Expandida, Redes de Atividades de Alto Nível e exemplo de cálculos com a Rede de Atividades não estruturada.

A aplicação prática da Rede de Atividades foi realizada com a metodologia XP modelada neste capítulo, e com o projeto SAD/da CHESF, e será apresentada no Capítulo 5 - Sistema de Gerência de Projetos e no Capítulo 6 – *Estudo* A*valiatório e Discussão de Resultado*, onde se encontram as etapas da rede criada de acordo com as atividades da metodologia XP 1 - *eXtreme Programming 1*, o planejamento, execução do estudo junto ao projeto SAD - Sistema de Apoio à Decisão da CHESF e resultados

Capítulo 5

GDP – Gerência de Desenvolvimento de Projetos

Neste capítulo é descrito o Sistema de Gerência de Projetos – GDP, uma visão geral sobre o sistema, a arquitetura, módulos do sistema, atores, descrição dos componentes com classes e atributos, requisitos funcionais, requisitos não funcionais e descrição da interface com o usuário. É apresenta também uma aplicação da Rede de Atividades na criação de um modelo para a Metodologia XP1 – eXtreme Programming 1 utilizada no projeto, adaptação e instanciação da rede (com dados reais), totalização e simulação.

5.1. Introdução

O GDP é um ambiente para gerência de desenvolvimento de projetos. Fornece uma maneira fácil e eficiente para determinar o tipo de produto a ser desenvolvido a fim de escolher a metodologia de desenvolvimento adequada, modelar metodologias de desenvolvimento de software e acompanhar projetos. Apesar de o sistema ser de propósito geral, para gerenciar qualquer tipo de projeto, é dado um enfoque maior no desenvolvimento de software.

Para realizar seus propósitos o GDP utiliza duas ferramentas ou módulos: o módulo ERA - Editor de Redes de Atividades, que permite criar, armazenar, instanciar e derivar Redes de Atividades e o SiSeM - Sistema de Seleção de Metodologias de Desenvolvimento de Software (Farias, 2005) que é um sistema de suporte a decisão para facilitar a escolha da metodologia de desenvolvimento de software mais adequada para

um determinado tipo de projeto e ambiente de desenvolvimento.

Os principais usuários do GDP são o gerente de processos, que irá definir qual o melhor processo a usar para um projeto em particular e modelar a Rede de Atividades do processo; o gerente de projeto, que cria a RA do projeto a ser criado e acompanha seu desenvolvimento; e a equipe de desenvolvedores, que fazem a atualização da RA do projeto em desenvolvimento.

O GDP mostra as etapas do processo a serem seguidas através do módulo gráfico da Rede de Atividades para auxiliar o gerente a acompanhar o desenvolvimento, e possibilita definir as atividades e desenvolvedores alocados, atribuir períodos e custos previstos. Com a ajuda do sistema, o gerente ainda pode realizar simulações e obter os valores dos custos previstos, realizados e remanescentes através de totalizações parciais e gerais. Assim como também, obter valores totais de tempos previstos, realizados e remanescentes.

5.2. Arquitetura do GDP

O GDP da suporte à gerência do desenvolvimento de projetos em quatro níveis: *escolha da metodologia, criação da Rede de Atividades da metodologia escolhida, instanciação da RA para o projeto real* e *execução da rede.*.

Desta maneira, a organização poderá definir a sua metodologia de acordo com características do seu ambiente de desenvolvimento, criar processos especializados a partir deste e instanciar os processos para projetos específicos. Estes níveis possibilitam uma padronização entre os processos, além de outras características de qualidade.

Descrevemos, brevemente, cada um destes níveis.

Nível 1 - Escolher metodologia: Por meio de um sistema de apoio à decisão, que utiliza uma árvore de decisão, e realiza em um diálogo com o usuário, são definidas as principais características do projeto a ser realizado a fim de determinar a metodologia mais adequada (Farias, 2005, Anexo II).

Nível 2 - Criar a RA de uma metodologia: Uma metodologia de desenvolvimento de sistemas consiste em uma proposta de uma série de atividades para o desenvolvimento de um projeto. Neste nível, o Gerente de Processo cria uma Rede de Atividades desta sistemática, determinando a interdependência funcional entre as diversas etapas propostas pela metodologia. A RA criada será armazenada na base de metodologias e servir como um modelo (*template)* que pode ser adaptado (customizado) de acordo com os passos já

adotados em uma instituição e para cada projeto concreto a ser desenvolvido segundo esta metodologia.

Nível 3 - Criar RA do projeto: O Gerente de Projetos cria a rede da aplicação a ser desenvolvida. Se o processo utilizar uma metodologia conhecida, que foi armazenada na base de metodologias, ele utilizará a RA correspondente que será instanciada para o projeto em questão. Serão determinadas as atividades concretas do projeto, contendo tempos, recursos e custos estimados.

Nível 4 - Executar projeto: Durante a execução do projeto os desenvolvedores inserem datas e custos de conclusão das atividades reais e o Gerente do Projeto pode fazer estimativas, acompanhar os tempos e custos reais de um projeto desenvolver e executar correções no projeto.

É apresentada na figura 5.1 a arquitetura geral do sistema.

5.2.1 Módulos do GDP

Os dois principais módulos da camada lógica do GDP são: SiSeM e ERA.

a. Módulo SISEM – o Sistema de Suporte à Escolha da Metodologia (Farias, 2005) é um Sistema de Suporte a Decisão (SSD) interativo que auxilia o gerente de processo para, antes do início do projeto, escolher a metodologia mais adequada ao tipo de sistema a ser desenvolvido.

Baseado em uma árvore de decisão o SISEM obtém, por meio de um diálogo, as principais características do sistema a ser desenvolvido. Atualmente ele já contém as metodologias XP, RUP, Catalysis, YP, GAIA, Bastos, AUML, COMPOR-M, POKER, COMMONKADS, MAS-COMMOMKADS, CoMoMAS. Para cada resposta, cada metodologia recebe uma pontuação, determinada de acordo com os estudos realizados em (Farias, 2004) e entrevistas realizadas com equipes de desenvolvimento de software. Além disto, a base de conhecimento contém regras contendo: critérios das metodologias, diálogos com os usuários, pesos atribuídos aos critérios e as metodologias indicadas (mais informações ver Anexo II e (Farias, 2005)). Como resultado o sistema oferece um *ranking* de metodologias adequadas ao projeto.

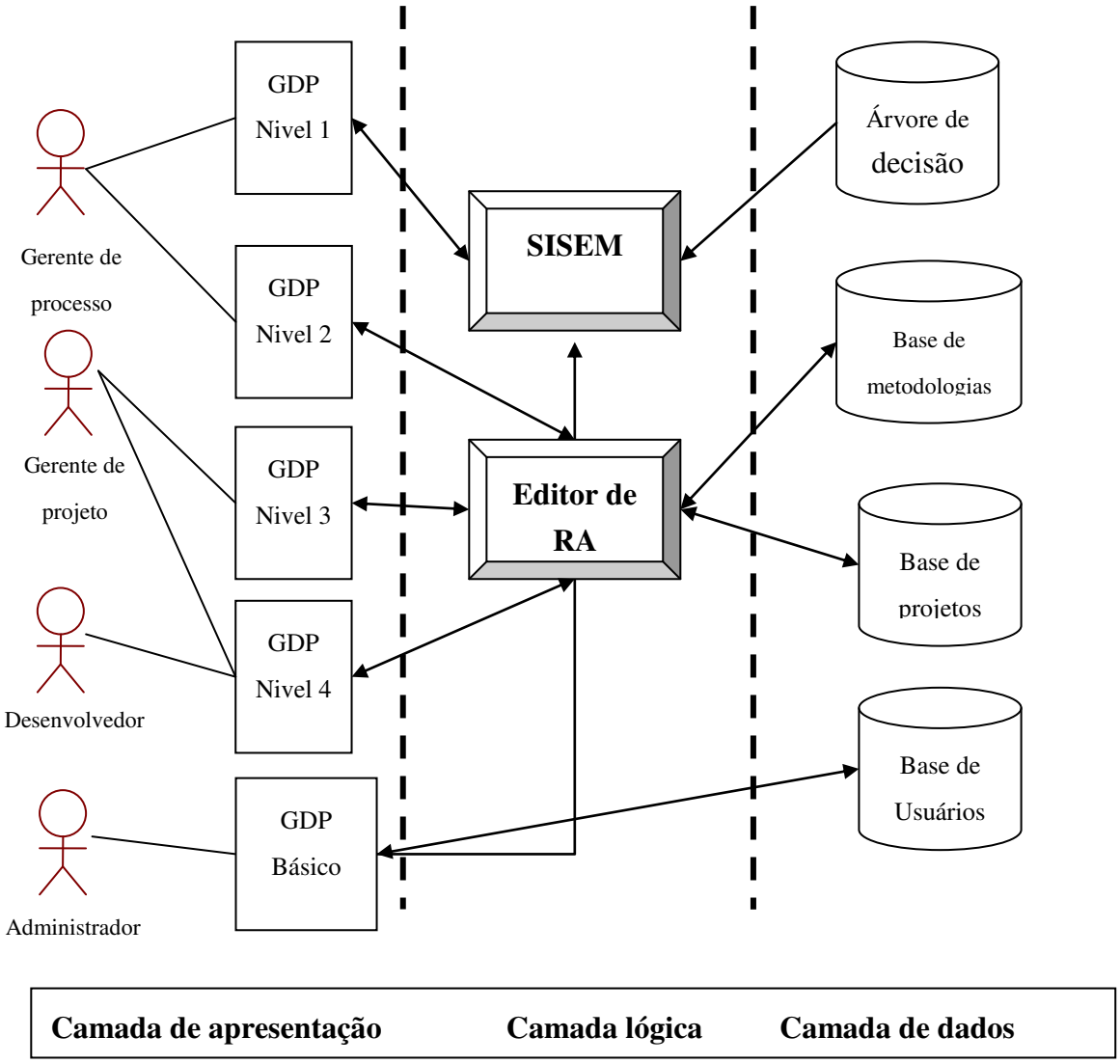


Figura 5.1: Arquitetura do sistema GDP

b. Módulo Editor de Rede de Atividades – ERA - neste módulo a Rede de Atividades pode ser criada, atualizada e executada. O ERA é composto dos seguintes sub-módulos (mais informações ver Anexo III):

- Interface – com este módulo é possível criar uma RA. Ele é um **editor gráfico** e para cada atividade contém: nome da atividade, pré e pós condições de atividades, eventos, sub-redes, artefatos gerados, tempos estimados, custos estimados e desenvolvedores alocados.
- Gerenciamento de recursos - o módulo de gerenciamento de recursos gerencia a alocação de pessoal, equipamentos, e etc. ao projeto.
- Armazenamento e recuperação da rede - este módulo permite o armazenamento e

recuperação de uma RA da Base de Projeto.

- Simulador - com o simulador, em cada ponto do projeto vários valores podem ser calculados, como o custo global estimado, custo real, custo remanescente, tempo global estimado, tempo real e tempo restante.
- Comunicação – este módulo contém uma coleção de eventos (início de uma atividade, artefato pronto,deadline próximo, etc.) de um processo para notificar usuários sobre o andamento do projeto. A comunicação deve ser realizada através de e-mail, anotações em artefatos, fórum e página web.

c. Base de projetos – para cada projeto armazena sua RA **e** contém dados de recursos humanos, recursos operacionais, de artefatos desenvolvidos e que poderão ser utilizados para consultas futuras em novos projetos. O banco de dados pode armazenar todas as informações criadas (planos de projeto, definição de processos, planos de riscos, alocação de recursos, modelos e diagramas) e todos os dados das atividades, dos tempos e custos previstos e reais.

d. Base de metodologias – contém as Redes de Atividades criadas pelo gerente de processos a ser usada pelo gerente de projetos. Atualmente a base contém RAs das metodologias (XP e XP1).

e. Base de usuários – é mantida pelo administrador e contêm dados e níveis de autorização dos diversos usuários do sistema.

5.2.2. Atores do Sistema

A seguir encontra-se a descrição das pessoas que interagem com o GDP e seus papéis fundamentais.

- **Administrador** – Responsável por gerenciar quem possui acesso ao sistema e qual o papel que será desempenhado por cada usuário do sistema ao "logar" no sistema. Outra atribuição será a atualização da árvore de decisão do SISEM. Esta árvore deverá sofrer constantes atualizações conforme novos paradigmas de sistemas vão sendo criados, em ampliações do GDP para outros domínios, ou em aperfeiçoamento dos valores atuais baseado na experiência de seu uso.
- **Gerente de processo** – Deve ser um profundo conhecedor dos processos da engenharia de software ou de outros sistemas e também das características do novo produto a ser criado. Ele determina a metodologia

mais adequada para um novo projeto e modela a Rede de Atividade de uma metodologia.

- **Gerente de projeto** – Responsável por criar a Rede de Atividades do projeto a ser desenvolvido. Ou ele cria uma rede inicial ou adapta a rede de uma metodologia para um projeto. A rede criada é preenchida com dados de atividades, tempos, custos e desenvolvedores. Durante a realização do projeto ele executa a rede e realiza simulações e totalizações.
- **Desenvolvedor** – Visualiza dados disponíveis no sistema, verificando quais atividades a fazer de acordo com o fluxo da Rede de Atividades, inserindo tempos e custos reais de conclusão de atividades.

5. 3. Diagramas de classes

A seguir são apresentados os diagramas de classe dos módulos SiSeM – Sistema para Seleção de Metodologias de Desenvolvimento de Software e ERA – Editor de Rede de Atividade.

5.3.1. SiSeM - Sistema para Seleção de Metodologias de Desenvolvimento de Software

O SiSeM possui um formulário que é uma aplicação web e mostra os diálogos de perguntas e respostas. Ele é composto por uma árvore de decisão que é composta por uma coleção diálogos baseados em critérios. Cada diálogo contém uma pergunta com sua respectiva resposta. Cada resposta contém a informação do peso de cada metodologia.

A Figura 5.2. apresenta o diagrama de classes do SiSeM.

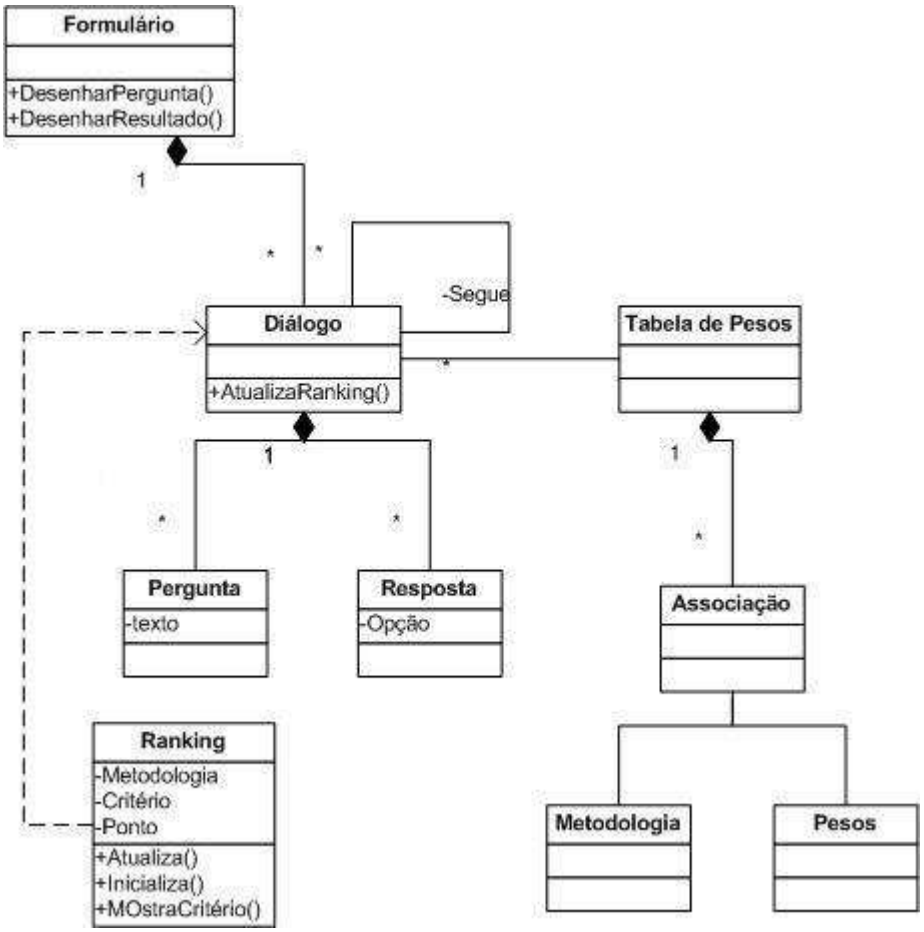


Figura 5.2: Diagrama de classes do SiSeM (Farias, 2005)

5.4. Modelo de classes – ERA – Editor de Rede de Atividades

O ERA possui um conjunto de elementos representativos da Rede de Atividades. A rede possui arestas e elementos representativos de atividades, transições, eventos e depósitos de artefatos. As atividades contêm uma coleção de agendamentos. Cada item da agenda possui a respectiva pessoa associada. As pessoas estão associadas a atividades simples e compostas da rede. As atividades compostas estão associadas a recursos operacionais.

A Figura 5.3 mostra o diagrama de classes do módulo ERA.

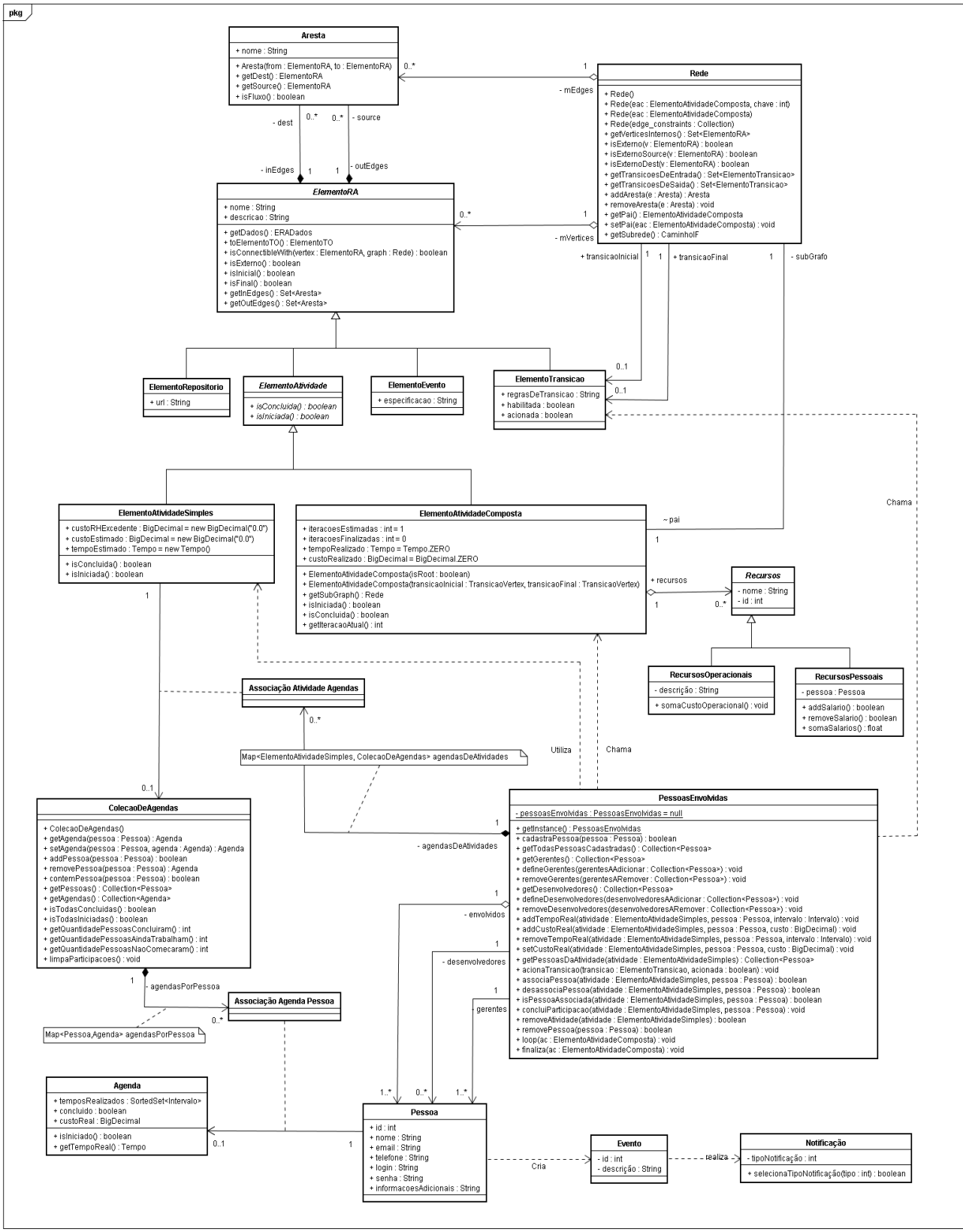


Figura 5.3.: Diagrama de classes – ERA

5.5. Requisitos Funcionais

A seguir é apresentado o diagrama de casos de uso do sistema GDP.

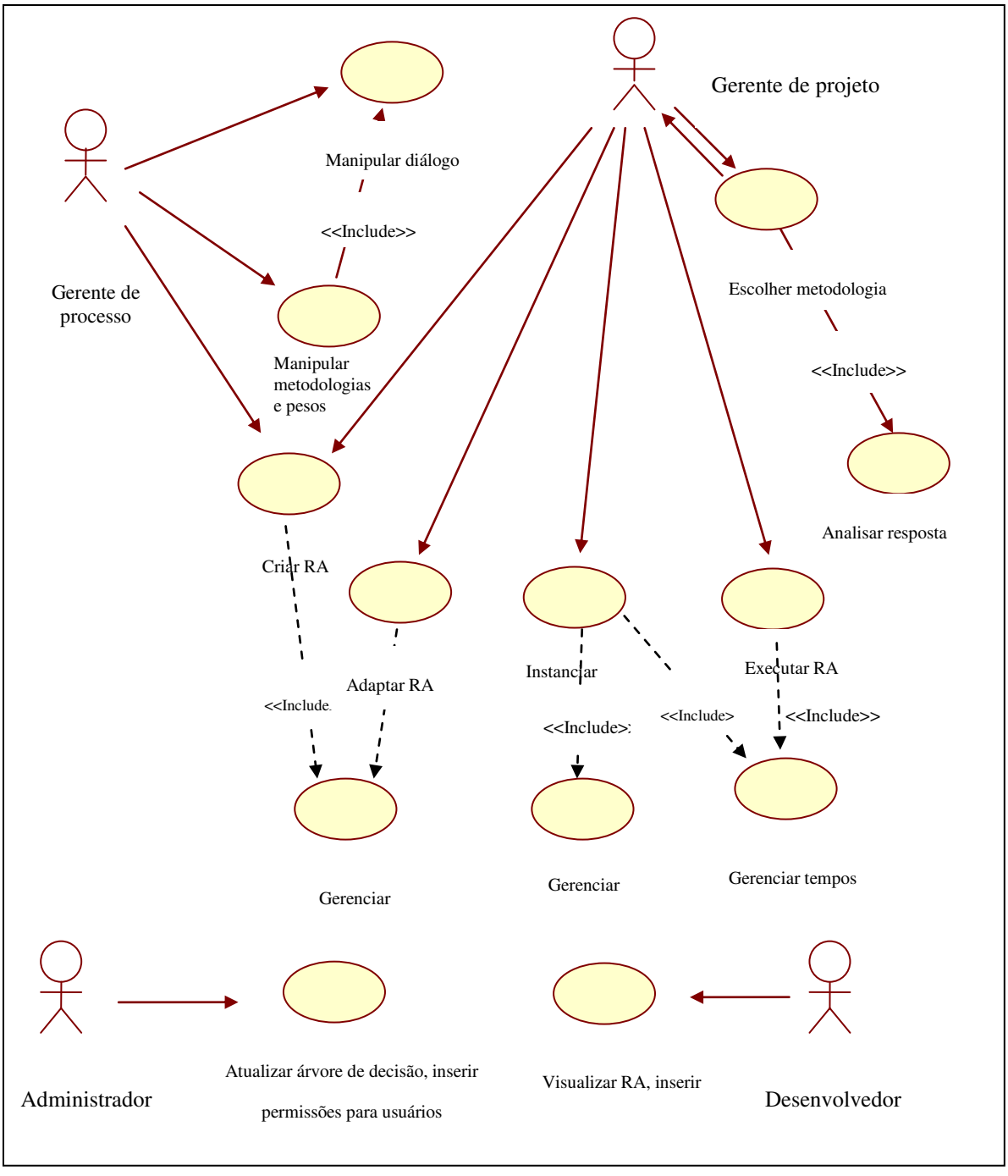


Figura 5.4: Diagrama de casos de uso – GDP

a) Módulo SiSeM

Segue abaixo uma especificação de cada caso de uso - módulo SiSeM.

RF01: Logar - Permite que o usuário tenha acesso ao sistema.

Entradas e pré-condições: nenhuma
Saídas e pós condições: o usuário consegue *logar* no sistema; mensagem de erro
Atores do sistema: Gerente de processo
Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta ao usuário uma tela (formulário) solicitando o *login* e senha do usuário.
2. O usuário informa os dados solicitados e confirma a operação clicando no botão OK.
3. O sistema aceita o *login* e senha do usuário como corretos e lhe apresenta a tela inicial principal do sistema.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, clicando no botão CANCEL.

Login não autorizado

No passo 3, o sistema pode não reconhecer o *login* e senha do usuário como válidos. Nesse caso, o sistema exibe uma mensagem indicando o ocorrido e retorna a tela de login.

RF02: Atualizar árvore de decisão - Este caso de uso permite que o gerente de processo insira características de metodologias e pesos.

Entradas e pré-condições: o usuário deve estar *logado* no sistema
Saídas e pós condições: metodologia cadastrada
Atores do sistema: Administrador
Prioridade: média

Fluxo de eventos principal

1. O sistema apresenta ao usuário um formulário para cadastro de dados do projeto.
2. O gerente preenche os dados.
3. O sistema confirma a ação.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, apagando os valores digitados e mudando de tela.

No passo 1, o sistema verifica que os dados não podem ser compilados, pois existe algum erro na sua estrutura. Uma mensagem indica o que há de errado.

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, clicando no botão cancelar.

No passo 1, o sistema verifica que o componente não pode ser compilado, pois existe algum erro na sua estrutura. Uma mensagem indica o que há de errado.

Login não autorizado

No passo 3, o sistema pode não reconhecer os dados do usuário como válidos. Nesse caso, o sistema exibe uma mensagem indicando o ocorrido.

RF03: Realizar Diálogo - Permite que o gerente de processo insira novas respostas sobre o tipo de aplicação a ser desenvolvida, equipe de desenvolvimento, ambiente de projeto, etc.

Entradas e pré-condições: o usuário deve estar *logado* no sistema
Saídas e pós condições: diálogo encerrado
Atores do sistema: Gerente de processo
Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta uma pergunta
2. O gerente de processo responde a pergunta.
3. O sistema analisa a resposta (Analisar Resposta) e atualiza resultados parciais
4. Se o diálogo não acabou, repete os passos acima,

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, mudando de tela.

No passo 1, o sistema verifica que os dados não podem ser compilados, pois e algum erro na sua estrutura. Uma mensagem indica o que há de errado.

RF04: Determinar Metodologia - Permite que o gerente de projeto leia as perguntas apresentadas na tela, responda-as e obtenha a resposta final (metodologia mais adequada para projeto e aplicação)

Entradas e pré-condições: o usuário deve estar *logado* no sistema

Saídas e pós condições: *ranking* de metodologias adequadas
Atores do sistema: Gerente de processos

Prioridade: importante

Fluxo de eventos principal

1. O sistema chama Realiza Diálogo
2. O sistema verifica o resultado recebido de Realiza Diálogo
3. O sistema mostra uma lista com as metodologias mais adequadas e suas pontuações.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar ou repetir a operação a qualquer momento, clicando no botão cancelar.

RF05: Analisar Resposta - Localiza a resposta obtida na árvore de decisão e retorna a respectiva pontuação e a próxima pergunta.

Entradas e pré-condições: o sistema deve receber como entrada uma respostas do gerente de processos
Saídas e pós condições: pesos da resposta obtidos da árvore de decisão e a próxima pergunta na árvore
Atores do sistema: Realizar Diálogo
Prioridade: importante

Fluxo de eventos principal

1. O sistema recebe a resposta
2. O sistema localiza a resposta na árvore.
3. O sistema obtém o peso para cada metodologia desta resposta e caminha na árvore obtendo a próxima pergunta
4. O sistema retorna os valores obtidos

Fluxo de eventos secundário (alternativos e de exceções)

Final do diálogo

Caso a resposta esteja em uma folha da árvore, no lugar da próxima pergunta é retornado uma mensagem de fim-de-diálogo

Dados incompletos/inconsistentes

Não existe.

b) Módulo ERA

Segue abaixo uma especificação dos principais casos de uso do módulo ERA.

RF01: Logar - Permite que o usuário tenha acesso ao sistema. De acordo com o seu perfil, o usuário pode acessar determinadas funcionalidades do sistema.

Entradas e pré-condições: nenhuma
Saídas e pós condições: o usuário consegue *logar* no sistema
Atores do sistema: todos menos o administrador
Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta ao usuário uma tela (formulário) solicitando *login*, senha do usuário, função e nome do projeto.
2. O usuário informa os dados solicitados e confirma a operação clicando no botão OK.
3. O sistema aceita os dados do usuário como corretos e lhe apresenta a tela inicial principal do sistema.

Fluxo de eventos secundário (alternativos e de exceções)
Cancelamento da operação
O usuário pode cancelar a operação a qualquer momento, clicando no botão CANCEL.
Login não autorizado
No passo 3, o sistema pode não reconhecer *login*, senha do usuário, função e nome do projeto
como válidos. Nesse caso, o sistema exibe uma mensagem indicando o ocorrido e retorna à tela de *login*.

RF02: Cadastrar Projeto - Permite que o gerente de projeto cadastre os dados do projeto

Entradas e pré-condições: o usuário deve estar *logado* no sistema
Saídas e pós condições: projeto cadastrado
Atores do sistema: Gerente de projeto
Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta ao usuário um formulário para cadastro de dados do projeto.
2. O gerente preenche os dados.
3. O sistema confirma a ação.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, clicando no botão cancelar.

No passo 1, o sistema verifica que o componente não pode ser compilado, pois existe algum erro na sua estrutura. Uma mensagem indica o que há de errado.

Login não autorizado

No passo 3, o sistema pode não reconhecer os dados do usuário como válidos. Nesse caso, o sistema exibe uma mensagem indicando o ocorrido.

RF03: Cadastrar Usuários - Permite que o DBA crie permissões para outros usuários acessarem o sistema.

Entradas e pré-condições: o usuário deve estar *logado* no sistema

Saídas e pós condições: usuários cadastrados

Atores do sistema: Administrador

Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta ao usuário um formulário para cadastro de usuários.
2. O DBA preenche os dados.
3. O confirma a ação.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, clicando no botão cancelar.

Login não autorizado

No passo 3, o sistema pode não reconhecer o *username* e senha do usuário como válidos. Nesse caso, o sistema exibe uma mensagem indicando o ocorrido e retorna a tela de *login*.

Dados incompletos/inconsistentes

No passo 1, o sistema verifica que o dado não pode ser inserido, pois existe algum erro na sua digitação. Uma mensagem indica o que há de errado.

RNF04: Modelar Rede de Atividades: Permite que o usuário criar e atualizar uma Rede de Atividades.

Entradas e pré-condições: o usuário deve estar *logado* no sistema
Saídas e pós condições: RA criada
Atores do sistema: gerente de projeto, gerente de processo
Prioridade: essencial

Fluxo de eventos principal

1. O sistema apresenta ao usuário uma área do editor de Redes de Atividades.
2. O usuário escolhe entre criar nova RA ou abrir RA existente
3. O usuário escolhe o ícone desejado.
4. O sistema insere o elemento no local desejado pelo usuário.
5. Caso o usuário não deseja criar mais elementos, salva a rede criada/alterada.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, clicando em qualquer lugar da tela.

Ações incorretas/inconsistentes

No passo 1, o sistema verifica que o elemento não pode ser inserido, ou que as conexões não podem ser realizadas pois existe algum erro de regras pertinentes à RA. Uma mensagem indica o que há de errado.

RF05: Adaptar RA - Permite que o usuário faça adaptações para uma rede previamente modelada a partir de uma metodologia

Entradas e pré-condições: o usuário deve estar *logado* no sistema
Saídas e pós condições: o usuário visualiza os elementos da rede desejada
Atores do sistema: gerente de projeto
Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta os elementos da rede conectados entre si.
2. O gerente escolhe o elemento desejado a ser alterado
3. O gerente altera o elemento
4. O sistema apresenta ao usuário uma área com ícones de menu.

5. O usuário escolhe o ícone desejado.
6. O sistema insere o elemento no local desejado pelo usuário
7. Gerente altera elementos ou exclui
8. Uma mensagem é exibida indicando o sucesso da operação.

Fluxo de eventos secundário (alternativos e de exceções)
Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento não clicando em nenhum elemento.

Ações incorretas/inconsistentes

No passo 1, o sistema verifica que o elemento não pode ser inserido, ou que as conexões não podem ser realizadas pois existe algum erro de regras pertinentes à RA. Uma mensagem indica o que há de errado.

RF06: Instanciar uma atividade - Permite que o usuário insira atributos e os visualize de acordo com um determinado elemento.

Entradas e pré-condições: deve receber como entrada a atividade cujas propriedades se deseja atualizar, os valores de tempo, custo e recursos a serem adicionados e tem como pré-condição o usuário estar *logado* no sistema
Saídas e pós condições: os atributos da atividade estão atualizados
Atores do sistema: gerente de projeto
Prioridade: importante

Fluxo de eventos principal

1. usuário escolhe o elemento da Rede de Atividades e clica nele com o botão direito.
2. usuário insere dados de tempo, custo e recursos.
3. O sistema verifica que o dado recebido como entrada está digitado corretamente.
4. O sistema executa alguns testes pré-definidos que verificam que o dado pode ser inserido no banco de dados.
5. Uma mensagem é exibida indicando o sucesso da operação.

Fluxo de eventos secundário (alternativos e de exceções)

Dado ainda não confirmado

No passo 1, o sistema verifica se o dado ainda não foi digitado corretamente. Nesse caso, o sistema mostra mensagens de erro.

Dados incompletos/inconsistentes

O sistema avisa que o dado não pode ser armazenado, pois está digitado incorretamente ou inconsistente.

RF07: Inserir dados reais na RA - Permite que o usuário altere os dados previstos do sistema para uma atividade realizada, modificando seus atributos

Entradas e pré-condições: deve receber como entrada o elemento cujos atributos serão apresentados e tem como pré-condição o usuário estar *logado* no sistema

Saídas e pós condições: atividades atualizadas com dados reais

Atores do sistema: desenvolvedor

Prioridade: importante

Fluxo de eventos principal

1. O desenvolvedor escolhe a atividade a ser atualizadas.
2. O sistema chama "Instanciar uma atividade" com a condições de somente os dados reais poderem ser atualizados
3. Uma mensagem é exibida indicando o sucesso da operação.

Fluxo de eventos secundário (alternativos e de exceções)

Dado ainda não confirmado

No passo 1, o sistema verifica se o dado foi digitado corretamente. Se não, o sistema mostra mensagens de erro.

Dados incompletos/inconsistentes

O sistema avisa que o dado não pode ser armazenado, pois está digitado incorretamente ou inconsistente.

RF08: Inserir Tempos Reais na RA - Permite que o usuário altere os tempos previstos do sistema para inserir os tempos reais da atividade realizada, modificando seus atributos.

Entradas e pré-condições: deve receber como entrada o elemento cujos atributos serão apresentados e tem como pré-condição o usuário estar *logado* no sistema

Saídas e pós condições: dados de tempo de atividades

Atores do sistema: desenvolvedor

Prioridade: essencial

Fluxo de eventos principal

1. O sistema verifica que o dado recebido como entrada está digitado corretamente.
2. O sistema apresenta uma seqüência de telas em que o usuário deve preencher os dados solicitados.
3. O sistema executa alguns testes pré-definidos que verificam que o dado pode ser inserido no banco de dados.
4. O sistema verifica que foram satisfeitas algumas regras básicas de cadastro.
5. Uma mensagem é exibida indicando o sucesso da operação.

Fluxo de eventos secundário (alternativos e de exceções)

Dado ainda não confirmado

No passo 1, o sistema verifica se o dado foi digitado corretamente. Se não, o sistema mostra mensagens de erro.

Dados incompletos/inconsistentes

O sistema avisa que o dado não pode ser armazenado, pois está digitado incorretamente ou inconsistente.

RF09: Executar RA - Permite que o usuário execute o processo de simulação de um elemento da rede e verifique tempos totais previstos, reais e remanescentes, custos previstos, reais e remanescentes.

Entradas e pré-condições: deve receber como entrada o elemento a ser executado e tem como pré-condição o usuário estar *logado* no sistema

Saídas e pós condições: totais compilados no sistema

Atores do sistema: gerente de projeto

Prioridade: importante

Fluxo de eventos principal

1. O sistema apresenta ao usuário a Rede de Atividades.
2. O usuário escolhe o elemento da rede a partir do qual devem ser feitos os cálculos.
3. O sistema apresenta tela de totais de tempos e custos (estimados e reais) indicando o sucesso da operação.

Fluxo de eventos secundário (alternativos e de exceções)

Cancelamento da operação

O usuário pode cancelar a operação a qualquer momento, clicando em outro elemento da rede.

