



Pós-Graduação em Ciência da Computação

**UMA METODOLOGIA DE PREDIÇÃO ESTATÍSTICA DE
PROJETOS BASEADA EM SIMULAÇÃO**

por

MARIANE MOREIRA DE SOUZA

Dissertação de Mestrado



Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, 2 DE MARÇO DE 2007

MARIANE MOREIRA DE SOUZA

**UMA METODOLOGIA DE PREDIÇÃO ESTATÍSTICA DE
PROJETOS BASEADA EM SIMULAÇÃO**

Dissertação apresentada ao Programa de Mestrado
em Ciência da Computação como requisito parcial
à obtenção do grau de Mestre em Ciência da
Computação
Universidade Federal de Pernambuco
Orientador: Prof. Dr. Alexandre Marcos Lins de
Vasconcelos

RECIFE, 2007

DEDICATÓRIAS

para minha família.

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus, que sempre me dá forças para seguir adiante. De maneira especial, agradeço aos meus pais, sem os quais eu nada seria. A meu namorado Humberto, por todos os momentos felizes e, principalmente, por estar sempre ao meu lado, me ajudando nas horas mais difíceis. Ao professor Alexandre Vasconcelos, por sua orientação, apoio e atenção, e por ter acreditado e confiado em mim. Ao amigo Sandro, pela orientação e amizade. A minha grande amiga Juliana, colega de trabalho, com quem compartilhei momentos felizes. A Regina, que me recebeu prontamente em sua casa em Recife. A todos que auxiliaram, direta ou indiretamente, no desenvolvimento desse trabalho.

Todos vocês são muito importantes para mim.

UMA METODOLOGIA DE PREDIÇÃO ESTATÍSTICA DE PROJETOS BASEADA EM SIMULAÇÃO

RESUMO

Muito se tem discutido sobre os problemas enfrentados pela indústria de software, os quais podem estar levando a mesma à estagnação. De acordo com pesquisas realizadas na área, além da baixa qualidade dos produtos, e produtividade não condizente com a demanda, a imprecisão de estimativas é um dos problemas cruciais enfrentados pela indústria de software.

O atraso na entrega de projetos e a extrapolação de custos são conseqüências diretas da baixa qualidade das estimativas definidas. Em muitos casos, quando não colaboram para o fracasso total de um projeto, a imprecisão de estimativas pode causar prejuízos à organização contratada, tais como a insatisfação de seus clientes e o pagamento de multas de contrato.

Desta forma, é importante que sejam fornecidos meios, para que a organização possa prever, com maior confiabilidade, as chances de um projeto futuro alcançar seus objetivos, dentro do prazo e custo definidos, facilitando, assim, o planejamento e a negociação com o cliente.

Neste sentido, esta dissertação apresenta, como principal contribuição, uma metodologia de predição de prazo de projetos, que utiliza simulação e métodos estatísticos. A metodologia tem como objetivo aumentar a qualidade do processo de estimativas das organizações, utilizando, para isso, informações de execuções passadas de projetos e características da equipe atual de desenvolvimento. Tais informações servem de base para a predição de um intervalo, ao qual, na prática, o término do projeto em análise terá chances de pertencer.

Os resultados de um estudo de caso, realizado em uma empresa de desenvolvimento de software, ao final da pesquisa, sugerem a capacidade da metodologia definida neste trabalho.

Palavras-chave: simulação, processo de software, predição, gerência de projetos, métodos estatísticos.

A METHODOLOGY OF STATISTICAL PREDICTION OF PROJECTS BASED ON SIMULATION

ABSTRACT

A lot have been discussed about the problems of software industry that can result in its stagnation. According to researches, besides bad quality of products and productivity not corresponded to demand, the definition of imprecise estimates is one of the crucial problems of software industry.

Deliveries out of stated periods and extrapolated costs are consequences of bad quality of estimates. In many cases, when it doesn't collaborate to the total failure of a project, it may cause damage to the organization, such as customer dissatisfaction and contract fines.

It's necessary, therefore, to provide ways, so that the organization can predict, with more trustworthiness, the possibilities of a future project to reach its objectives, in the defined stated period of time and cost, thus facilitating planning and negotiation with customers.

In this way, this dissertation presents, as its main contribution, a methodology of prediction of projects, using simulation and statistical methods. This methodology aims to improve the quality of estimates process, using, for this, information about past projects execution and characteristics of current development team. Such information is used to provide an interval, to which, in practice, the signoff date of the project will have possibilities to belong.

The results of a case study, performed on a software development organization, after the end of the research, suggest the capacity of the methodology defined in this work.

Keywords: simulation, software process, prediction, project management, statistical methods.

SUMÁRIO

1 INTRODUÇÃO.....	12
1.1 MOTIVAÇÃO E JUSTIFICATIVA.....	12
1.2 OBJETIVOS DO TRABALHO	14
1.3 CONTEXTUALIZAÇÃO.....	15
1.4 METODOLOGIA DE PESQUISA	17
1.5 ORGANIZAÇÃO DA DISSERTAÇÃO	17
2 GERÊNCIA DO PROCESSO DE SOFTWARE	19
2.1 TERMINOLOGIA DE PROCESSO DE SOFTWARE	19
2.2 CICLO DE VIDA DO PROCESSO DE SOFTWARE	21
2.3 AMBIENTES DE DESENVOLVIMENTO DE SOFTWARE ORIENTADOS A PROCESSO.....	24
2.4 CONSIDERAÇÕES FINAIS	26
3 MODELAGEM E SIMULAÇÃO DO PROCESSO DE SOFTWARE	28
3.1 MODELAGEM DE SISTEMAS.....	28
3.1.1 <i>Classificação de Modelos</i>	29
3.1.2 <i>Simulação de Modelos</i>	31
3.1.2.1 Simulação Discreta	31
3.1.2.2 Simulação Contínua	32
3.1.3 <i>Técnicas de Modelagem e Simulação</i>	33
3.1.3.1 Modelo de Eventos Discretos.....	33
3.1.3.2 Modelo Baseado em Diagrama de Estados	35
3.1.3.3 Modelo Baseado em Dinâmica de Sistemas.....	36
3.2 SIMULAÇÃO DE PROCESSO DE SOFTWARE.....	36
3.3 TRABALHOS RELACIONADOS	38
3.3.1 <i>Articulator</i>	39
3.3.2 <i>AgentProcess</i>	40
3.3.3 <i>SESAM</i>	42
3.3.4 <i>Virtual Team</i>	43
3.4 CONSIDERAÇÕES FINAIS	44
4 PREDIÇÃO DE PROJETOS DE SOFTWARE	46
4.1 VISÃO GERAL SOBRE PREDIÇÃO DE PROJETOS	46
4.2 CONCEITOS ESTATÍSTICOS APLICADOS À PREDIÇÃO DE PROJETOS.....	49
4.3 SIMULAÇÃO APLICADA À PREDIÇÃO DE PROJETOS.....	54
4.4 CONSIDERAÇÕES FINAIS	56
5 UMA METODOLOGIA PARA PREDIÇÃO DE PROJETOS.....	58
5.1 MODELO DE SIMULAÇÃO DO PROCESSO DE SOFTWARE INSTANCIADO	58
5.1.1 <i>Plano de Projeto</i>	63
5.1.2 <i>Tratamento de Habilidades e Afinidades dos Agentes de Simulação</i>	64

5.1.3 Algoritmo de Simulação	68
5.2 MODELO DE PREDIÇÃO DE PROJETOS.....	71
5.2.1 Escolha da Amostra de Projetos da Organização.....	72
5.2.2 Definição da Capacidade e Maturidade	73
5.2.3 Simulação: Predição de Término de Execução do Projeto.....	75
5.2.4 Predição do Intervalo de Término do Projeto.....	76
5.3 AVALIAÇÃO DO INTERVALO DE TÉRMINO.....	77
5.4 CONSIDERAÇÕES FINAIS	78
6 PROSIMULATOR: FERRAMENTA DE SIMULAÇÃO INTEGRADA A UM PSEE.....	79
6.1 O AMBIENTE IMPPROS	79
6.2 PROANALYSER: FERRAMENTA DE ANÁLISE DE ITENS DE PROCESSO DE SOFTWARE	83
6.3 PROSIMULATOR: FERRAMENTA DE SIMULAÇÃO DE PROCESSO DE SOFTWARE.....	85
6.3.1 Seleção do Processo Instanciado	86
6.3.2 Execução da Simulação.....	88
6.3.3 Análise e Validação do Modelo de Processo	89
6.3.4 Registro de Lições Aprendidas.....	91
6.3.5 ProSimulator: Exemplo de Aplicação	92
6.4 CONSIDERAÇÕES FINAIS	95
7 ESTUDO DE CASO: APLICAÇÃO DA METODOLOGIA DE PREDIÇÃO NO DOMÍNIO DE PROJETOS DE SOFTWARE	97
7.1 OBJETIVO DO ESTUDO DE CASO	97
7.2 A ORGANIZAÇÃO SwFACTORY	98
7.3 APLICAÇÃO DA METODOLOGIA DE PREDIÇÃO	98
7.3.1 Escolha dos Projetos.....	99
7.3.2 Cálculo do Atraso dos Projetos da Amostra	100
7.3.3 Definição da Capacidade e Maturidade	101
7.3.4 Simulação dos Projetos-Piloto	102
7.3.5 Predição do Intervalo de Término dos Projetos-Piloto.....	104
7.4 AVALIAÇÃO DA METODOLOGIA DE PREDIÇÃO NA ORGANIZAÇÃO SwFACTORY	105
7.5 CONSIDERAÇÕES FINAIS	106
8 CONSIDERAÇÕES FINAIS	108
8.1 CONTRIBUIÇÕES DO TRABALHO.....	108
8.2 PROPOSTA DE TRABALHOS FUTUROS.....	109
APÊNDICE A.....	116
MODELO DE OTIMIZAÇÃO APLICADO À ALOCAÇÃO DE DESENVOLVEDORES.....	116
APÊNDICE B.....	122
PROANALYSER: FERRAMENTA DE ANÁLISE DE ITENS DE PROCESSO DE SOFTWARE	122

LISTA DE FIGURAS

<i>Figura 1.1 – Metodologia de predição de projetos.</i>	14
<i>Figura 1.2 – Arquitetura do ambiente ImPProS [Oliveira et al. 2005].</i>	16
<i>Figura 2.1 – Meta-processo de software [Oliveira et al. 2005], adaptado de [Humphrey 1989].</i>	22
<i>Figura 3.1 – Modelo evento-discreto [Davies 1979].</i>	34
<i>Figura 4.1 – Aumento da maturidade e melhoria na previsibilidade, controle e efetividade [Paulk et al. 1994].</i>	49
<i>Figura 4.2 – Exemplo de histograma: distribuição normal de frequência do percentual de atraso de projetos de uma organização (adaptado de [Oliveira 2006]).</i>	51
<i>Figura 4.3 – Gráfico de dispersão: percentual de atraso de cada elemento de uma amostra de projetos.</i>	51
<i>Figura 4.4 – Curva de distribuição normal de probabilidades (adaptado de [Oliveira 2006]).</i>	52
<i>Figura 4.5 – Distribuição normal: probabilidade de ocorrência de elementos (adaptado de [Oliveira 2006]).</i>	54
<i>Figura 4.6 – Predição de projetos baseada em simulação: detecção de falhas de estimativas</i>	55
<i>Figura 5.1 – Arquitetura do Modelo de Simulação de Processo de Software.</i>	59
<i>Figura 5.2 – Modelo relacional: estrutura do Plano de Projeto.</i>	64
<i>Figura 5.3 – Algoritmo de simulação em pseudocódigo detalhado.</i>	69
<i>Figura 5.4 – Arquitetura do Modelo de Predição de Projetos.</i>	72
<i>Figura 5.5 – Definição da capacidade e maturidade da organização relacionada ao atraso na execução de projetos.</i>	74
<i>Figura 5.6 – Exemplo de cronograma simulado: diferença entre prazo de projeto estimado e simulado.</i>	75
<i>Figura 5.7 – Predição do intervalo de término do projeto.</i>	76
<i>Figura 5.8 – Avaliação do intervalo de término.</i>	77
<i>Figura 6.1 – Fluxo de execução da integração entre o ImPProS e o ProSimulator: simulação direta do Plano de Projeto [Oliveira 2005a].</i>	82
<i>Figura 6.2 – Estrutura do arquivo XML (Simulação direta do Plano de Projeto) [Oliveira 2005a].</i>	82
<i>Figura 6.3 – ProSimulator: fluxo de atividades.</i>	86
<i>Figura 6.4 – Diagrama de caso de uso geral: seleção do processo instanciado.</i>	87
<i>Figura 6.5 – Diagrama de caso de uso detalhado: seleção do processo instanciado.</i>	88
<i>Figura 6.6 – Diagrama de caso de uso geral: simulação do processo de software instanciado.</i>	89
<i>Figura 6.7 – Diagrama de caso de uso: análise e validação da simulação do modelo de processo.</i>	90
<i>Figura 6.8 – Diagrama de caso de uso: registro de lições aprendidas e encerramento da simulação.</i>	92
<i>Figura 6.9 – ProSimulator: exemplo de gráfico de gantt.</i>	93
<i>Figura 6.10 – ProSimulator: exemplo de relatório textual de simulação.</i>	93
<i>Figura 6.11 – ProSimulator: análise do processo instanciado no ProAnalyser após a simulação.</i>	94
<i>Figura 6.12 – ProSimulator: avaliação do ProAnalyser de item não-atendido na simulação do processo.</i>	94
<i>Figura 7.1 – Gráfico de dispersão: percentual de atraso dos projetos da organização.</i>	101
<i>Figura 7.2 – Relatório de teste de Shapiro-Wilk: verificação da normalidade da amostra de projetos.</i>	101
<i>Figura 7.3 – Projeto-piloto 2: predição pontual de término na ferramenta ProSimulator.</i>	103
<i>Figura A.1 – Cronograma de atividades e habilidades dos desenvolvedores.</i>	117
<i>Figura A.2 – Execução do cronograma: heurística de maior habilidade x busca local.</i>	117