Ações incorretas/inconsistentes

No passo 1, o sistema verifica que o elemento (atividade) não pode ser totalizado. Os campos aparecem em branco ou uma mensagem indica o que há de errado.

5.6. Requisitos Não Funcionais

RNF01: Robustez

Em casos de situações excepcionais, o sistema deve garantir a integridade dos dados persistentes manipulados pelo sistema.

RNF02: Extensibilidade

O sistema vem sendo desenvolvido de modo a ser extensível para contemplar alterações atuais e futuras que por ventura sejam necessárias do decorrer do uso prático.

RNF 03: Usabilidade

O sistema GDP tem uma interface amigável para usuário principiante e experiente. As interfaces do sistema possuem áreas de menus, e ícones representativos de elementos da Rede de Atividades com representatividade do elemento real.

O sistema é facilmente utilizado após um treinamento. O usuário já deve ter familiaridade com o uso básico de computadores, interface Windows e navegação Web.

Browser: O sistema deve ter interface amigável e ergonômica, e deve necessariamente ser acessível via *web browser.*

Erros: O sistema deve mostrar mensagens explicativas em caso de erro

Ajuda: O sistema deve prover informações de auxílio *help on-line*.

RNF 04: Segurança

É de fundamental importância que o sistema seja seguro para que seus dados não sejam alterados ou excluídos impropriamente ou ainda danificados. Assim, para acessar o sistema GDP o usuário possui login, senha e permissões específicas.

RNF 05: Requisitos de Portabilidade

A interface do SiSeM executa em plataforma MS-Windows com *browser* Internet

Explorer, embora seja altamente desejável que a independência de browser seja alcançada.

O sistema GDP módulo ERA é programado em Java e usa interfaces padronizadas J2SE e/ou J2EE e Frameworks de desenvolvimento Web (Jung). Embora seja desejável que venha a ser acessado no browser. O sistema GDP mantém informações em banco de dados MySQL v. 5.0.

RNF 06: Software

O módulo ERA é implementado em Java, e a utilização da Interface Gráfica é por meio de applets. O acesso aos dados persistentes é feito remotamente através de Sockets implementados em Java e seu servidor faz conexão com MySQL versão 5.0. O applet utiliza a versão 1.6.4 do framework JUNG para produzir e manipular os grafos na tela.

RNF 07: Requisitos de Escala

Quantidade de Usuários: o sistema não deve ter limite intrínseco para a quantidade de usuários cadastrados.

RNF 08: Requisitos de Desempenho

Tempo Resposta LAN: O sistema deve ter tempo de resposta inferior a 5 segundos para apresentar resultados na interface em qualquer interação, estando o cliente e o servidor na mesma rede local.

RNF 09: Requisitos de Custo

Hardware: O sistema deve ser executável em plataforma de hardware de baixo custo, tipo microcomputador, com processador de desempenho médio (tipo Pentium Core 2 Duo, 2 GHz), com memória de 2.0 GB, com interface de rede 10/100 Mbps.

Software Básico: O sistema deve ser executável em plataforma de software tipo Windows, com software de apoio (Tomcat, Apache, etc.) gratuito.

5.7. Descrição da Interface com o Usuário

A seguir é apresentada o protótipo da tela principal do GDP (Figura 5.6.). Essa tela é vista pelos quatro atores (administrador, gerente de processo, gerente de projeto e desenvolvedor), mas cada um deles tem seu nível de permissão. Outras telas do sistema e relato de caso de uso são apresentados na próxima seção.

A estrutura dos menus é apresentada a seguir.

Menu Escolha de metodologias

- **Escolher -** escolha de metodologia
- **Abrir diálogo** - começar um diálogo
- **Diálogo** - manipular diálogo
- **Inserir metodologia -** insere nova metodologia na base de conhecimento
- **Sair** - termina a execução do programa.

Menu Rede

- **Criar** - cria uma nova rede de metodologia
- **Abrir** - abre uma rede de metodologia
- **Fechar** - fecha uma rede
- **Salvar** - grava uma rede
- **Exportar** - exporta uma rede criada com outro formato
- **Imprimir** - imprime rede

Menu Projeto

- Cadastrar - cadastra dados do projeto
- Visualizar - visualiza dados do projeto
- Excluir - exclui projeto
- Alterar - altera dados do projeto
- Adaptar - faz adaptação (customização) de rede criada
- Instanciar - instancia metodologia com dados de um projeto (tempo e custos)
- Simular - permite realizar simulações
- Totalizar - permite calcular totais de tempo e custo

Menu participantes

- **Cadastrar** - cadastra participantes do projeto
- **Editar -** edita dados de participantes do projeto
- **Excluir** - exclui participante
- **Alterar** - altera dados do participantes
- **Definir gerentes** - define gerente(s) do participantes
- **Definir desenvolvedores** - define desenvolvedores(s) do projeto

Menu Recursos

- **Cadastrar** – cadastra recursos operacionais do projeto
- **Editar –** edita recursos do projeto

Menu com ícones de elementos

- **Atividade simples** – desenha elemento de atividade
- **Transição** – elemento de conexão entre atividades e eventos
- **Atividade composta** – elemento para representar atividade composta
- **Evento** - elemento para representar um evento
- **Depósito** – elemento para armazenar artefato
- **Mover** – ícone para mover a tela de desenho
- **Seta** – ícone para selecionar elemento

Relatórios – mostra tela com opções de relatórios e permite impressão.

5.8. Relatórios e Consultas pré-definidos

Com os relatórios, várias informações servirão de auxílio para o gerente, para que mais decisões possam ser tomadas, além das já disponíveis atualmente através nos módulos gráficos e de simulação. Abaixo, encontram-se alguns tipos de relatórios que serão implementados:

- Relatório de atividades por desenvolvedor para cada membro da equipe, apresentando tarefas e sub-tarefas para planejamento individual de pessoas.
- Armazenamento das tarefas do projeto a seguir, as iterações, as histórias do usuário
- Acompanhamento do tempo das tarefas e geração de relatórios individual/equipe
- Acompanhamento dos custos das tarefas e gastos com recursos operacionais
- Relatório de atividades com tempos previstos e reais gastos
- Relatório de atividades com custos previstos e reais gastos
- Gráficos para a velocidade da iteração, distribuição de tipos da tarefa, etc.
- Emissão de relatórios e consultas sobre controle de produtos e processos

- Relação de artefatos produzidos
- Totalização de métricas (velocidade da equipe, horas individuais, em grupo, etc.) em uma semana, mês ou ano.

5.9. Exemplo de utilização do GDP – Projeto SAD da CHESF-Companhia Hidro-Elétrica do São Francisco

O Sistema de Apoio à Decisão da Operação - SAD tem o objetivo de fornecer, em tempo real, acesso fácil e rápido às informações que são relevantes aos fatores críticos de sucesso dos operadores.

O SAD deverá cumprir os seguintes requisitos:

- localizar, extrair e filtrar, de diversas fontes de dados, dados importantes para a atuação dos operadores em situações de contingência;
- integrar dados estruturados, semi-estruturados e não-estruturados;
- apresentar a informação por meio de interfaces amigáveis, que requeiram pouco treinamento para seu uso;
- apresentar a informação de forma concisa e relevante;
- utilizar tecnologias da informação de ponta.

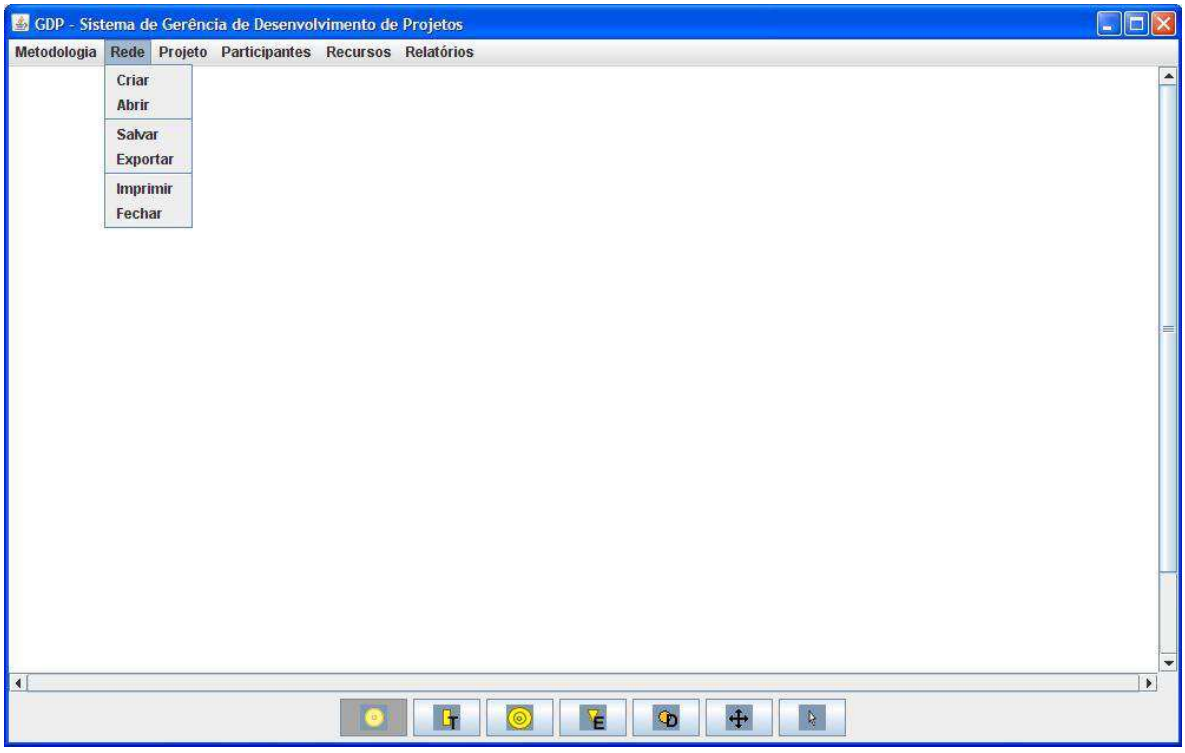


Figura 5.6: Tela principal do sistema GDP

Entre os benefícios esperados, são citados: maior velocidade nas soluções dos estados inseguros em contingências, reduzindo ao mínimo os riscos da operação;

redução do tempo de restabelecimento da rede elétrica; obtenção rápida das informações ajudando na avaliação da causa das ocorrências do sistema; minimização da perda de receita associada à parcela variável por indisponibilidade de equipamentos; e sistema padrão aplicável em todos os Centros de Operação, sub-estações e usinas da CHESF.

5.9.1. Metodologia adotada

No sentido de se determinar uma metodologia para a realização do projeto, os *tutores* do projeto observaram observar que o projeto tem uma carga pesada de desenvolvimento e características próprias. Portanto, uma metodologia adequada de desenvolvimento de software foi escolhida (XP1), lembrando que o projeto possui as seguintes características especiais:

- Os requisitos funcionais do software ainda são vagos;
- Há um forte componente de pesquisa envolvido, principalmente no que diz respeito a:
- técnicas de integração de dados estruturados, semi-estruturados e não-estruturados;
- construção de repositório ODS com um provável novo modelos de dados;
- concepção de uma linguagem de consulta ao ODS segundo o seu modelo de dados;
- e agentes inteligentes do tipo "WRAPPER" que possam operar também com dados semi-estruturados e não-estruturados.

Portanto, os principais fatores críticos de sucesso do SAD são:

- A utilização de um processo de desenvolvimento de software extremamente ágil e *risk-confronting*;
- Adoção de um rápido ciclo requisitos-desenvolvimento-avaliação no sentido de nortear o esclarecimento dos requisitos e a avaliação de alternativas.

5.9.2. Etapas da realização do estudo de caso

1) Treinamento

No estudo de caso com o projeto do SAD, foi realizado inicialmente um treinamento para a equipe de desenvolvimento do projeto apresentando a Rede de Projeto, elementos, regras e exemplos de utilização das redes com processos de desenvolvimento, além de demonstrar o sistema GDP e seus módulos.

2) Criação da Rede de Atividades da metodologia XP1

Em seguida a gerente do projeto realizou a criação da rede de Atividades da metodologia XP1 (XP1, 2007) - uma extensão da metodologia XP (2004) com o sistema GDP, módulo gráfico.

Para inserir elementos da RA (atividades, transições, arestas, eventos, repositórios, etc.) deve-se selecionar o ícone do elemento no menu de elementos na parte central inferior da tela e clicar na área de modelagem (Figura 5.7). A Rede de Atividades principal encontra-se na Figura 5.7. Na Figura 5.8 encontra-se a sub-rede da etapa de *integração*. No banco de dados do GDP encontram-se todas as sub-redes do projeto SAD.

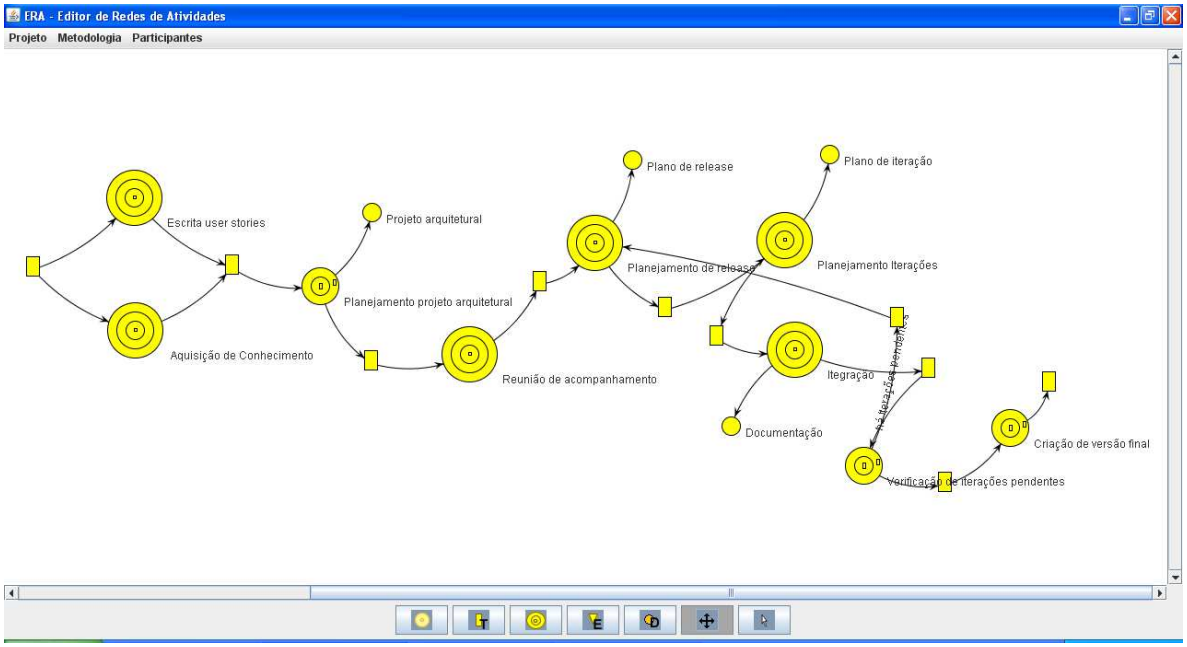


Figura 5.7. Rede de Atividades da metodologia XP1

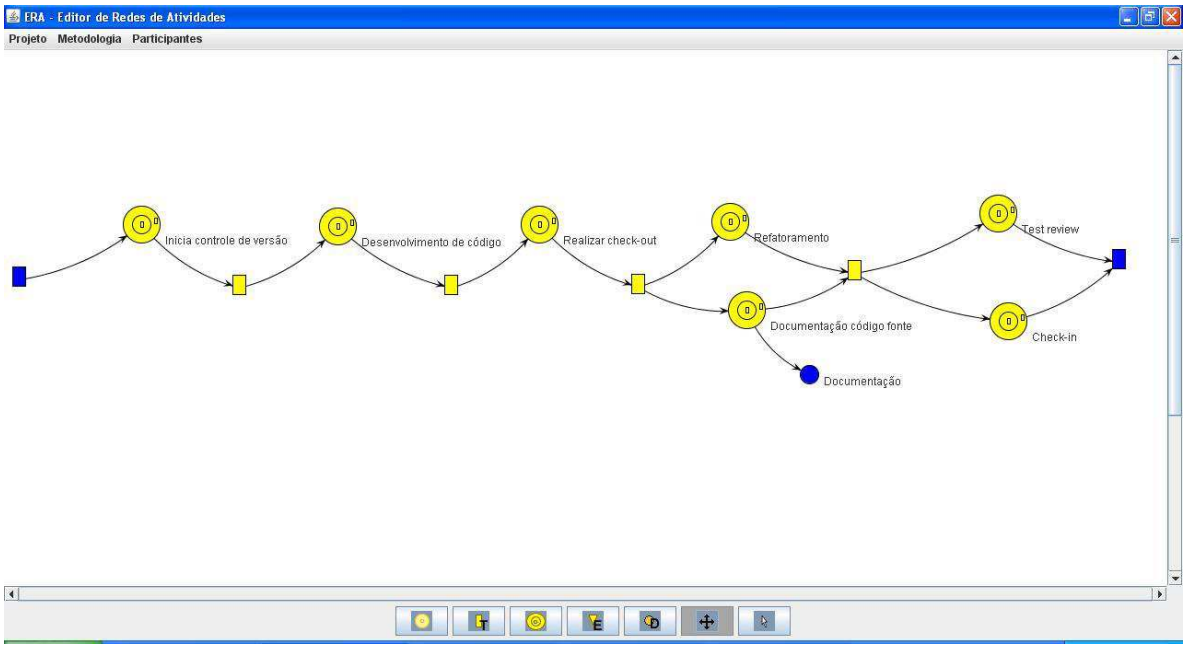


Figura 5.8. Sub-rede da etapa de integração de XP1

3) Adaptação da rede criada

O próximo passo foi adaptar a rede criada de acordo com as etapas do planejamento do projeto SAD (Tabela 5.1). Estas etapas foram definidas pelos "tutores" do projeto em conjunto com gerente de projeto para os dois anos de duração do projeto. A gerente do projeto recuperou (abriu) a rede de XP1 e realizou a adaptação da rede, modificando o desenho e descrevendo as atividades reais de acordo com a documentação da Tabela 5.1. na rede de XP1. Algumas das redes criadas para o projeto SAD estão nas Figuras 5.9 e 5.10.

Projeto SAD
ANO I

Tabela 5.1: Tabela de Etapas, Artefatos, Cronograma

Etapas	Atividades	Deliverables	Cronograma	Artefatos	Responsá-veis	Custo (R$)
Aquisição de conhecimento	Aquisição de Conhecimento sobre *tecnologias ODS e* WRAPPERS	Relatório sobre o estado-da-arte em tecnologia *ODS* Versão 1 Versão 2	04/2008 (160 h) 05/2008 (160 h)	Relatório sobre o estado-da-arte em "ODS"	Carlos/ José	50.000,00
		Análise sobre o estado-da-arte em tecnologia WRAPPERS		Relatório sobre o estado-da-arte em "WRAPPERS".	Vitor/ Ângela	

		Versão 1 Versão 2	04/2008 (160 h) 05/2008 (160 h)			
Análise	Descrição de user stories		04/2008 (160 h)	Relatório de User stories criadas	Ernani/ Pedro	50.000,00
	Levanta-mento do escopo, contexto e requisitos	Versão 1 *ODS* WRAPPERS	04/2008 (160 h)	Relatório sobre o escopo do projeto, e sobre os requisitos funcionais e não funcionais	João/ Maria Leonardo /Humberto	25.000,00
		Versão 2 *ODS* WRAPPERS	05/2008(160 h)		João/ Maria	
		Versão 3 *ODS* WRAPPERS	06/2008(160 h)		Maria/João	
Primeira Integração de Dados			06/2008 (160 h)	Relação sobre Integração de Dados	Ernani/ Pedro	10.000,00
Planejamento						
Projeto	Arquitetura e escolha de tecnologias		03/2008 04/2008 (1760 h)	Relatório contendo o projeto arquitetural e descrição de tecnologias escolhidas	Leonardo / Humberto	15.000,00
	Projeto e População do Banco de dados		04/2008 a 02/2009 (1760 h)	- Relatório com levantamento de requisitos sobre o banco de dados utilizado. -Relatório com definição da arquitetura do serviço ODS	João/ Maria	80.000,00
	Desenvolvimento do núcleo da ferramenta		07/2008 a 03/2009 (1440 h)	Relatório com descrição de User Stories e priorização	João/ Maria	60.000,00
Testes	Elaboração de Testes		07/2008 a 03/2009	Relatório com testes de aceitação para as user stories	Leonardo / Humberto	30.000,00
Documentação			07/2008 a 03/2009 (1440 h)	User Stories com divisão em 5	João/Maria	60.000,00

				releases		
					Total ANO I	**380.000,00**
ANO II						
Etapas	**Atividades**	**Deliverables**	**Cronograma**	**Artefatos**	**Responsá-veis**	**Custo (R$)**
Integração	Desenvolvi-mento e integração de dados		04/2009 a 12/2009 (1440 h)	Relatório com user stories, testes de aceitação, código e documenta-ção do software. Implantação	Leonardo/ Humberto	100.000,00
	Medição da eficácia do piloto		12/2009 (1440 h)	Relatório com plano de Medição.	Leonardo / Humberto	90.000,00
					Total ANO II	**190.000,00**
					Total geral	**570.000,00**

OBS: Os dados apresentados nesta tabela (nomes de desenvolvedores, valores de tempo e custo do projeto real) foram modificados para preservar o sigilo do projeto.

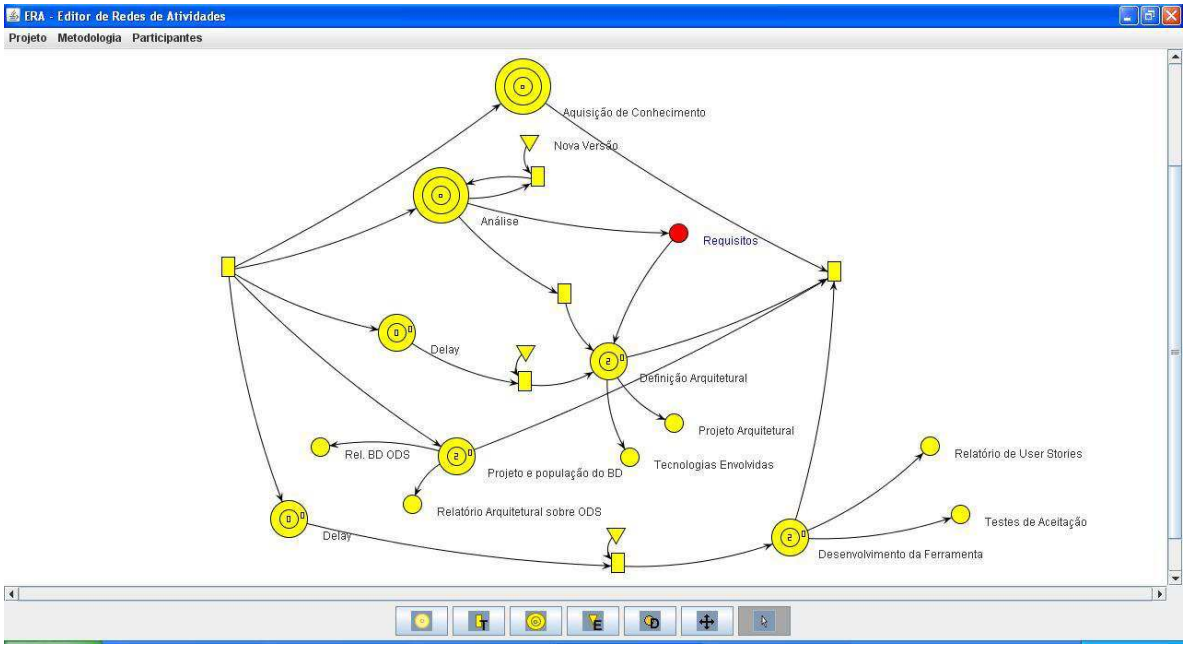


Figura 5.9. Rede de Atividades de XP1 – Projeto SAD Ano I

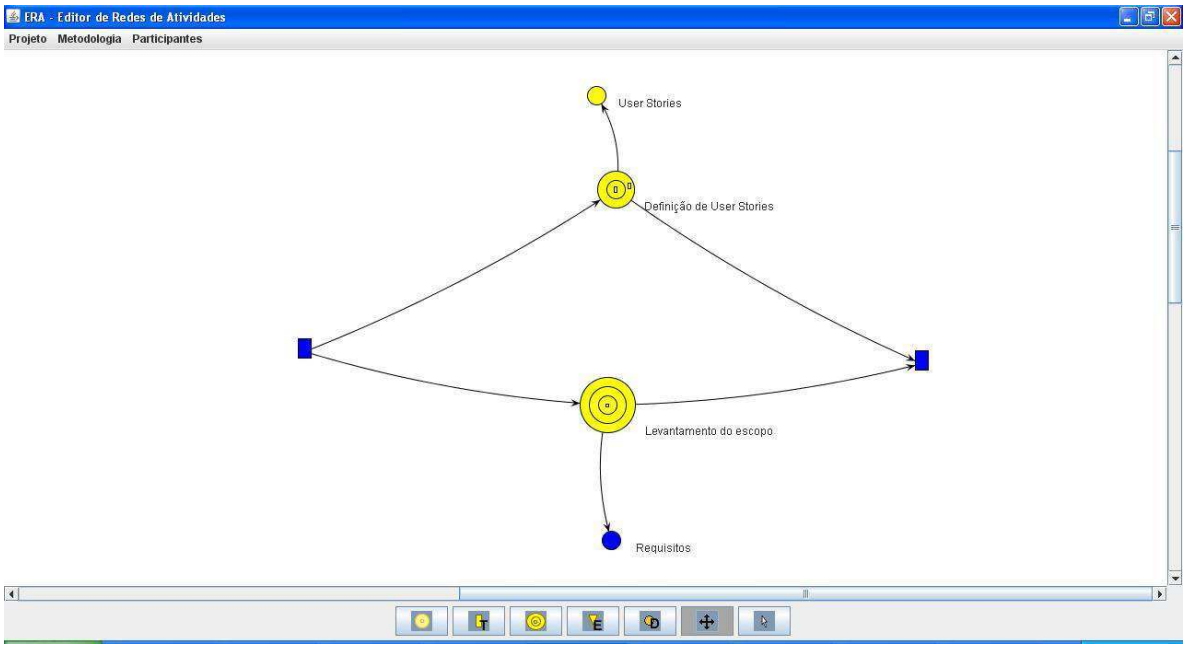


Figura 5.10: Rede de Atividades de XP1 – Sub-rede de aquisição de conhecimento

4) Instanciação da rede SAD

Após a rede de XP1 ter sido criada, o próximo passo foi inserir os atributos de recursos (humanos e operacionais), tempos e custos. Para isto foi utilizado o botão direito do mouse para editá-los (Figura 5.11).

A gerente incluiu os valores de tempo, custos e definição dos recursos humanos para as diversas atividades do projeto. (de acordo com a Tabela 5.1.). Após a instanciação dos itens do projeto, os desenvolvedores foram convidados a participar do estudo de caso, em seções individuais.

Cada desenvolvedor observou a Rede de Atividades criada para o projeto e verificou quais as suas funções a realizar. Após a conclusão da atividade eles inseriam os tempos realizados (data e hora). Enquanto isso a gerente podia acompanhar a realização das atividades da primeira atividade do projeto através do módulo gráfico do editor ERA. As Figuras 5.11 e 5.12. mostram algumas telas capturadas durante a instanciação.

A Figura 5.11 mostra os tempos iniciais (hora e data) de cada membro da equipe que está alocado à atividade "Aquisição de conhecimento em ODS". Estes valores são apresentados na tabela de tempos no formulário maior. O *tempo real* será apresentado pelo sistema após a conclusão da tarefa (Figura 5.13).

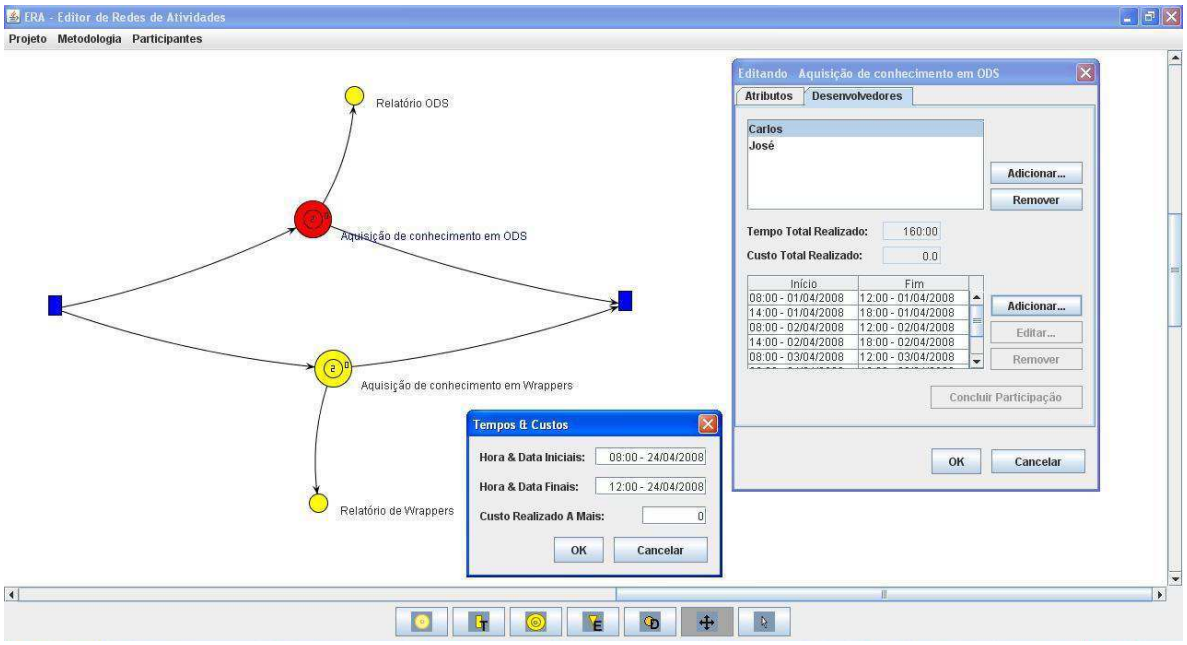


Figura 5.11.: Instanciação de valores de tempo para desenvolvedor

A Figura 5.12 mostra os valores de custos e tempos associados à atividade "Definir *user stories*".

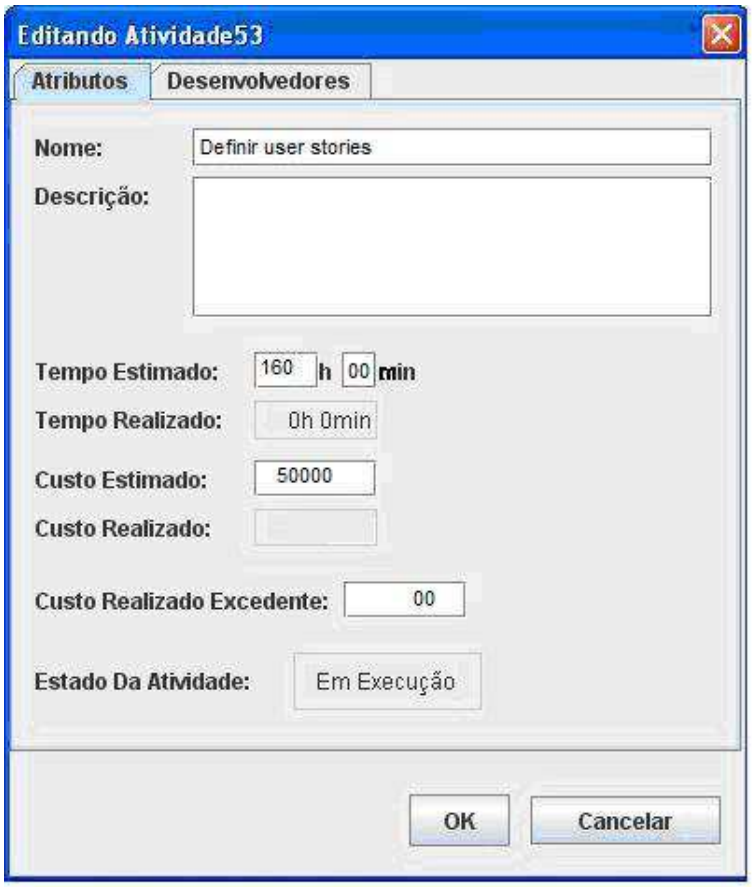


Figura 5.12: Tela de atributos alocados à atividade “Definir user stories”.

5) Totalização e Simulação da rede SAD

Após a conclusão das atividades, a gerente e sua equipe puderam observar os totais realizados para a primeira etapa do projeto (aquisição), através do módulo de totalização e simulação do ERA (Figura 5.13.). Vale a pena ressaltar que os tempos totais previstos são iguais aos tempos realizados, pois esta é uma regra do projeto SAD: totais de custo não podem exceder o orçamento previsto.

Neste módulo podem ser realizadas simulações, por exemplo, retirar ou incluir algumas atividades ou pessoas e verificar o andamento do projeto.

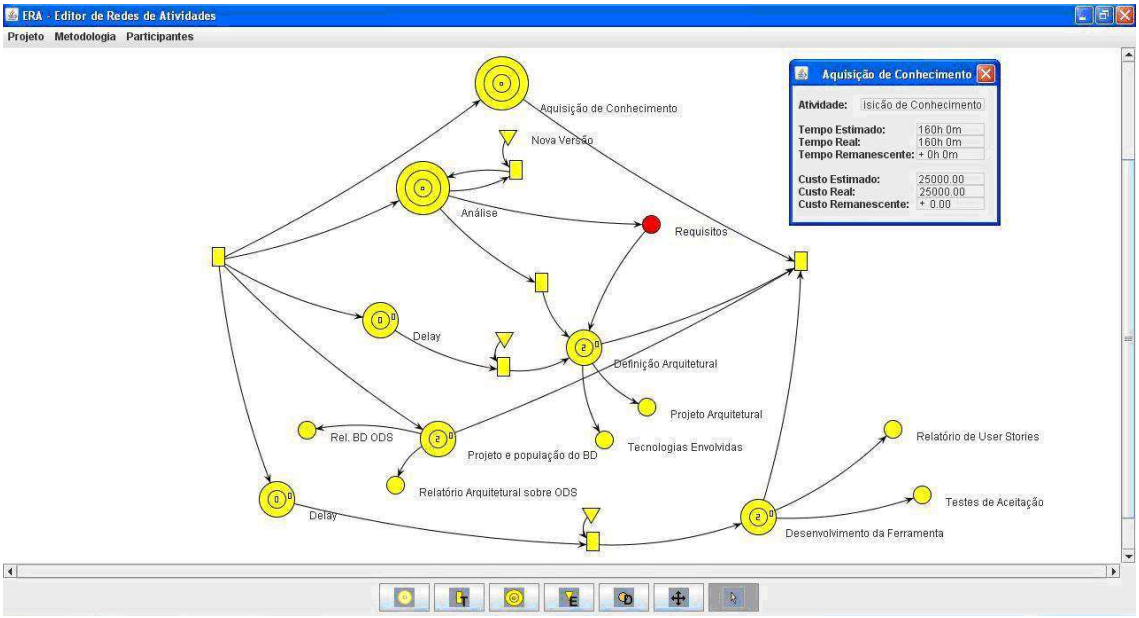


Figura 5.13: Totalização de valores de tempo e custo para a atividade “Aquisição de conhecimento”.

Durante a execução do estudo de caso algumas dificuldades foram encontradas, como por exemplo, dificuldade de modelar o processo XP1, pois a gerente do projeto não o havia utilizado ainda e foi preciso entender o processo para depois modelá-lo.

Durante as etapas de adaptação e instanciação não houve dificuldades, a não ser na inclusão de valores de tempos e custos no editor de RA, pois ele não possui help *on-line*.

Uma dificuldade encontrada pela gerente durante a simulação foi lembrar-se de acionar as transições antes da simulação. Os demais integrantes da equipe não sentiram dificuldades visto que o trabalho maior ficou para a gerente do projeto.

5.10. Comentários

Neste capítulo foi apresentado o sistema GDP - Gerência de Desenvolvimento de Projetos. Ele permite o planejamento e acompanhamento de projetos através dos módulos: **SiSeM** – Sistema para Seleção de Metodologias de Desenvolvimento de Software e **ERA** – editor de Rede de Atividades. O SiSeM é um sistema para sugerir a adoção de uma metodologia de desenvolvimento adequada ao tipo de projeto a ser desenvolvido. O Editor de Rede de Atividades por sua vez, permite modelar metodologias e projetos além de acompanhar o desenvolvimento do projeto. Em seu **sub-módulo gráfico** (interface) o editor permite criar uma RA e, para cada atividade, determinar as pré e pós condições de atividades, eventos, sub-redes, artefatos gerados, tempos estimados e reais, custos estimados e reais. O sub-módulo de **gerenciamento de recursos** gerencia a alocação de pessoal, equipamentos ao projeto e o sub-módulo **de armazenamento e recuperação da rede** que permite o armazenamento e recuperação de uma RA da Base de Projeto. Finalmente, com o **simulador** é possível, em cada ponto do projeto, simular vários cenários de continuidade e finalização do projeto.

O acompanhamento de determinado projeto é apresentado na forma de uma rede que exibe, através de seus componentes, o fluxo das atividades, e para cada atividade são mostradas informações tais como: tempos e custos estimados, remanescentes, reais e recursos alocados. Dessa forma, a equipe de desenvolvimento pode acompanhar o andamento dos projetos.