<i>Figura A.3 – Modelo de Otimização de Alocação de Desenvolvedores.</i>	119
<i>Figura A.4 – ProSimulator: aplicação do Modelo de Otimização.</i>	120
<i>Figura B.1 – Diagrama de caso de uso da ferramenta ProAnalyser.</i>	123
<i>Figura B.2 – Fluxo de atividades: avaliação da relevância de itens de processo de software.</i>	123
<i>Figura B.3 – Fluxo de atividades: análise e avaliação de itens de processo de software.</i>	124
<i>Figura B.4 – ProAnalyser: apresentação.</i>	131
<i>Figura B.5 – ProAnalyser: análise automática de itens.</i>	132
<i>Figura B.6 – ProAnalyser: resultado da análise automática de itens.</i>	132
<i>Figura B.7 – ProAnalyser: resultado da avaliação do item “relacionamento entre as atividades”.</i>	133

LISTA DE TABELAS

<i>Tabela 5.1 – Cálculo da variação do tempo em função da habilidade do agente [Silva 2001].</i>	67
<i>Tabela 5.2 – Cálculo da variação do tempo em função da afinidade do agente [Silva 2001].</i>	67
<i>Tabela 6.1 – Conjunto de itens relacionados ao comportamento do processo.</i>	90
<i>Tabela 7.1 – Percentual de atraso dos projetos da organização.</i>	100
<i>Tabela 7.2 – Média e desvio padrão da amostra de projetos da organização.</i>	102
<i>Tabela 7.3 – Projeto-piloto 2: habilidades dos membros da equipe.</i>	102
<i>Tabela 7.4 – Projeto-piloto 2: afinidade entre membros da equipe.</i>	102
<i>Tabela 7.5 – Projetos-piloto: predição de término pela simulação.</i>	103
<i>Tabela 7.6 – Projetos-piloto: predição de término pela capacidade.</i>	104
<i>Tabela 7.7 – Projetos-piloto: predição do intervalo de término.</i>	105
<i>Tabela 7.8 – Projetos-piloto: avaliação do intervalo de término.</i>	106
<i>Tabela B.1 – Relação de itens analisados pela ferramenta ProAnalyser.</i>	127
<i>Tabela B.2 – Relação de itens analisados pelo usuário.</i>	127
<i>Tabela B.3 – Exemplo de resultados, consequências e soluções de um item.</i>	128
<i>Tabela B.4 – Gestão de problemas: mapeamento CMMI e ProAnalyser/ImPProS.</i>	129
<i>Tabela B.5 – Gestão de problemas: mapeamento MPS/BR e ProAnalyser/ImPProS.</i>	130

1

Introdução

Este capítulo introduz o trabalho descrito nesta dissertação, apresentando em linhas gerais a pesquisa desenvolvida. Na Seção 1.1 é apresentada a motivação e justificativa do trabalho realizado. A Seção 1.2 descreve, de maneira geral, os objetivos do trabalho, e a abordagem utilizada para alcançar tais objetivos. A Seção 1.3 apresenta um ambiente de implementação de processos, que contextualiza parte desta pesquisa. A Seção 1.3 apresenta a metodologia de pesquisa, que consiste de um estudo de caso, realizado para avaliação do trabalho desenvolvido. A Seção 1.5 apresenta a organização desta dissertação, definindo o assunto abordado em cada um dos capítulos.

1.1 Motivação e Justificativa

Atualmente, muito tem se discutido a cerca dos problemas enfrentados pela indústria de software. Segundo Pressman [Pressman 2002], existem problemas cruciais no desenvolvimento de software, que não se limitam ao funcionamento não adequado do produto, mas estão relacionados à maneira não adequada de se desenvolver o mesmo. Além da baixa qualidade, a produtividade não condizente com a demanda, e a imprecisão de estimativas são problemas críticos do desenvolvimento de software.

O estudo realizado pelo *Standish Group* [Standish 1995], chamado relatório do “*Chaos*”, examinou 8.000 projetos de software desenvolvidos por empresas americanas,

em meados da década de 90, identificando que, em média, um projeto concluído excedia seu orçamento em 90% e seu cronograma em 222%.

Segundo Paula [Paula 2003], ter estimativas de prazo e custo é uma expectativa mais do que razoável de clientes e gerentes. Porém, muitos gerentes não conhecem métodos técnicos de estimativas, outros não sabem estimar, e os que sabem, muitas vezes, trabalham em organizações onde não existe tal cultura.

Jones [Jones 1999] destaca que a ausência de um processo de gerenciamento apropriado, aliado às estimativas deficientes de custo e de tempo, são as principais causas das falhas dos projetos de software. Ainda que não resultem em fracasso total, a entrega de projetos fora do prazo combinado com o cliente pode gerar insatisfação, além de prejuízos financeiros para a organização contratada.

Alguns trabalhos, descritos na literatura, procuram resolver problemas de planejamento de projetos, utilizando abordagens para antever o comportamento dos mesmos, antes de serem executados na prática. Os trabalhos de Scacchi [Scacchi 1999] e Silva [Silva 2001] utilizam simulação para antever o comportamento de projetos, auxiliando no refinamento e melhoria contínua de processos; os trabalhos de Drappa e Ludewig [Drappa e Ludewig 2000] e Guedes [Guedes 2006] também utilizam simulação, porém, aplicada no contexto do treinamento, aprendizado e capacitação em gerência de projetos. Estes trabalhos serão apresentados de forma mais detalhada na Seção 3.3.

O trabalho descrito nesta dissertação define uma abordagem que procura amenizar o problema crítico da imprecisão de estimativas, contextualizado na disciplina de Planejamento de Projetos [PMI 2004], da Engenharia de Software. A abordagem definida é baseada em simulação e utilização de métodos estatísticos, e tem como objetivo auxiliar gerentes de projetos na definição de estimativas de prazo, com certo grau de confiança, facilitando o planejamento, e a negociação com o cliente.

1.2 Objetivos do Trabalho

Tomando como base o problema da imprecisão de estimativas, este trabalho apresenta, como principal contribuição, uma metodologia de predição estatística de prazo de projetos, baseada em simulação. Tal metodologia tem por objetivo aumentar a qualidade das estimativas, e pode ser utilizada como auxílio aos gerentes, durante a fase de planejamento dos projetos.

A metodologia consiste na aplicação de dois modelos, ilustrados na Figura 1.1.

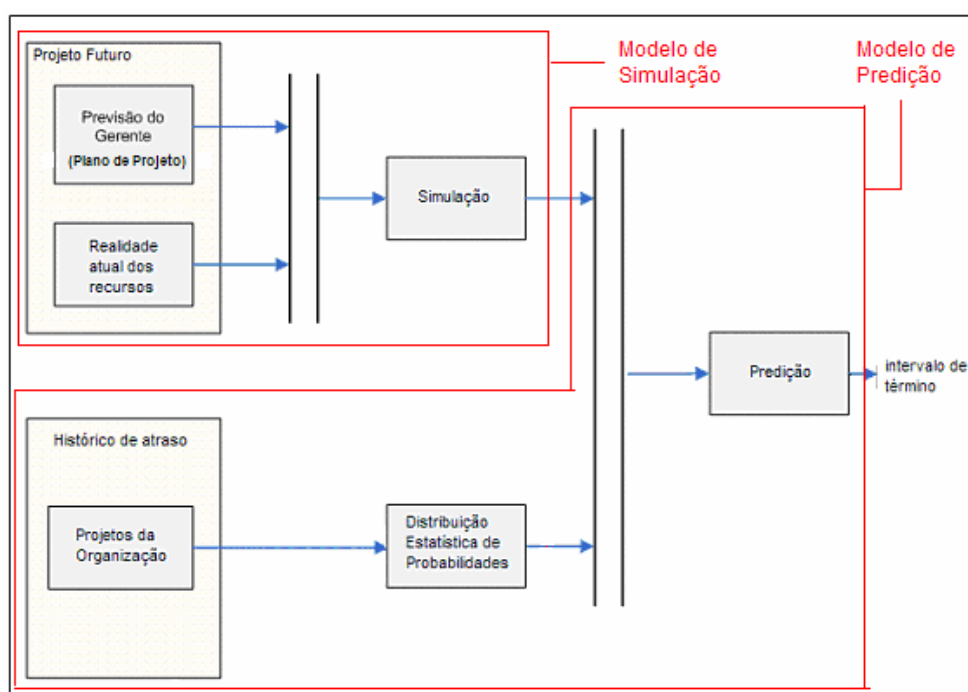


Figura 1.1 – Metodologia de predição de projetos.

O Modelo de Simulação realiza a simulação da execução de um projeto, combinando a estimativa do mesmo, definida pelo gerente, com a realidade da organização. Ao final de sua aplicação, o modelo prediz uma data de término para o projeto em análise.

O Modelo de Predição está relacionado com a análise estatística de uma amostra de projetos, já executados pela organização, que representam a mesma com relação ao

atraso. Tais informações são utilizadas junto à simulação para fornecer um intervalo de término para o projeto em questão.

De acordo com a Figura 1.1, inicialmente, a previsão do gerente de projetos (estimativas de prazo e custo) é definida em um plano de projeto, que será simulado, de acordo com a realidade atual dos recursos da organização (características pessoais de membros da equipe, disponibilidade de recursos financeiros, etc...).

Ao mesmo tempo, o histórico de projetos já executados pela organização é analisado, através da utilização de métodos estatísticos. Tal análise consiste na identificação de uma distribuição normal de probabilidades, que representa o comportamento da organização, com relação ao atraso na execução de projetos.

Finalmente, a predição de término fornecida pela simulação é utilizada junto à distribuição de probabilidades identificada, permitindo que o mecanismo de predição possa fornecer um intervalo ao qual, na prática, o projeto em questão terá, estatisticamente, 95% de chances de pertencer.

1.3 Contextualização

Parte da metodologia de predição, apresentada em linhas gerais na seção anterior, foi definida no contexto de um ambiente de implementação progressiva de processo de software (ImPProS). O ambiente ImPProS procura auxiliar as organizações na definição e implementação de seus processos, de maneira gradativa, de acordo com características específicas da mesma e dos tipos de projetos desenvolvidos [Oliveira et al. 2005]. A Seção 6.1 descreve com mais detalhes o ambiente ImPProS.

O Modelo de Simulação, que representa parte da metodologia de predição já apresentada, foi automatizado a partir do desenvolvimento da ferramenta *ProSimulator*. A ferramenta *ProSimulator* representa o módulo de simulação do ambiente ImPProS, ilustrado na Figura 1.2, que define a arquitetura do ambiente. O *ProSimulator* está relacionada como a fase de simulação de um ciclo de vida do processo de software. A

Seção 2.2 descreve com mais detalhes as fases de um ciclo de vida do processo, adotado pelo ambiente ImPProS, com base no trabalho de Humphrey [Humphrey 1989].

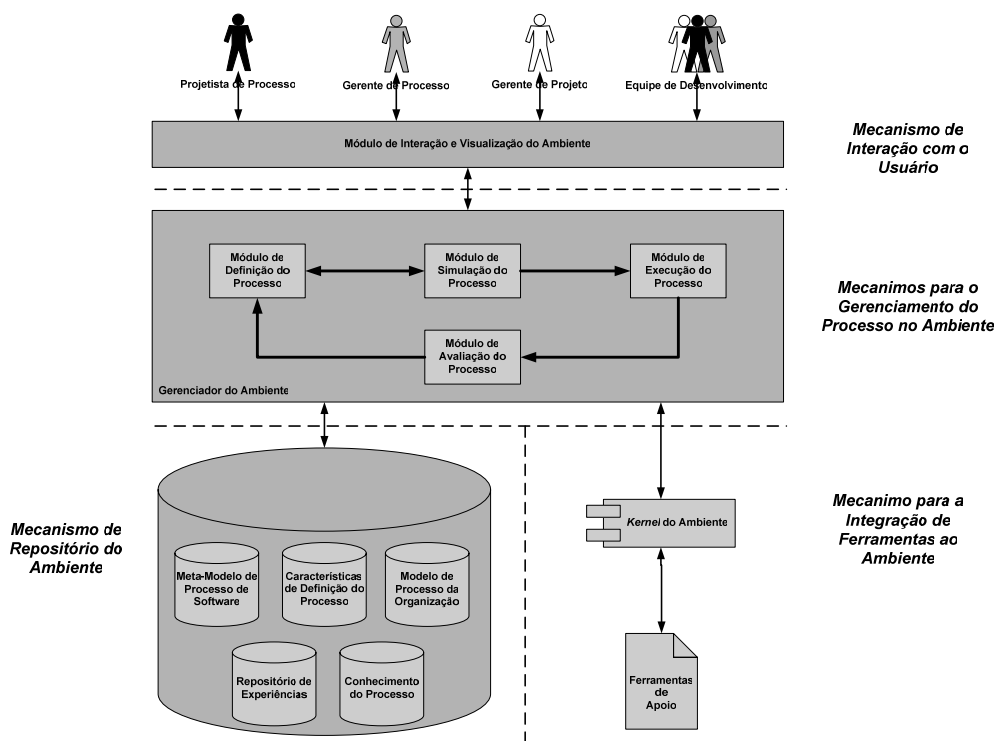


Figura 1.2 – Arquitetura do ambiente ImPProS [Oliveira et al. 2005].

A ferramenta *ProSimulator* realiza a simulação de processos, instanciados para projetos, no ambiente ImPProS, e fornece ao mesmo: uma pré-validação do modelo de processo; um conjunto de falhas detectadas durante a simulação, inerentes ao cronograma estimado pelo gerente de projetos; e a predição de uma data de término do projeto em análise. Mais detalhes sobre a ferramenta *ProSimulator* são descritos na Seção 6.3.

Um objetivo secundário da pesquisa realizada, definido no contexto do ambiente ImPProS, é auxiliar na melhoria contínua de processos. Ou seja, a partir da detecção de problemas comuns, durante a simulação de vários projetos, realizados seguindo um mesmo processo, é possível inferir melhorias, a serem realizadas em algum ponto deste processo, para que os problemas comuns detectados não sejam mais propagados em outros projetos.

1.4 Metodologia de Pesquisa

A fim de verificar o potencial da metodologia de predição, definida durante a pesquisa, foi realizado um estudo de caso, em uma empresa denominada *SwFactory*. A *SwFactory* é uma empresa de desenvolvimento de software de pequeno porte, situada na cidade de Lavras, ao Sul de Minas Gerais.