O módulo SiSeM está pronto, e foi utilizado na empresa *Light Infocon Tecnologia S.A*. para escolha da metodologia mais adequada ao projeto Cheque Moradia (Moradia, 2006). Apesar da empresa já possuir sua própria metodologia, o cliente desejava ter uma documentação do projeto e uma série de necessidades adicionais e então a empresa necessitava saber qual seria a metodologia mais adequada para esta aplicação.

Com relação ao módulo ERA, estão prontos os sub-módulos: gráfico, armazenamento e recuperação da rede, de recursos humanos e simulação, utilizados no estudo de caso do projeto SAD, a ser descrito no próximo capítulo. Será desenvolvido posteriormente o sub-módulo de controle de recursos operacionais e o módulo de controle de comunicação.

Capítulo 6

Estudo Avaliatório e Discussão de Resultados

Neste capítulo é apresentado o experimento prático com o ambiente GDP (Rede de Atividades e software editor de RA) no projeto SAD – Sistema de Apoio à Decisão da CHESF - Companhia Hidro-Elétrica do São Francisco. É mostrada a metodologia de avaliação, sua apresentação e aplicação, os instrumentos criados e utilizados, e a descrição da condução do ensaio. Este capítulo integra dois enfoques avaliatórios que envolveram a participação de usuários no ambiente: *a sondagem da satisfação subjetiva* e a *mensuração do desempenho do usuário*. São apresentados também os resultados obtidos individualmente com cada enfoque e em seguida eles são confrontados, mostrando os resultados da avaliação do desempenho, assim como o processamento estatístico dos dados, uma discussão das dificuldades registradas pelos usuários e as recomendações a partir dos enfoques adotados. São delineados os perfis das categorias de usuários consideradas nesta pesquisa e apresentados os resultados das opiniões sobre o ambiente, respaldados pelo processo de observação e pela análise dos registros das sessões de teste.

6.1. Metodologia de Avaliação

Dentre as técnicas de avaliação de produtos de software estão àquelas propostas por

Nielsen (2001), Redsih e Hackos (1998), Schneiderman (1998), ISO 9126 (2000) e Queiroz, 2001. Neste trabalho foi adotada a metodologia de avaliação de softwares desenvolvida por Queiroz.

Esta metodologia confronta os enfoques avaliatórios centrados na mensuração do desempenho do usuário, avaliação com base em experimentos laboratoriais, mensuração da satisfação do usuário em torno da sua opinião e inspeção de conformidade do produto a um padrão.

A **mensuração do desempenho** é fundamentada em metas quantitativas de usabilidade. Mayhew (1999 *apud* Queiroz, 2001) (e.g. tempo de execução de uma tarefa, número de ações e escolhas incorretas cometidas durante a execução de uma tarefa), e metas relativas, quando envolvem comparações de ações de usuários apoiadas por produtos da mesma natureza (e.g. tempos de execução de uma tarefa utilizando três produtos similares e concorrentes no mercado).

Por outro lado, a **mensuração da satisfação do usuário** trata de aspectos subjetivos fundamentais na opinião do participante sobre o produto avaliado. Este enfoque possibilita a aquisição de informações relativas a atitudes, opiniões e preferências do usuário.

A metodologia de Queiroz foi adotada por este trabalho porque apresenta viabilidade dos procedimentos associados a cada uma de suas etapas, clareza dos objetivos a atingir em cada uma delas, facilidade de aprendizado e execução, além de respaldo dos resultados em outras iniciativas de natureza similar.

A abordagem metodológica possui diretrizes para coletar dados em *software*s e material: DePerUSI – Delineamento do Perfil do Usuário e OPUS - Opinião do Usuário de Software, Declaração de Conhecimento das Condições de Teste, Ficha de Cadastro de Participante e Fichas de Registros de Eventos, com base nos estudos de Nielsen e outros citados em (Queiroz, 2001), abordando aspectos relativos ao sistema como um todo (funcionalidade, confiabilidade dos resultados, usabilidade, motivação de uso, facilidade de memorização, etc.). Cabe ao avaliador que pretende fazer uso da metodologia, adaptá-la à sua realidade.

Nesta pesquisa de tese, o levantamento da satisfação do usuário quanto ao uso da Rede de Atividades e do software GDP – Gerência de Desenvolvimento de Produtos, módulo ERA – Editor de Redes de Atividades, fundamentou-se no instrumento de sondagem para avaliação com base no OPUS (onde foram incluídas novas perguntas e retiradas outras) e nas fichas de registros de eventos.

Esta metodologia foi adaptada para a avaliação do sistema SAD - Sistema de Apoio à Decisão.

6.1.1. Definição do Processo Avaliatório

Objeto de Estudo: utilização do ambiente (Rede de Atividades + Rede de Projeto + software Editor de RAS - ERA) no gerenciamento de um projeto de desenvolvimento de software.

Objetivo geral: identificar a viabilidade de utilização do ambiente no gerenciamento de projetos de desenvolvimento de software.

Objetivos específicos: observação da facilidade de uso do ambiente, facilidade de execução das tarefas, mensuração do tempo de conclusão das tarefas de usuários experientes e principiantes, mensuração do número de ações incorretas durante a execução da tarefa e mensuração do número de escolhas incorretas de elementos durante a execução das tarefas.

Perspectiva: o estudo é desenvolvido sob a visão do gerente de projeto e de desenvolvedores, avaliando a viabilidade de utilização da Rede de Atividades no planejamento e acompanhamento de um projeto real.

Contexto: o estudo prático foi realizado com os gerentes e desenvolvedores do projeto SAD - Sistema de Apoio à Decisão da CHESF - Companhia Hidro-Elétrica do São Francisco que trabalham no laboratório SAD do Departamento de Sistemas e Computação, na Universidade Federal de Campina Grande - UFCG.

O Sistema Utilizado no Estudo Experimental: Sistema de Apoio à Decisão da Operação - SAD, cujo requisito maior é fornecer à operação, em tempo real, acesso fácil e rápido às informações que são relevantes aos fatores críticos de sucesso dos operadores.

6.1.2 Aplicação da Metodologia

6.1.2.1 Caracterização do Alvo de Estudo

Nesta pesquisa, o sistema produto-alvo do estudo foi utilizado em função das categorias de usuários (experientes e principiantes) disponíveis para fins de recrutamento como participantes dos ensaios avaliatórios.

6.1.2.2 Definição das Metas e Interesses

Foram especificadas as metas e interesses gerais e específicos que pautaram a condução do processo avaliatório através do qual serão verificadas as hipóteses formuladas.

6.1.2.3 Caracterização do Universo Amostral

Durante esta sub-etapa estabeleceu-se, com base nas metas e interesses especificados. Tal definição possibilitou tanto o planejamento do modo de partição do universo amostral em sub-grupos de usuários (experientes e principiantes).

6.1.2.4. Definição do Modo de Recrutamento dos Participantes

Além do contato com a equipe do projeto SAD, esta sub-etapa caracterizou-se pela definição e estruturação de uma estratégia no processo de recrutamento de outros participantes (contato oral, divulgação do trabalho nos laboratórios da Universidade Federal de Campina Grande, salas de aula da UFCG e da Universidade Estadual de Campina Grande).

6.1.2.5. Decisão do Número de Participantes do Ensaio Avaliatório

Foi pretendido, desde o início do planejamento, trabalhar com um universo amostral de pelo menos, 100 usuários. Inicialmente foram distribuídos 60 questionários para sondagem do perfil, entre desenvolvedores do projeto SAD, gerentes de projeto e alunos das universidades UFCG e UEPB. Dentre os questionários analisados foi verificada a pouca quantidade de usuários com experiência em gerenciamento de projetos. Outro problema foi conseguir disponibilidade de tempo por parte dos gerentes. Ainda houve dificuldade de conseguir que os professores liberassem os alunos no horário da aula para participarem dos testes. Então, a decisão tomada no tocante ao número de usuários de teste dos ensaios de usabilidade considerou um universo amostral de 50 participantes, e de acordo com a sua experiência, foi dividido em (principiantes e experientes).

Dos 50 usuários, apenas cerca de 72% do contingente considerado participaram das sessões de testes. Deste montante, os resultados obtidos nos testes de 16% dos participantes foram excluídos do universo amostral, pela falta de tempo disponível da parte deles ou por não terem devolvido os questionários pós-teste.

Assim, o universo amostral efetivo deste experimento totalizou 30 indivíduos, de ambos os sexos, distribuídos em duas categorias: *principiantes* e *experientes*, na

proporção de *15:15*, respectivamente. A Tabela 6.1 sumariza os aspectos gerais do ensaio avaliatório.

Tabela 6.1.a - Aspectos gerais do ensaio

ASPECTOS GERAIS DO ENSAIO	
Estrutura de rede utilizada	Redes de Atividades
Produto	GDP – Gerência de Desenvolvimento de Produtos, módulo ERA, v. 1.0
Natureza do ambiente	Ambiente dedicado ao gerenciamento de projetos e processos de desenvolvimento
Objetivo geral	Identificar a viabilidade de utilização do ambiente no gerenciamento de projetos no desenvolvimento de softwares.
Objetivos específicos	-Observação da facilidade de uso do ambiente; -Observação da facilidade de execução das tarefas; -Mensuração do tempo de conclusão das tarefas; -Mensuração do número de ações incorretas durante a execução da tarefa; -Mensuração do número de escolhas incorretas de elementos durante a execução da tarefa; -Número de pedidos de ajuda -Investigação da satisfação subjetiva dos usuários de teste para a confrontação com os demais fatores de avaliação considerados. -Verificação da análise quantitativa das repostas ao experimento.
Natureza da avaliação	Somativa objetiva/subjetiva e qualitativa/quantitativa
Natureza dos testes	Campo e laboratorial
Número de avaliadores	01
Natureza das tarefas avaliadas	Mais freqüentes
Dimensão do universo amostral	30 usuários
Estratégias	1. Questionários 2. Observação direta 3. Verbalização de procedimentos (*Think Aloud*)
Duração da sessão de teste	2:30 min (em média)
Duração completa do ensaio	25 semanas (do planejamento até totalização de resultados)
Tipos de resultados previstos	-Estudo discriminatório dos respondentes dos questionários pré-teste - Bateria de tarefas de testes do ensaio -Estudo analítico da satisfação subjetiva dos respondentes dos questionários pós-teste - Verificação de existência de problemas - Verificação de aspectos positivos - Totalização de dados quantitativos.

Tabela 6.1.b - Aspectos gerais do ensaio

Indicador(es)	**Subjetivo(s)**	- Facilidade de uso do produto - Facilidade de aprendizado do produto - Facilidade de execução de tarefas - Facilidade de localização de informações - Facilidade de visualização das informações no grafo. - Eficácia do tipo de mensagens *on line*	
	Objetivo(s)	- Número de opções incorretas - Número de ações incorretas - Tempo gasto na conclusão de uma tarefa - Número de pedidos de ajuda - Observações de frustração - Observações de confusão - Observações de satisfação	
Número de tarefas		04	
Especificação das tarefas		Tarefa 0	Ativação do aplicativo
		Tarefa 1	Modelagem da Rede de Atividades da metodologia XP1.
		Tarefa 2	Inserção das atividades reais na Rede de Atividades (adaptação)
		Tarefa 3	Inserção de pessoal, tempos e custos na rede. (instanciação)
		Tarefa 4	Totalização e simulação de tempos e custos

Tabela 6.2. – Recursos materiais utilizados na avaliação

NATUREZA	ESPECIFICAÇÃO
Hardware	Notebook Intel Core 2 Duo CPU T730, 2 GHz, 1,99 RAM, 230 GB.
Software	*GDP – módulo ERA v. 1.0*
Outros	Cronômetros (01) Ficha de Cadastro do Participante Questionários (Perfil do usuário e OPUS) Roteiros de Tarefas (versões usuário de teste e avaliador) Fichas de Registros de Eventos Bloco de notas

6.2. Elaboração do Material e Condução do Ensaio

6.2.1 Planejamento e Estruturação de Tarefas de Teste

Neste trabalho foi possível modelar as tarefas do projeto com o editor *ERA versão 1.0* com o propósito de identificar problemas de modelagem, de adaptação, de instanciação e simulação, além de problemas de uso, aprendizado, execução de tarefas, localização de informações, visualização da rede e verificação de mensagens de erro do produto e, por

conseguinte, refinar o processo de seleção de tarefas para compor o roteiro de testes do ensaio de avaliação planejado (mensuração do desempenho do usuário durante o uso do produto).

Treinamento: o treinamento dos participantes teve duração de aproximadamente de meia hora. Na primeira fase, foi realizada uma exposição sobre a Rede de Atividades, sobre os elementos e regras propostas, utilizando um conjunto de *slides* que também foram distribuídos para os participantes. Na segunda fase foi realizada uma apresentação sobre o sistema GDP e o módulo ERA.

Projeto Piloto: antes da execução do estudo, foi realizado um projeto piloto. Para este projeto, foi selecionado apenas um participante tido como experiente e outro principiante. Os participantes foram treinados na utilização dos elementos e regras da RA e as utilizaram com o projeto. Os participantes realizaram a modelagem da RA de acordo com a metodologia utilizada no projeto (XP1), a adaptação da metodologia ao projeto real, a instanciação com dados reais e uma totalização/simulação sobre totais de tempos e custos. O projeto piloto foi um estudo baseado em observação. Mas ele não visou extrair qualquer resultado dos participantes, mas encontrar problemas no material planejado para o estudo, permitindo que este material fosse aprimorado antes de sua utilização e para sondar o tempo de duração do teste e qual a duração real do experimento em comparação com o tempo previsto inicialmente.

Participantes: desenvolvedores de software e o gerente do projeto da CHESF, e outros gerentes de projetos e alunos do curso de Computação da UFCG e UEPB. O estudo não foi executado em um ambiente da indústria, mas em um ambiente acadêmico que produz o sistema SAD a ser utilizado na CHESF e laboratórios da UFCG e UEPB.

6.2.2. Condução do Ensaio e Coleta de Dados

Procedimento de Participação: foi elaborado um procedimento de participação para o experimento, abaixo descrito:

Procedimento de escolha dos participantes com a Utilização do Ambiente Proposto

- Responsável pelo experimento distribui questionários para equipe do projeto SAD e gerentes de projeto e alunos da UFCG e UEPB
- Participante preenche o questionário pré-teste
- Participante entrega o questionário para o responsável pelo experimento

- Responsável pelo experimento guarda o questionário na pasta do experimento e escolhe os participantes de acordo com o perfil obtido no questionário

Procedimento de Participação com a Utilização do Ambiente Proposto

- Responsável chega ao local do experimento
- Participante assina o termo de adesão ao experimento
- Responsável pelo experimento associa um número ao participante
- Participante recebe treinamento
- Participante realiza tarefa1- lê a descrição da metodologia a ser utilizada no projeto (XP1)
- Participante cria a Rede de Atividades da metodologia adotada na instituição
- Responsável pelo experimento anota número de ações incorretas, número de opções incorretas, tempo de execução da tarefa e número de pedidos de ajuda em seu registro
- Participante realiza a tarefa 2- adaptação da rede de acordo com a metodologia adotada no projeto SAD
- Responsável pelo experimento anota número de opções incorretas, número de ações incorretas, tempo de execução da tarefa e número de pedidos de ajuda
- Participante observa a rede criada com a metodologia
- Participante realiza a tarefa 3- instancia a rede e insere as informações do projeto e da equipe, colocando atividades, recursos, tempos e custos
- Responsável pelo experimento anota número de opções incorretas, número de ações incorretas, tempo de execução da tarefa e número de pedidos de ajuda em seu registro
- Participante realiza a tarefa 4- simulação dos temos e custos totais reais, previstos e remanescentes
- Responsável pelo experimento anota número de opções incorretas, número de ações incorretas, tempo de execução da tarefa e número de pedidos de ajuda em seu registro
- Responsável pelo experimento distribui questionário pós-teste e formulário de perguntas subjetivas
- Participante entrega o questionário pós-teste respondido para o responsável pelo experimento
- Responsável pelo experimento guarda o questionário na pasta do

experimento

- Participante entrega o questionário de perguntas subjetivas para o responsável pelo experimento
- Responsável pelo experimento guarda o formulário na pasta do experimento.

6.3. Resultados da Mensuração de Desempenho

Este item mostra os resultados obtidos a partir da mensuração de desempenho do usuário durante a modelagem da Rede de Atividades e do uso do software ERA - Editor de Rede de Atividades.

Primeiro serão apresentados os indicadores quantitativos coletados a partir dos experimentos, fundamentados na observação direta das anotações das fichas de registros de tarefas.

6.3.1. Análise Preliminar dos Indicadores Quantitativos

A mensuração do desempenho resultou em dados sobre três aspectos do procedimento de teste: o tempo de execução das tarefas, número de ações e de escolhas incorretas. O processo de observação direta permitiu uma análise do indicador de **tempo de execução** ao longo das tarefas de teste que explica o porquê das diferenças nos valores registrados, como pode ser visto na Tabela 6.3.

Observa-se que há diferenças significativas nos mínimos e máximos obtidos para as duas categorias de usuários, principalmente na tarefa 1 - *Modelagem da Rede de Atividades da metodologia XP1:* o valor mínimo obtido foi 17 min. e máximo 42 min. no grupo dos participantes experientes, já no grupo de principiantes, os valores foram: mínimo de 60 min. e máximo de 93 min. Deve-se considerar que, no início de cada sessão, o usuário de teste foi instruído a executar a tarefa como se fizesse em seu ambiente de trabalho, sem uma preocupação excessiva com o tempo de execução.

Nas tarefas 2 - *Inserção das atividades reais na Rede de Atividades* e 3 - *Inserção de pessoal, tempos e custos na rede* existe pouca diferença de valores entre os grupos de experientes e principiantes. Na tarefa 2 os valores mínimos variam entre 8 e 6 entre principiantes e experientes, respectivamente e valores máximos variam entre 20 e 22. Na tarefa 3 os valores estão entre 5 e 7 (mínimos) e 12 e 16 (máximos).

Na tarefa 4 – *Totalização e Simulação de custos e tempos* existe uma diferença um pouco maior entre os valores registrados: 10 min. e 29 min. entre os experientes e 11 min. e 30 min. dos principiantes.

A Tabela 6.3. apresenta os indicadores quantitativos de tempo coletados através do processo da observação direta com os participantes dos experimentos: gerentes de projetos de software e alunos da graduação de computação. Na última coluna encontra-se o somatório de tempo para cada usuário. Ao final de cada grupo de usuários, encontra-se o somatório total dos tempos para cada tarefa, o tempo médio de execução das tarefas e os valores máximos e mínimos de tempo de cada tarefa.

Foi observado que tanto as reações individuais ao ambiente quanto as posturas dos participantes face ao grau de experiência implicaram diferenças observáveis no uso do tempo delimitado para a execução de cada tarefa, em função de seu grau de dificuldade (as tarefas 1 e 3 por exemplo, foram as mais extensas.)

No tocante ao número de **ações incorretas**, observa-se na Tabela 6.4. principalmente para os valores totais, que à medida que aumenta o grau de dificuldade das tarefas, aumenta a diferença entre os resultados associados aos usuários de teste. Aqui foi confirmada também a expectativa de melhores desempenhos dos usuários experientes, seguidos dos principiantes. Os valores médios das ações incorretas dos principiantes são maiores do que os valores dos experientes. Os valores variam de 1,07 a 1,93 entre experientes, e 1,53 a 3,60 entre principiantes, isto aconteceu devido à falta de familiarização dos principiantes com o processo, visto que apenas 13,3 % (2 de 15) usuários já haviam tido experiência com processos de desenvolvimento.

O número de **opções incorretas** refere-se mais especificamente ao modo de interação via ícones ou itens de menus ou outros itens relacionados à interação da função de cada um dos elementos da Rede de Atividades. As médias dos valores observados (Tabela 6.5.) das tarefas 2, 3 e 4 mostram que existiu diferença entre as médias das categorias de usuários experientes e principiantes respectivamente: com médias 0,60 a 1,33 dos experientes, contra 1,40 a 2,20 dos principiantes com opções incorretas. A diferença entre categorias foi um pouco maior na tarefa1: 1,33 (experientes) e 2,20 (principiantes).

O indicador de **números de pedidos de ajuda**, conforme apresentado na Tabela 6.6., apresenta a conclusão de que os usuários de teste tiveram que pedir ajuda devido ao fato de que estavam realizando a modelagem da Rede de Atividades pela primeira vez. Os principiantes só possuíam um único contato prévio com o processo XP na disciplina que estavam ou tinham cursado no curso de computação e, além disso, o software em questão ainda não possui ajuda on-line. Mas observa-se que a quantidade de pedidos não foi muito alta: o menor valor dos mínimos é de 0 pedidos entre os principiantes e máximo de 6 pedidos. Entre os experientes, os valores vão de 0 a 3 pedidos de ajuda. A maior parte dos pedidos de ajuda dos usuários é com relação às dúvidas no entendimento do processo XP1 representados na Tabela 6.6.

Tabela 6.3: Tempo de execução de tarefas

Indicadores quantitativos obtidos a partir do experimento (observação direta com fichas de registros de eventos).

Usuário	Categoria	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	∑
U1	P	60	8	7	11	86
U2	P	65	9	8	19	101
U3	P	70	8	7	15	100
U4	P	87	14	11	23	135
U5	P	72	12	9	16	109
U6	P	76	18	8	21	123
U7	P	92	17	11	25	145
U8	P	85	15	10	25	135
U9	P	79	22	9	23	133
U10	P	76	17	8	30	131
U11	P	90	19	16	27	152
U12	P	93	20	13	18	144
U13	P	88	16	8	15	127
U14	P	84	11	14	23	132
U15	P	78	8	8	11	105
	TOTAL	**1195**	**214**	**147**	**302**	
	Média	**79,67**	**14,27**	**9,80**	**20,13**	123,87
	MÍNIMO	**60**	**8**	**7**	**11**	
	MÁXIMO	**93**	**22**	**16**	**30**	
U16	E	27	16	5	14	62
U17	E	30	20	7	21	78
U18	E	24	6	8	13	51
U19	E	17	7	7	11	42
U20	E	22	7	5	12	46
U21	E	40	16	10	25	91
U22	E	42	18	11	29	100
U23	E	34	8	7	14	63
U24	E	35	7	8	18	68
U25	E	42	12	12	28	94
U26	E	17	12	6	10	45
U27	E	30	20	9	21	80
U28	E	19	8	5	11	43
U29	E	30	6	10	17	63
30	E	20	6	7	12	45
	TOTAL	**429**	**169**	**117**	**256**	
	Média	**28,60**	**11,27**	**7,80**	**17,07**	64,73
	MÍNIMO	**17**	**6**	**5**	**10**	
	MÁXIMO	**42**	**20**	**12**	**29**	

Legenda: P - principiante
E - experiente

Tabela 6.4: Número de ações incorretas

Usuário	Categoria	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	∑
U1	P	3	1	1	1	6
U2	P	3	2	1	2	8
U3	P	4	2	0	1	7
U4	P	3	2	1	1	7
U5	P	4	1	1	2	8
U6	P	5	1	4	3	13
U7	P	4	1	3	3	11
U8	P	4	1	2	1	8
U9	P	3	3	1	2	9
U10	P	2	1	4	2	9
U11	P	3	2	3	1	9
U12	P	4	0	2	1	7
U13	P	3	2	1	1	7
U14	P	5	2	3	1	11
U15	P	4	2	1	2	9
	TOTAL	**54**	**23**	**28**	**24**	
	Média	**3,60**	**1,53**	**1,87**	**1,60**	
	MÍNIMO	**2**	**0**	**0**	**1**	
	MÁXIMO	**5**	**3**	**4**	**3**	
U16	E	0	0	3	1	4
U17	E	1	1	1	1	4
U18	E	1	0	0	2	3
U19	E	2	2	1	2	7
U20	E	1	1	2	1	5
U21	E	3	2	1	1	7
U22	E	1	0	1	2	4
U23	E	3	1	2	1	7
U24	E	3	0	1	0	4
U25	E	3	2	1	2	8
U26	E	2	3	0	1	6
U27	E	3	1	0	0	4
U28	E	1	1	1	0	3
U29	E	2	1	2	0	5
U30	E	3	2	1	2	8
	TOTAL	**29**	**17**	**17**	**16**	
	Média	**1,93**	**1,13**	**1,13**	**1,07**	
	MÍNIMO	**0**	**0**	**0**	**0**	
	MÁXIMO	**3**	**3**	**3**	**2**	

Legenda: P - principiante
E - experiente

Tabela 6.5: Número de opções incorretas

Usuário	Categoria	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	∑
U1	P	1	0	1	0	2
U2	P	3	1	2	1	7
U3	P	1	1	2	0	4
U4	P	3	1	1	4	9
U5	P	1	2	1	2	6
U6	P	3	1	3	2	9
U7	P	1	1	3	1	6
U8	P	1	2	2	2	7
U9	P	5	3	2	1	11
U10	P	2	2	3	1	8
U11	P	1	2	0	3	6
U12	P	2	1	1	2	6
U13	P	3	0	2	1	6
U14	P	2	2	2	3	9
U15	P	4	2	2	2	10
	TOTAL	**33**	**21**	**27**	**25**	
	Média	**2,20**	**1,40**	**1,80**	**1,67**	
	MÍNIMO	**1**	**0**	**0**	**0**	
	MÁXIMO	**5**	**3**	**3**	**4**	
U16	E	0	0	1	0	1
U17	E	1	1	1	1	4
U18	E	1	0	0	0	1
U19	E	2	1	1	1	5
U20	E	1	1	2	1	5
U21	E	3	1	1	0	5
U22	E	1	0	1	0	2
U23	E	2	1	2	1	6
U24	E	0	0	1	1	2
U25	E	3	0	1	1	5
U26	E	1	0	0	1	2
U27	E	1	1	0	1	3
U28	E	2	1	2	0	5
U29	E	1	0	0	1	2
U30	E	1	2	1	1	5
	TOTAL	**20**	**9**	**14**	**10**	
	Média	**1,33**	**0,60**	**0,93**	**0,67**	
	MÍNIMO	**0**	**0**	**0**	**0**	
	MÁXIMO	**3**	**2**	**2**	**1**	

Legenda: P - principiante

E - experiente

Tabela 6.6: Número de pedidos de ajuda

Usuário	Categoria	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	∑
U1	P	2	0	0	1	3
U2	P	3	1	1	0	5
U3	P	3	2	0	0	5
U4	P	6	1	1	1	9
U5	P	4	3	1	1	9
U6	P	4	2	3	2	11
U7	P	5	2	3	2	12
U8	P	3	1	3	3	10
U9	P	2	2	3	2	9
U10	P	4	1	3	1	9
U11	P	5	2	3	3	13
U12	P	2	3	2	1	8
U13	P	3	1	2	1	7
U14	P	3	1	0	2	6
U15	P	4	2	2	1	9
	TOTAL	**53**	**24**	**27**	**21**	
	Média	**3,53**	**1,60**	**1,80**	**1,40**	**2,08**
	MÍNIMO	**2**	**0**	**0**	**0**	
	MÁXIMO	**6**	**3**	**3**	**3**	
U16	E	2	0	1	1	4
U17	E	1	1	1	1	4
U18	E	2	1	0	0	3
U19	E	2	2	1	1	6
U20	E	2	1	0	2	5
U21	E	3	1	2	1	7
U22	E	3	2	2	2	9
U23	E	3	2	1	0	6
U24	E	2	1	1	0	4
U25	E	3	2	2	1	8
U26	E	1	1	0	0	2
U27	E	3	1	1	2	7
U28	E	0	1	0	0	1
U29	E	1	0	2	1	4
U30	E	2	2	1	1	6
	TOTAL	**30**	**18**	**15**	**13**	
	Média	**2,00**	**1,20**	**1,00**	**0,87**	**1,27**
	MÍNIMO	**0**	**0**	**0**	**0**	
	MÁXIMO	**3**	**2**	**2**	**2**	

Legenda: P - principiante
E - experiente

De um modo geral, as sessões com os usuários *principiantes* foram caracterizadas por um dispêndio de grande parte dos tempos pré-definidos para a execução das tarefas de teste com relação à:

a) interpretação a respeito do processo XP1 descrito no material bibliográfico adotado;

b) interpretação, nem sempre correta, das solicitações dos roteiros de tarefas;

c) identificação dos elementos adequados à modelagem dentre àqueles disponibilizados pela Rede de Atividades ou pelo software editor;

d) análise, nem sempre satisfatória, das mensagens do software;

e) tentativa infrutífera de reversão de ações.

Os comentários orais por parte dos participantes *principiantes* ao avaliador foram relacionados à:

a) dúvidas sobre as etapas e atividades do processo XP1;

b) dúvidas sobre valores das solicitações dos roteiros de tarefas (e.g. necessidade de previsão de valores de tempos e custos na faixa de tempo pré-definida, data e hora no formulário de edição do software);

c) estratégias mais constantemente usadas no tocante à reversão de situações de erro;

d) dúvidas sobre as mensagens de gravação da metodologia e do projeto;

e) recursos mais adequados a situações específicas (e.g. inserção de emissão de sons para situações de erros, mudança de cor para atividades já concluídas, uso de *hints* para ícones e sugestão de novos ícones para o software editor).

As seções com os usuários *experientes* obtiveram um dispêndio de grande parte dos tempos pré-definidos para a execução das tarefas de teste devido à:

a) leitura e análise das etapas e atividades da metodologia XP1;

b) leitura detalhada e seqüencial das ações associadas a cada tarefa do roteiro de tarefas;

c) ponderação de diferentes estratégias possíveis para a execução da adaptação da metodologia ao projeto da CHESF;

d) análise sobre a previsão de valores previstos de tempos e custos dentro da

faixa de valores pré-definida para o projeto da CHESF;

e) análise sobre os resultados da simulação dos totais (previstos, reais e remanescentes) do projeto da CHESF.

Os comentários orais por parte dos participantes *experientes* ao avaliador foram relacionados à:

a) dúvidas sobre as etapas e atividades do processo XP1 descrito no material bibliográfico adotado;

b) perguntas relativas à aplicação de ações comumente utilizadas no trabalho de origem aplicadas à tarefa em questão;

c) formas de obtenção de determinados resultados;

d) questionamentos sobre valores descritos nos roteiros de tarefas (e.g. necessidade de digitação de valores de data e hora na tela de edição do software);

e) estratégias apropriadas à reversão de situações de erro;

f) formas de uso de ícones do software e sugestão de novos ícones para o software editor;

g) visualização e navegação na hierarquia geral dos níveis da rede;

h) dúvidas sobre as mensagens apresentadas no software sobre gravação da metodologia e do projeto.

Nas sessões de teste com usuários experientes, o avaliador atuou essencialmente como observador, mediador e como instrutor, nos momentos de solicitação de ajuda por parte dos usuários. Enquanto observador, anotou o tempo de execução das tarefas, o número de escolhas e ações incorretas, além de observar as expressões faciais e comentários dos participantes através da verbalização informal. Estas informações contribuíram de forma esclarecedora em alguns momentos da análise dos dados, confrontando com a mensuração do desempenho e facilitando o entendimento dos resultados obtidos.

De um modo geral, as categorias de usuários principiantes e experientes demonstraram bom desempenho no uso de elementos da Rede de Atividades e nas suas regras, e sugeriram formas de disponibilização de resultados e melhorias de usabilidade para o software editor.

6.3.2. Análise Estatística dos Indicadores Quantitativos

As anotações referentes aos indicadores objetivos (fichas de eventos utilizadas nos experimentos), aliadas à verbalização de ações pelos usuários, possibilitaram não só a

contextualização dos dados quantitativos coletados, como também a caracterização das sessões de teste relativas a cada sub-grupo de usuários.

É sabido que não é possível fazer estatísticas sem amostras aleatórias das populações de interesse. No entanto, a amostragem aleatória das populações tem se mostrado impraticável nas pesquisas acadêmicas segundo (Levine, Berenson e Stephan; 2000) e (Costa Neto, 1977) que citam a dificuldade dos experimentos e o fato deles não permitirem inferências estatísticas válidas sobre as populações de interesse.

"Na impossibilidade da amostragem aleatória, as inferências estatísticas sobre os efeitos de tratamento deveriam ser restritas às pessoas envolvidas no experimento, enquanto que as inferências sobre os efeitos de tratamento para outras pessoas deveriam ser denominadas inferências não estatísticas (sem base em probabilidades, uma vez que se generaliza as unidades experimentais a outras que são bastante semelhantes com relação às características consideradas)" (Levine, Berenson e Stephan, 2000).

No entanto, não se pode deixar de considerar questões relativas à validade do processamento estatístico dos dados coletados a partir do ensaio de usabilidade conduzido nesta pesquisa, face às dificuldades enfrentadas, principalmente no recrutamento dos participantes e à predisposição deles à participação não remunerada e tempo a ser despendido.

Apesar de aqui a amostra não ter sido selecionada de forma aleatória, dados os motivos já expostos, os participantes (profissionais graduados, gerentes de projetos, desenvolvedores e gerentes do projeto SAD e alunos) apresentam características bastante distintas, como será apresentado posteriormente na Seção 6.4.1., e isto possibilita a composição de um universo amostral representativo da população considerada. O tamanho da amostra selecionada excedeu 30 unidades amostrais, o que pode ser considerada grande amostra ($n_A > 30$) segundo (Levine, Berenson e Stephan, 2000; Sabo, 1999) .

Os valores obtidos para os indicadores estatísticos encontram-se nas Tabelas 6.3, 6.4., 6.5 e 6.6. anteriormente apresentadas. O propósito da análise estatística é verificar se as diferenças comentadas na seção anterior são estatisticamente aceitáveis, de modo que se possa incrementar as inferências não estatísticas com conclusões estatísticas sobre a população a partir da amostra selecionada. Além disto, procurar evidências suficientes para verificar se há um efeito de tratamento significativo nos fatores de interesse (indicadores objetivos medidos) com os grupos de participantes (principiantes e experientes), com 4 tarefas de teste, graus de dificuldades e conhecimentos diferentes.

O procedimento estatístico mais adequado ao nosso propósito foi o ***Teste t*** (Barbetta, 2008), pois este verifica diferenças entre duas categorias, possibilitando comparações entre médias de dois grupos e verificar se há diferenças significativas. A

hipótese nula de nenhuma diferença das médias aritméticas das populações, isto é,

$$\mathbf{H_0: \mu_P = \mu_E}$$

será testada em relação à alternativa de haver diferenças entre elas

$$\mathbf{H_A: \mu_P \neq \mu_E}$$

Onde: μ_P representa a média do grupo dos principiantes e μ_E representa a média do grupo dos experientes.

O *Teste t* é utilizado para testar a igualdade das médias de dois conjuntos de números. Para isto calcula-se a estatística *t* que é dada pela seguinte fórmula:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\mathrm{SE}_{12}}$$

O denominador é calculado com:

$$\mathrm{SE}_{12} = \mathrm{S}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

onde

$$\mathrm{S}^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}$$

Na equação acima $\bar{x}_1$ e $\bar{x}_2$ são as médias aritméticas dos dois conjuntos de números, s_1 e s_2 são as raízes quadradas das variâncias dos dois conjuntos, e n_1 e n_2 são os números de pontos em cada conjunto.

O procedimento estatístico dos dados foi realizado com auxílio do MS Excel 2003, que consistiu das etapas descritas a seguir, onde foram comparados os valores dos dados dos experientes com principiantes para cada uma das variáveis (tempo, escolhas incorretas, ações incorretas e número de pedidos de ajuda).

A Tabela 6.7. apresenta os resultados do *Teste t* para a variável apresentada na Tabela 6.3 (tempo de execução de tarefas).

Tabela 6.7.: Resultados da aplicação do *Teste t* para as variâncias da Tabela 6.3.

Teste t: duas amostras presumindo variâncias equivalentes		
Indicador - Tempo de execução de tarefas		
α=0,05		
Tarefa 1		
	Principiantes	***Experientes***
Média	3,642857143	2
Variância	1,324175824	0,923077
Observações	14	14
Variância agrupada	1,123626374	
Hipótese da diferença de média	0	
gl	26	
Stat t	4,100509838	
P(T<=t) uni-caudal	0,000179776	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	**0,000359553**	
t crítico bi-caudal	2,055529418	
Tarefa 2		
	Principiantes	***Experientes***
Média	14,71428571	10,92857143
Variância	20,52747253	29,14835165
Observações	14	14
Variância agrupada	24,83791209	
Hipótese da diferença de média	0	
gl	26	
Stat t	2,009737384	
P(T<=t) uni-caudal	0,027471537	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	**0,054943074**	
t crítico bi-caudal	2,055529418	
Tarefa 3		
	Principiantes	***Experientes***
Média	10	8
Variância	7,230769231	4,615385
Observações	14	14
Variância agrupada	5,923076923	
Hipótese da diferença de média	0	
gl	26	
Stat t	2,174229226	
P(T<=t) uni-caudal	0,019488626	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	**0,038977252**	
t crítico bi-caudal	2,055529418	

Tarefa 4		
	Principiantes	***Experientes***
Média	20,13333333	17,06667
Variância	32,83809524	40,49524
Observações	15	15
Variância agrupada	36,66666667	
Hipótese da diferença de média	0	
gl	28	
Stat t	1,386952185	
P(T<=t) uni-caudal	0,088199706	
t crítico uni-caudal	1,701130908	
P(T<=t) bi-caudal	**0,176399412**	
t crítico bi-caudal	2,048407115	

Observa-se na Tabela 6.7 a estatística do teste t com relação ao identificador **tempo de execução de tarefas**. Na tarefa 1 (Modelagem da RA de XP1) o valor t = 4,100509838, é maior que o valor crítico bi-caudal 2,055529418 ($t > t_{crit}$), de modo que a hipótese nula foi rejeitada. E o valor *p* para o teste bicaudal 0,000359553 é menor que 0,05 ($p < \alpha$), conferindo que a hipótese nula pode ser então rejeitada também. Na tarefa 3 (Instanciação da RA) ocorre o mesmo, visto que o valor t = 2,174229226 > 2,055529418, conferindo diferença entre as categorias de usuários e rejeitando a hipótese.