O estudo de caso foi realizado a partir da aplicação da metodologia de predição em projetos reais da organização. Seis projetos foram escolhidos para serem projetos-piloto, e outros quatorze foram escolhidos para compor a amostra de projetos, analisada segundo o atraso. As predições foram feitas com base no nível de confiança de 95%, considerado razoável para testes estatísticos.

Ao final do estudo de caso, verificou-se que o término de todos os projetos-piloto, executados na prática, pertencia aos intervalos estimados pela metodologia, sugerindo a capacidade da mesma. O estudo de caso realizado é apresentado com mais detalhes no Capítulo 7.

1.5 Organização da Dissertação

Esta dissertação está organizada em oito capítulos, sendo o presente (Capítulo 1), uma introdução sobre o trabalho realizado. O Capítulo 2 descreve os principais conceitos relacionados ao domínio de processos de software, que servem de base para o entendimento do assunto abordado nesta dissertação; o Capítulo 3 apresenta uma visão geral sobre modelos, e descreve as principais técnicas de simulação, existentes na literatura, bem como os trabalhos relacionados na área de simulação de processos de software; o Capítulo 4 fornece uma visão geral sobre predição de projetos, descrevendo o uso de simulação e de conceitos estatísticos neste contexto; o Capítulo 5 descreve, com maiores detalhes, a Metodologia de Predição de Projetos, definida neste trabalho, a partir da descrição do Modelo de Simulação e do Modelo de Predição, que a compõem; o Capítulo 6 apresenta a ferramenta de simulação *ProSimulator*, desenvolvida com base

no Modelo de Simulação (Capítulo 5), e utilizada no contexto do ambiente de implementação de processos ImPProS; o Capítulo 7 apresenta o estudo de caso, realizado em uma pequena empresa de desenvolvimento de software, a partir da aplicação da Metodologia de Predição de Projetos. Finalmente, o Capítulo 8 apresenta uma conclusão sobre o trabalho desenvolvido, resumindo as principais contribuições do mesmo, bem como a proposta de possíveis trabalhos futuros.

2

Gerência do Processo de Software

Neste capítulo são apresentados conceitos relacionados ao gerenciamento de processos de software, necessários para o entendimento do trabalho descrito nesta dissertação. A Seção 2.1 apresenta os conceitos básicos que definem a terminologia de processo de software. A Seção 2.2 descreve as fases de um possível ciclo de vida do processo de software. A Seção 2.3 caracteriza os ambientes de desenvolvimento de software orientados a processo, que contextualiza parte deste trabalho. A Seção 2.4 apresenta as considerações finais sobre o capítulo.

2.1 Terminologia de Processo de Software

Uma definição de processo de software criada e comumente aceita pelo SEI (*Software Engineering Institute*) é a de que um **processo de software** representa um conjunto de atividades, métodos, práticas e transformações usadas por pessoas para desenvolver software [Paulk et al. 1993]. Reis, no trabalho [Reis 1998], define processo de software como um conjunto de passos de processo, parcialmente ordenados, relacionados com conjuntos de artefatos, pessoas, recursos, estruturas organizacionais e restrições, e tem como objetivo produzir e manter os produtos de software finais requeridos.

Uma **atividade** do processo é um passo que produz mudanças de estado visíveis externamente no produto de software. A descrição de uma atividade define suas características, que formam um perfil de atividade, seu cronograma (datas iniciais e finais previstas), o conjunto de artefatos relacionados e suas relações de dependência com outras atividades [Oliveira et al. 2005]. Uma atividade possui ainda recursos a serem alocados (por exemplo, máquinas e orçamento), é escalonada, monitorada e atribuída a desenvolvedores, que podem utilizar ferramentas para executá-la [Reis 2003]. Existem atividades que não necessitam de intervenção humana, sendo chamadas atividades automáticas.

Um **agente** está relacionado com as atividades de um processo, e pode ser uma pessoa ou uma ferramenta automatizada [Oliveira et al. 2005]. Os agentes podem assumir perfis, que serão analisados para atribuição de atividades, durante a execução do processo de software. Como exemplo de perfis assumidos pelos agentes, pode-se citar Gerente de Projetos e Engenheiro de Software.

Um **artefato** é um produto criado ou modificado por uma atividade, durante um processo, podendo ser reutilizado pela mesma ou por outra, para fabricação de novos produtos. Desta forma, uma atividade pode consumir artefatos (de entrada), e gerar novos artefatos (de saída) [Oliveira et al. 2005]. Frequentemente, os artefatos são persistentes e possuem versões [Reis 1998]. Geralmente estão relacionados com marcos de entrega de um processo em execução.

As entidades descritas acima, bem como seus relacionamentos e restrições formam os componentes básicos que definem um modelo de processo de software. Segundo Acuña e Ferré [Acuña e Ferré 2001], um modelo de processo de software é uma representação abstrata da arquitetura, projeto ou definição do processo de software.

Após a definição de um modelo de processo, é importante que o mesmo possa ser validado e executado na prática. Um modelo de processo instanciado, ou processo executável, é um modelo passível de ser verificado e executado por uma máquina de processo, em um ambiente de desenvolvimento de software (ADS), com capacidade de

suportar a modelagem e execução do processo (ADS orientados a processo) [Oliveira et al. 2005].

Segundo Pressman [Pressman 2002], um projeto é a instância de um processo com objetivos e restrições específicas. O *Project Management Body of Knowledge Guide* (PMBOK *Guide*) [PMI 2004] define projeto como o emprego de um esforço temporário para criar um produto, serviço ou resultado único. Neste contexto, a Gerência de Projetos tem como objetivo o planejamento, controle e monitoramento de um processo em execução, de acordo com os objetivos do projeto em questão.

Já a Gerência de Processos procura construir, analisar e verificar modelos de processo, para isso obtendo informações durante a execução desse processo, instanciado para um dado projeto, a fim de evoluir tais modelos para que possam ser usados posteriormente [Oliveira et al. 2005].

Desta maneira, a partir da definição de um modelo de processo e sua instanciação, uma organização pode iniciar a fase de gerenciamento do seu processo, permitindo sua execução, controle e monitoramento, promovendo uma melhoria contínua.

2.2 Ciclo de Vida do Processo de Software

Um processo de software é definido segundo as necessidades iniciais da organização. Porém, uma vez definido, o processo se encontra em constante evolução, causada por mudanças planejadas ou não dentro da organização. Desta maneira, existe um ciclo de vida para processo de software, análogo ao ciclo de vida de um produto de software. As atividades do ciclo de vida do processo de software são chamadas meta-atividades, e o processo de desenvolvimento e evolução do processo de software é chamado meta-processo.

A Figura 2.1, descrita no trabalho [Oliveira et al. 2005] (adaptado de Humphrey [Humphrey 1989]), mostra a interação entre as atividades básicas do meta-processo de software.

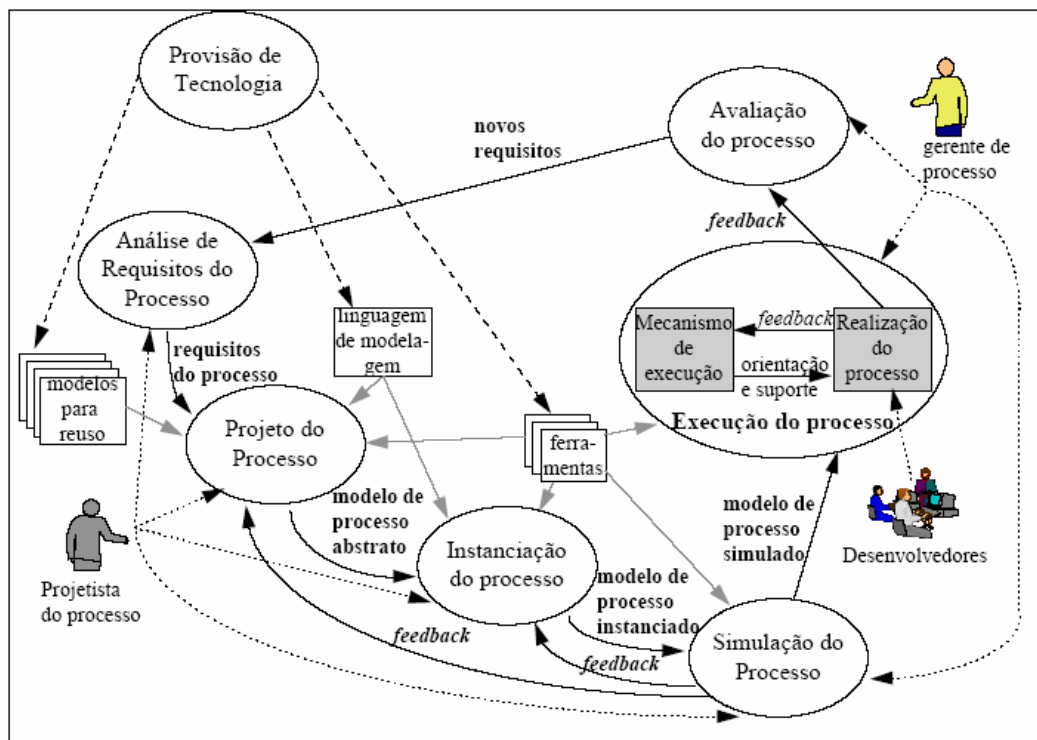


Figura 2.1 – Meta-processo de software [Oliveira et al. 2005], adaptado de [Humphrey 1989].

O projetista de processo é responsável pelas atividades relacionadas à modelagem, instanciação e simulação do modelo de processo a ser executado; Os agentes desenvolvedores atuam durante a fase de execução do processo; O gerente de processo é responsável pelo acompanhamento da execução e avaliação do processo, analisando seu desempenho. Os parágrafos seguintes resumem, com base no trabalho [Oliveira et al. 2005], as principais meta-atividades do meta-processo de software.

Durante a fase de Provisão de Tecnologia, é necessário suprir as necessidades tecnológicas, relacionadas ao desenvolvimento do produto de software e definição do modelo de processo, incluindo linguagens para modelagem, modelos para reuso e ferramentas de suporte ao gerenciamento do processo de software.

Na fase de Análise de Requisitos do Processo, são identificados requisitos para projeto de um novo processo ou de um processo existente. Isso pode ser feito através da utilização de sistemas de informação convencionais, ou mecanismos de aquisição de conhecimento, que procuram descobrir os principais requisitos e propriedades de um dado domínio, no caso, processos de software.

Durante a fase Projeto do Processo, são utilizadas as linguagens de modelagem já especificadas, para se definir a arquitetura do processo, de acordo com as propriedades e requisitos já definidos na fase anterior.

Na fase de Instanciação do Processo, o modelo de processo definido é então instanciado, adicionando-se informações detalhadas sobre prazos, agentes e recursos utilizados por cada atividade definida no processo.

Após a instanciação, a fase Simulação do Processo, que contextualiza parte do trabalho descrito nesta dissertação, assume um papel chave de verificação e validação dos processos definidos, antes de sua execução na prática no ambiente organizacional. Nesta fase, problemas relacionados à estrutura e execução do modelo, tais como atividades desencadeadas e prazos e custos extrapolados, podem ser identificados a priori, permitindo o refinamento do processo, e a replicação de casos de sucesso, a partir da reutilização de modelos verificados e validados. No caso deste trabalho, além de prover a validação de um modelo de processo, a simulação é utilizada como base, para se prover um intervalo ao qual o término de um dado processo em execução (projeto) terá chances de pertencer. Para isso serão utilizados ainda dados históricos relacionados a execuções reais de projetos da organização. Maiores detalhes sobre a Metodologia de Predição, definida neste trabalho, estão descritos no Capítulo 5.

A fase de Execução do Processo representa a realização do processo instanciado no mundo real. Isso é feito a partir da invocação de ferramentas de auxílio ao desenvolvimento do produto de software. O *feedback* de informações sobre o andamento do processo é coletado e analisado durante a execução;

Por fim, a fase de Avaliação do Processo, paralela à fase de Execução do Processo, procura avaliar quantitativamente e qualitativamente, o modelo de processo definido, a partir do *feedback* fornecido durante sua execução. As informações adquiridas a partir da avaliação são utilizadas na atividade Análise de Requisitos, realimentando o ciclo.

2.3 Ambientes de Desenvolvimento de Software Orientados a Processo

Atualmente, grande parte das organizações de desenvolvimento de software possui dificuldade em entregar produtos de qualidade, dentro dos prazos e custos definidos. Tais problemas estão diretamente relacionados à baixa qualidade do processo de desenvolvimento de software, e não do produto em si [Paula 2003].

Com o objetivo de solucionar este problema, várias tecnologias têm surgido, visando disciplinar o processo de desenvolvimento, estabelecendo etapas bem definidas, fornecendo um mecanismo de controle do processo de software. Inicialmente, a tecnologia CASE (*Computer Aided Software Engineering*) se destacou com a proposta de se utilizar software para auxiliar na produção de software. A partir da integração de ferramentas CASE, aliada à definição de uma estrutura unificadora de serviços, onde várias ferramentas, de diferentes métodos podem ser integradas, permitindo a comunicação e cooperação entre as mesmas, surgiram os ambientes de desenvolvimento de software (ADS) [Reis 2000].

A evolução dos ADS continuou com a utilização de outras tecnologias, tais como inteligência artificial, sistemas distribuídos, banco de dados não convencionais, dentre outras [Oliveira et al. 2005]. Porém, a definição de ambientes centrados em processo representa uma evolução significativa nos ADS. A partir da automação do processo de software, incorporada aos ADS mais recentes, surgiram os ambientes de

desenvolvimento de software orientados a processo, também conhecidos na literatura como PSEE (*Process-Centered Software Engineering Environment*) [Gimenez 1994].

Os PSEEs surgiram como uma abordagem para disciplinar um processo de software, a partir da definição de mecanismos para controle, acompanhamento, automatização e melhoria contínua do processo. Segundo Reis [Reis 2000], esses ambientes fornecem desde ferramentas integradas, para prover apoio ao desenvolvimento dos projetos de software, até mecanismos de implementação e controle das fases de um ciclo de vida do processo [Humphrey 1989]: definição, simulação, execução e avaliação do processo de software.