Nas tarefas 2 (Adaptação da RA) e 4 (Totalização e simulação) a hipótese nula foi aceita, pois os valores *stat t* são menores do que o valor *t crítico bi-caudal* ($t < t_{crit}$), e o valor *p* não é menor que 0,05 ($p < \alpha$), isto significa que não houve diferenças significativas nas tarefas 2 e 4 para os usuários principiantes e experientes, pois as duas categorias conseguiram utilizar sem diferenças significativas o ambiente.

Conclui-se então que nas tarefas onde se exige maior conhecimento e experiência por parte dos usuários, as diferenças foram mais significativas entre as categorias (tarefas 1 e 3).

Este mesmo "diagnóstico" se repete nas próximas Tabelas 6.8, 6.9 e 6.10 onde os valores obtidos apresentam similaridade com os valores discutidos nos três parágrafos anteriores com diferenças significativas (de tempo, ações, opções e ajuda) apenas nas tarefas 1 e 3, entre principiantes e experientes, conforme a apresentação resumida da Tabela 6.11.

Tabela 6.8.: Resultados da aplicação do *Teste t* para as variâncias da Tabela 6.4. (indicador objetivo - número de ações incorretas).

Teste t: duas amostras presumindo variâncias equivalentes entre principiantes e experientes		
Indicador - Ações incorretas		
α=0,05		
Tarefa 1		
	Principiantes	***Experientes***
Média	3,642857143	2,071429
Variância	0,708791209	0,840659
Observações	14	14
Variância agrupada	0,774725275	
Hipótese da diferença de média	0	
gl	26	
Stat t	**4,723563987**	
P(T<=t) uni-caudal	0,0000348614	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,0000697229	
t crítico bi-caudal	**2,055529418**	
Tarefa 2		
	Principiantes	***Experientes***
Média	1,571428571	1,214286
Variância	0,571428571	0,796703
Observações	14	14
Variância agrupada	0,684065934	
Hipótese da diferença de média	0	
gl	26	
Stat t	**1,14246264**	
P(T<=t) uni-caudal	0,131836882	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,263673764	
t crítico bi-caudal	**2,055529418**	
Tarefa 3		
	Principiantes	***Experientes***
Média	1,928571429	1
Variância	1,60989011	0,461538
Observações	14	14
Variância agrupada	1,035714286	
Hipótese da diferença de média	0	
gl	26	
Stat t	**2,414039396**	
P(T<=t) uni-caudal	0,01155823	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,02311646	
t crítico bi-caudal	**2,055529418**	

Tarefa 4		
	Principiantes	***Experientes***
Média	1,6	1,066667
Variância	0,542857143	0,638095
Observações	15	15
Variância agrupada	0,59047619	
Hipótese da diferença de média	0	
gl	28	
Stat t	**1,900763853**	
P(T<=t) uni-caudal	0,03383633	
t crítico uni-caudal	1,701130908	
P(T<=t) bi-caudal	0,06767266	
t crítico bi-caudal	**2,048407115**	

Tabela 6.9. : Resultados da aplicação do *Teste t* para as variâncias da Tabela 6.5. (indicador objetivo- número de opções incorretas).

Teste t: duas amostras presumindo variâncias equivalentes		
Indicador - Opções incorretas		
α=0,05		
Tarefa 1		
	Principiantes	***Experientes***
Média	2,285714286	1,42857143
Variância	0,835164835	0,72527473
Observações	14	14
Variância agrupada	0,78021978	
Hipótese da diferença de média	0	
gl	26	
Stat t	**2,567401273**	
P(T<=t) uni-caudal	0,008173679	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,016347358	
t crítico bi-caudal	**2,055529418**	
Tarefa 2		
	Principiantes	***Experientes***
Média	1,142857143	1,071428571
Variância	0,901098901	0,532967033
Observações	14	14
Variância agrupada	0,717032967	
Hipótese da diferença de média	0	
gl	26	
Stat t	**0,223178021**	
P(T<=t) uni-caudal	0,412570558	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,825141116	
t crítico bi-caudal	**2,055529418**	

Tarefa 3		
	Principiantes	***Experientes***
Média	1,928571429	1
Variância	1,302197802	0,615384615
Observações	14	14
Variância agrupada	0,958791209	
Hipótese da diferença de média	0	
gl	26	
Stat t	**2,509009554**	
P(T<=t) uni-caudal	0,009336278	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,018672556	
t crítico bi-caudal	**2,055529418**	

Tarefa 4		
	Principiantes	***Experientes***
Média	1,214285714	0,857143
Variância	0,796703297	0,43956
Observações	14	14
Variância agrupada	0,618131868	
Hipótese da diferença de média	0	
gl	26	
Stat t	**1,201850425**	
P(T<=t) uni-caudal	0,120127199	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,240254398	
t crítico bi-caudal	**2,055529418**	

Tabela 6.10: Resultados da aplicação do *teste t* para as variâncias da Tabela 6.6. (indicador objetivo- número de pedidos de ajuda).

Teste t: duas amostras presumindo variâncias equivalentes		
Indicador – Pedidos de Ajuda		
α=0,05		
Tarefa 1		
	Principiantes	***Experientes***
Média	3,642857143	2
Variância	1,324175824	0,923077
Observações	14	14
Variância agrupada	1,123626374	
Hipótese da diferença de média	0	
gl	26	
Stat t	**4,100509838**	
P(T<=t) uni-caudal	0,000179776	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,000359553	
t crítico bi-caudal	**2,055529418**	
Tarefa 2		
	Principiantes	***Experientes***
Média	1,714285714	1,285714286
Variância	0,527472527	0,373626374
Observações	14	14
Variância agrupada	0,450549451	
Hipótese da diferença de média	0	
gl	26	
Stat t	**1,689277519**	
P(T<=t) uni-caudal	0,051562479	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,103124958	
t crítico bi-caudal	**2,055529418**	
Tarefa 3		
	Principiantes	***Experientes***
Média	1,928571429	1
Variância	1,302197802	0,615385
Observações	14	14
Variância agrupada	0,958791209	
Hipótese da diferença de média	0	
gl	26	
Stat t	**2,509009554**	
P(T<=t) uni-caudal	0,009336278	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,018672556	
t crítico bi-caudal	**2,055529418**	

Tarefa 4		
	Principiantes	***Experientes***
Média	1,428571429	0,857143
Variância	0,879120879	0,593407
Observações	14	14
Variância agrupada	0,736263736	
Hipótese da diferença de média	0	
gl	26	
Stat t	**1,761952784**	
P(T<=t) uni-caudal	0,044916397	
t crítico uni-caudal	1,705617901	
P(T<=t) bi-caudal	0,089832793	
t crítico bi-caudal	**2,055529418**	

Conforme foi comentado, dentre os valores discutidos, as diferenças significativas ocorrem apenas nas tarefas 1 e 3, entre principiantes e experientes, como mostra a Tabela 6.11.

Tabela 6.11: Síntese gráfica dos resultados do procedimento *Teste t* com indicadores objetivos

CATEGORIAS – *Teste t*				
Tarefa	**T_e**	**N_{ai}**	**N_{oi}**	**N_{pa}**
T1	TP > TE	$Np_{ai} > N_{ai}E$	$N_{oip} > N_{oie}$	$N_{pa} > N_{pa}$
T2	TP = TE	$N_{ai} = N_{ai}E$	$N_{oi}P = N_{oi}E$	$N_{pa} = N_{pa}$
T3	TP > TE	$Np_{ai} > N_{ai}E$	$N_{oi}P > N_{oi}E$	$N_{pa} > N_{pa}$
T4	TP = TE	$N_{ai} = N_{ai}E$	$N_{oi}P = N_{oi}E$	$N_{pa} = N_{pa}$

Legenda:
P - principiante
E - experiente

6.3.3. Problemas identificados a partir da mensuração do desempenho do usuário

Os experimentos com os usuários e os registros nas fichas de teste possibilitaram a viabilidade do ambiente proposto nesta tese e a identificação de uma série de falhas e aspectos positivos. Nesta seção serão apresentados e discutidos os problemas identificados a partir da mensuração do desempenho dos usuários e na próxima seção apresentaremos os resultados da sondagem da Satisfação do Usuário com questionários. Na última seção, será apresentado um sumário dos problemas e um parecer sobre o produto, com base nos dois enfoques adotados e recomendações para melhoria.

A seguir são listados os problemas mais críticos identificados.

6.3.3.1. Falta de início automático da simulação

Sobre a simulação e totalização da rede, os usuários cometeram 28 % das ações incorretas, pois não *concluíram* uma *atividade* para em seguida verificar a sua *totalização* e inicializar a próxima atividade. Para amenizar este problema, 20 % de usuários sugeriram através da verbalização, registradas nas fichas de registros de eventos, que deveria existir um *start* automático para início de atividades seguintes àquelas já finalizadas. Outro problema verificado com relação à apresentação dos resultados da simulação diz respeito à sobreposição da tela de resultados (totais reais, parciais e remanescentes) nos elementos da Rede de Atividades que estão sendo simulados. A sobreposição da tela oculta ao usuário a leitura dos elementos que estão próximos (Figura 6.1.) e passíveis de totalização/simulação, gerando 29 % de pedidos de ajuda. Para resolver este problema, a tela de totais poderia aparecer no canto inferior da tela de modelagem, para não tirar a visualização do fluxo da rede.

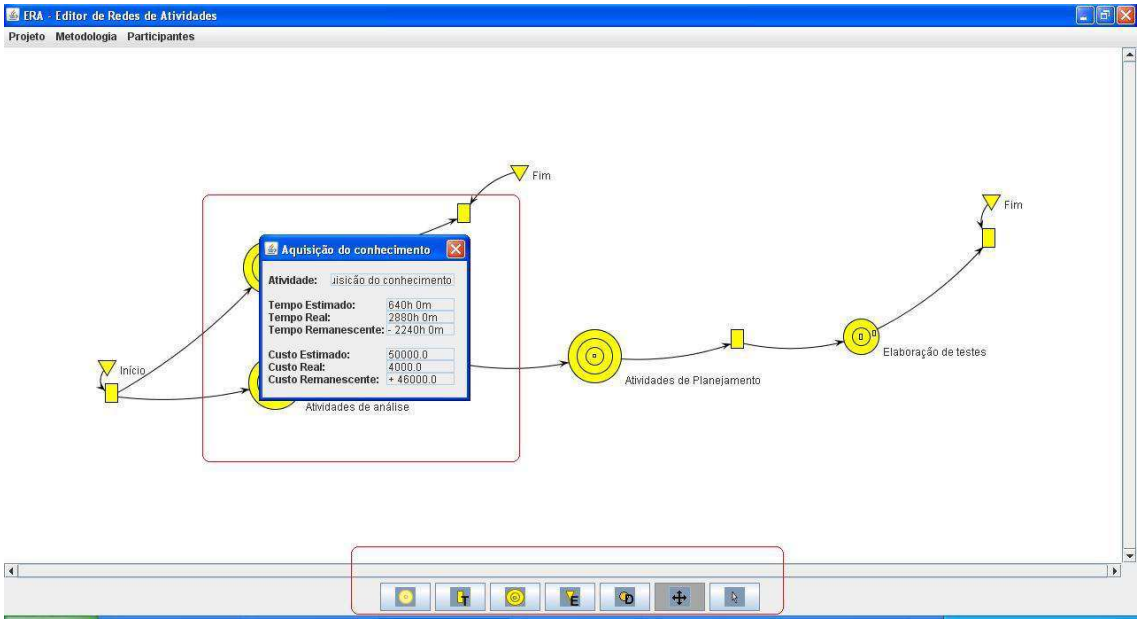


Figura 6.1: Tela de Rede de Atividades com totalização de tempos e custos e ícones representativos

6.3.3.2. Visualização de hierarquia de níveis e sub-níveis

Sobre a visualização da RA, 20% dos usuários verbalizaram que gostariam de visualizar a hierarquia total de níveis e sub-níveis em uma única tela para ver o processo geral. Observa-se que o editor já permite a visualização total de um nível por tela, e com opção de deslocamento para esquerda e direita e *zoom*. Na Figura 6.2. é possível

visualizar o primeiro nível da Rede de Atividades do XP1, e na Figura 6.3. é que pode ser visto o segundo nível (Sub-rede).

Para visualização de todos os níveis em uma única tela, poderia ser incluída uma sobreposição de nível na tela, de acordo com a seleção do usuário, ao clicar na atividade desejada.

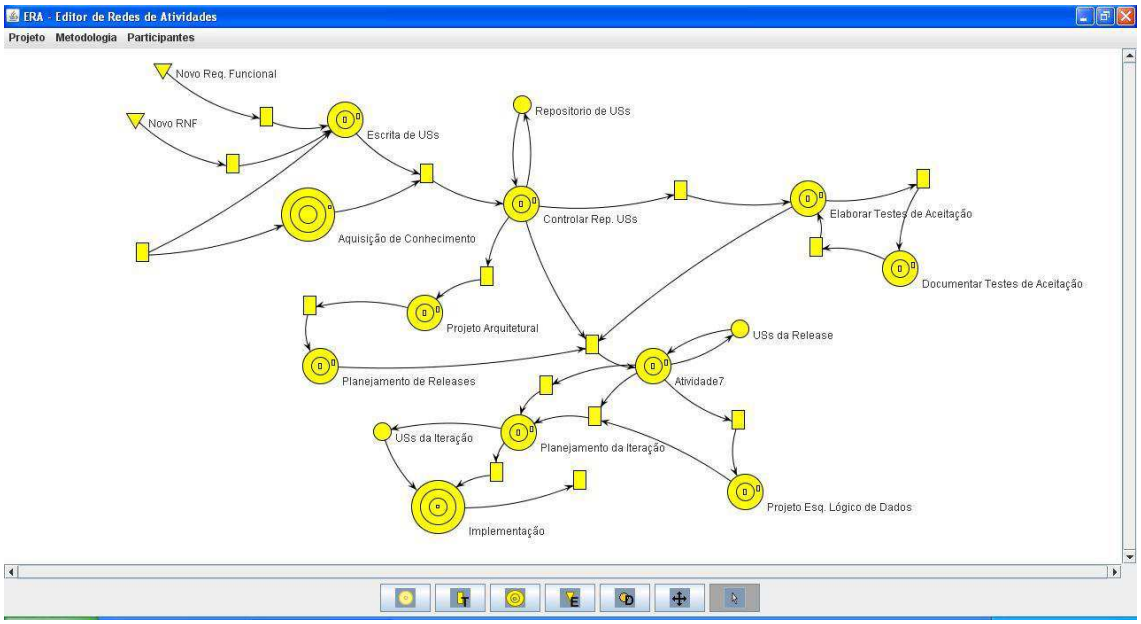


Figura 6.2: Tela de Rede de Atividades com visualização do primeiro nível da rede de XP1

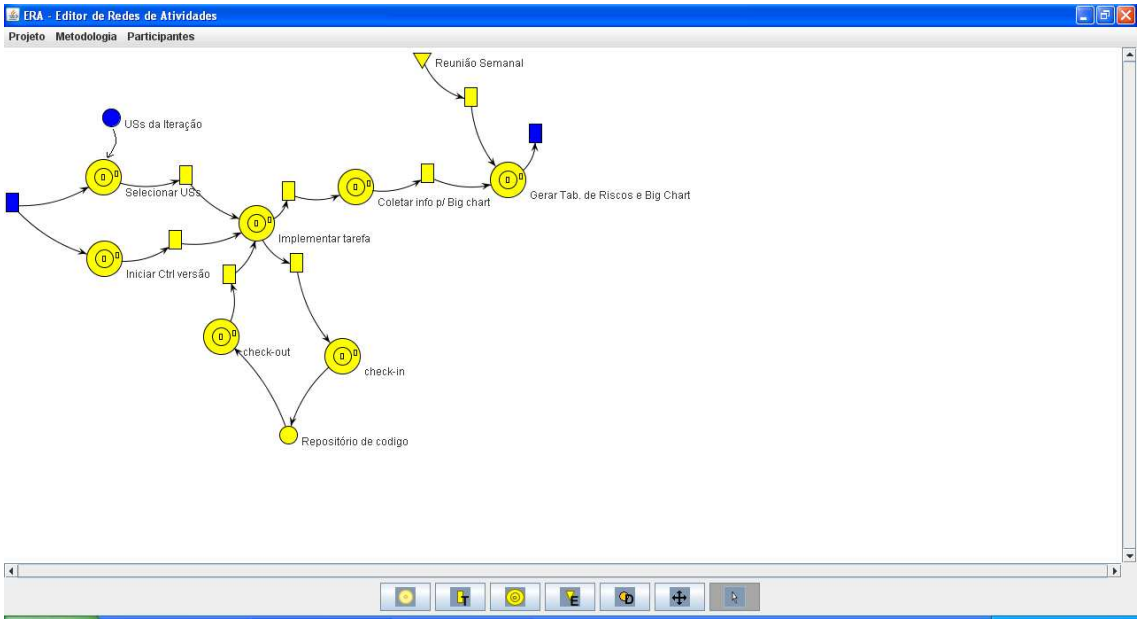


Figura 6.3: Tela de Rede de Atividades com visualização do segundo nível da rede de XP1.

6.3.3.4. Ausência de Atalhos

Não existe nenhuma forma para encurtar os passos de atividades repetitivas. As

falhas identificadas dizem respeito à falta de algumas teclas de atalho por exemplo, CONTROL+X, CONTROL+C, CONTROL+V e CONTROL + Z, para desfazer ações incorretas ou deleções impróprias na modelagem de elementos da Rede de Atividades, isto acarretou 24 % das opções incorretas. Outra questão levantada foi com relação à falta do funcionamento da tecla DELETE para apagar elementos inseridos por engano na rede, isto resultou em 31 % das ações incorretas. Observa-se que a área de ícones de elementos da RA (situados na parte inferior central da tela da Figura 6.2.) poderia conter o ícone representando *aresta* de ligação entre elementos da rede.

Outra falha é falta da funcionalidade de clique duplo em um elemento para editar suas propriedades (nome, ou atributos) foi verificado por 25 % dos usuários, resultando no total de 27 % das ações incorretas, resultantes de um hábito próprio de ferramentas com interface Windows.

6.3.3.5. Ausência de alguns ícones e *hints (dicas)*

Mais alguns ícones são necessários para deixar a interface mais usável, por exemplo: Ícone para *zoom*, opção já existente através do botão direito do mouse, mas que foi sentida a falta por 22 % dos usuários.

A primeira dificuldade encontrada diz respeito às tentativas de ligação de elemento *atividade* para outra *atividade.* 17 % dos usuários (5 de 30) tentaram conectar os elementos sem colocar o elemento de *transição* entre eles.

Observou-se que é necessário um ícone no editor de Rede de Atividade para inserção de setas entre elementos. Após verificação da impossibilidade da ação (o software não permite esta falta), 33% (9 de 30) dos usuários realizaram pedidos de ajuda para saber se existia um ícone no editor de Rede de Atividade para inserção de setas entre os elementos.

Hints ou descrições (dicas) para os ícones também foram sugeridos por 29% dos usuários.

6.3.3.6. Falta de clareza nas mensagens de gravação

As mensagens de erro e de gravação (Figura 6.4.) no software editor de RA –confundem no momento do usuário gravar a RA da metodologia modelada. Elas estão associadas à pelo menos, 43 % de ações incorretas na tarefa 1 e

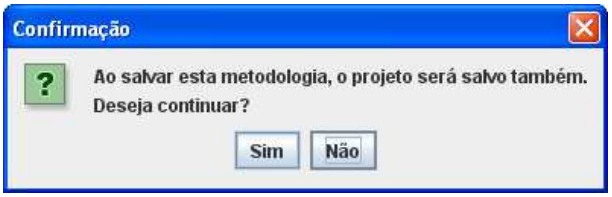


Figura 6.4: Tela de mensagem de gravação de metodologia modelada

39 % na tarefa 2 pelos usuários.

6.3.3.7. Falta de instruções *on line*

Outra falha trata da inserção da data de início e fim de atividade (Figura 6.5.) pois não existe um auxílio na tela (*help* ou instrução *on line*) descrevendo como deve ser a formatação da data (exemplo: dd/mm/aaaa) e do custo (Figura 6.6.). Esta falha fez com que 35% dos participantes cometessem a mesma ação incorreta pelo menos uma vez (cada usuário) e pediram ajuda pelo menos uma vez durante a tarefa 3.

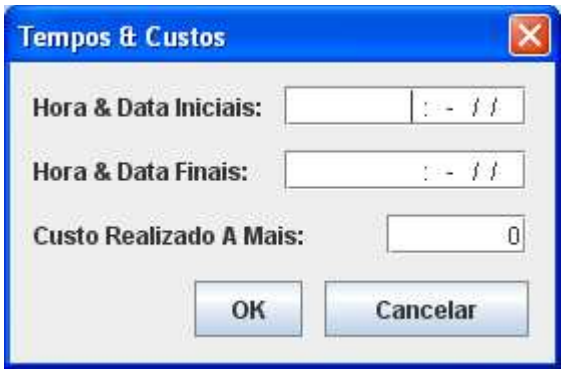


Figura 6.5: Formulário para digitação de horários, datas e custos extras

6.3.3.8. *Layout* com cores na tela

O software editor de RA proporciona que os elementos selecionados fiquem com outra cor (vermelha) e os elementos iniciais e finais de uma sub-rede são apresentados automaticamente na cor azul (Figura 6.3.). Alguns usuários (26%) associaram isto ao fato de que as atividades já realizadas e concluídas também poderiam contar com este "artifício" de mudança de cores de acordo com a conclusão da atividade.

A mensuração do desempenho do usuário através do ensaio de usabilidade demonstrou que apesar dos recursos e regras propostos na Rede de Atividades para modelagem de processos e atividades para solução de diversos tipos de problemas relacionados a tempos, custos e pessoal, propiciarem uma bom uso, o software editor apresenta uma série de falhas na interface (anteriormente citadas), as quais merecem ser convenientemente analisadas e corrigidas.

6.4. Sondagem da Opinião do Usuário

A sondagem da satisfação do usuário foi realizada através da aplicação de dois instrumentos: questionário de sondagem do perfil do usuário aplicado antes do experimento e OPUS – opinião do usuário aplicado após o teste.

6.4.1. Resultados da Análise do Perfil do Usuário

A aplicação do formulário de sondagem do perfil do usuário (Anexo IV) levantou

o perfil participante do experimento, buscando estabelecer sua relação com o perfil estabelecido para o universo amostral e visando estabelecer a categoria de inserção do participante.

Os Quadros 1 e 2 no Anexo 5apresentam as respostas dos usuários de teste aos itens do questionário para delineamento do perfil do usuário de teste.

Perfil dos Participantes

Verificou-se entre os usuários uma incidência de 76,6% de estudantes, dos quais, 50% indivíduos (15 de 30) são alunos do curso de graduação em computação ou licenciatura em computação, 10% são alunos de mestrado (3 de 30), 16,6 % são alunos de doutorado (5 de 30). Dentre eles, 13,3% são graduados (4 de 30), e 10% são mestres (3 de 30). Como demonstra a Figura 6.2, a maioria dos participantes ou 70 % (21 de 30) é do sexo masculino e 30 % do sexo feminino (9 de 30).

A maioria dos participantes (40%) tem experiência do desenvolvimento de software no âmbito industrial (12 de 30), seguidos por 36,6 % (11 de 30) usuários que possuem experiência de desenvolvimento de software como parte de disciplinas de curso de graduação.

Quanto à experiência com gerência de projetos de software, 33,33% foi líder em poucos projetos (menos de três) na indústria de desenvolvimento de software (10 de 30), 26,66 % nunca teve experiência com projetos de software (8 de 30), 23,3% apenas foi líder de projetos em disciplinas de curso (7 de 30) e 16,66 % foi líder em muito projetos na indústria (5 de 30).

No que se refere à experiência com processos de software, de acordo com a Figura 6.5, 43,33% dos usuários (13 de 30) não possuía nenhuma experiência (os usuários principiantes), 13,33% dos usuários havia estudado um processo (4 de 30), 33,3% dos usuários (10 de 30) havia utilizado processos na indústria de software, e 10 % (3 de 30) havia utilizado processos no ambiente industrial.

46,6% dos usuários não tinham experiência com projetos anteriores (14 de 30), seguida por 33, 3% usuários (10 de 30) que havia participado de menos de três projetos e apenas 20% deles (6 de 30) havia participado de mais de três projetos.

Com relação ao desenvolvimento de software orientado a objetos, 43,33% dos usuários desenvolvem, 23,33% já utilizam na indústria (7 de 30), 13,33% ou (4 de 30) já utilizaram em vários projetos (mais de 3) na indústria de software, 16,66% possuem conhecimento (5 de 30) adquirido através de cursos ou livros e apenas 3,3% ou (1 de 30) não tem nenhum conhecimento.

Dentre os usuários, a maioria 83,33% não possui conhecimento (25 de 30) de Rede PERT para auxílio no gerenciamento de projetos. Apenas 16,66% ou 5 de 30, dentre experientes e principiantes, conhece esta rede através de cursos ou livros.

A maioria dos usuários (73,33% ou 22 indivíduos) não possui nenhum conhecimento de rede de Petri. 23,33% deles viu o assunto em disciplina de curso de graduação e 1 usuário (3,3 %) desenvolveu projeto de disciplina nesta área.

Com relação à experiência com definição de tempos e custos para projetos. 10 usuários ou 33,3% não possuem nenhuma experiência, 7 usuários leram sobre isto em livros ou disciplinas de curso, 10% (3 de 30) realizaram algum tipo de previsão em projetos acadêmicos e 9 usuários 30% já estimaram tempos e custos em projetos de software.

6.4.2. Resultados da análise dos questionários pós-teste

Nesta seção serão apresentados os dados resultantes da sondagem da opinião do usuário de software após os experimentos. A Tabela 6.12. mostra a totalização das respostas dos usuários de teste.

No primeiro bloco, *Você e a Rede de Atividades*, com 8 itens, as opções de resposta estão estruturadas em duas escalas semântico-numéricas de 5 pontos – concordo totalmente, concordo parcialmente, nem concordo nem discordo, discordo parcialmente, discordo totalmente. No segundo bloco Você e o Software editor de Redes de Atividades – ERA, permanece a mesma estrutura semântico-numérica original: muito fácil, fácil, nem fácil nem difícil, difícil, muito difícil. A esta configuração foi atribuído +1 ponto, e quando desfavorável -1 ponto.

De acordo com (Queiroz, 2001) "...o método da somatória de valores (*method of summated ratings*), concebido por Likert (1932, *apud* Aiken, 1996) para uma escala típica de 5 pontos (escala adotada no OPUS), as respostas expressas positivamente devem ser computadas como segue: 0 para a posição mais desfavorável da escala (e.g., discordo totalmente e muito difícil), 1 para a posição parcialmente desfavorável (e.g. discordo parcialmente e difícil), 2 para a posição central (e.g. nem concordo nem discordo e nem fácil nem difícil), 3 para a posição parcialmente favorável (e.g. concordo parcialmente e fácil) e 4 para a posição mais favorável (e.g. concordo totalmente e muito fácil)." Este método também foi utilizado neste experimento.

Foram adicionados á estrutura do OPUS duas colunas: ESCORE BRUTO, contendo os resultados não ponderados da computação de cada item e as faixas de escores, às quais foram atribuídos os referenciais semânticos BAIXO (0-40), RAZOÁVEL (41-80), BOM (81-120) E MUITO BOM (121-160), conforme a legenda apresentada no canto superior direito da Tabela 6.10. e ESCORE PONDERADO. Ao final de cada seção do OPUS são apresentados os escores esperados (mínimo (0), médio (50) e máximo (100)) e os escores brutos e ponderados obtidos por seção.

É possível visualizar de forma detalhada a opinião dos respondentes do OPUS sobre cada tema investigado, incluindo o uso da Rede de Atividades e do aplicativo Editor de RA – ERA. Tanto na seção 1, com escore total **104** que trata dos aspectos gerais da RA, como na seção 2, com escore total **99,55**, que verifica a opinião do usuário quanto ao software Editor de RA, quanto a seção 3, com escore total de **87,4**, que trata de uso e navegação no software, o escore total foi considerado como **BOM**, demonstrando que os respondentes foram favoráveis aos aspectos investigados no processo avaliatório.

Tabela 6.12: Síntese dos resultados da sondagem do universo amostral com o questionário OPUS

	ASPECTO	Escala											
		1		2		3		4		5			
		Concordo totalmente		Concordo parcialmente		Nem concordo nem discordo		Discordo parcialmente		Discordo totalmente		ESCORE BRUTO	
1	A utilização da Rede de Atividades é adequada à modelagem do processo utilizado.	17	56,67%	13	43,33%	0	0,00%	0	0,00%	0	6,67%	107	
2	Os elementos que compõem a rede são suficientes para modelar o processo adotado.	15	50,00%	10	33,33%	1	3,33%	4	13,33%	0	0,00%	96	
3	Não existiu dificuldade na representação das etapas e sub-etapas do processo.	10	33,33%	18	60,00%	2	6,67%	0	0,00%	0	0,00%	98	
4	Através da seqüência de atividades representadas, é possível acompanhar as etapas do processo modelado e verificar quais etapas e atividades a seguir.	18	60,00%	11	36,67%	1	3,33%	0	0,00%	0	0,00%	107	
5	A visualização do processo através do grafo (da RA) é muito útil durante o processo de modelagem.	22	73,33%	7	23,33%	1	3,33%	0	0,00%	0	0,00%	111	
6	A visualização do processo através do grafo (rede) é muito útil durante o acompanhamento do processo.	21	70,00%	7	23,33%	2	6,67%	0	0,00%	0	0,00%	109	
7	Eu utilizaria a Rede de Atividades novamente para descrever as atividades do processo de desenvolvimento.	27	90,00%	0	0,00%	2	6,67%	1	3,33%	0	0,00%	113	
8	O treinamento aplicado para o uso da rede foi suficiente.	11	36,67%	12	40,00%	4	13,33%	3	10,00%	0	0,00%	81	

		Escores esperados			ESCORE BRUTO (seção 1)	Média ESCORE PONDERADO (seção 1)
		MÍNIMO	MÉDIO	MÁXIMO		
		0	50	100	822	104

Você e o Software Editor de Redes de Atividades - ERA													
ASPECTO		Escala											
		1		2		3		4		5			
		Concordo totalmente		Concordo parcialmente		Nem concordo nem discordo		Discordo parcialmente		Discordo totalmente		ESCORE BRUTO	
9	**Com a utilização do editor se obtém automatização na modelagem da RA.**	14	46,67%	15	50,00%	1	3,33%	0	0,00%	0	0,00%	103	
10	**Não existem dificuldades em interpretar os ícones que representam os elementos da RA.**	9	30,00%	20	66,67%	0	0,00%	1	3,33%	0	0,00%	97	
11	**Não existiu dificuldade na adaptação de etapas da RA modelada inicialmente.**	12	40,00%	15	50,00%	2	6,67%	1	3,33%	0	0,00%	98	
12	**Não existiu dificuldade na instanciação de atividades e sub-atividades do projeto na RA.**	14	46,67%	8	26,67%	2	6,67%	5	16,67%	1	3,33%	89	
13	**Não existiu dificuldade na inclusão de valores (tempo e custo) do projeto para a RA.**	17	56,67%	10	33,33%		0,00%	3	10,00%	0	0,00%	101	
14	**Não existiu dificuldade na simulação de totais (tempo e custo) do projeto.**	13	43,33%	12	40,00%	2	6,67%	3	10,00%	0	0,00%	95	
15	**Eu utilizaria o módulo Editor de RAs novamente para modelar um processo de desenvolvimento.**	18	60,00%	9	30,00%	1	3,33%	2	6,67%	0	0,00%	103	
16	**Eu utilizaria o módulo Editor de RAs novamente para acompanhar meu processo de desenvolvimento.**	16	53,33%	13	43,33%	1	3,33%	0	0,00%	0	0,00%	105	
17	**Eu utilizaria novamente o módulo Editor de RA para dar suporte aos desenvolvedores e suas atividades.**	19	63,33%	10	33,33%	1	3,33%	0,00%	0,00%	0	0,00%	108	
18	**Eu utilizaria o módulo Editor de RAs novamente para gerenciamento de projetos.**	15	50,00%	14	46,67%	1	3,33%	0	0,00%	0	0,00%	104	

19	**O treinamento aplicado para o uso do software foi suficiente.**	14	46,67%	9	30,00%	2	6,67%	5	16,67%	0	0,00%	92	

	Escores esperados			**ESCORE BRUTO (seção 2)**	**Média ESCORE PONDERADO (seção 2)**
	MÍNIMO	**MÉDIO**	**MÁXIMO**		
	0	**50**	**100**	**1095**	**99,55**

Uso e Navegação do Software													
ASPECTO		**Escala**											
		1		**2**		**3**		**4**		**5**			
		Muito fácil		**Fácil**		**Nem fácil nem difícil**		**Difícil**		**Muito difícil**		**ESCORE BRUTO**	
20	**Uso do software na realização das tarefas de interesse.**	5	16,67%	24	80,00%	1	3,33%	0	0,00%	0	0,00%	94	
21	**Localização dos itens de menu associados às tarefas.**	11	36,67%	12	40,00%	3	10,00%	4	13,33%	0	0,00%	90	
22	**Uso das funcionalidades mais comuns encontradas no software.**	6	20,00%	21	70,00%	3	10,00%	0	0,00%	0	0,00%	93	
23	**Compreensão das mensagens de erro apresentadas e recuperação destas situações.**	3	10,00%	11	36,67%	14	46,67%	2	6,67%	0	0,00%	75	
24	**Processo de entrada e saída de dados durante o uso do software.**	7	23,33%	12	40,00%	10	33,33%	1	3,33%		0,00%	85	

Legenda:

Baixo 0-40

Razoável 41-80

Bom 81-120

Escores esperados			**ESCORE BRUTO (seção 3)**	**Média ESCORE PONDERADO (seção 3)**
MÍNIMO	**MÉDIO**	**MÁXIMO**		
0	**50**	**100**	**437**	**87,4**

Tabela 6.12: Síntese dos resultados da sondagem do universo amostral com o questionário OPUS

6.4.3. Análise Preliminar dos Escores Obtidos

Como se pode observar na Tabela 6.11., a listagem dos escores por seção e por categoria do ambiente avaliado possibilita a visualização mais detalhada da opinião dos respondentes do OPUS sobre cada um dos temas investigados: a relação do usuário com a Rede de Atividades, a relação do usuário com o software editor de Redes de Atividades – ERA e o uso e navegação do usuário no software.

A análise dos resultados obtidos por seção para cada, por categoria de usuário, resulta na constatação de que os escores relativos às categorias principiantes (76,73) e experientes (79,87) têm valores muito próximos. Esta ordenação se mantém em relação aos escores globais ponderados das categorias consideradas.

Na seção 1 (Você e a Rede de Atividades), verifica-se que os escores relativos às categorias principiantes (26,47) e experientes (27,87) são praticamente iguais e crescentes, apesar dos usuários principiantes terem apresentado tempo de execução de tarefas maior, número de ações incorretas, de escolhas incorretas e pedidos de ajuda maiores.

Na seção 2 (Você e o Software Editor de Redes de Atividades), verificam-se também escores relativos bem próximos entre principiantes (36,13) e experientes (37,00).

Na seção 3 (Uso e Navegação no Software), os escores de principiantes (14,13) e experientes (15,00) foram praticamente iguais entre si.

Conforme se verificou com estes resultados, os usuários das categorias conseguem utilizar o ambiente e obtiveram pequenas diferenças de médias nas respostas obtidas na sondagem da satisfação.Estes resultados são consonantes com aqueles obtidos a partir da mensuração do desempenho do usuário durante o uso do produto, exceto quando as tarefas exigem mais experiência da parte dele. Em suma, as categorias de principiantes (76,73) e experientes (79,87) se mostraram, favoráveis ao ambiente em questão, conforme se observa na última linha da Tabela 6.13.

6.4.4. Análise Estatística dos Indicadores Quantitativos do OPUS

Similarmente à análise dos indicadores objetivos registrados nos testes de usabilidade realizada no item 6.3.2., a análise da Tabela 6.13. apresentada na sub-seção anterior teve o propósito de verificar diferenças entre os escores registrados em função das categorias de usuários do instrumento de sondagem utilizado nesta pesquisa (questionário OPUS). Nesta seção, os dados da tabela 6.13. são utilizados no *Teste t* para verificar se as diferenças registradas são estatisticamente aceitáveis. Neste caso, as variáveis analisadas foram o escore por seção e por categoria de usuário.

Tabela 6.13: Síntese da comparação dos itens do questionário OPUS, por seção e por categoria de usuário.