Reis, no trabalho [Reis 2000], define as principais funções genéricas que podem ser suportadas por PSEEs:

- Engenharia de Processos: definição e manutenção dos modelos de processo, provendo facilidade nas atividades de definição, análise e simulação dos processos de software.
- Engenharia de Software: desenvolvimento e manutenção de um produto de software através do seguimento de um processo de software.
- Gerência de Projetos: coordenação e monitoramento das atividades de engenharia de software, a fim de garantir que o processo está sendo seguido.

A mudança de um processo em uma organização não é uma tarefa simples. Daí a necessidade de prover mecanismos que auxiliem a organização na implementação de seus processos de acordo com sua realidade. O ambiente descrito por Oliveira, no trabalho [Oliveira et al. 2005], representa um ambiente de Implementação Progressiva de Processo de Software (ImPProS), que tem como objetivo principal fornecer um apoio automatizado para implementação dos processos de uma organização. O termo “progressiva” decorre do fato de que a implementação do processo é aperfeiçoada com as experiências aprendidas na sua definição, simulação, execução e avaliação. O

trabalho descrito nesta dissertação tem como um dos seus resultados a ferramenta *ProSimulator*, que representa o módulo de simulação do ambiente ImPProS. Maiores detalhes sobre a ferramenta *ProSimulator* e o ambiente ImPProS estão descritos no Capítulo 6.

2.4 Considerações Finais

Este capítulo teve como objetivo explicar os principais conceitos relacionados a processo de software, destacando a importância de uma organização em definir e conhecer seu(s) processo(s), para que a partir daí possam ser identificadas as principais deficiências relacionadas ao(s) mesmo(s).

Nos dias atuais, uma organização que desenvolva projetos com um mínimo de complexidade, comumente exigida pelo cliente, dificilmente irá sobreviver sem a cultura de processos. Ainda que não estejam formalizados, processos fazem parte da rotina de todo tipo de organização, e muitas vezes são seguidos intuitivamente pelas pessoas.

É comum que os colaboradores apresentem resistência na formalização e seguimento de processos mais burocráticos, especialmente em pequenas organizações, onde a informalidade é uma característica nata. Muitos não entendem os benefícios do seguimento e controle dos processos, enxergando tais atividades como entraves à sua capacidade e imaginação.

Tal conduta pode ser evitada, através do envolvimento de todos os membros da organização em programas de implantação e melhoria de processos. Neste caso, um conjunto de ferramentas que facilitem o trabalho dos colaboradores pode ser adotado, facilitando a visualização de melhorias a curto prazo.

Neste contexto, o trabalho descrito nesta dissertação contribui diretamente para a melhoria dos processos organizacionais, fundamental para manter a competitividade das

organizações, nos dias atuais. O ambiente ImPProS, através de um conjunto de ferramentas automatizadas, dentre elas a ferramenta *ProSimulator*, desenvolvida neste trabalho, permite que os membros da organização tenham mais facilidade em lidar com processos, diminuindo a resistência com relação aos mesmos, resultando na maior qualidade dos produtos.

Além disso, a longo prazo, a organização tende a obter uma maior maturidade na execução de seus processos, a partir da constante monitoração e avaliação dos mesmos, promovendo uma melhoria contínua.

3

Modelagem e Simulação do Processo de Software

Neste capítulo são definidos conceitos relacionados à modelagem e simulação de processos, encontrados na literatura. Também são apresentados alguns trabalhos desenvolvidos na área de Simulação de Processo de Software, analisados para a definição do modelo de simulação descrito neste trabalho. A Seção 3.1 apresenta uma visão geral sobre modelagem de sistemas, incluindo uma classificação geral de modelos, as técnicas de simulação discreta e contínua, relacionadas a modelos formais, e as técnicas de modelagem de processos por eventos discretos, diagrama de estados e dinâmica de sistemas. Na Seção 3.2 é fornecida uma visão geral sobre simulação de processo de software, apresentando as diferentes abordagens em que a mesma é geralmente utilizada. A Seção 3.3 apresenta os trabalhos Articulator, AgentProcess, SESAM, e Virtual Team, analisados durante o desenvolvimento deste trabalho. A Seção 3.4 apresenta as considerações finais sobre o capítulo.

3.1 Modelagem de Sistemas

Um sistema pode ser entendido como uma parte da realidade [Barros 2001]. A modelagem de sistemas visa à criação de modelos, que procuram definir os componentes de um dado sistema, bem como seus relacionamentos e restrições, com o objetivo de permitir a visualização de sua estrutura, e facilitar o entendimento do seu

comportamento, o que não seria simples pela observação do sistema real, infinitamente mais complexo.

Qualquer representação de um sistema é um modelo desse sistema [Silva 2001]. A grande vantagem da utilização de modelos está no fato dos mesmos representarem uma simplificação da realidade. Neste caso, pontos relevantes de um sistema (como, por exemplo, um processo de software) podem ser estudados e compreendidos de maneira mais facilitada, sem levar em conta características muito específicas do mundo real, consideradas muitas vezes irrelevantes para a compreensão do funcionamento do sistema como um todo.

Práticas de modelagem têm sido freqüentemente aplicadas em ambientes de desenvolvimento de software, possibilitando a geração de soluções, e a detecção prévia de problemas relacionados ao produto e processo de software [Sendall e Kuster 2004]. A modelagem de processo auxilia principalmente na fase de definição do seu ciclo de vida, incluindo as etapas de análise de requisitos, projeto e instanciação do processo.

Segundo Curtis [Curtis et al. 1992] e Armenise [Armenise et al. 1992], a definição de modelos de processo visa facilitar a comunicação e compreensão entre as pessoas da organização, a cerca de seus processos, promovendo o aperfeiçoamento e reutilização dos mesmos. Além disso, a modelagem fornece suporte à orientação, gerência e execução automatizada do processo.

3.1.1 Classificação de Modelos

Modelos possuem diversas classificações: os chamados modelos explícitos são descritos em uma linguagem, que permite seu compartilhamento entre indivíduos, além de facilitar a representação de restrições mais complexas do modelo do mundo real, não reconhecidas em modelos mentais (representados na mente humana). Modelos explícitos são úteis para difusão e validação do conhecimento, a cerca do domínio em questão, bem como para a previsão do comportamento do sistema representado [Barros 2001].

Modelos explícitos podem ser classificados quanto ao seu formalismo de representação, podendo ser: orientados a linguagens, como PML (*Process Modeling Language*), em que são definidos de maneira formal, por meio de regras ou *scripts*; e diagramáticos, em que são representados informalmente, por meio de *workflows*, existindo em alguns casos a abordagem híbrida [Murta 2002].

Quanto à sua execução, modelos explícitos podem ser classificados como formais ou informais. Modelos formais são passíveis de serem verificados e validados, uma vez que podem ser transformados em postulações matemáticas e lógicas. Um trecho de código, escrito em uma linguagem de programação, é um exemplo de modelo formal [Barros 2001]. Modelos informais permitem a definição estrutural de um dado sistema, porém, não permitem sua execução. No domínio de processo de software, tais modelos funcionam como guias para realização de atividades relacionadas ao processo. Exemplos de modelos informais são diagramas básicos UML (*Unified Modeling Language*) [OMG 2005a] ou modelos descritos na notação SPEM (*Software Process Engineering Meta-Model*) [OMG 2005b]. Apesar de não ser passível de execução, a representação informal de modelos explícitos já é de extrema importância, permitindo uma definição unificada de conceitos do modelo, facilitando o compartilhamento de experiências a cerca do mesmo.