ESCORES DA SEÇÃO 1						
	Escore Principiantes		Escore Experientes		TOTAL	
Item	bruto	ponderado	bruto	ponderado	BRUTO	PONDERADO
1	54	3,60	53	3,53	107	3,57
2	43	2,87	53	3,53	96	3,20
3	44	2,93	49	3,27	93	3,10
4	52	3,47	46	3,07	98	3,27
5	55	3,67	53	3,53	108	3,60
6	53	3,53	56	3,73	109	3,63
7	55	3,67	58	3,87	113	3,77
8	41	2,73	50	3,33	91	3,03
TOTAIS	**397**	**26,47**	**418**	**27,87**	**815**	**27,17**
	ESCORES DA SEÇÃO 2					
	Escore Principiantes		Escore Experientes		TOTAL	
Item	bruto	ponderado	bruto	ponderado	BRUTO	PONDERADO
9	43	2,87	52	3,47	95	3,17
10	51	3,40	49	3,27	100	3,33
11	47	3,13	51	3,40	98	3,27
12	51	3,40	46	3,07	97	3,23
13	54	3,60	50	3,33	104	3,47
14	56	3,73	48	3,20	104	3,47
15	55	3,67	52	3,47	107	3,57
16	37	2,47	51	3,40	88	2,93
17	56	3,73	52	3,47	108	3,60
18	55	3,67	49	3,27	104	3,47
19	37	2,47	55	3,67	92	3,07
TOTAIS	**542**	**36,13**	**555**	**37,00**	**1097**	**33,40**
	ESCORES DA SEÇÃO 3					
	Escore Principiantes		Escore Experientes		TOTAL	
Item	bruto	ponderado	bruto	ponderado	BRUTO	PONDERADO
20	47	3,13	47	3,13	94	3,13
21	43	2,87	47	3,13	90	3,00
22	45	3,00	48	3,20	93	3,10
23	37	2,47	38	2,53	75	2,50
24	40	2,67	45	3,00	85	2,83
TOTAIS	**212**	**14,13**	**225**	**15,00**	**437**	**14,57**
	SOMATÓRIO DOS ESCORES DAS 3 SEÇÕES					
	Escore Principiantes		Escore Experientes		TOTAL	
	bruto	ponderado	bruto	ponderado	BRUTO	PONDERADO
	1151	**76,73**	**1198**	**79,87**	**2349**	**75,13**

A Tabela 6.14. ilustra os resultados obtidos com os *Testes t* com α = 0,05, realizado sobre os escores de cada seção do questionário pós-teste. Conforme pode ser observado a partir dos resultados dos testes das três seções do questionário pós-teste, os valores *t* não são maiores do que os valores t crítico bi-caudal, ($t < t_{crit}$), e os valores p são maiores do que α =0,05. Pode-se observar isto visto que não existe diferença estatística significante nas respostas dos questionários dos dois grupos participantes do experimento.

Tabela 6.14: Resultados da aplicação do *Teste t* para as variâncias do questionário OPUS

Teste-t: duas amostras presumindo variâncias equivalentes		
α =0,05		
	Principiantes	*Experientes*
Média	47,69565217	49,7826087
Variância	45,22134387	17,54150198
Observações	23	23
Variância agrupada	31,38142292	
Hipótese da diferença de média	0	
gl	44	
Stat t	**-1,263356751**	
P(T<=t) uni-caudal	0,106557171	
t crítico uni-caudal	1,680229977	
P(T<=t) bi-caudal	**0,213114341**	
t crítico bi-caudal	**2,015367547**	

Na Tabela 6.15. é visto que o procedimento de teste para as seções 1, 2 e 3 tem a hipótese nula aceita pois os valores *p* bi-caudal não são menores do que o valor de α (α = 0,213114341 > 0,05).

Os resultados referem-se às categorias de usuário para as quais as médias dos escores de usuários experientes são maiores do que as médias dos usuários principiantes, e apesar de se apresentarem com valores médios absolutos maiores, não são significativamente diferentes. Isto vai de acordo com os resultados obtidos a partir da mensuração de desempenho em alguns pontos descritos na seção 6.3.2., tais como avaliação com indicadores de tempo, ação, opção e ajuda nas tarefas que não exigem grande experiência do usuário (tarefas 2 e 4). E contradiz os resultados da mensuração das tarefas 1 e 3 que exigem mais experiência. Este fato reforça a vantagem de se usar dois métodos de avaliação diferentes, devido às respostas dos usuários através de questionários algumas vezes tenderem ao lado positivo. Por isso é bom usar dois métodos- desta maneira se obtém uma comparação dos métodos usados.

Tabela 6.15. - Resultados da aplicação do TESTE-T para as variâncias do questionário OPUS

Teste-t: duas amostras presumindo variâncias equivalentes		
Seção 1	*Principiantes*	*Experientes*
Média	49	52,14285714
Variância	37	17,14285714
Observações	7	7
Variância agrupada	27,07142857	
Hipótese da diferença de média	0	
gl	12	
Stat t	-1,130064093	
P(T<=t) uni-caudal	0,140265723	
t crítico uni-caudal	1,782287548	
P(T<=t) bi-caudal	**0,280531446**	
t crítico bi-caudal	2,178812827	
Seção 2		
	Principiantes	*Experientes*
Média	49,9	50,3
Variância	54,1	6,233333333
Observações	10	10
Variância agrupada	30,16666667	
Hipótese da diferença de média	0	
gl	18	
Stat t	-0,162847588	
P(T<=t) uni-caudal	0,436226373	
t crítico uni-caudal	1,734063592	
P(T<=t) bi-caudal	**0,872452747**	
t crítico bi-caudal	2,100922037	
Seção 3		
	Principiantes	*Experientes*
Média	41,25	44,5
Variância	12,25	20,33333333
Observações	4	4
Variância agrupada	16,29166667	
Hipótese da diferença de média	0	
gl	6	
Stat t	-1,138716467	
P(T<=t) uni-caudal	0,149117958	
t crítico uni-caudal	1,943180274	
P(T<=t) bi-caudal	**0,298235916**	
t crítico bi-caudal	2,446911846	

6.4.5. Problemas identificados a partir da sondagem da Satisfação do Usuário

Do mesmo modo como se procedeu na seção 6.3, serão apresentados e discutidos ao longo desta seção os problemas evidenciados a partir do questionário OPUS. Ao final desta sub-seção será apresentada uma descrição dos problemas e um parecer sobre o ambiente com base no enfoque avaliatório tratado nesta seção.

A partir dos escores dos aspectos sondados na Tabela 6.12, a Tabela 6.16. apresenta uma lista de aspectos do processo interativo usuário-produto que apresentaram problemas, sob o ponto de vista dos respondentes do OPUS.

Tabela 6.16. Sumário de problemas e comentários sobre o ambiente a partir da sondagem da satisfação (questionário OPUS).

Problema	Item do OPUS	Solução	Escore do item
1	3	Revisar aspectos da comunicação usuário-produto, tais como linguagem, retorno da informação e das ações em geral, a fim de aperfeiçoar o processo interativo.	90
2	14	Revisar o mecanismo de acesso ao início da funcionalidade de *simulação*.	95
3	8 19	Inserir um tipo de ajuda *on-line* e *off-line* a fim de tornar mais fácil a superação de dúvidas e problemas encontrados pelo usuário durante o uso do produto.	91 92
4	23	Revisar as mensagens de erro apresentadas e redigí-las em estilo mais claro.	75
5	24	Revisar a estruturação a forma de apresentação adotada na entrada e saída do software, a fim de tornar mais fácil sua compreensão pelo usuário.	85

O sumário de problemas da Tabela 6.16 se fundamentou no cruzamento das informações das Tabelas 6.12. e 6.13.

A última coluna da Tabela 6.13.mostra os escores brutos dos itens comentados do OPUS, assim como a classificação semântico-gráfico interpretável a partir da legenda inclusa na Tabela 6.12. É importante observar que as faixas 0-40,41-80, 81-120 e 121-160 da legenda, associadas aos escores brutos dos itens do OPUS, correspondem às faixas ponderadas 0–25, 26-50, 51-75 e 76 – 100. Assim, o escore global ponderado 75,13 corresponde à classificação semântica BOM da legenda.

Também foi entregue aos participantes um formulário de sondagem com respostas subjetivas, de caráter opcional de preenchimento. Os comentários envolvem questões particulares da Rede de Atividades e questões relativas ao uso ou falta de funcionalidades no software editor de RA (falta de botão DELETE, CONTROL+Z, etc.) alguns destes já descritos e exemplificados anteriormente na seção 6.4.

Os principais comentários obtidos a partir de sondagem da satisfação da rede de Atividades, através de formulário pós-teste (questões abertas) estão descritos na Tabela 6.17

Tabela 6.17. - Sumário de comentários sobre o ambiente

Item	Comentário	Quantidade de respostas	
		Principiantes	Experientes
1) Você sentiu falta de algum elemento da RA para modelar o processo adotado ?	"A única dificuldade que senti foi de uma forma de distinguir o retorno de um *loop*."	2	1
	"Senti falta de um elemento com várias atividades sem sub-rede."		3
	"Deveria existir um ícone para representar uma transição (aresta+transição+aresta) para conectar duas atividades automaticamente."	1	3
2) Você considera que existiu dificuldade na representação das etapas e sub-etapas do processo adotado com a RA ?	"A única dificuldade foi de interpretação das etapas e sub-etapas da metodologia de desenvolvimento."	5	
	"Não houve dificuldades em relação aos elementos da RA, mas poderia haver melhorias de usabilidade no software."		1
3) Existiu dificuldade na instanciação de das atividades, recursos, tempos e custos na RA ?	"Poderia colocar custos fixos para determinadas etapas do processo para obter maior agilidade na edição de custos."		1
4) Você sentiu falta de algo para visualização das informações apresentadas na rede sobre o projeto (atividades, recursos, tempos e custos) ?	"Creio que deve ser pensado algo para representar visualmente tarefas que são feitas inúmeras vezes, por exemplo descrever *user stories*, *tasks*, etc."		3
	"Poderia incluir visualização de tempos e custos por desenvolvedor, além de ser por atividade"		2
5) Você considera que existiu que existiu alguma dificuldade na simulação dos tempos e custos?	"Poderia haver um modo de simulação onde o clique duplo sobre uma transição a dispare."		1
	"Poderia colocar controle de riscos, e observações acerca de insucessos."		1

Na Tabela 6.15. observa-se que de acordo com os comentários dos usuários, a Rede de Atividades em si não necessita de muitos ajustes. Mas a interface do software editor ainda precisa ser

re-analisada de acordo com os problemas listados nos resultados deste experimento para ficar otimizada e de acordo com as opiniões de usuários de projetos e processos de software.

6.5. Comparação dos Resultados Apresentados

Esta última seção apresenta uma síntese dos resultados dos dois enfoques avaliatórios adotados nesta pesquisa, apresentados e discutidos nas seções (6.3. Mensuração do Desempenho) e 6.5. (Sondagem da Satisfação do Usuário).

Para melhor visualizar os resultados dos enfoques avaliatórios, os temas serão dispostos com:

a) base na estrutura de itens do OPUS;

b) os dados da mensuração de desempenho;

c) a sondagem da satisfação do usuário (concordo totalmente, concordo parcialmente, nem concordo nem discordo, discordo parcialmente e discordo totalmente / muito fácil, fácil, nem fácil nem difícil, difícil e muito difícil).

Ao final de cada tema serão comentados os resultados da confrontação dos enfoques avaliatórios, apontados os problemas e descritas sugestões de melhoria.

6.5.1. Uso da Rede de Atividades

Aspectos de modelagem e visualização

1) A utilização da rede de Atividades é adequada à modelagem do processo utilizado

- A tabela 6.10 (totalização do OPUS) apresenta a quantidade de usuários 17 (56,67%) que concordaram totalmente no uso da Rede de Atividades para modelagem, 13 usuários (43,33%) concordaram e nenhum discordou.
- Os dados analisados da mensuração do desempenho em relação à utilização da Rede de Atividades para modelagem de processos foram extraídos da Tabela 6.4. que apresenta uma quantidade média de apenas 1,93 ações incorretas dos usuários experientes, e 3,60 ações incorretas de usuários principiantes na tarefa 1 (modelagem do processo XP1).

  Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existe um aspecto que precisa ser levada em consideração: para modelar uma Rede de Atividades é preciso ter um conhecimento prévio sobre o processo a ser utilizado.

2) Os elementos que compõem a rede são suficientes para modelar o processo adotado

- Os dados do OPUS mostram que 15 usuários (50 %) concordaram totalmente que os elementos que compõem a rede são suficientes para modelar e mais 10 usuários (33,3 %) concordaram parcialmente no sentido de que os elementos são da rede são suficientes para modelagem do processo XP1, apenas 1 usuário de 30 discordou, e 4 discordaram totalmente.
- Na mensuração de desempenho, foram totalizados no máximo 3 opções incorretas de seleção de elementos para modelagem na tarefa 1 dentre os usuários experientes e 5 opções incorretas dos usuários principiantes.

Os resultados dos dois enfoques avaliatórios adotados são convergentes.

Problemas identificados: não foram detectados problemas significativos que possam ser mencionados.

3) Não existiu dificuldade na representação das etapas e sub-etapas do processo.

- A partir dos questionários foi verificado que 10 usuários (3,3%) concordaram totalmente, 18 (60 %) concordaram e apenas 2 usuários ficaram neutros.
- Na técnica de mensuração de desempenho foi observado que houve uma média de apenas 2 pedidos de ajuda na tarefa 1 de usuários experientes e 3,53 pedidos dos principiantes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são divergentes, visto que existem duas dificuldades que precisam ser levadas em consideração:

Problemas identificados: a) através da verbalização informal de ações, foi comentado por 3 usuários que sentiram "...falta de um elemento com várias atividades sem sub-rede." A modelagem de tarefas repetitivas do processo e com grande número de sub-itens (exemplo 300 *user-stories*) tornaria necessária a adoção de um artifício no software editor de RAs para agilizar a modelagem.

Apesar dos usuários (63,3%) terem respondido que não existiu dificuldade na representação das etapas e sub-etapas do processo, o tempo médio de execução de tarefas por usuário principiante (79,67min.) excedeu o tempo previsto algumas vezes e foi bem maior com relação aos usuários experientes (28,60min.) conforme já havíamos previsto antes da realização dos experimentos.

Entre os participantes principiantes, com pouco conhecimento do processo XP1, 5 tentaram modelar a rede, e não conseguiram distinguir as várias etapas e sub-etapas. A indicação de ajuda foi de re-leitura do material bibliográfico. O problema neste caso, não parecia ser na modelagem da RA, mas no entendimento das etapas descritas sobre XP1, problema registrado na Tabela 6.3. através do tempo de execução de tarefas e da verbalização de ações .

4) Através da seqüência de atividades representadas, é possível acompanhar as etapas do processo modelado e verificar quais etapas e atividades a seguir.

- Foi analisado nos questionários que 18 usuários (60%) concordaram totalmente com a afirmação do questionário, 11 usuários concordaram e apenas 1 usuário (3,33%) ficou neutro.
- Na totalização das fichas de registro, na tarefa 2, consta a pequena média de 1,53 ações incorretas (usuários principiantes) de acompanhamento ou realização de atividade de modelagem do processo e 1,13 opções incorretas de experientes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existe uma dificuldade que precisa ser levada em consideração:

Problema identificado: mesmo tipo de linha para representar ligações entre atividades e transições, e atividades e artefatos. Através da verbalização de informal de ações, dois usuários sugeriram que estas linhas poderiam ter espessuras diferentes.

5) A visualização do processo através do grafo (da RA) é muito útil durante o processo de modelagem.

- Nos questionários foi totalizado o valor de 22 usuários (73,3 %) que concordaram totalmente com esta afirmação, 7 usuários (23,3 %) concordaram parcialmente e apenas 1 usuário (3,33) se posicionou de maneira neutra.
- Através da técnica da observação direta e de registros em blocos de notas e formulários, foi verificado que os usuários gostam da apresentação gráfica da rede e que no máximo foram realizados 6 pedidos de ajuda de principiantes e 3 de experientes para entendimentos a respeito de dificuldades de visualização do grafo.

Problemas identificados: não foram detectados problemas significativos que possam ser mencionados.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes.

6) A visualização do processo através do grafo (rede) é muito útil durante o acompanhamento do processo.

- A análise dos questionários pós-teste mostra que 21 usuários (70%) concordam totalmente com isto, 7 usuários (23,3 %) concordam parcialmente e apenas 2 usuários (6,67) estão neutros.
- A partir dos registros de pedidos de ajuda com média de 1,60 por usuário principiante e 1,20 pedidos de experientes, sendo que apenas 2 pedidos tratam da falta de boa visualização do processo através do grafo.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes,

porém existe uma dificuldade que precisa ser levada em consideração:

Problema identificado: falta representar mudança de cores em atividades concluídas ou em execução, de acordo com comentários de um usuário.

7) Eu utilizaria a Rede de Atividades novamente para descrever as atividades do processo de desenvolvimento.

- Os questionários apresentam que a partir do total de usuários (30), 27 (90,0 %) disseram que utilizariam a Rede de Atividades novamente para descrever as atividades do processo de desenvolvimento, apenas 2 usuários (96,67 %) ficaram neutros e só 1 usuário (3,33 %) discordou totalmente.
- A pequena quantidade de valores de ações incorretas (média de 1,13 e 1,93 entre os experientes) e (média de 1,53 e 3,60 de principiantes) respectivamente encontradas nos registros nas tarefas 1 e 2 já mostra um indício de que os usuários estavam conseguindo descrever suas atividades.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existe uma dificuldade que precisa ser levada em consideração:

8) O treinamento aplicado para o uso da rede foi suficiente.

- A partir da totalização dos questionários foi verificado que entre os usuários experientes, 8 concordaram (43,3 %) com esta afirmação, 4 (27 %) concordaram parcialmente, e 3 (20 %) se mantiveram neutros. Mas dentre os principiantes, apenas 3 usuários (20 %) concordaram totalmente, 8 concordaram parcialmente (26,6 %), 1 se manteve neutro e 3 (20 %) discordaram parcialmente.
- Os questionários pré-teste mostram que 46,6 % dos usuários principiantes não possuíam conhecimento anterior sobre processos de desenvolvimento nem sobre gerência de projetos (53,3 % usuários).

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existe uma dificuldade que precisa ser levada em consideração:

Problema identificado: treinamento curto para usuários principiantes. O pouco conhecimento prévio de processos de desenvolvimento gerou dificuldade de compreensão da abordagem da RA em uma única seção de treinamento.

6.5.2. Aspectos do uso da RA com o software editor

1) Com a utilização do editor se obtém automatização na modelagem da RA.

- As respostas dos usuários indicam valores positivos no lado esquerdo do questionário, entre principiantes e experientes: 46,67 % concordaram totalmente, 50 % concordaram parcialmente e apenas 1 usuário ficou neutro.
- Na Tabela 6.5. são observados pequenos valores médios de números de opções incorretas na tarefa 1 para principiantes: 2,20 e 1,33 para experientes.

Problemas identificados: o software Editor ajuda no desenho da rede, mas um conhecimento prévio sobre as etapas do processo a modelar é importante para a modelagem ser mais rápida.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém o tempo médio de realização da tarefa 1 (modelagem da rede) para usuários principiantes foi de 79,27min. (um pouco além do tempo previsto de 60min.).

2) Não existem dificuldades em interpretar os ícones que representam os elementos da RA.

- Verificando–se os questionários pós-testes, foram encontradas 9 respostas de usuários (30%) concordando totalmente, 20 concordando parcialmente (66,6 %) e apenas 1 resposta de desaprovação (3,33%).
- Nas fichas de registros da mensuração de desempenho, foram encontradas apenas 6 pedidos (20 %) de ajuda neste sentido.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existem dificuldades que precisam ser levadas em consideração e ajustadas na interação:

Problemas identificados: falta de *hint* (dica) nos ícones, falta ícone para inserir seta (apesar desta funcionalidade já surgir automaticamente após a inserção de um elemento da RA na tela de modelagem). Não existem ícones para recortar e colar, e as teclas de atalho não funcionam, conforme comentários de 8 usuários.

3) Não existiu dificuldade na adaptação de etapas da RA modelada inicialmente.

- De acordo com os questionários pós-teste, é visto que 12 usuários (ou 40%) responderam concordar totalmente com esta afirmação, 15 usuários (50% concordaram parcialmente), 2 ficaram neutros e apenas 1 usuário discordou.
- Foi recebida uma média de 1,6 pedidos de ajuda de usuários principiantes e 1,20 de experientes na tarefa 2 do roteiro (adaptação da rede original ao projeto SAD).

Problemas identificados: não foram detectados problemas significativos que possam ser mencionados.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes.

4) Não existiu dificuldade na instanciação de atividades e sub-atividades do projeto na RA.

- Observando a Tabela 6.10, vê-se que 14 usuários (46,67 %) concordaram totalmente com a afirmação, 8 usuários (26,67 %) concordaram parcialmente, 5 ficaram neutros e 1 usuário discordou.
- Nos registros de teste podem ser vistos em média, apenas 1 pedido de ajuda (de experientes) a 1,80 pedidos (de principiantes) na tarefa 3 (instanciação), e 1,13 ações incorretas de experientes e 1,87 ações incorretas de principiantes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes.

5) Não existiu dificuldade na inclusão de valores (tempo e custo) do projeto para a RA.

- Nos questionários pós-teste encontram-se 17 respostas (56,67 %) concordando totalmente com esta afirmação, 10 respostas (33,3 %) concordando parcialmente e apenas 3 respostas (10 %) discordando.
- Nos registros de número de opções incorretas da tarefa 3 do roteiro de teste, existiram apenas 1,80 opções incorretas de principiantes e 0,93 de experientes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existem dificuldades que precisam ser levadas em consideração e ajustadas na interação:

Problemas identificados: falta instrução *on line* com exemplo de formatação de datas e horários no software editor.

6) Não existiu dificuldade na simulação de totais (tempo e custo) do projeto.

- Na Tabela 6.10. pode ser observado que 43,33 % dos usuários (13 de 30) concordaram totalmente com a afirmação, 40 % (12 usuários) concordaram parcialmente, 6,67 % (2 de 30) se mantiveram neutros, e 10 % discordaram (3 de 30).
- Nas fichas de registros de eventos da tarefa 4 (Tabela 6.4) existem dados médios de 1,60 ações incorretas de usuários principiantes na tarefa 4 (simulação), e 1,07 ações incorretas de experientes, o que demonstra que foram cometidas poucas ações negativas neste sentido. No registro de tempo de execução de tarefas é visto que os usuários experientes gastaram em média até 7,80min. na tarefa 4 contra 9,80min. dos principiantes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existem dificuldades que precisam ser levadas em consideração e ajustadas na interação:

Problemas identificados: tela de *totalização por atividade* surge sobreposta à atividade sendo analisada, além disto, os nomes dos desenvolvedores não estão sendo mostrados na tela de resultados

e não há um botão ou uma forma de *start* geral (acionamento automático) para simulação (atualmente é preciso que o gerente clique na transição anterior à atividade para que possa dar início à próxima atividade e realizar a simulação).

7) Eu utilizaria o módulo Editor de RAs novamente para modelar um processo de desenvolvimento.

- Os resultados dos questionários pós-teste mostram que 18 usuários (60 %) utilizariam novamente a rede, 9 usuários (30%) também concordam (parcialmente), 1 se manteve neutro e 2 usuários (6,67 %). Discordaram.
- A Tabela 6.3. apresenta o tempo médio de modelagem da tarefa 1 por parte dos usuários principiantes (variou de 60 à 93min.), e (17 a 28,60min.) para usuários experientes.

Porém, ainda existe uma forma de deixar a modelagem mais rápida: incluir um ícone representativo de uma seta automática (composta de aresta+transição+aresta) para conectar duas atividades, o que tornaria a modelagem mais rápida, conforme expressaram 4 usuários através da verbalização de informações e descrição no formulário pós-teste.

Os resultados da mensuração de desempenho e da sondagem da satisfação são divergentes com relação a variável tempo. Porém foi observado que ambas as categorias de usuários, modelaram as redes de uma forma correta e se mostraram motivados para tal.

8) Eu utilizaria o módulo Editor de RAs novamente para acompanhar meu processo.

- 16 usuários (53,33 %) confirmaram esta afirmação fortemente nos questionários, 13 usuários (43,33 %) concordaram parcialmente e só 1 usuário ficou neutro.
- Nas fichas de registros da mensuração de desempenho (ver Tabela 6.6), foram registradas as pequenas médias de 1,20 pedidos de ajuda de usuários experientes na tarefa 2 (adaptação da rede ao projeto) e 1,87 pedidos de ajuda de principiantes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existe uma dificuldade que precisa ser levada em consideração e ajustadas na interação:

Problema identificado: falta de visualização da hierarquia total dos níveis e sub-níveis em uma única tela para facilitar a visualização geral e melhor acompanhamento do processo total.

9) Eu utilizaria novamente o módulo Editor de RAs para dar suporte aos desenvolvedores e suas atividades.

- Os resultados dos questionários OPUS apresentam 63,33 % dos usuários (19 de 30) confirmando totalmente a utilização do editor de RAs, 33,3 % usuários (10 de 30) concordando

parcialmente e apenas 1 usuário neutro.

- Na Tabela 6.4. é observada uma média de 1,87 ações incorretas de inexperientes e 1,13 ações incorretas de principiantes na tarefa 3 (instanciação da rede).
- Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existem dificuldades que precisam ser levadas em consideração e ajustadas na interação:

Problemas identificados: existem dificuldades identificadas com relação ao desenho de muitas atividades rotineiras e de nível mais baixo da rede, de acordo com a verbalização informal de 3 usuários e comentários descritos no formulário pós-teste que afirmaram que "...pode ser pensado algo para representar visualmente no módulo ERA a Rede de Atividades, talvez uma tabela, para descrever tarefas que são feitas inúmeras vezes, por exemplo descrever *user stories*, *tasks*, etc."

10) Eu utilizaria o módulo Editor de RAs novamente para gerenciamento de projetos.

- Verificando–se os questionários pós-testes, foram encontradas 15 respostas (50 % dos usuários) que concordam totalmente com a utilização do editor para gerenciamento de projetos, 14 respostas (46,67 % dos usuários) concordeo parcialmente e uma resposta neutra.
- Na Tabela 6.5. existe a pequena média de 1,67 opções incorretas por usuários principiantes e de 0,67 opções incorretas de usuários experientes na tarefa 4 (simulação e totalização do projeto), além de comentários verbais e escritos por 52% dos participantes sobre a facilidade de gerenciamento permitida pelo ambiente.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existe uma dificuldade que precisa ser levada em consideração e ajustada na interação:

Problemas identificados: não existem (ainda) relatórios gráficos de barra e relatórios descritivos sobre a situação do projeto por desenvolvedores, o que proporcionará um tipo adicional de visão geral do projeto.

11) O treinamento aplicado para o uso do software foi suficiente.

- A partir da totalização dos questionários pós-teste foi verificado que entre os 14 usuários (46,67 %) que responderam *concordo totalmente* 11 (73 %) são experientes e 3 (20%) são principiantes. Dentre 9 usuários (30%) que concordaram parcialmente 6 são principiantes (40 %) e 3 (20 %) são experientes. Dos 2 que responderam de maneira neutra, 1 usuário é principiante e outro experiente, e ainda existe a resposta de 5 usuários principiantes que discordaram totalmente sobre a suficiência do treinamento.
- Apesar do treinamento fornecido antes do experimento para cada usuário, ainda foram registrados pedidos de ajuda da parte dos usuários (média de 2,08 pedidos de ajuda na realização

de tarefas entre principiantes) e 1,27 pedidos de usuários experientes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são divergentes.

6.5.3. Uso e Navegação no Software

1) Uso do software na realização das tarefas de interesse.

- Na tabela 6.10. pode ser observado que 16,67 % dos usuários (5 de 30) acham muito fácil o software para realizar as tarefas de interesse, 80 % (24 usuários) acham fácil, e 3,33 % (1 de 30) se manteve neutro.
- De um modo geral, a opinião dos usuários em relação ao software, registrada nos formulários pós-teste foi de que "Não houve dificuldades em relação aos elementos da RA, mas poderia haver melhoria de usabilidade no software." (comentários de 3 usuários), "A dificuldade foi simplesmente de ír usando algo que tínhamos acabado de aprender no treinamento. Mas a notação da RA é simples e a existência de poucos elementos deixa isto tranqüilo." (comentário de 1 usuário) e que "Da forma como está sendo feito no Software está legal, ele tenta se aproximar ao máximo da realidade prevista." (1 usuário).

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes.

2) Localização dos itens de menu associados às tarefas.

- Dentre os usuários de teste, 11 (36,67%) disseram que a localização dos itens de menu é *muito fácil*, 12 responderam *fácil* (40%), 3 (10%) responderam *nem fácil nem difícil* e 4 usuários (13,33 %) responderam que é *difícil*.
- A Tabela 6.5. mostra os valores mínimos e máximos de opções incorretas (localização de itens de menu, de ícones, etc.) realizadas por usuários. Os valores de opções incorretas registrados por usuários principiantes variam de 1 a 5, e entre usuários experientes variam de 0 a 3.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes.

Comentário adicional: no registro de verbalização de informações foi encontrado o comentário de 1 usuário informando que o menu de ícones poderia ficar no lado direito da tela do editor, para ficar mais acessível para o usuário destro.

3) Uso das funcionalidades mais comuns encontradas no software.

- De acordo com a Tabela 6.10, 20 % dos usuários (6 de 30) acham muito fácil usar as funcionalidades do software, 70 % (21 de 30) acham fácil e apenas 10 % (3 de 30) se mantiveram neutros.

- Na Tabela 6.2 são encontrados valores mínimos e máximos de quantidade de ações incorretas no produto. Usuários principiantes realizaram 5 ações incorretas no máximo e usuários experientes realizaram 3 ações incorretas no máximo.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes.

Problemas identificados: não foram detectados problemas significativos que possam ser mencionados.

4) Compreensão das mensagens de erro apresentadas e recuperação destas situações.

- Na tabela 6.10. pode ser observado que 10 % dos usuários (3 de 30) acham muito fácil e compreendem as mensagens de erro apresentadas nos software Editor de RA, 36,67 % (11 usuários) acham fácil, e 46,67 % (14 de 30) responderam *nem fácil nem difícil*. Ainda pode ser visto que 6,67% (2 de 30) usuários acham difícil compreender as mensagens de erro e recuperação.
- Na Tabela 6.6 são apresentadas as médias de pedidos de ajuda nas tarefas 1, 2 e 3 que envolvem estas mensagens. As médias de pedidos foram respectivamente: 3,53, 1,60 e 1,80 pelos usuários principiantes e 2,20, 1,20 e 1,00 pelos usuários experientes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são divergentes, porém existem dificuldades que precisam ser levadas em consideração e ajustadas na interação: as mensagens de confirmação de gravação de Redes de Atividades dos níveis de metodologia e de projeto (Figura 6.1) não estão claras.

5) Processo de entrada e saída de dados durante o uso do software.

- Dentre os usuários de teste, 7 (23,33 %) responderam no questionário pós-teste que é muito fácil o processo de entrada e saída de dados durante o uso do software, 12 usuários (40,00 %) responderam que é fácil e 10 (33,33 %) responderam de maneira neutra, e apenas 1 usuário (3,33 %) respondeu que é difícil.
- A tarefa 3 do roteiro de tarefas envolvia este processo de entrada/saída de dados, na Tabela 6.4. estão registradas as médias das ações incorretas: 1,87 ações por usuários principiantes e 1,13 por usuários experientes.

Os resultados da mensuração de desempenho e da sondagem da satisfação são convergentes, porém existem dificuldades que precisam ser levadas em consideração e ajustadas na interação:

Dificuldades identificadas: falta de instruções *on line* com exemplo para formatação de datas, horários e custos no software editor, conforme comentários verbais de 10 usuários registrados nas

fichas de eventos.

6.6. Comentários

Este capítulo apresentou a metodologia de pesquisa utilizada no experimento prático, baseada em dois enfoques avaliatórios que envolvem a participação do usuário: a *sondagem da satisfação subjetiva* e a *mensuração do desempenho do usuário*. A integração dos dois enfoques fundamentou a parte experimental da avaliação de usabilidade do ambiente desta tese (Rede de Atividades + Software Editor de Redes de Atividades).

Foram apresentados os instrumentos utilizados na sondagem da satisfação subjetiva: os questionários desenvolvidos e adotados nesta pesquisa para delineamento do perfil e sondagem da opinião do usuário, e os intrumentos para mensuração do desempenho: roteiro de tarefas e fichas de registros de eventos.

Foram descritos os procedimentos da metodologia adotada, os dois enfoques utilizados e a descrição do planejamento do ensaio, elaboração do material, condução do ensaio e coleta dos dados.

Após isto, foram apresentadas a tabulação e análise dos dados obtidos a partir dos dois enfoques avaliatórios adotados na pesquisa: os resultados da avaliação do desempenho, com os dados coletados a partir dos testes de usabilidade conduzidos no Departamento de Sistemas e Computação da UFCG, assim como o procedimento estatístico destes dados, uma discussão dos problemas registrados e as recomendações sugeridas. Foram discutidos também os resultados da sondagem da satisfação subjetiva, com o delineamento do perfil das categorias de usuários consideradas nesta pesquisa, os resultados referentes às opiniões registradas nos instrumentos de sondagem, o procedimento estatístico destes dados, os problemas encontrados através deste enfoque e sugestões de melhorias.

Por fim, a última seção tratou da confrontação dos resultados obtidos a partir dos dois enfoques adotados, discutindo as similaridades e diferenças resultantes do experimento. Foi mostrado que os usuários principiantes conseguem executar as mesmas tarefas dos usuários experientes, com baixa diferença de médias aritméticas, com a Rede de Atividades no software editor de RAs, ainda que às vezes em período maior de duração, e números de escolhas incorretas quando as tarefas exigem experiência e conhecimento prévio (tarefas 1 – modelagem da rede e 3 – instanciação da rede com valores estimados).

O próximo capítulo apresenta as conclusões desta pesquisa de doutorado, comentando a aplicabilidade, contribuições e propondo a continuidade desta linha de pesquisa.

Capítulo 7

Conclusões

7.1. Considerações finais

De acordo com os problemas e pressupostos discutidos no Capítulo 1, ocorrem desperdícios, defeitos e prazos extrapolados no desenvolvimento de projetos. Os projetos complexos são desenvolvidos com dificuldade e as instituições necessitam de modelos de gerenciamento de projetos que permitam um acompanhamento detalhado do projeto, modelagem dos processos, acompanhando de tempos, custo e recursos envolvidos na realização das atividades. As atividades do projeto podem ser modeladas através de métodos formais e assim, pode-se descrever com precisão as dependências e interdependências existentes entre elas, as atividades paralelas e concorrentes e os tempos e custos associados.

Enquanto a literatura pesquisada revelou a existência de alguns trabalhos e ferramentas relacionadas com funções isoladas, a investigação através de pesquisas nacionais e internacionais revelou a existência de problemas no gerenciamento de processos e projetos de software.

Desta forma, esta pesquisa teve como objetivo principal desenvolver uma estrutura e uma ferramenta para modelar dinamicamente processos de desenvolvimento de software, acompanhar, e simular os projetos.

Desta maneira, contribuímos para o gerenciamento de projetos com uma estrutura de rede formal, baseada em redes de Petri, e com um ambiente de desenvolvimento de qualquer tipo de projeto denominado Gerência do Desenvolvimento de Projetos - GDP.

Com a **Rede de Atividades** pode-se realizar a modelagem das etapas de metodologias de desenvolvimento de software e definir atividades. Baseado em uma Rede de Petrí específica desenvolvida, a **Rede de Projetos**, é possível alocar as atividades com os respectivos custos, prazos e recursos alocados, e calcular tempos globais e um confrontar dados reais com dados previstos, possibilitando previsões e simulações de custos e prazos para um projeto.

O ineditismo do trabalho foi verificado na carência de estudos e ferramentas de gerenciamento que controlem simultaneamente **processos** de desenvolvimento de software e tempo, custos e recursos de **projetos**, baseados em métodos formais e com acompanhamento dinâmico e controle mais detalhado das variáveis envolvidas.

Para atingir o primeiro objetivo proposto '*extensão da estrutura da Rede de Atividades para acompanhamento de variáveis de um projeto*' foi realizada a definição de novas variáveis, para a Rede de Atividades incorporar tempos e custos (seção 4.3.1.), e a rede foi expandida (seção 4.6).

Para atingir o segundo objetivo proposto, '*definição formal de um tipo de rede de Petri, denominada **Rede de Atividades**, para permitir descrever as principais interdependências entre as atividades de um projeto e determinar as características de um projeto bem estruturado',* foi realizada a definição formal da Rede de Atividades com atributos de custo e tempo, descrita na seção 4.3.1. com os novos elementos e atributos criados que permitem descrever interdependências entre as atividades de um processo de desenvolvimento e armazenar os produtos criados a cada atividade.

Para garantir o terceiro objetivo, '*para garantir a composição precisa das atividades e calcular custos e prazos globais*', foi criado um tipo especial de rede de Petri, denominada **Rede de Projetos,** descrita na seção 4.2, base formal das Redes de Atividades. Para a Rede de Atividade modelar adequadamente os processos e projetos, realiza-se a sua modelagem de acordo com as regras da Rede de Projeto.

Para atingir o quarto objetivo proposto '*criação de uma ambiente de suporte ao processo de gerência do desenvolvimento de aplicações em quatro níveis'*, foi criado o ambiente GDP - Gerência do Desenvolvimento de Projetos, contendo quatro níveis de desenvolvimento de projetos. Para escolha da metodologia adequada ao projeto foi desenvolvido o sistema SiSeM, para modelagem da metodologia, adaptação e instanciação da metodologia para o projeto, e execução e simulação da rede foi criado o módulo Editor de Redes de Atividades – ERA, ambos descritos no Capítulo 5.

Para atingir o quinto objetivo proposto '*validar o ambiente proposto num estudo de caso*

real', foi realizada a avaliação da rede e do ambiente GDP no projeto SAD – Sistema de Apoio à Decisão da CHESF e também com outras pessoas com perfil semelhante à equipe da CHESF, descritos na seção 5.9. e no Capítulo 6.

O ambiente proposto (GDP - Gerência do Desenvolvimento de Projetos) integra SISEM e ERA. Os objetivos específicos foram atingidos, e o objetivo da tese foi atendido, atendendo os problemas alencados. Mas para que o apoio computacional às redes e aos objetivos se tornem plenos, a implementação do GDP deverá ser concluída.

7.2. Validação dos Resultados

O uso do GDP em um projeto real em andamento, o projeto SAD da CHESF, está servindo como experiência bastante proveitosa para avaliar os conceitos criados.

Está sendo realizado um estudo junto à equipe de desenvolvimento do projeto (gerente e desenvolvedores) e foram realizados experimentos com usuários com perfil semelhante aos primeiros, recrutados no Departamento de Sistemas e Computação da UFCG, nos laboratórios de projetos e na UEPB. O experimento consistiu em analisar a realização de tarefas no ambiente a partir dos dois enfoques avaliatórios adotados na pesquisa: mensuração de desempenho, e sondagem da satisfação subjetiva. Através da coleta e procedimento estatístico destes dados, foram encontrados pontos positivos, negativos e sugestões de melhorias.

Foi observado no experimento avaliatório (descrito no Capítulo 6) que os usuários principiantes do GDP conseguem executar as mesmas tarefas dos usuários experientes, tanto com a Rede de Atividades em si, quanto no uso do software editor de RAs, com baixa diferença de médias aritméticas de número de ações e escolhas incorretas, ainda que às vezes em período maior de duração, em especial quando as tarefas exigem experiência e conhecimento prévio.

Para ambientes distintos da engenharia de software foi feito uma apresentação do ambiente a um escritório de arquitetura e foram detectadas muitas semelhanças ao ambiente estudado neste tipo de projeto.

7.3. Contribuições da Pesquisa

As principais contribuições desta pesquisa são:

- Extensão e definição formal de novos atributos e elementos de um tipo especial de rede

denominada **Rede de Atividade**, para permitir a modelagem de diversos processos de desenvolvimento de software e controlar custos, tempos e recursos associados às atividades de um projeto.

- Para garantir a composição precisa das atividades para calcular custos e prazos globais, foi criado um tipo especial de rede de Petri, denominada **Rede de Projetos** que é uma base formal das Redes de Atividades.
- Definição de um ambiente de suporte ao gerenciamento de projetos, denominado GDP, considerando quatro níveis de uso, desde a caracterização do tipo do projeto e escolha da metodologia mais adequada até sua execução;
- Desenvolvimento de um Editor de Redes de Atividades para o ambiente GDP a fim de modelar as etapas do processo de desenvolvimento de um projeto, alocar atividades e recursos humanos, auxiliar no controle de custos, prazos, totalizar tempos e custos previstos, realizados e remanescentes;
- Criação de um sistema de suporte a decisão, o módulo SiSeM - Sistema para Seleção de Metodologias, que, por meio de uma árvore de decisão ponderada auxilia a escolha da metodologia mais adequada ao desenvolvimento de sistemas de software.

7.4. Trabalhos Futuros

A continuidade desta pesquisa pode ser dada, no nível de aperfeiçoamento das redes utilizadas e no incremento de novas funcionalidades para o ambiente GDP. A seguir estão relacionados alguns trabalhos futuros.

- Criar o modelo formal das RAs de alto nível. Como o ERA já permite a alocação de recursos humanos às atividades, detalhes adicionais deste modelo poderão determinar o uso de regras para o controle de recursos operacionais;
- Aperfeiçoar o ERA para manutenção de múltiplas RAs, integração de RAs, e outras.
- Integrar os módulo SiSeM e ERA em uma interface única para o GDP via web (de acordo com a Figura 5.6);
- Criar um manual de ajuda *on-line* para o GDP;
- Criar uma interface *estilo planilha* no GDP para acessar os dados da RA;
- Gerar relatórios de totalização de valores de custos e tempos para impressão.
- Desenvolver módulo de comunicação do ERA

- Melhorar a usabilidade da interface do ambiente GDP. De acordo com os resultados dos experimentos de usabilidade definidos no capítulo anterior.

7.5. Limitações Gerais

Como a concepção do GDP foi fortemente influenciada pela engenharia de software, poderá ter limitações quando aplicado a outros domínios. Por exemplo, os níveis 1 e 2 do ambiente só e aplicam a domínios que possuem metodologias bem definidas para dar suporte ao processo de desenvolvimento;

Não foi estudado a fundo o comportamento de uma RA não bem estruturada. Para projetos que exigem este tipo de estrutura, o cálculo de custos e prazos pode falhar;

O GDP não possui um controle de recursos operacionais e, neste ponto, poderia ser integrado com propostas específicas para este fim, também baseadas em Redes de Petri;

O GDP controla processos e projetos, mas não oferece suporte ao acompanhamento do código fonte da aplicação que está sendo desenvolvida pelo programador.

Referências Bibliográficas

Aiken, L. R. Rating Scales and Checklists – Surveying Opitinions and Assessing Personality. John-Wiley & Sons, Inc. New York, 1996

Almeida, H., COMPOR - Desenvolvimento de Software para Sistemas Multiagentes, Dissertação de Mestrado, UFCG, Campina Grande, Paraíba, 2004.

Alpan, G.; Jafari, M. A. Dynamic analysis of timed Petri nets: A case of two processes and a shared resource, IEEE Trans. Robot. Autom., vol. 13, no. 3, pp. 338–346, Jun, 1997.

ANALYSER, 2003 Analyzer All Clear. Disponível em: <http://www.simtel.net/product.download.mirrors.php?id=55157> Acesso em setembro, 2003.

APSEE – Prosoft – Projeto de Ambiente de Desenvolvimento de Software. UFRGS, <http://www.inf.ufrgs.br/prosoft> Acesso em março, 2008.

Aversano, L.; Gaeta, M; DeLucia, A.; Ritrovato, P.; Stafanucci, S.; Villani, M.L. Managing Coordination and Cooperation in Distributec Software Process: the GENESIS Environment. Published on line in Wiley InterScience. Disponível em: www.interscience.wiley.com. Acesso em dezembro, 2005.

Barbetta, P. A.; Reis,M.M.; Bornia, A.C. Estatística para cursos de engenharia e Informática., 2ª Edição, São Paulo, Editora Atlas S.A. , 2008.

Barcaui, A.; Borba, D.; Silva, I.; Neves R. Gerenciamento do Tempo em Projetos. FGV Management. FGV Editora, Rio de Janeiro, 2006.

Barroca, L.; Hall, J.; Hall, P. Ed. Software Architectures Advances and Applications, Springer-Verlag, Londres, 1999.

Bastos, R. M. O Planejamento de Alocação de Recursos Baseado em Sistema Multi-Agentes. Porto Alegre: CPGCC da UFRGS, Tese de Doutorado, 1998.

Bastos, R. M.; Oliveira, J. P. M. A Conceptual Modeling Framework For Multi-Agent Information Systems. In: Proceedings of the Xix International Conference on Conceptual Modeling, 2000, Salt

Lake City, USA. Anais. 2000.

Bause, F.; Kritzinger, P. S. Stochastic Petri Nets: An Introduction to the Theory. Advanced Studies in Computer Science, Verlag Vieweg. 1996.

Blaha, W.; Eddy, L. F. W.; Rumbaugh, J. ; Blaha, M.; Premerlani, E. W. F.; Lorensen, W. Object-Orienttid Modeling and Design,Prentice-Hall Inc., New Jersey, 1991.

Boaventura Netto; Oswaldo, P. **Grafos - Teoria, Modelos e Algoritmos**. Editora Edgard Blucher, 2006.

Booch, G.; Rumbaugh, J.; Jacobson, I. **Unified Modeling Language Manual**, The., Addison-Wesley Publishing Company, 2004.

Caldas, C. H. S. Sistemas de Planejamento e Controles Operacionais de Empreendimentos: a integração tempo, custos e recursos. Dissertação e Mestrado, UFF, 1998.

Campos, F. M. Métricas de software como ferramenta de apoio ao gerenciamento de projetos. Disponível em: <http://www.imasters.com.br/artigo.php?cn=2557&cc=160 2004> Acesso em setembro, 2004.

Cates, G. R. GERT and Q-GERT in Improving Project Management with Simulation and Completion Distribution Functions. Dissertation submitted for the degree of Doctor of Philosophy. Department of Industrial Engineering and Management Sciences in the College of Engineering and Computer Sciences at the University of Central Florida, Orlando, Florida, 2004.

Chaos Demographics – 2004 Third Quarter Research Report. The Standish Group International Inc. Disponível em: <http://www.standishgroup.com/> Acesso em janeiro, 2000.

Chaos Demographics – 2006 First Quarter Research Report. The Standish Group International Inc. Disponível em: <http://www.standishgroup.com/> Acesso em junho, 2008.

Chen, Y.L.; Hsu, P.Y.; Chang, Y.B. A Petri Net Approach to Support ResourceAssignment in Project Management. IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems And Humans, Vol. 38, No. 3, May, 2008.

Choren, R.; Garcia, A.; Lucena, C.; Romanovsky, **A. Software Engineering for Multi-Agent**

Systems III, Springer, 2004.

Cleland D. **Project Management – Strategic Design and Implementation**. 3 ed. New York: McGraw-Hill, 1999.

Cleland, D. I.; Ireland, L. R. Gerência de projetos. Rio de Janeiro. Reichmann & Affonso, 2002.

CMM-Capability Maturity Model Integration 2006 Disponível em: <http://www.sei.cmu.edu/cmm/> Acesso em dezembro, 2007.

COSMOS 4.1, 1998. Disponível em: <http://www.cs.umt.edu/RTSL/SoftEng/cosmos41/> Acesso em setembro, 2005.

Cronkovic, I. Component-based Software Engineering - New Challenges in Software Development. Software Focus, volume 4, págs. 127–133. Wiley, dezembro, 2001.

Crystal Family Disponível em <http://alistair.cockburn.us/index.php/Crystal_methodologies> Acesso em agosto, 2004.

Cuckierman, Z. S. **O modelo PERT-CPM aplicado a projetos**. Editora Qualitymark, 1993.

D'souza, W. D. A. Objects, Components and Frameworks with UML – The Catalysis Approach, Addison-Wesley Publishing Company, 1999.

Dogru, A.H.; Tanik, M.M. A Process Model for Component-Oriented Software Engineering. IEEE Transactions of Software Engineering, V.20, 2003 Disponível em: <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1184164>, 2003.

DotProject Disponível em: < http://www.dotproject.net/> 2006. Acesso em janeiro, 2008.

Dowson, M., Integrated Project Support with STAR. IEEE Software, pages 6-15, november, 1987.

Elmaghraby, S. E., Baxter, E. I. and Vouk, M. A. An approach to the modeling and analysis of software production process, Int. Trans. Oper.Res., vol. 2, no. 1, pp. 117–135, 1995.

Engineering and Managing Knowledge – ComonnKADS Disponível em

<http://www.commonkads.uva.nl/index.html>. Acesso em abril, 2004.

Farias, C.B.A. Análise comparativa de Metodologias de Desenvolvimento de Software. Relatório Projeto e Pesquisa I. UFCG/CCEI//DEE, junho, 2004.

Farias, C.B.A. SiSeM - Sistema Interativo para Seleção de Metodologias de Desenvolvimento de Software. Relatório Projeto e Pesquisa II. UFCG/CCEI/DEE, junho, 2005.

Farias, C.B.A.; Schiel, U. Um Sistema para Seleção de Metodologias de Desenvolvimento de Software. Revista Científica Integr@ção, ano I, nº. I, ISSN 1809-8320. Fundação Francisco Mascarenhas, PB, junho, 2006.

Farias, C.B.A.; Schiel, U. Gerenciamento de Projetos. Relatório Projeto e Pesquisa III. UFCG/CCEI/DEE, setembro, 2006.

Farias, C.B.A.; Schiel, U. SiSeM - Sistema Interativo para Seleção de Metodologias de Desenvolvimento de Software. IV Semana Pedagógica, Científica e Tecnológica do CCT – IVSPCT. UEPB, Campina Grande, PB, novembro, 2006.

Farias, C.B.A.; Schiel, U. Gerenciamento de Projetos com Redes de Atividades, Qualificação de Doutorado, UFCG/CCEI/DEE/COPELE, maio, 2007.

Farias, C.B.A.; Schiel, U. GDP – Sistema de Gerenciamento de Desenvolvimento de Projetos, XIV SIMPEP - Unesp FEB - Departamento de Engenharia de Produção - Campus Bauru, UFSCAR - Universidade Federal de São Carlos, EESC USP - Escola de Engenharia de São Carlos, PUCPR - Pontifícia Universidade Católica do Paraná. Baurú, SP, novembro, 2007.

Farias, C.B.A.; Schiel, U. Análise Comparativa de Metodologias de Desenvolvimento de Software. Revista Científica Integr@ção, ano II V. II, ISSN 1809-8320. Fundação Francisco Mascarenhas, PB, dezembro, 2007.

Farias, C.B.A.; Schiel, U. Gerenciamento de Projetos com Redes de Atividades. SBSI 2008 – IV Simpósio Brasileiro de Sistemas de Informação, Rio de Janeiro, RJ, abril, 2008.

Farias, C.B.A.; Schiel, U. GDP - Um Ambiente para Gerenciamento de Desenvolvimento de Projetos SIMS 2008 - XIII Simpósio de Informática, VIII Mostra de Software Acadêmico PUC RS

- FACI, Campus Uruguaiana, RS, Outubro, 2008. Publicação na Revista Hífen ISSN1983-6511.

FDD - Feature Driven Development. Disponível em: <http://www.featuredrivendevelopment.com> Acesso em setembro, 2007.

Ferraz. A. M. S. Implementando o PMBOK utilizando a ferramenta Methodology Explorer. UFPE Centro de Informática – CIn, dezembro, 2004.

Firth, R. A Classification Scheme for Software Development Methods. Software Engineering Institute Technical Report 87-TR-41. Software Engineering Institute, Carnegie-Mellon University, Pittsburgh, PA, 1988.

Fuggetta, A. Software Process: a Readmap. In: The Future of Software Engineering, A. Finkelstein (ed), 2000.

Furtado, M. Elizabeth. Uma Metodologia para Projeto de Banco de Dados Temporal orientado a Objetos" Dissertação de Mestrado, UFPB, Copin, 1993.

Garcia, F. YP – easY Process. Disponível em: < www.dsc.ufcg.edu.br/~garcia>, 2005. Acesso em fevereiro, 2007.

Genrich, H. J. Predicate/Transition Nets. In W. Brauer, W. Reisig and G. Rozenberg (Eds.). Petri Nets: Central Models and Their Properties – Advances in Petri Nets 1986, pp. 207-247. Lecture Notes in Computer Science, 254. Springer-Verlag, 1986.

Genrich, H. J.; Lautenbach, K. System modeling with high-level petri nets. Theoretical Computer Science, 13(1):109-136, January 1981.

Glaser, N.; Haton, M.; Haton, J.P., Models and knowledge acquisition cycles for multiagent systems. In Proc. Of 9th KAW, 20-24, Banff, Canada, 1995.

Glaser, Norbert. Contribution To Knowledge Modelling In A Multi-Agent Framework (The CoMomas Approach). Phd Thesis, L'Universtit´E Henri Poincar´E, Nancy I, France, November, 1996.

Glaser, N. Conceptual Modelling of Multi-Agent Systems: the CoMoMas Engineering Environment

Kluwer Academic Publishers: Dordrecht, 2002.

Hasegawa, K.; Miyagi, P. E.; Santos Filho, D. J. , Takahashi, K.L. Ma, and Sugisawa, M. "On resource arc for Petri net modeling of complex resource sharing system," J. Int. Robot. Syst. Theory Appl., vol. 26, no. 3/4, pp. 423–437, 1999.

Hebert, J.E. Critical path analysis and a simulation program for resource-constrained activity scheduling in GERT project networks, Purdue University, PhD dissertation, 1975.

Huang, Wei; El-Darzi, E.; Jin, Li. Extending the Gaia Methodology for the Design and Development of Agent-Based Software Systems, Computer Software and Applications Conference, 2007.

Iglesias, C. A.; Garijo, M.; Gonzalez, J. C.; Velasco, J. R. Analysis And Design Of Multiagent Systems Using MAS-Commonkads. Lecture Notes In Computer Science, V.1365, 1998.

Iglesias, C. A.; Garijo, M.; González, J. C.; Velasco, J. R. A Methodological Proposal for Multiagent Systems Development extending CommonKADS. Dep. de Teoría de la Señal, Comunicaciones e Ing. Telemática, E.T.S.I. Telecomunicación Universidad de Valladolid, E-47011 Valladolid, Spain. Disponível em <http://ksi.cpsc.ucalgary.ca/KAW/KAW96/iglesias/Iglesias.html#CoM>. Acesso em março, 2004.

ISO 9126 Software Quality Characteristics. Disponível em: <http://www.sqa.net/iso9126.html> Acesso em fevereiro, 2008.

Jacobson, I.; Booch, G.; Rumbaugh, J. **The Unified Software Development Process**, Addison-Wesley Publishing Company, 1999.

Jensen, K., **Coloured Petri Nets: Basic Concepts, analysis, methods and Practical use**, volume 1, EATCS Monographs on Theoretical Computer Science, Spronger-Verlag, 1992.

Jeetendra, V.A.; Krishnaiah, O.V.; Reddy, P. Petri Nets for Project Management and Resource Levelling. Springer-Verlag London Limited, Indian Institute of Technology Madras, Chennai, India, 2000.

Jianxin, G.; Jianwei, Q. Approach to workflow based on Petri net. Computer Engineering and

Design, 2005, vol.26, no.2:513-515.

Jongwook, K.; Desrochers, A. A.; Sanderson, A. C. Task planning and project management using Petri nets, in Proc. IEEE Int. Symp. Assem.Task Planning, 1995, pp. 265–271.

Kerzner, H. **Gestão de Projetos: as melhores práticas.** São Paulo: Bookman, 2002.

Kidd, J. B. A Comparison Between the VERT Program and Other Methods of Project Duration Estimation, OMEGA International Journal of Management Science, Vol. 15, No.2, pp 129-134, 1987.

Kumar, V. K.; Ganesh, L. S. Fuzzy operations and Petri nets, Proj. Manage. J., vol. 30, no. 3, pp. 13–22, September. 1998.

Levine, D.; Berenson, M.; Stephan, D. **Estatística: Teoria e Aplicações**, Editora LTC, 2000.

Liu, L. C.; Horowitz, E. **A formal model for software project management** IEEE Trans. Softw. Eng., vol. 15, no. 10, pp. 1280–1293, October. 1989.

Magott, J. Combined generalized stochastic Petri Nets and PERT networks for the performance evaluation of concurrent processes, in Proc. 3rd Int. Workshop on Petri Nets and Performance Models, 1989, pp. 249–256

MANAGE PRO. Disponível em: <http://www.managepro.com/> 1999. Acesso em setembro, 2005.

Marsan, M. A.; Conte, G.; Balbo, G. A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. ACM Transactions on Computer Systems, 2(2): 93-122. May, 1984.

Maturidade Organizacional e o Modelo de Avaliação PMI-OPM3. Project Management Institute, Inc. Disponível em: <http://opm3online.pmi.org/Overview.aspx> Acesso em junho, 2007.

Mayhew, D. J. **The Usability Engineering Lifecycle**. Morgan Kaufmann Publishers Inc., San Francisco, 1999.

Medeiros, E. **Desenvolvendo Software com UML 2.0**, Pearson Makron Books, 2004.

Medina, D.; González, M., J.L.; Drake, J.M.; González, M. UML-MAST: Modeling and Analysis

Methodology for Real-Time Systems Developed with UML CASE Tools. Group of Computers and Real-Time. University of Cantabria. Spain, 1997.

Medina, M.A.; Sanchez, A.; Castellanos, N., Ontological agents model based on MAS-CommonKADS methodology. Huajuapan de Leon, Univ. Tecnologica de la Mixteca, Puebla, México. Electronics, Communications and Computers, 2004. CONIELECOMP 2004. 14th International Conference on, Fev. 2004.

Meredith, J. R; Mantel, S. J. **Project Management: A Managerial Approach**, 4th ed. New York: Wiley, 2000.

MINUTE MAN. Disponível em: <www.minuteman-systems.com> Acesso em outubro, 2003.

Moder, J. J., Phillips, C.R.; Davis, E. W. Project Management With Cpm, Pert and Precedence Diagramming), (3rd edition) Publisher: Van Nostrand Reinhold Company, 1983.

Moeller, G.L.; Digman, L.A. Operations Planning with VERT, Operations Research, Vol. 29, Issue 4, July-Aug. 1982, pp. 676-697

Molloy, M.K. **Performance analysis using stochastic Petri Nets.** IEEE Transactions on Computers, 31(9): 913-917. 1982.

Morris, P.W.G. The Management of Projects, London, Thomas Telford, first published 1994, paperback edition, 1997.

MPS.BR - Melhoria de Processos do Software Brasileiro. Disponível em: <http://www.softex.br/mpsbr/_home/default.asp> 2003. Acesso em dezembro, 2007.

MS Project. Disponível em: <http://www.microsoft.com/brasil/office/project/default.asp>. Acesso em outubro, 2005.

Murata, T. Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, 77(4): 541-580. Abril, 1989.

Nielsen, J.,Coyne, K. Thahir, M. Make it Usable. PC Magazine, Janeiro, 2001.

ODE - Um Ambiente de Desenvolvimento de Software Baseado em Ontologias. Disponível em:

<http://www.inf.ufes.br/~labes/ode/> Acesso em outubro, 2005.

Odell, J.; Parunak, B. S. V. Extending UML for agents. in Proc. of the agent oriented in information systems workshop at the 17th National Conference on Artificial Intelligence, 2000.

Oliveira, J. F. **Metodologia para Desenvolvimento de Projetos de Sistemas** – Guia Prático, Editora Erica, 2002.

Paula, O.L.F. Desenvolvimento de PCP Para Micro e Pequenas Empresas Utilizando como Base o Pert-CPM. Dissertação de Mestrado. UFSC. Florianópolis, maio de 2001. Disponível em: <http://teses.eps.ufsc.br/defesa/pdf/5209.pdf>

PERTMASTER 7.8. Disponível em: <http://www.pertmaster.com/> Acesso em setembro, 2005.

PMBOK - A Guide to the Project Management Body of Knowledge, PMBOK, Newtown Square, PA, 2000 Edition.

PMBOK - A Guide to the Project Management Body of Knowledge. PMBOK® Guide, Third Edition, Project Management Institute, 2004 Edition, p. 5.

Pinto, F. Introdução ao CMMI. Disponível em: <http://www.pontodatecnologia.com.br/2006/07/introduo-ao-cmmi.html> 2006. Acesso em dezembro, 2007.

PLANVIEW. Disponível em: <www.myplanview.com> Acesso em outubro, 2005.

PMI - Project Management Institute. PMBOK Guide: a guide to the Project management body of knowledge. PMI Standards Committee, 2004.

Possi, M. Gerenciamento de Projetos, Rio de Janeiro, Brasport, 2006.

Pressman, R. Engenharia de software. São Paulo, MacGraw-Hill, 2006.

PRIMAVERA - Systems Helps Corporate IT Organizations Improve Resource Management - Have the Right People Work On the Right Projects With Web-Enabled Primavera TeamPlay 3.5. Disponível em: <http://findarticles.com/p/articles/mi_m0EIN/is_2002_Oct_7/ai_92526471?tag=rel.res3> Acesso em julho, 2007.

PRINCE2. Disponível em: <http://www.prince2.com/> 2005. Acesso em março, 2006.

Pritsker A. A. B.; Whitehorse, G. E. GERT: Graphical evaluation and review technique, Part II, probabilistic and industrial engineering applications," Journal of Industrial Engineering, Vol. 17 (6), 293-301, 1968.

Pritsker A.A.B.; Happ, W.W. GERT: Graphical evaluation and review technique, Part I, fundamentals, Journal of Industrial Engineering, Vol. 17 (5), 267-274, 1966.

PSP - Personal Software Process. SEI Interactive. Disponível em: <http://interactive.sei.cmu.edu/Features/1999/June/Background/Background.jun99.htm> 1999. Acesso em março, 2006.

PSP - Personal Software Process. Disponível em: <www.cin.ufpe.br/~processos/TAES3/slides/PspTsp2004.ppt> 2005. Acesso em abril, 2006.

Projeto Cheque Moradia, Governo do Estado da Paraíba, novembro, 2006.

Putnam, L. A General Empirical Soluiton to the Macro Software Sizing and Estimating Problem. IEEE Trans. Software Engineering, vol. 4, nº 4, 1978, pp. 345-361.

Queiroz, J. E. R. Abordagem Híbrida para Avaliação de Usabilidade com Interfaces com o Usuário. Tese (Doutorado em Engenharia Elétrica), Universidade Federal de Campina Grande, CEEI, DEE, COPELE, Campina Grande, 2001.

Quezado, P. C. A. Programação do Fluxo Produtivo de Máquinas e Equipamentos para Moinhos sob/encomenda Utilizando PERT/CPM e Heurísticas. Dissertação de Mestrado, UFSC, 1999.

RUP BUILDER - Rational Unified Process Builder. Process Manager's Guide, Disponível em: <http://sdt. web.cern.ch/sdt/RUP/builderguide.pdf>, janeiro, 2004. Acesso em setembro, 2004.

Redish, J.; Hackos, J. User and task Analyses for Interface Design, 1998.

Reisig, W. Petri Nets – An Introduction, Springer Verlag, 1985.

SAD – Sistema de Apoio à Decisão, CHESF/UFCG/DSC/Laboratório SAD. Disponível em: < http://www.chesf.gov.br/downloads/pesquisa%20e%20desenvolvimento/Projetos/PP_83.pdf> 2007.

Acesso em junho, 2007.

Schiel, U.; Mistrik, I. POKER - Petri-Net Oriented Knowledge Engineering Research, Anais X Simp. Bras. de Banco de Dados, Recife, outubro, 1995.

Schiel, U; Carvalho A. F. TOM – Rules a Uniform and Flexible Approach to Events, Constraints and Derived Information, Anais 8. Simp. Bra. de Banco de Dados, Campina Grande, PB, 1993.

Schneiderman, B. **Designing the User Interface: strategies for effective human - computer interaction**. Third Edition. ISBN 0-201-69497-2, 1998.

Schreiber, A. T.; Wielinga, B.J.; Van de Velde, J. M. A. W. CommonKADS: A comprehensive methodology for KBS development. Deliverable DM1.2a KADS-II/M1/RR/UvA/70/1.1, University of Amsterdam, Netherlands Energy Research Foundation ECN and Free University of Brussels. ftp://swi.psy.uva.nl/pub/CommonKADS/reports/CK-UvA-70.ps.gz. 1994. Acesso em maio, 2003.

Scott, A. O'Malley; Scott A. DeLoack, Determing When to Use an Agent-Oriented Software Engineering Paradigm. Proceedings of the Second International Workshop on Agent-Oriented Software Engineering (AOSE-2001), Montreal, Canada, may 29th 2001.

SCRUM. Disponível em: <www.scrum.com/>, 2005. Acesso em fevereiro, 2006.

Sommerville, I. **Engenharia de Software**, 6ª Edição, Pearson Addison Wesley, 2003.

SPICE. Disponível em: <http://sqi.gu.edu.au/spice> Acesso em fevereiro, 2005.

TABA - UFRJ. Meta-Ambiente para instanciação de Ambientes de Desenvolvimento convencionais e Orientados a Domínios, Disponível em: <http://www.cos.ufrj.br/taba>. Acesso em janeiro, 2004

TFS - Team Foudation Server. Disponível em: <http://msdn2.microsoft.com/en-us/teamsystem/aa718934.aspx> 2005. Acesso em setembro, 2006.

Torres, J. B. Uma Ferramenta de Gerência de Projeto – GEPRO, Dissertação de Mestrado, COPIN - UFPB, Campina Grande, 1996.

TRILLIUM. University of Houston Clear Lake. Disponível em:

<http://www.sqi.gu.edu.au/trillium/> Acesso em março, 2005.

TSP - The Team Software Process. Carnegie Mellon University Disponível em: <http://www.sei.cmu.edu/tsp/tsp.html> 28 April 2006. Acesso em maio, 2006.

Valeriano, D. **Moderno Gerenciamento de Projetos**. L. Prentice Hall Brasil, 2005.

Van Der Aalst, W. M. P. The application of Petri nets to workflow management, J. Circuits Syst. Comput., vol. 8, no. 1, pp. 21–66, 1998.

Vargas, R. V. **Gerenciamento de projetos: estabelecendo diferenciais competitivos**. Rio de Janeiro: Brasport, 2002.

Vianna, M. Conheça o Microsoft Solutions Framework - MSF. 31 março 2004. Disponível em: <http://www.linhadecodigo.com.br/artigos.asp?id_ac=78&sub=0>. Acesso em agosto, 2004.

VSTS - Visual Studio Team System. Disponível em: <http://www.microsoft.com/brasil/msdn/teamsystem/Default.mspx.> 2008. Acesso em fevereiro, 2008.

Wang, C.; Xue, L.J.; Yang, L. Research of College's Scientifi'c Research Management Workflow Model Based on Petri Net. School of Management, Xi'an University of Architecture & Technology, 2007 International Conference on Management Science & Engineering (14th), August 20-22, 2007 Harbin, P.R.China.

Wills, A. C. Designing Component Kits and Architectures with Catalysis, in (Barroca, Hall e Hall, 1999).

Wooldridge, M. Intelligent Agents. In: WEISS, Gerhard (Ed.). Multiagent Systems - A Modern Approach. (S.L.): MIT Press, 1997.

Wooldridge, M.; Jennings, N. R.; Kinny, D., The Gaia Methodology For Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems Springer Netherlands, October, 2004.

XP - eXtreme Programming. Disponível em: <www.extremeprogramming.org>, Acesso em setembro, 2003.

XP1 - Adaptação de eXtreme Programming Disponível em: <http://dsc.ufcg.edu.br/~jacques/projetos/common/xp1/xp1.html> Acesso em março, 2008

XPLANNER. Disponível em: <http://www.xplanner.org> Acesso em outubro, 2005.

Yang, L.; Jun, Z. Scientific research Project management of the university based on workflow. Information Technology, 2006, no.2:11-13.

Zhu, P.; Schnieder, E. Holistic Modeling of Complex Systems with Petri Nets Institute of Control and Automation Engineering, Technical University of Braunschweig, Braunschweig, Germany. Systems, Man, and Cybernetics, 2000 IEEE International Conference on, Volume: 4, On page(s): 3075-3080 vol.4. Location: Nashville, TN, USA, 2000.

Zinga, J. S. L.; Gomes, M. D. S.; Lopes, R. C. ERA - Editor de Rede de Atividades, Monografia de Projeto I e II, UFCG, DSC, maio, 2007.

ANEXOS

ANEXO I

Survey - desenvolvimento de software no Brasil

Resultados da Survey

Os resultados são baseados nas respostas de 20 gerentes de projetos de TI de empresas que desenvolvem software sob encomenda ou pacotes de software.

Estado

Tabela 2: Localização das empresas

Resposta	*Quantidade*	*%*
PB	6	30%
PE	2	10%
SP	5	25%
RS	2	10%
MG	5	25%
Total	**20**	**100**

Foco da Empresa

Tabela 3: Respostas para a questão: "Qual o foco da empresa ?"

Resposta	*Quantidade*	*%*
Desenvolvimento de Pacotes	13	65%
Desenvolvimento sob encomenda	7	35%
Total	**20**	**100**

Metodologia de Desenvolvimento

Tabela 4: Respostas para a questão: "Qual a metodologia de desenvolvimento de software?".

Resposta	*Quantidade*	*%*
XP	3	15%
RUP	7	35%
SCRUM	4	20%
CRYSTAL	1	50%
CATALYSIS	0	0%
Nenhuma	2	10%
Outra- MSF, met. própria, UML	3	15%
Total	**20**	**100**

Etapas de Desenvolvimento

Tabela 5: Respostas para a questão: “Existem problemas durante a(s) etapa(s) de?”.

Resposta	*Quantidade*	*%*
Análise	5	25%
Projeto	6	30%
Implementação	4	15%
Testes	5	25%
Total	**20**	**100**

Padrão de Gerenciamento/Método de Maturidade

Tabela 6: Respostas para a questão: “A empresa usa qual padrão de gerenciamento de projetos de software?”.

Resposta	*Quantidade*	*%*
CMM	3	15%
PSP	0	0%
PMBOK	6	30%
TSP	0	0%
PRINCE2	0	0%
ISO/IEC 1550	1	5%
Nenhum	7	35%
Outro (MPS-BR)	3	15%
Total	**20**	**100**

Métrica de Planejamento

Tabela 7: Respostas para a questão: “A empresa usa qual métrica de planejamento?”.

Resposta	*Quantidade*	*%*
Pontos-por-função	8	40%
COCOMO	3	15%
PUTNAM	3	15%
Outro (homem-hora, caso de uso)	6	30%
Total	**20**	**100**

Software de Gerência de Projeto

Tabela 8: Respostas para a questão: “A empresa usa qual software de gerência de projetos?”.

Resposta	*Quantidade*	*%*
MS project	13	65%
Manage Pro	0	0%
Minute Man	0	0%
AllClear Analyser	0	0%
Hydra	0	0%
Xplanner	3	15%
Visual Studio	2	10%
Outra (RPM, Work Banch)	2	10%
Total	**20**	**100**

Faturamento da Empresa

Tabela 9: Respostas para a questão: “Qual o faturamento de sua empresa ?”

Resposta	*Quantidade*	*%*
Menos de 10 milhões	17	85%
Entre 10 milhões e 100 milhões	2	10%
Entre 100 milhões e 1 bilhão	1	5%
Mais de 1 bilhão	0	0%
Total	**20**	**100**

Ranking de Problemas

Tabela 10: Ranking de problemas".

Ranking		Problemas possíveis
1	20%	Falta de um controle efetivo de custos.
	20%	Ferramentas de gerenciamento de projetos não possuem todas as funcionalidades necessárias.
2	15%	Falta de um controle de acompanhamento de múltiplas atividades paralelas.
	15%	Falta de um controle efetivo de cronogramas.
3	11%	Metodologia inadequada ao tipo de aplicação/ ambiente de desenvolvimento.
4	6%	Conflitos entre analistas e programadores.
5	5%	Padrão de gerenciamento de projeto inadequado.
6	3%	Falta de um processo de desenvolvimento claramente
	3 %	Conflitos entre analistas e clientes.
7	2%	Clientes insatisfeitos, má usabilidade.

ANEXO II

SiSeM - Um Sistema para Seleção de Metodologias de Desenvolvimento de Software

O sistema SiSeM é um sistema de apoio à decisão que dá suporte ao desenvolvedor de software ou gerente de projeto para que ele possa escolher uma metodologia adequada ao seu ambiente de desenvolvimento, e tipo de aplicação.

Através de um diálogo interativo, as características do futuro sistema são informadas e confrontadas com as características inerentes às metodologias de desenvolvimento de software. Como resultado é apresentada *uma lista de metodologias* mais adequadas, organizadas na forma de um *ranking* das melhores.

Para o desenvolvimento do SiSeM foram elaboradas **árvores de decisão** (Figuras 10-17) (Farias, 2004) contendo perguntas e possíveis respostas (definidas através de questões objetivas) e os critérios que diferenciam as metodologias entre si, de acordo com as respostas dadas. Os critérios foram escolhidos de acordo com a Tabela 2.1 do Capítulo 2: *Agilidade, Robustez¸ Quantidade de Artefatos, Uso de padrões, Reusabilidad, Escala-bilidade, Ambiente, Ferramentas de Apoio e Custo.*

Para cada um destes critérios, são apresentadas perguntas ao usuário. O mapeamento destes diálogos se baseou no conhecimento adquirido em pesquisas bibliográficas sobre metodologias de desenvolvimento de software (Farias, 2004), na experiência prática de projetos anteriores e em entrevistas com equipes de desenvolvimento.

Esta árvore serve também como estrutura de suporte ao somatório da pontuação de metodologias.

De acordo com cada resposta, uma pontuação é atribuída através de uma escala de pontos: +5, +10, + 15 ou +20 para cada metodologia. Cada metodologia recebe mais pontos a medida que as respostas vão passando pelo caminho da árvore.

Toda a árvore de decisão está armazenada como arquivo XML (Figura 19). Para incorporar novas metodologias dentro do escopo das existentes, basta acrescentar a pontuação delas ao diálogo.

Para incorporar novas tendências (Orientação a Objetos, Orientação a Agentes, etc.) implica em uma alteração da própria árvore de decisão, é preciso incluir as novas características e rever todas as pontuações existentes.

Como o sistema é acessado via web*, as pessoas interessadas podem ter acesso à pesquisa e obtenção dos resultados, mas apenas o gerente de processo poderá fazer inclusão e alteração de dados na árvore de decisão.

A utilização do sistema facilita o trabalho dos engenheiros de projeto, além de melhorar a produtividade desses profissionais e a qualidade dos projetos de software definidos e, principalmente, criando produtos finais de melhor qualidade.

1. Projeto

A próxima Figura mostra os subsistemas do SiSeM e suas interfaces.

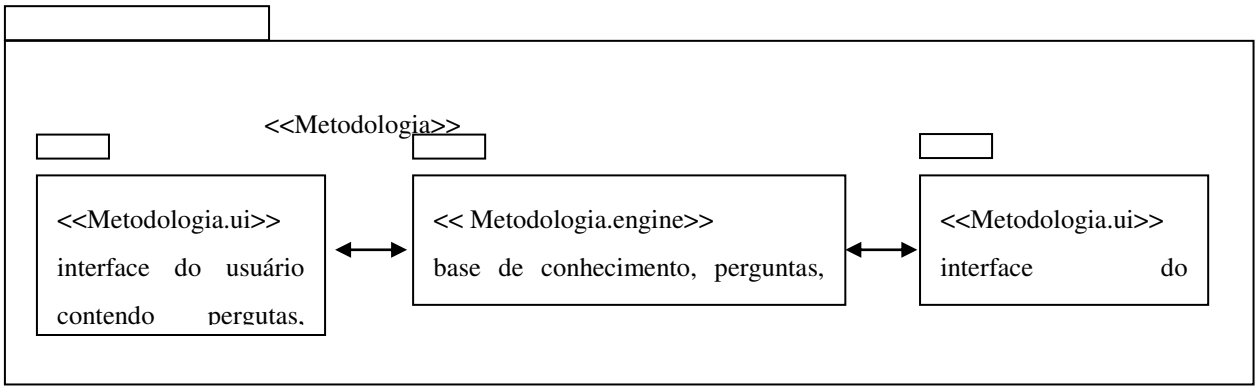


Figura 1 – Interfaces do sistema

Para a criação da árvore de decisão, foi realizado um estudo sobre metodologias de desenvolvimento de software (Farias, 2004), e de acordo com critérios selecionados a partir do *Software Engineering Institute,* foi realizada uma análise comparativa entre metodologias orientadas a objetos, baseadas em conhecimento e agentes (ver capítulo 3). Foi criada a tabela (Tabela 3.1) comparativa e de acordo com estes critérios foi construída uma árvore de decisão para conduzir o diálogo com o usuário. Pesos de 0 à 20 (zero a vinte) foram atribuídos às metodologias de acordo com as características, por exemplo, RUP ganhou mais pontos (+20) na característica de "documentação-quantidade de artefatos" do que XP e YP (que receberam + 10), pois ele possui 25 artefatos enquanto XP possui 10.

Os pesos forma inseridos na árvore de acordo com o caminho percorrido, por exemplo, se o gerente está a procura de uma metodologia que seja baseada em conhecimento, as metodologias baseadas em conhecimento: CommonKADS, ComoMAS, MAS-CommonKADS e POKER ganharam mais pontos do que XP, YP e Catalysis. A Figura 10 mostra uma parte da árvore referente à condução do diálogo do exemplo de uso do módulo. Nas Figuras 2-9 encontram-se as sub-árvores de decisão criadas para as outras categorias de aplicações e em (Farias, 2005) encontra-se a documentação completa do sistema.

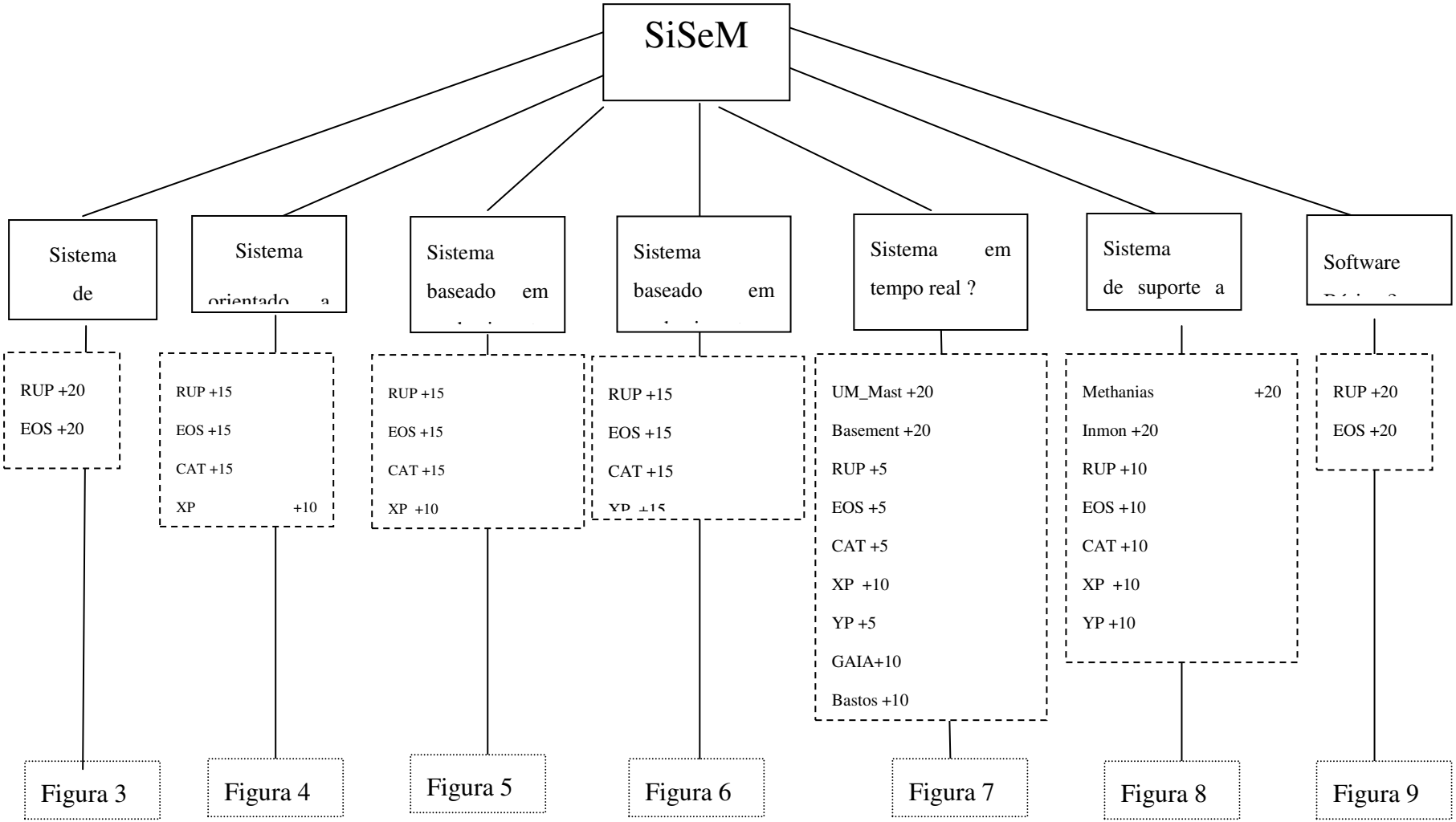


Figura 2: Diagrama de árvore com estrutura de perguntas e pontuações do SISEM

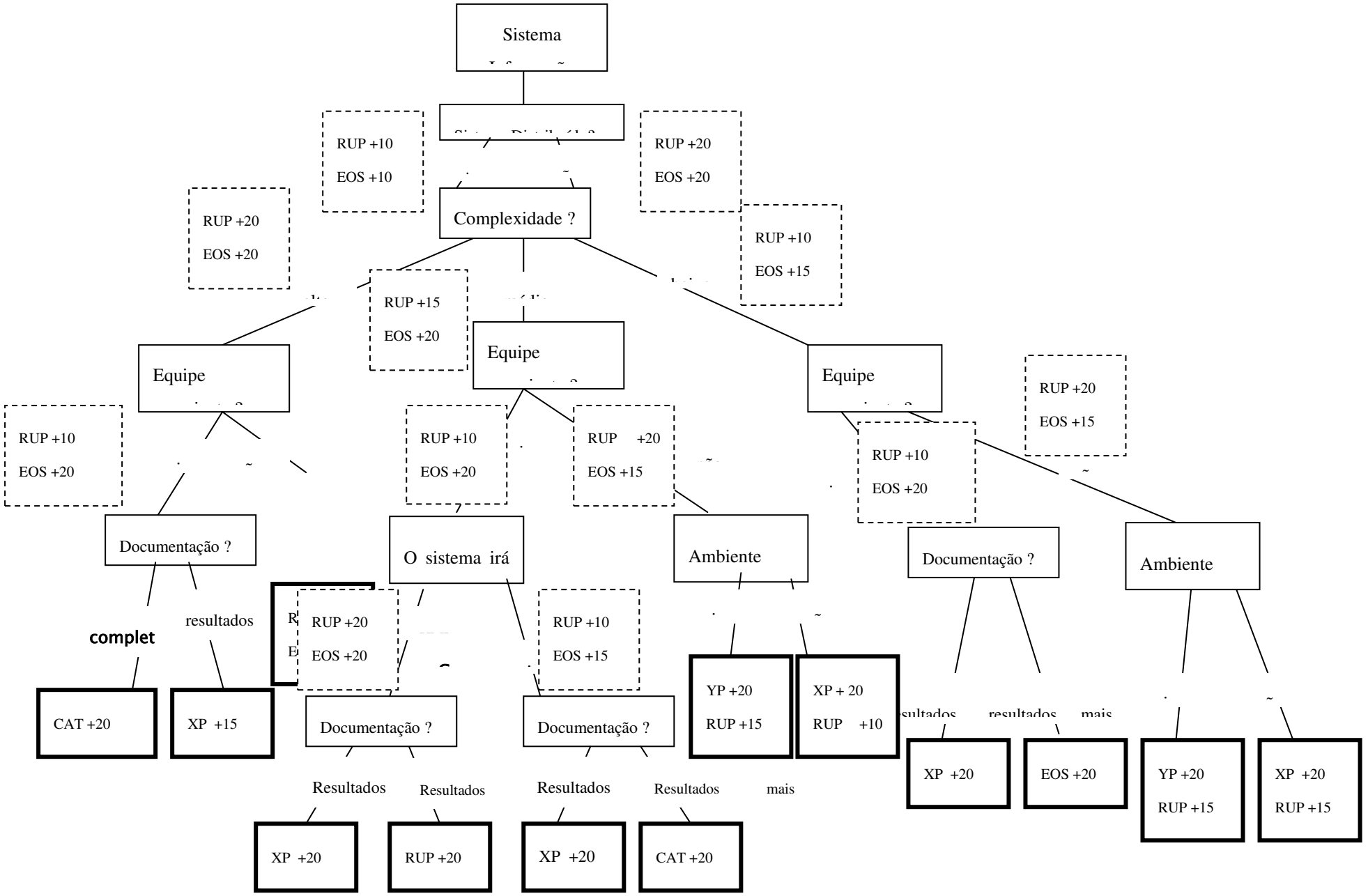
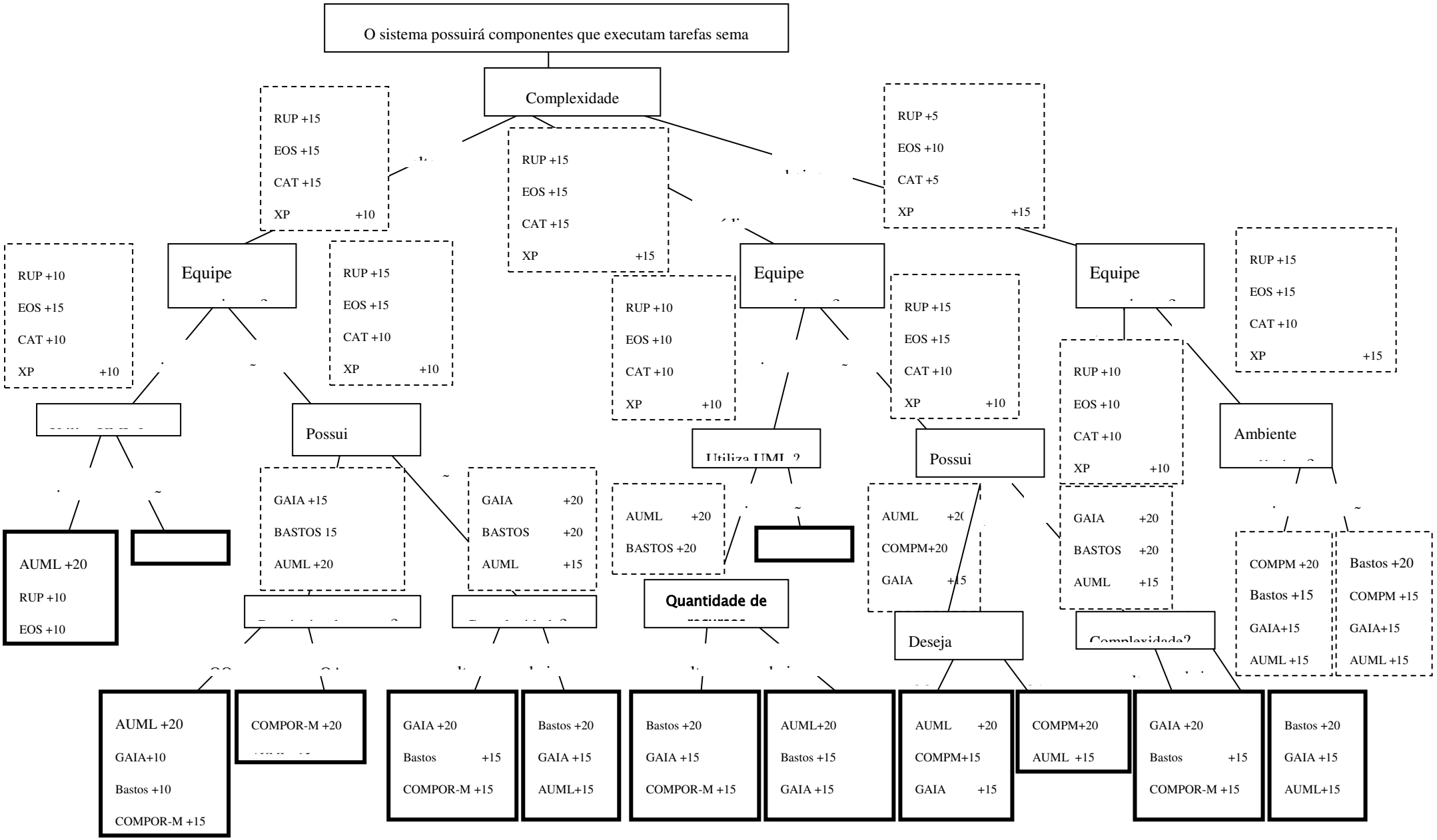
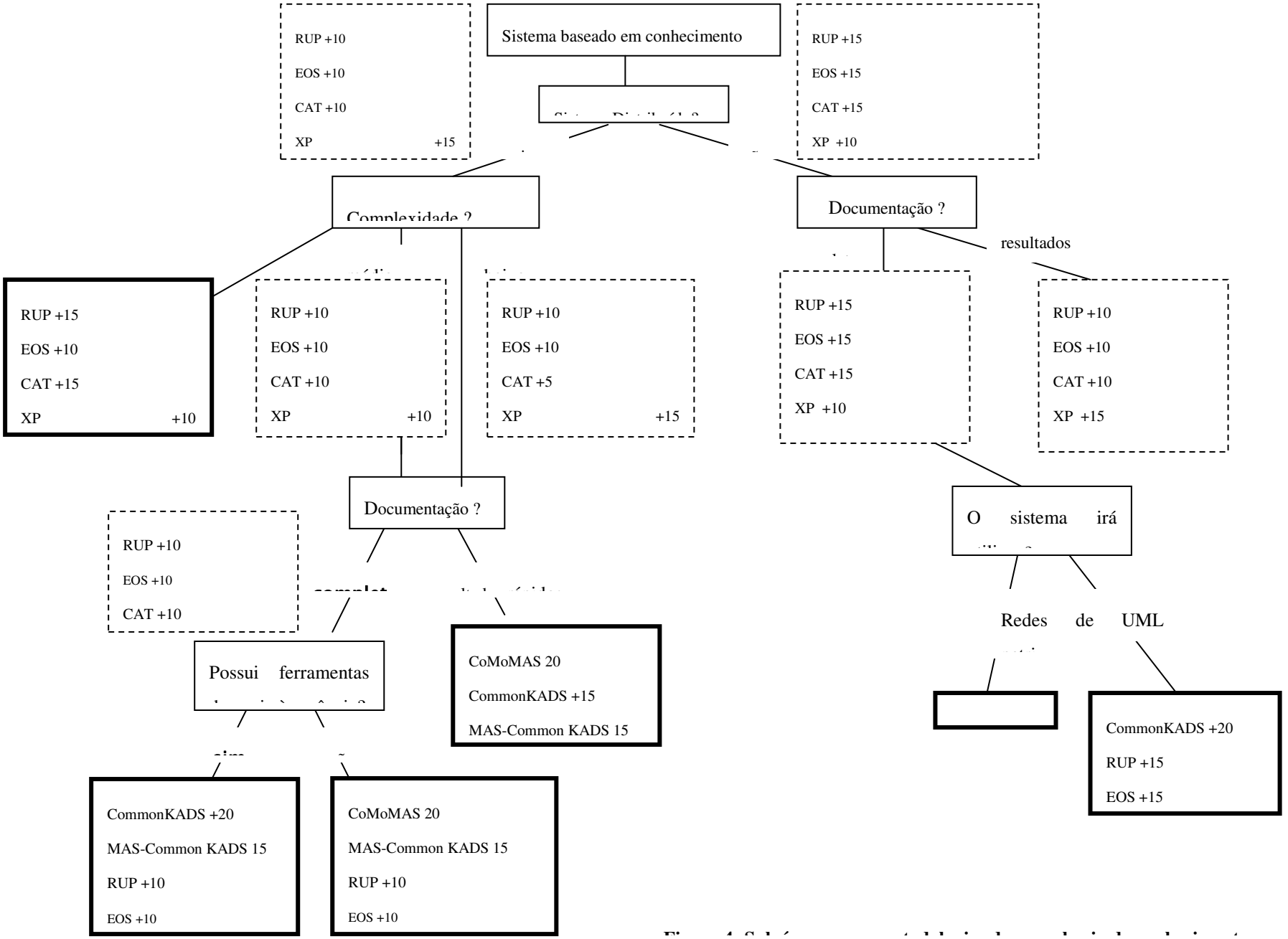

Sistema
RUP +10
EOS +10
RUP +20
EOS +20
Complexidade ?
RUP +20
EOS +20
RUP +10
EOS +15
RUP +15
EOS +20
Equipe
Equipe
Equipe
RUP +10
EOS +20
RUP +10
EOS +20
RUP +20
EOS +15
RUP +20
EOS +15
RUP +10
EOS +20
Documentação ?
O sistema irá
Ambiente
Documentação ?
Ambiente
complet
resultados
RUP +20
EOS +20
RUP +10
EOS +15
CAT +20
XP +15
Documentação ?
Documentação ?
YP +20
RUP +15
XP + 20
RUP +10
XP +20
EOS +20
YP +20
RUP +15
XP +20
RUP +15
Resultados
Resultados
Resultados
Resultados
mais
XP +20
RUP +20
XP +20
CAT +20

Figura 3: Sub árvore com metodologias orientadas a agentes




RUP +10
EOS +10
CAT +10
XP +15
Sistema baseado em conhecimento
RUP +15
EOS +15
CAT +15
XP +10
Complexidade ?
Documentação ?
resultados
RUP +15
EOS +10
CAT +15
XP +10
RUP +10
EOS +10
CAT +10
XP +10
RUP +10
EOS +10
CAT +5
XP +15
RUP +15
EOS +15
CAT +15
XP +10
RUP +10
EOS +10
CAT +10
XP +15
Documentação ?
O sistema irá
RUP +10
EOS +10
CAT +10
Possui ferramentas
CoMoMAS 20
CommonKADS +15
MAS-Common KADS 15
Redes de UML
CommonKADS +20
RUP +15
EOS +15
CommonKADS +20
MAS-Common KADS 15
RUP +10
EOS +10
CoMoMAS 20
MAS-Common KADS 15
RUP +10
EOS +10

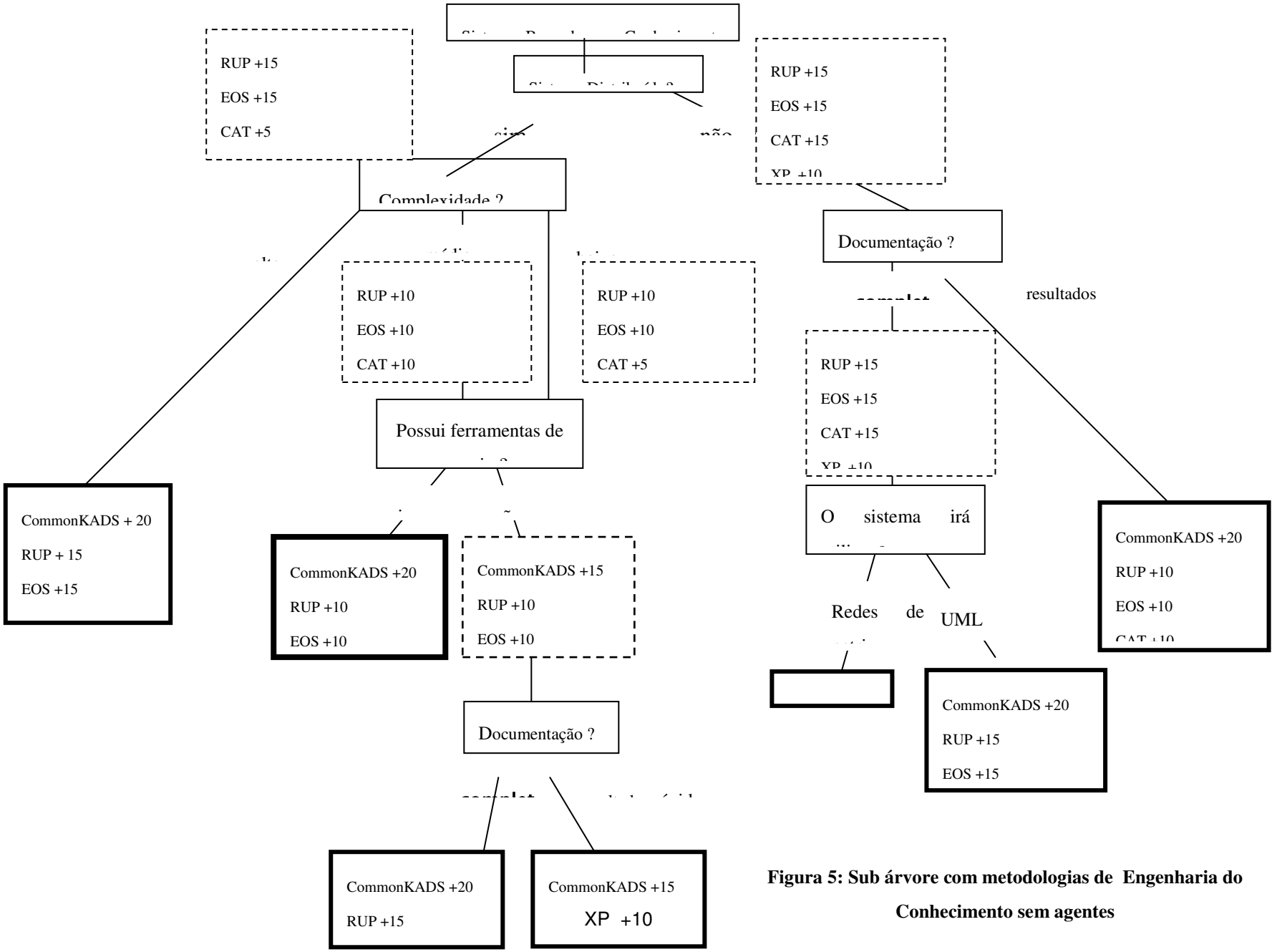


Figura 5: Sub árvore com metodologias de Engenharia do Conhecimento sem agentes

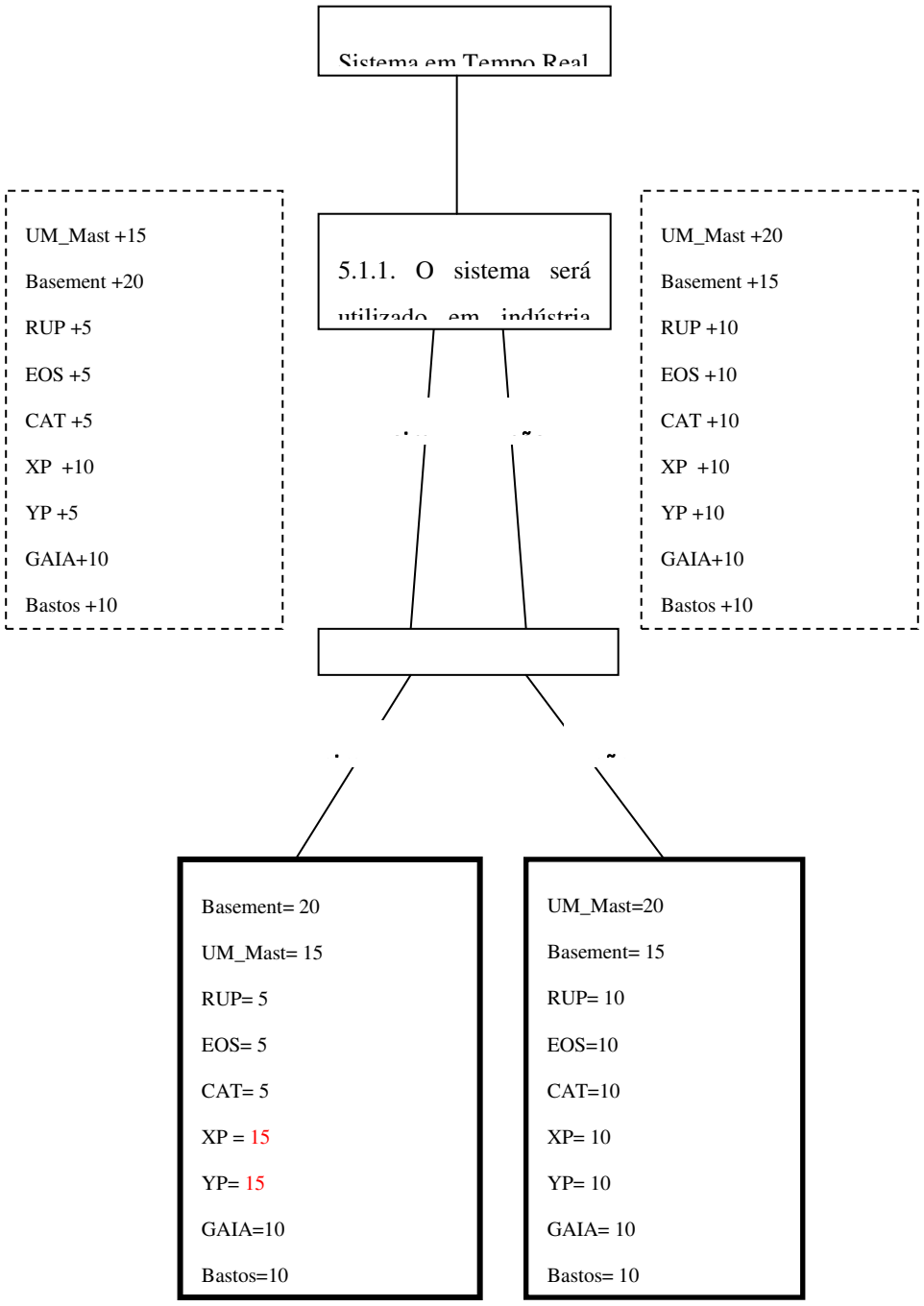


Figura 6: Sub árvore com metodologias para Sistemas em

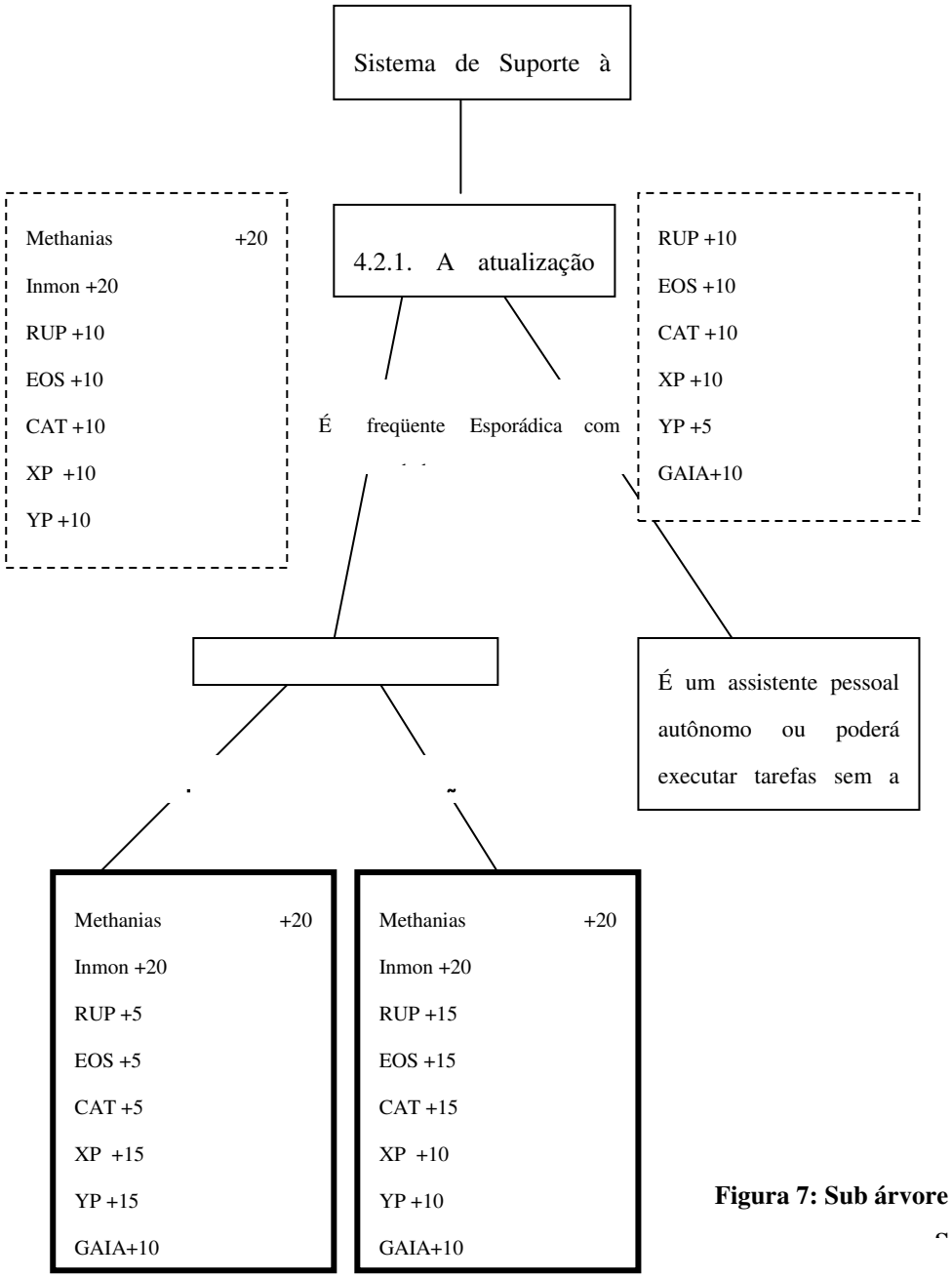


Figura 7: Sub árvore com metodologias para Sistemas de

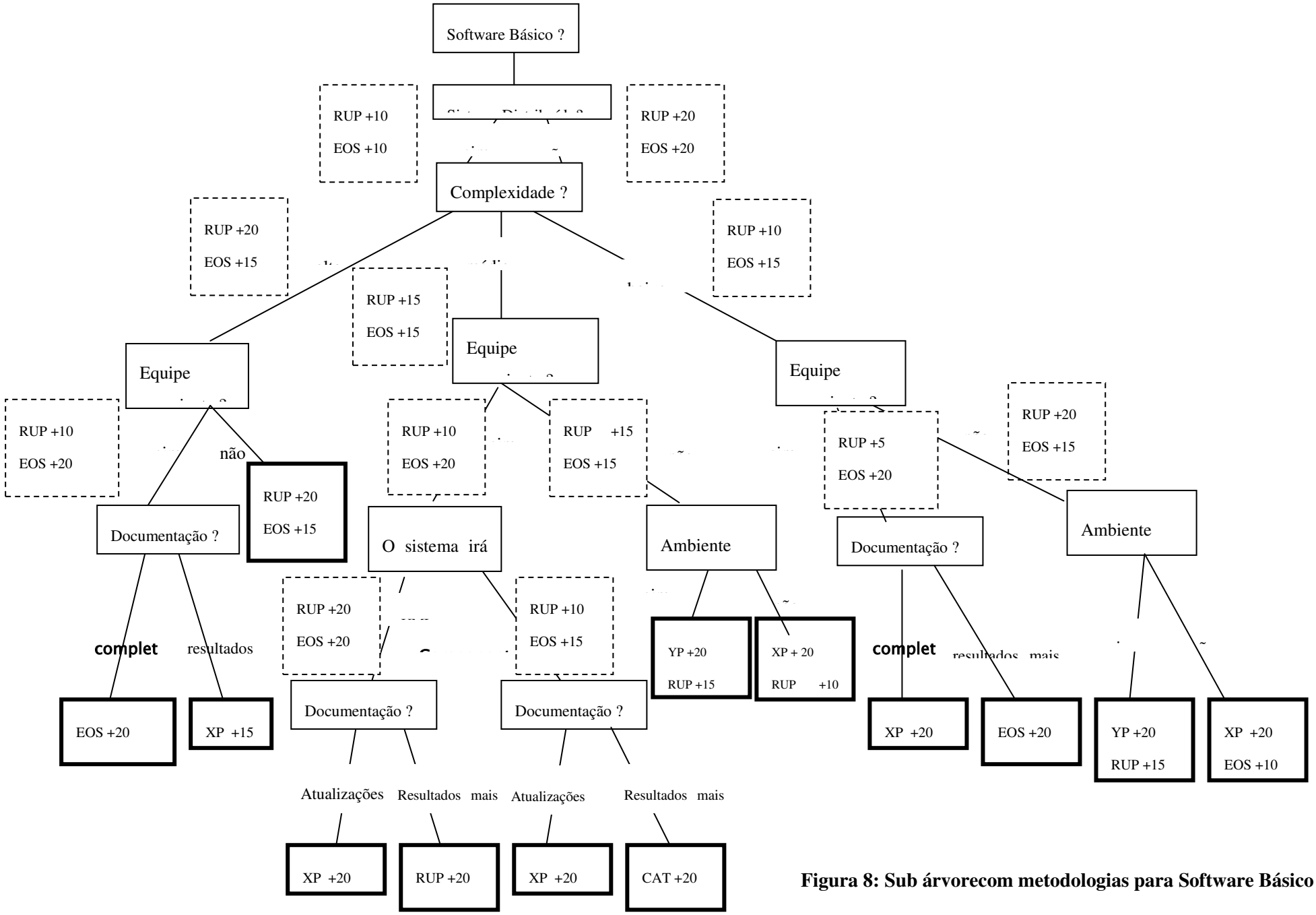


Figura 8: Sub árvorecom metodologias para Software Básico

2. Implementação

Os diálogos de interação são páginas ASP.NET implementadas com a linguagem C#.

A Figura 9 mostra uma parte do código, onde é obtida a última pergunta que foi respondida e mostra a pergunta seguinte (se houver).

```
// Obtem a pergunta que foi respondida
XmlNode QuestionNode = GetActualNode();
XmlNode AnswerNode = QuestionNode.SelectNodes ("//answer[@goto=" +
this.rbAnswers.SelectedValue+"]")[0];
int iCriteria = Convert.ToInt32 (QuestionNode.Attributes
["criteria"].Value);
objResults = (CriteriaCollection) Session["Results"];
foreach (XmlAttribute objAttribute in AnswerNode.Attributes )
{
      if ((objAttribute.Name != "goto") && (objAttribute.Name !=
"text"))
            {
                  int iValue = Convert.ToInt32
```

Figura 9: Parte do código da engine em C#

Os dados sobre metodologias, diálogos e resultados ficam em uma árvore de decisão codificada em XML. Na Figura 10 exemplificamos parte do código XML com o diálogo referente ao nível de perguntas sobre complexidade do sistema.

A plataforma do sistema sevidor é Windows, utiliza interface gráfica (GUI). Com relação à segurança o usuário não poderá alterar a base de conhecimento, só quem poderá fazê-lo será o gerente de processo.

```
<question id="31" text="Qual o nível de complexidade ? "
criteria="8">
      <answers>
            <answer goto="-1" text="Alta" RUP="10" Catalysis="15"
XP="10"       YP="10"       Gaia="0"       Bastos="0"       AUML="0"
COMPORM="0"CommomKADS="15"      MASCommonKADS="15"      COMOMAS="15"
POKER="20" />
            <answer goto="312" text="Média" XP="10" Catalysis="10"
RUP="10"   YP="10"   Gaia="0"   Bastos="0"   AUML="0"   COMPORM="0"
CommomKADS="15" MASCommonKADS="20" COMOMAS="20" POKER="15" />
```

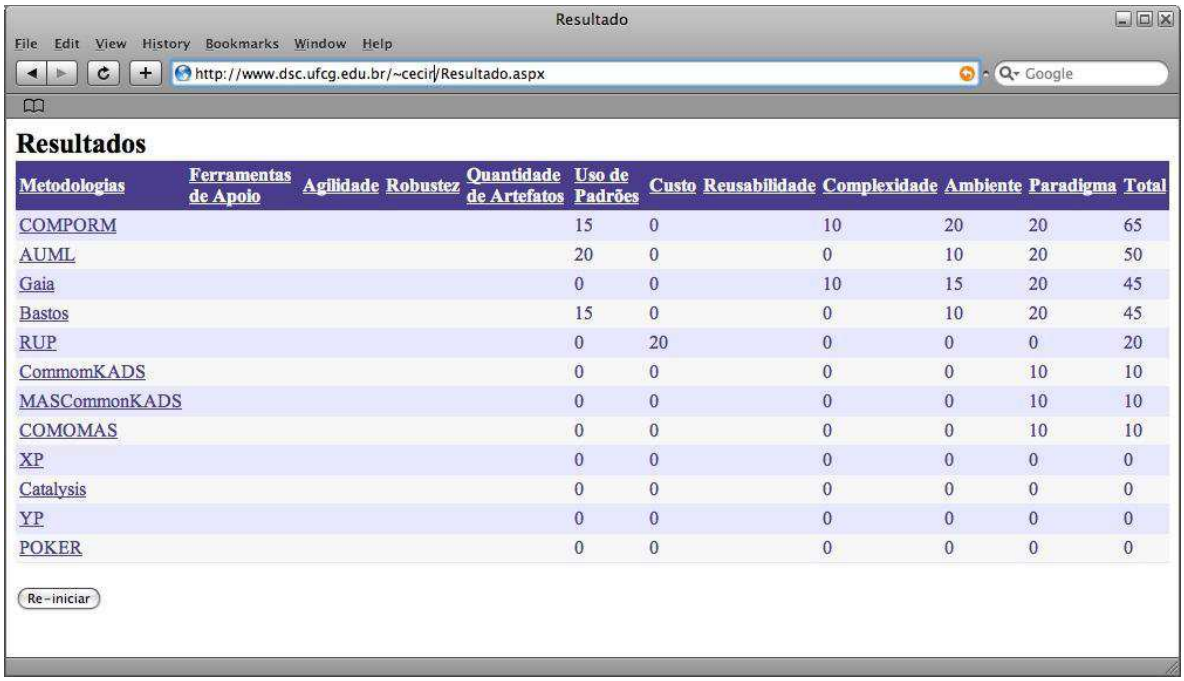
Figura 10: Parte do código XML da árvore de decisão

2.1. Telas da interface

Na Figura 11 é apresentada a tela inicial do SiSeM contendo a primeira pergunta: *Como pode ser caracterizado seus sistema*? E mostra várias opções de respostas. A pergunta mostrada na Figura 12 questiona sobre o nível de complexidade da aplicação e permite 3 respostas: alta, média ou baixa. Outras perguntas podem ser apresentadas ainda, dependendo da resposta anterior. Neste exemplo, na Figura 13 (Farias, 2005; Farias e Schiel, 2006c) é apresentado o resultado de um diálogo completo, listando um *ranking* de metodologias, destacando atributosdas metodologias e as respectivas pontuações.

Figura 11: Interface inicial do SiSeM

Figura 12: Tela para preenchimento de respostas



Metodologias	Ferramentas de Apoio	Agilidade	Robustez	Quantidade de Artefatos	Uso de Padrões	Custo	Reusabilidade	Complexidade	Ambiente	Paradigma	Total
COMPORM					15	0		10	20	20	65
AUML					20	0		0	10	20	50
Gaia					0	0		10	15	20	45
Bastos					15	0		0	10	20	45
RUP					0	20		0	0	0	20
CommomKADS					0	0		0	0	10	10
MASCommonKADS					0	0		0	0	10	10
COMOMAS					0	0		0	0	10	10
XP					0	0		0	0	0	0
Catalysis					0	0		0	0	0	0
YP					0	0		0	0	0	0
POKER					0	0		0	0	0	0

Figura 13: Resultado - Ranking com as metodologias, critérios e pontuações

3. Conclusão

Neste anexo foi apresentado o SiSeM - Sistema para Seleção de Metodologias de Desenvolvimento de Software. Este sistema auxilia o engenheiro de software a identificar qual a metodologia de desenvolvimento de software é mais adequada ao projeto e o ambiente em questão.

O trabalho desenvolvido é um sistema de suporte à decisão interativo e automatizado. O resultado do diálogo apresenta um *ranking* das metodologias mais adequadas à aplicação a ser desenvolvida considerando o respectivo ambiente de desenvolvimento. Ele prevê a incorporação de novas metodologias por meio de uma rotina de atualização da ávore de decisão que está em XML, de forma que possa acompanhar novas tendências e paradigmas de sistemas de software.

O uso do sistema possibilitará que o engenheiro de software evite a aplicação de um processo inadequado, que poderá resultar em uma sobrecarga significativa de atividades supérfluas. Possibilitará um aumento significativo na produtividade da equipe, produzindo um software com mais qualidade, além de prazo e orçamento menores.

Como trabalho futuro o sistema deverá considerar como melhor solução para o engenheiro de software, não somente metodologias integrais, mas também, a composição de processos específicos para certas fases do desenvolvimento, como engenharia de requisitos, métodos de teste de software, etc.

ANEXO III - ERA

Editor de Redes de Atividades

O *Editor de Rede de Atividades* possibilita a manipulação das atividades de um processo de desenvolvimento de software, seguindo as especificações e regras de uma Rede de Projetos para modelar Redes de Atividades e acompanhar tempos e custos das atividades e das pessoas a elas relacionadas. Essa representação gráfica, unida com as métricas de tempo, custo e recursos é o grande diferencial deste editor, em relação à outros (*Microsoft Vison, Meta Design, Design CPN*, etc.) pois permite ter noção visual e total do projeto.

No Capítulo 5 foram apresentados as principais funcionalidades do ERA na forma de diagramas de casos de uso e as respectivas explicações. Neste anexo são destacados detalhes do projeto e da implementação do editor.

O Editor de Redes de Atividades é composto por 5 pacotes (Figura 1) relacionados entre si.

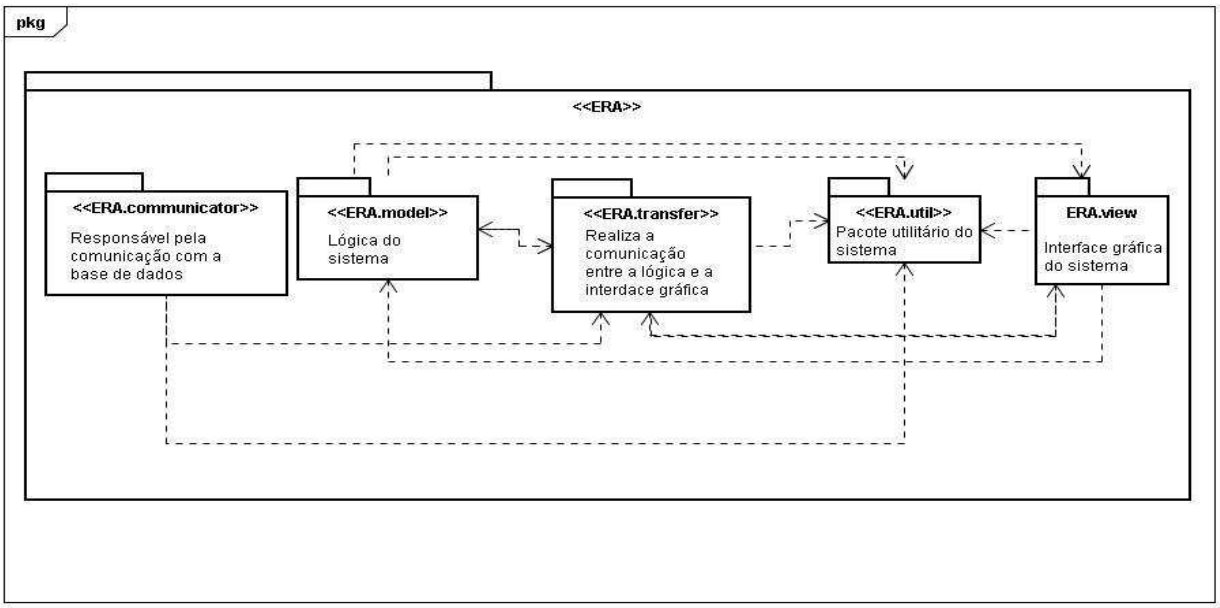


Figura 1: Sub-sistemas do ERA

Na Figura 2, a seguir, é mostrado o modelo conceitual do sistema.

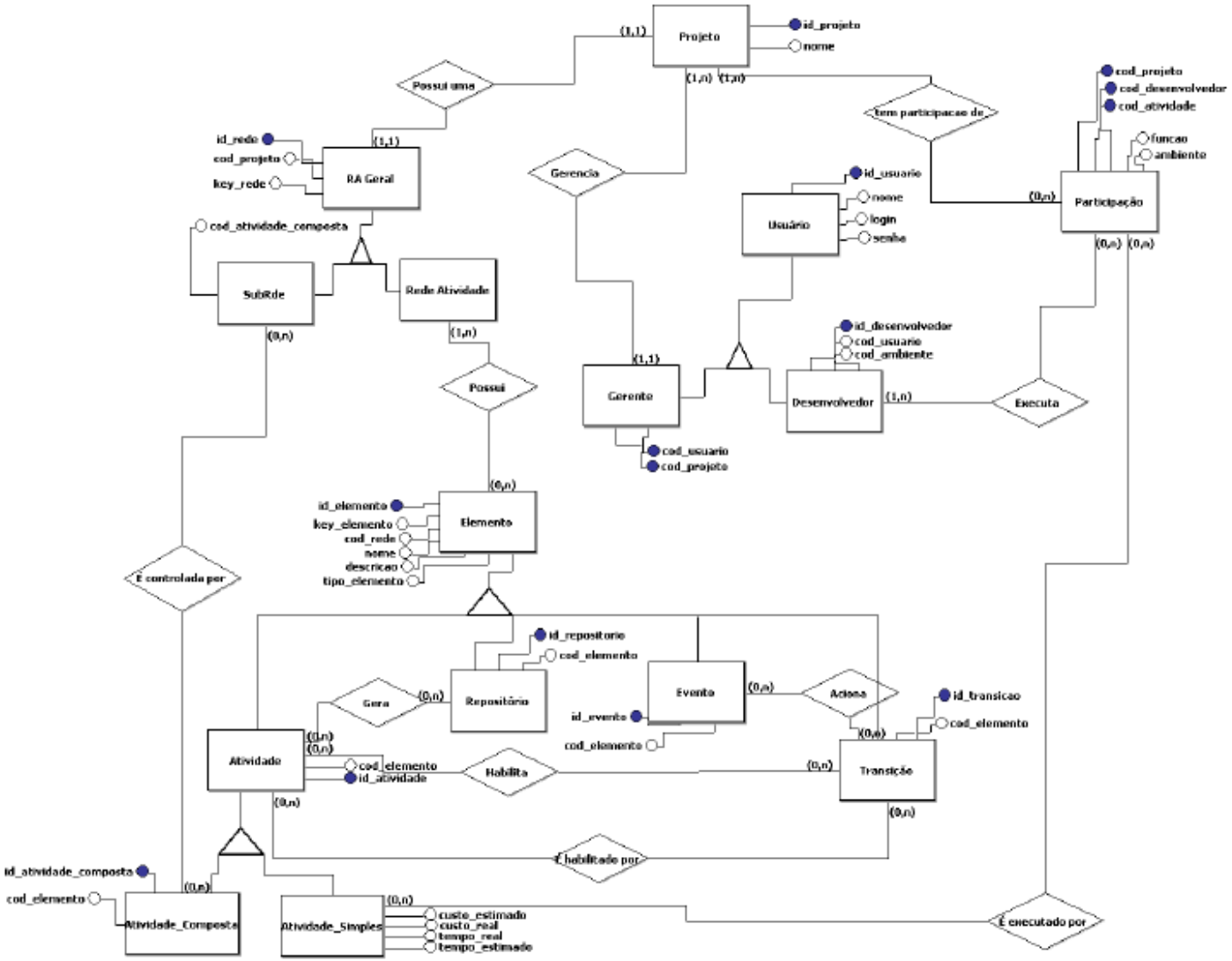


Figura 2: Modelo conceitual do ERA

2. Implementação

Na implementação do módulo ERA participaram três alunos do curso de Ciência da Computação da UFCG: Rodrigo Cerqueira Lopes, Júlio Zinga Suzuki e Márcio Dorimar que cursaram as disciplinas Projeto I e II (Zinga, Lopes e Dorimar, 2007) e Rodrigo Pinheiro também aluno de graduação, participante da etapa de análise.

O sistema foi implementado em Java, e a utilização da Interface Gráfica. É por meio de um applet acessado em um browser, deixando-o suficientemente portável para qualquer sistema operacional que suporte Java (que possui Java Virtual Machine).

O acesso aos dados persistentes é feito remotamente através de Sockets implementados em Java e seu servidor faz conexão com MySQL versão 5.0. É necessário que o banco de dados esteja na mesma máquina que o servidor de onde o *browser* carrega o *applet*.

O *applet* utiliza a versão 1.6.4 do *framework* JUNG para produzir e manipular os grafos na tela.

Em seguida é apresentado o Dicionário de Dados do ERA.

2.1 Dicionário de Dados

Tabela: **Projeto** **Nº:** E – 01

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada:

Descrição: Armazena as informações relacionadas a um projeto.

Atributo	Tipo	Tamanho	Requerido	Descrição
#+ Id_projeto	Int	10	Não nulo	Código único de identificação do projeto
Nome	varchar	45	Não nulo	Nome do projeto
Descricao	text	-	-	Descrição do projeto
dataInicial	date	-	Não nulo	Data inicial do projeto
dataFinal	date	-	Não nulo	Data final do projeto
ultimoIdGlobal_ERAGraph	int	10	Não nulo	Armazena a última chave do componente gráfico do ERA
ultimoIdGlobal_ElementoAtividade	int	10	Não nulo	Armazena a última chave do elemento Atividade da Rede de Atividades
ultimoIdGlobal_ElementoAtividadeComposta	int	10	Não nulo	Armazena a última chave do elemento Atividade Composta da Rede de Atividades
ultimoIdGlobal_ElementoTransicao	int	10	Não nulo	Armazena a última chave do elemento Transição da Rede de Atividades
ultimoIdGlobal_ElementoEvento	int	10	Não nulo	Armazena a última chave do elemento Evento da Rede de Atividades
ultimoIdGlobal_ElementoRepositorio	int	10	Não nulo	Armazena a última chave do elemento Repositório da Rede de Atividades

Tabela: **Usuário** **Nº:** E – 02

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada:

Descrição: Armazena as informações relacionadas a usuários.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ Id_usuario	Int	10	Não nulo	Código único de identificação do usuario
Nome	varchar	100	Não nulo	Nome do usuário
Descricao	text	-	-	Representa uma descrição do usuário
email	varchar	45	Não nulo	Representa o email do usuário
telefone	varchar	15	Nulo	Representa o telefone do usuário
login	varchar	45	Não nulo	Representa o login do usuário.
senha	varchar	255	Não nulo	Representa a senha do usuário.

Tabela: **Função** **N°:** E – 03

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada:

Descrição: Armazena a função do usuário em uma determinada atividade, ou seja, se ele é um programador ou gerente.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ Id_funcao	Int	10	Não nulo	Código único de identificação da função
funcao	varchar	50	Não nulo	Descrição da função do usuário.

Tabela: **Desenvolvedor** **N°:** E – 04

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada: usuário, ambiente

Descrição: Armazena informações dos usuários que são desenvolvedores do projeto.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ Id_desenvolvedor	Int	10	Não nulo	Código único de identificação do desenvolvedor
* cod_usuario	int	10	Não nulo	Código que relaciona o desenvolvedor a um usuário do sistema.
* cod_ambiente	int	10	-	Código que relaciona o desenvolvedor a um ambiente de projeto, ou seja, a um ambiente de software ou de hardware.

Tabela: **Gerente** **N°:** E – 05

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada: usuário, ambiente

Descrição: Armazena informações dos usuários que são gerentes do projeto.

Nome	Tipo	Tamanho	Requerido	Descrição
#*+ cod_usuario	int	10	Não nulo	Código que relaciona o desenvolvedor a um usuário do sistema.
#*+ cod_ambiente	**int**	10	-	Código que relaciona o gerente a um ambiente de projeto, ou seja, a um ambiente de software ou de hardware.

Tabela: **Elemento** **N°:** E – 06

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada: rede

Descrição: Armazena informações referentes a cada elemento da Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_elemento	int	10	Não nulo	Código único de identificação do elemento
tipo_elemento	int	10	Não nulo	Relaciona qualquer elemento da Rede de Atividades a um tipo
* cod_rede	int	10	Não nulo	Relaciona cada elemento a uma Rede de Atividades
x	double	-	Não nulo	Posição x de um elemento.
y	double	-	Não nulo	Posição y de um elemento.
Nome	varchar	50	Não nulo	Nome do elemento.
descricao	tinytext	-	-	Descrição de um elemento.
key_elemento	int	10	Não nulo	Armazena um código que representa qual componente está sendo instanciado, se é uma atividade, atividade composta, evento, transição ou repositório.

Tabela: Transição **N°:** E – 07

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada: elemento

Descrição: Representa a componente transição de uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_transicao	int	10	Não nulo	Código único de identificação da componente transição
* cod_elemento	int	10	Não nulo	Relaciona a componente transição a um elemento da Rede de Atividades.
eh_inicial	tinyint	1	-	Indica se uma transição está iniciando.
eh_final	**tinyint**	**1**	-	Indica se uma transição está terminando.

Tabela: Repositório **N°:** E – 08

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada: elemento

Descrição: Representa a componente repositório de uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_repositorio	int	10	Não nulo	Código único de identificação da componente repositório
* cod_elemento	int	10	Não nulo	Relaciona a componente repositório a um elemento da Rede de Atividades.
*** cod_elemento_real**	**int**	**10**	**-**	Código de onde se encontra o elemento repositório.

T**abela**: **Evento** **N°:** E – 09

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada: elemento

Descrição: Representa a componente evento de uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_evento	int	10	Não nulo	Código único de identificação da componente evento
* cod_elemento	int	10	Não nulo	Relaciona a componente evento a um elemento da Rede de Atividades.

Tabela: **Atividade_composta** **N°:** E – 10

Nome Simbólico: Atividade Composta **Tipo**: Entidade

Tabela Referenciada: elemento

Descrição: Representa a componente atividade composta de uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_atividade_composta	int	10	Não nulo	Código único de identificação da componente atividade composta
* cod_elemento	int	10	Não nulo	Relaciona a componente atividade composta a um elemento da Rede de Atividades.

Tabela: **Atividade** **N°:** E – 11

Nome Simbólico: Atividade **Tipo**: Entidade

Tabela Referenciada: elemento

Descrição: Representa a componente atividade de uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_atividade	int	10	Não nulo	Código único de identificação da componente atividade
* cod_elemento	int	10	Não nulo	Relaciona a componente atividade a um elemento da Rede de Atividades.
tempo_estimado	datetime	-	Não nulo	Armazena o tempo estimado para o término da atividade
tempo_real	datetime	-	Não nulo	Armazena o tempo real do término da atividade
custo_estimado	float	-	Não nulo	Armazena o custo estimado para o término da atividade
custo_real	float	-	Não nulo	Armazena o custo real do término da atividade

Tabela: **Aresta** **N°:** E – 12

Nome Simbólico: Aresta **Tipo**: Entidade

Tabela Referenciada: elemento

Descrição: Representa a componente aresta de uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#* cod_elemento_origem	int	10	Não nulo	Código que relaciona um elemento a origem da aresta
#* cod_elemento_destino	int	10	Não nulo	Código que relaciona um elemento ao destino da aresta
nome	varchar	50	-	Armazena o nome da aresta

Tabela: **Rede** **N°:** E – 13

Nome Simbólico: Rede de Atividades **Tipo**: Entidade

Tabela Referenciada: projeto, atividade_composta

Descrição: Representa uma Rede de Atividades.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_rede	int	10	Não nulo	Código único de identificação de uma Rede de Atividades
* cod_projeto	int	10	Não nulo	Código que relaciona uma rede de atividades a um projeto
* cod_atividade_composta	int	10	-	Relaciona uma rede a uma atividade composta, caso haja.
key_rede	**int**	**10**	Não nulo	Chave de identificação, única para cada projeto, gerado pela aplicação

Tabela: Participação **N°:** E – 14

Nome Simbólico: **Tipo**: Relacionamento

Tabela Referenciada: projeto, atividade, desenvolvedor, função, ambiente

Descrição: Armazena informações sobre um projeto.

Nome	Tipo	Tamanho	Requerido	Descrição
#*+ cod_projeto	int	10	Não nulo	Chave estrangeira para o projeto ao qual está inserido o elemento associado.
#* cod_atividade	int	10	Não nulo	Chave estrangeira para a atividade que esta participação está associada
#* cod_desenvolvedor	int	10	Não nulo	Chave estrangeira para o usuário desenvolvedor que faz esta participação.
* funcao	int	10	Não nulo	Relaciona os desenvolvedores a uma função no projeto

* **ambiente**	**int**	**10**	Não nulo	Armazena o tipo de ambiente, se é um ambiente de software ou de hardware.

T**abela**: Administrador **N°:** E – 15

Nome Simbólico: DBA **Tipo**: Entidade

Tabela Referenciada:

Descrição: Armazena informações sobre o usuário DBA do sistema.

Nome	Tipo	Tamanho	Requerido	Descrição
#+ id_dba	int	11	Não nulo	Código único de identificação de usuário DBA do sistema.
login	varchar	16	Não nulo	Armazena o login do DBA
senha	varchar	16	Não nulo	Armazena a senha do DBA

Tabela: XML **N°:** E – 17

Nome Simbólico: **Tipo**: Entidade

Tabela Referenciada:

Descrição:

Nome	Tipo	Tamanho	Requerido	Descrição
Cod_projeto	int	45	Não nulo	Código único para identificação de projeto.
lista_de_desenvolvedores	text	-	Não nulo	Armazena todos os desenvolvedores que participam de um projeto.
lista_de_gerentes	text	-	Não nulo	Armazena todos os gerentes que participam de um projeto.
lista_de_atividades	text	-	Não nulo	Armazena todas as atividades de um projeto.
lista_de_envolvidos	**text**	**-**	Não nulo	Armazena todos as pessoas que participam de um projeto.

ANEXO IV

Material utilizado no estudo avaliatório

Projeto SAD

CONTATO

Nome: _____

E-mail: _____

Telefone: ( ) _____

Celular: ( ) _____

Laboratório da Universidade: _____

Telefone trabalho: ( ) _____

Disponibilidade de fazer o teste nos seguintes dias da semana:_____

Horários preferenciais: _____

DECLARAÇÃO DE CONHECIMENTO DAS CONDIÇÕES DE TESTE

Você foi convidado a participar da avaliação do procedimento avaliatório de um ambiente integrado de gerência do desenvolvimento de projetos para acompanhamento, simulação e controle de custos, prazos e recursos de projetos utilizando Redes de Atividades (RA). Este procedimento está sendo conduzido por Cecir Almeida Farias doutoranda da COPELE – Coordenação de Pós-Graduação em Engenharia Elétrica na UFCG, sob a orientação do prof. Dr. Ulrich Schiel. Este formulário tem como objetivo informá-lo quanto aos detalhes desta pesquisa, listar seus direitos enquanto participante deste ensaio e cadastrá-lo formalmente como participante desta pesquisa.

No primeiro momento, você irá realizar a modelagem de um processo de desenvolvimento de software- eXtreme Programming - XP1 com a Rede de Atividades proposta no trabalho de tese, de acordo com a descrição metodológica elaborada pelo prof. Jaques Sauvé, e entregue durante este procedimento.

No segundo momento, lhe será entregue um roteiro de tarefas para realizar a modelagem com o auxílio de um *software* denominado GDP – *Gerenciamento de Projetos de Software*, onde será utilizado o módulo ERA- Editor de RAs.

O propósito deste procedimento avaliatório é coletar informações que nos permitam inferir recomendações destinadas à otimização da Rede de Atividades e do *software*, tornando-os tão eficazes e usáveis quanto possível. É importante ter em mente que não será você o alvo do procedimento e sim o ambiente em questão.

A sessão de testes será de 120 minutos, durante a qual serão feitos registros escritos e audíveis de detalhes pertinentes ao contexto da avaliação. Estes registros serão usados apenas para fins de avaliação e otimização do sistema. Não serão distribuídos nem consultados por indivíduos alheios a este procedimento avaliatório. Seu nome não será associado a quaisquer dados coletados neste procedimento.

Seus direitos enquanto participante são:

1. Você poderá solicitar a exclusão de sua participação da sessão de teste em qualquer instante por qual(ais)quer razão(ões) que você julgar convenientes.
2. Ao final da sessão, você poderá ver seus dados, caso julgue necessário. Se você decidir excluí-los do contexto do ensaio avaliatório, por favor, informe imediatamente sua decisão ao avaliador. Caso contrário, nos comprometemos de inviabilizar qualquer tentativa de identificação de seus dados por parte de terceiros.
3. Durante a realização do teste, você só poderá esclarecer dúvidas com o avaliador. Caso os esclarecimentos do avaliador ao seu questionamento possam comprometer a integridade dos dados ou polarizar sua opinião de algum modo, você será informado pelo avaliador, que poderá omitir a resposta.

Solicitamos que seja evitada qualquer discussão desta sessão com outros indivíduos, pertencentes ou não, ao grupo de usuários de teste deste ensaio avaliatório.

Finalmente, queremos agradecer-lhe o tempo e esforço despendidos durante a condução deste procedimento avaliatório. Tenha em mente que não há respostas certas ou erradas no contexto dos testes que você realizará, assim como não estão envolvidos aqui os conceitos de bom e mau desempenho. Esta sessão visa tão somente à identificação de problemas de usabilidade associados ao produto!

Se ainda houver questões relativas ao teor deste documento, formule-as verbalmente e poderei esclarecer quaisquer dúvidas relativas ao procedimento avaliatório, ou use o espaço abaixo, se preferir formulá-las por escrito.

QUESTÕES REMANESCENTES

_____

O preenchimento dos dados solicitados no Cadastro de Participação, em anexo, assim como a aposição de sua assinatura no espaço abaixo indicado, implicam o término da leitura deste documento e sua concordância total e voluntária na participação deste ensaio avaliatório da Universidade Federal de Campina Grande, *Campus I.*

Asssinatura	**Data**

	_____/_____/_____

Roteiro das Tarefas do experimento

Roteiro da pré-tarefa

Pré-tarefa:	**Verificação e organização do ambiente**
Tempo estimado:	
Roteiro:	Você está participando da equipe de testes de usabilidade de um trabalho de doutorado sobre o ambiente de Gerenciamento de Projetos com Redes de Atividades. Uma das metas é encontrar aspectos positivos ou dificuldades na abordagem proposta e na ferramenta que está sendo desenvolvida. Verifique se os papéis e os três formulários a serem utilizados estão disponíveis para seu uso e se sistema computacional está ativo. Caso não estejam, chame o observador dos testes e inicialize o sistema.

Roteiro da Tarefa 1

Tarefa 1:	Modelagem da Rede de Atividades da metodologia XP1.
Tempo Estimado:	60 min.
Roteiro:	Utilizando a descrição de XP1 (em anexo), desenhe a rede com os elementos adequados vistos durante o treinamento para representar as etapas (e sub-etapas) do processo e os respectivos artefatos gerados. OBS.: Caso encontre alguma dificuldade, avise o observador. Se o seu tempo para esta tarefa de teste for maior do que o tempo estimado, avise o observador. Ao terminar, vá para o roteiro da próxima tarefa.

Roteiro da Tarefa 2

Tarefa 2:	Inserção das atividades reais na Rede de Atividades (Adaptação)
Tempo Estimado:	10 min.
Roteiro:	Consulte a Tabela de Etapas (em anexo), abra a rede da tarefa 1, e altere a rede de atividades RA criada anteriormente. Você deve clicar com o botão direito do mouse em cada atividade, escolher a opção "editar atributo" e digitar o nome da atividade. Se o seu tempo para esta tarefa de teste for maior do que o tempo estimado, avise o observador. Ao terminar, vá para o roteiro da próxima tarefa.

Roteiro da Tarefa 3

Tarefa 3:	Inserção de pessoal, tempos e custos na rede.
Tempo Estimado:	20 min.
Roteiro:	1. Consulte a Tabela de Etapas (que possui atividades, cronograma e custos de um projeto real) e insira estes valores na RA criada na tarefa 1. Para isto, você deve clicar em uma atividade com o botão direito do mouse e escolher a opção ***editar atributos.*** Os valores de tempo e custos para a atividade devem ser inseridos na tela na ficha de ***atributos,*** nos respectivos campos de acordo com os valores da tabela anexa. 2. Os participantes devem ser alocados na ficha de ***desenvolvedores***, aqui também serão inseridos os nomes da equipe alocada, os valores de tempos e custos individuais. OBS: Você irá digitar os nomes dos desenvolvedores que já constam na tabela em anexo. Mas os tempos e custos individuais previstos devem ser digitados de acordo com seu bom senso e experiência em outros projetos (desde que estejam na faixa de tempo e custo já digitados na ficha de atributos do passo 1). OBS: Se o tempo para esta atividade for maior do que o tempo estimado, avise o observador. Ao terminar, vá para o roteiro da próxima tarefa.

Roteiro da Tarefa 4

Tarefa 4:	Totalização de valores de custos e tempos e simulação.
Tempo Estimado:	10 min.
Roteiro:	Verifique quem é o responsável por cada tarefa no sistema, finalize as atividades de um caminho de sua preferência, clicando no botão *concluir tarefa* e insira alguns dados de tempo e custos para finalizá-las. Ao terminar, passe o mouse por cima de algumas atividades e visualize os totais previstos, reais e remanescentes. Obs.: Ao terminar, indique ao observador que já terminou a tarefa, preencha o questionário sobre *Opinião do Usuário* e se desejar descrever sugestões ou mais comentários sobre o experimento, descreva na folha de anotações a parte.

Muito obrigado por sua participação !

Ficha de Registro de Eventos (Tarefas Individuais)

	Número do Participante:	**Data do Ensaio**	**Horário**	**Tempo de Execução Total das Tarefas**

Registro de Eventos de Teste

Legenda:

EI - Nº de escolhas incorretas
E - Nº de erros cometidos
PA - Número de pedidos de ajuda
T- Tempo de execução da tarefa

Tarefa	**Evento**				**Comentário**
1.	**EI**	AI	**PA**	**T**	
2.					
3.					
4.					

Questionário Pós-teste

Você e a Rede de Atividades											
ASPECTO		Escala									
		1		2		3		4		5	
		Concordo totalmente		Concordo parcialmente		Nem concordo nem discordo		Discordo parcialmente		Discordo totalmente	
1	A utilização da Rede de Atividades é adequada à modelagem do processo utilizado.										
2	Os elementos que compõem a rede são suficientes para modelar o processo adotado.										
3	Não existiu dificuldade na representação das etapas e sub-etapas do processo.										
4	Através da seqüência de atividades representadas, é possível acompanhar as etapas do processo modelado e verificar quais etapas e atividades a seguir.										
5	A visualização do processo através do grafo (da RA) é muito útil durante o processo de modelagem.										
6	A visualização do processo através do grafo (rede) é muito útil durante o acompanhamento do processo.										
7	Eu utilizaria a Rede de Atividades novamente para descrever as atividades do processo de desenvolvimento.										
8	O treinamento aplicado para o uso da rede foi suficiente.										
Você e o Software Editor de Redes de Atividades - ERA											
ASPECTO		Escala									

		1		2		3		4		5	
		Concordo totalmente		Concordo parcialmente		Nem concordo nem discordo		Discordo parcialmente		Discordo totalmente	
9	Com a utilização do editor se obtém automatização na modelagem da RA.										
10	Não existem dificuldades em interpretar os ícones que representam os elementos da RA.										
11	Não existiu dificuldade na adaptação de etapas da RA modelada inicialmente.										
12	Não existiu dificuldade na instanciação de atividades e sub-atividades do projeto na RA.										
13	Não existiu dificuldade na inclusão de valores (tempo e custo) do projeto para a RA.										
14	Não existiu dificuldade na simulação de totais (tempo e custo) do projeto.										
15	Eu utilizaria o módulo Editor de RAs novamente para modelar um processo de desenvolvimento.										
16	Eu utilizaria o módulo Editor de RAs novamente para acompanhar meu processo de desenvolvimento.										
17	Eu utilizaria novamente o módulo Editor de RA para dar suporte aos desenvolvedores e suas atividades.										
18	Eu utilizaria o módulo Editor de RAs novamente para gerenciamento de projetos.										
19	O treinamento aplicado para o uso do software foi suficiente.										
Uso e Navegação do Software											
ASPECTO		Escala									
		1		2		3		4		5	
		Muito fácil		Fácil		Nem fácil nem difícil		Difícil		Muito difícil	
20	Uso do software na realização das tarefas de interesse.										

21	**Localização dos itens de menu associados às tarefas.**										
22	**Uso das funcionalidades mais comuns encontradas no software.**										
23	**Compreensão das mensagens de erro apresentadas e recuperação destas situações.**										
24	**Processo de entrada e saída de dados durante o uso do software.**										

Quadro 1: Questionário com síntese de sondagem dos participantes principiantes do experimento

Nas perguntas abaixo, marque a alternativa que se aplica ao seu caso.

1. Determine seu nível de formação.
[15] Graduando [ ] Graduado
[ ] Mestrando [ ] Mestre
[ ] Doutorando [ ] Doutor

2. Determine a sua experiência no desenvolvimento de software
[7] Eu nunca desenvolvi software
[ ] Eu desenvolvi software para uso pessoal
[8] Eu desenvolvi software como parte de trabalhos de curso
[ ] Eu desenvolvi software em ambiente industrial

3. Determine a sua experiência na gerência de projetos de desenvolvimento de software
[8] Eu nunca fui gerente de projetos de software
[4] Eu fui líder de grupos de trabalho em projetos de curso
[3] Eu fui líder em poucos (≤ 3) projetos na indústria
[ ] Eu fui líder em muitos (> 3) projetos na indústria

4. Determine sua experiência com processos de desenvolvimento de software (XP, RUP, etc.)
[13] Eu nunca usei um processo de software
[2] Eu desenvolvi software com uso de um processo
[ ] Eu desenvolvi software com este(s) processo(s) em ambiente acadêmico
[ ] Eu desenvolvi software com este(s) processo(s) em ambiente industrial

5. Determine sua experiência com projetos de software
[14] Eu nunca participei de um projeto
[1] Eu utilizei o processo em poucos (≤ 3) projetos
[ ] Eu utilizei o processo em muitos (> 3) projetos

6. Por favor, indique sua experiência nas questões abaixo, assinaleo uma das alternativas da escala de 1 a 5, segundo os dados abaixo:
1 – nenhuma
2 – estudo em cursos ou livros
3 – praticado em projetos de curso
4 – praticado na indústria
5 – praticado em diversos (> 3) projetos da indústria

Item \ escala	1	2	3	4	5
Experiência em desenvolvimento de software orientado a objetos	4	1	10		
Experiência na definição de redes de tarefas (redes de PERT	13	2			
Experiência na definição de redes de PETRI	14	1			
Experiência em estimativas de tempo e custo de software	9	4	1	1	
Experiência na definição de atividades para equipes de trabalho	5	2	6	2	

Quadro 2: Questionário com síntese de sondagem dos participantes experientes

Nas perguntas abaixo, marque a(s) alternativa(s) que se aplica(m) ao seu caso.

1. Determine seu nível de formação.
[ ] Graduando [4] Graduado
[3] Mestrando [3] Mestre
[5] Doutorando [ ] Doutor

2. Determine a sua experiência no desenvolvimento de software
[ ] Eu nunca desenvolvi software
[ ] Eu desenvolvi software para uso pessoal
[3] Eu desenvolvi software como parte de trabalhos de curso
[12] Eu desenvolvi software em ambiente industrial

3. Determine a sua experiência na gerência de projetos de desenvolvimento de software
[ ] Eu nunca fui gerente de projetos de software
[3] Eu fui líder de grupos de trabalho em projetos de curso
[7] Eu fui líder em poucos (≤ 3) projetos na indústria
[5] Eu fui líder em muitos (> 3) projetos na indústria

4. Determine sua experiência com processos de desenvolvimento de software (XP, RUP, YP, etc.)
[ ] Eu nunca usei um processo de software
[2] Eu desenvolvi software com uso do(s) processo (s)_____
[3] Eu desenvolvi software com este(s) processo(s) em ambiente acadêmico
[10] Eu desenvolvi software com este(s) processo(s) em ambiente industrial

5. Determine sua experiência com projetos de software
[ ] Eu nunca participei de um projeto
[9] Eu utilizei o processo em poucos (≤ 3) projetos
[6] Eu utilizei o processo em muitos (> 3) projetos

6. Por favor, indique sua experiência nas questões abaixo, assinaleo uma das alternativas da escala de 1 a 5, segundo os dados abaixo:
1 – nenhuma
2 – estudo em cursos ou livros
3 – praticado em projetos de curso
4 – praticado na indústria
5 – praticado em diversos (> 3) projetos da indústria

Item \ escala	1	2	3	4	5
Experiência em desenvolvimento de software orientado a objetos	1		3	7	4
Experiência na definição de redes de tarefas (redes de PERT	12	3			
Experiência na definição de redes de PETRI	8	6	1		
Experiência em estimativas de tempo e custo de software	1	3	2	8	1
Experiência na definição de atividades para equipes de trabalho			3	8	4

Folha de anotações

Utilizada para anotação de observações do usuário sobre a Utilização da Rede de Atividades e do software ERA

1. Você sentiu falta de algum elemento da Rede de Atividades para modelar o processo adotado? Qual elemento? Justifique.

_____
_____
_____
_____
_____

2. Você considera que existiu alguma dificuldade na representação das etapas e sub-etapas, do processo adotado? Qual foi a dificuldade ?

_____
_____
_____
_____
_____

3. Você considera que existiu alguma dificuldade na inserção das atividades, recursos, tempos e custos na RA? Justifique sua resposta.

_____
_____
_____
_____
_____

4. Você sentiu falta de alguma informação para visualização e acompanhamento das informações apresentadas na rede sobre o projeto (atividades, recursos, tempos e custos)?

_____
_____
_____
_____

5. Você considera que existiu alguma dificuldade na simulação dos tempos e custos?

_____
_____
_____
_____