De acordo com seu estado, modelos formais podem ser classificados em estáticos, em que os valores das variáveis do modelo não variam ao longo do tempo, ou dinâmicos, em que as variáveis podem apresentar valores distintos ao longo de sua execução. Modelos dinâmicos podem ser classificados em discretos e contínuos, de acordo com a variação do estado do modelo. Em modelos discretos, as variáveis mudam seus valores em determinados instantes de tempo não constantes, determinados pela ocorrência de eventos. Já em modelos contínuos, as variáveis alteram seus valores continuamente, em um intervalo de tempo constante, ao longo de sua execução [Barros 2001].

Modelos formais podem ser classificados ainda como determinísticos, em que uma mesma configuração dos parâmetros do sistema sempre produz o mesmo resultado,

ou estocásticos, onde é possível a representação de incerteza, de maneira que uma mesma configuração de parâmetros pode produzir diferentes resultados, de acordo com uma probabilidade de ocorrência de certos eventos [Barros 2001].

3.1.2 Simulação de Modelos

A importância da utilização de modelos formais está no fato dos mesmos serem passíveis de serem executados. Tais modelos podem ter seu comportamento simulado. Hoover e Perry [Hoover e Perry 1989] definem simulação como “o processo de desenvolver um modelo matemático ou lógico de um sistema real e então conduzir experimentos baseados em computador, usando o modelo para descrever, explicar e prever o comportamento de um sistema real”.

Técnicas de simulação são geralmente aplicadas em previsão e planejamentos quantitativos, na verificação de conseqüências, relacionadas à implantação de novos métodos, no estudo de novos sistemas, a fim de reprojeta-los ou refiná-los, na comunicação e aprendizado sobre um sistema real, dentre outros [Shimizu 1975]. Segundo Madachy e Boehm [Madachy e Boehm 1999], a simulação torna-se uma ferramenta importante para avaliação de um modelo, quando este não possui uma solução analítica, ou seja, uma solução através de uma fórmula fechada. Além disso, o papel da simulação é fundamental em modelos, onde o efeito da interação entre os componentes do mesmo, e o modelo como um todo, estão separados no tempo e no espaço.

A simulação baseada em computador se divide, na literatura, em dois segmentos distintos: Simulação Discreta e Simulação Contínua. As próximas seções irão caracterizar cada uma dessas técnicas.

3.1.2.1 Simulação Discreta

Segundo Barros [Barros 2001], na simulação discreta, o simulador possui uma fila de eventos, ordenada pelo tempo para ocorrência de cada um dos eventos. No domínio de processos de software, alguns exemplos de eventos relevantes seriam: início da

execução de uma tarefa, conclusão de uma tarefa, contratação de um desenvolvedor, chegada ou falta de um recurso, dentre outras.

Em cada ciclo de simulação, o simulador trata o primeiro evento da fila, alterando o relógio interno para o tempo de ocorrência desse evento, atualizando as variáveis do modelo de acordo com o evento. Neste caso, o tempo de ocorrência entre eventos não pode ser previamente determinado, e a simulação pode ocorrer em intervalos de tempo não constantes. A alteração nos valores das variáveis pode provocar a geração de novos eventos, ocasionando uma nova atualização das variáveis, e conseqüente geração de novos eventos. A simulação prossegue até que nenhum evento seja gerado ou até um limite de tempo pré-determinado [Barros 2001].

No domínio de processos de software, um exemplo da ocorrência de novos eventos, a partir de um dado evento, seria a execução de uma atividade, que consome todos os recursos do projeto, provocando falta de recursos. Isso acarreta no atraso de uma próxima atividade, e a conseqüente atualização de sua data inicial, para a data em que existam recursos suficientes disponíveis.

Exemplos de aplicações que utilizam simulação evento-discreta incluem escalonamento de processos e verificação de filas em gargalos de tráfego [Silva 2001]. Outros domínios de aplicação incluem planejamento de processos e *layouts* de fábricas, sistemas de transporte, redes de telecomunicações, dentre outros.

3.1.2.2 Simulação Contínua

A simulação contínua descreve sistemas por conjuntos de equações, para serem resolvidas numericamente, de forma algébrica ou diferencial, onde geralmente existe o tempo como uma variável independente [Rus et al. 1998].

Segundo Barros [Barros 2001], a simulação de modelos contínuos ocorre em intervalos infinitesimais, constantes e previamente definidos. As variáveis do modelo são alteradas a cada intervalo de simulação. Por exemplo, no caso de processos de software, a taxa de produtividade da equipe resultará em uma taxa de conclusão de tarefas, e ambas podem variar ao longo do tempo.

Em cada iteração do processo de simulação contínua, o simulador adianta um relógio interno, por um tempo equivalente ao intervalo de simulação. Em seguida, os valores das variáveis do modelo são recalculados. A simulação termina após a realização de um número de iterações previamente estabelecido [Barros 2001].

A simulação contínua possui aplicações em diversas áreas, incluindo representação do desenvolvimento de ecossistemas; representação da evolução de sistemas estelares; resolução de problemas relacionados ao balanço térmico de reatores nucleares, e ao escoamento de líquidos; validação de sistemas estocásticos na área de pesquisa operacional, dentre outros.

3.1.3 Técnicas de Modelagem e Simulação

As próximas seções apresentam, sucintamente, um conjunto de técnicas aplicadas à modelagem e simulação de processos, bem como alguns trabalhos em que as mesmas foram aplicadas.

3.1.3.1 Modelo de Eventos Discretos

O modelo evento-discreto descreve um sistema em termos de relações lógicas, que causam alterações de estado em pontos discretos, ao invés de atuar continuamente sobre o tempo. A natureza desse modelo está no fato de que seu estado se altera somente em conjuntos de pontos discretos, mas possivelmente randômicos conhecidos como “tempos de evento” [Silva 2001]. A Figura 3.1, descrita no trabalho [Davies 1979], ilustra o funcionamento de um modelo evento-discreto, onde a variável “T” indica o intervalo de mudança de estado do sistema de acordo com a ocorrência de um evento.

Um caso particular de Modelos de Eventos Discretos são Modelos de Simulação Baseada em Conhecimento ou Simulação Baseada em Regras. A simulação baseada em conhecimento resulta da combinação de simulação com tecnologias de sistemas baseados em conhecimento, utilizadas para implementar comportamentos inteligentes de especialistas humanos [Russel e Norvig 2003]. Uma das formas mais comuns de representar o conhecimento é através de regras de produção, baseadas na lógica. Essas

regras definem restrições, relacionadas ao domínio do problema, a serem armazenadas em uma base de conhecimento, que será acessada pelo mecanismo de inferência do sistema. Como um caso especial de modelos de eventos discretos, em um modelo de simulação baseado em regras, o estado do modelo é representado por fatos e suas ações por predicados, ambos definidos na base de conhecimento.

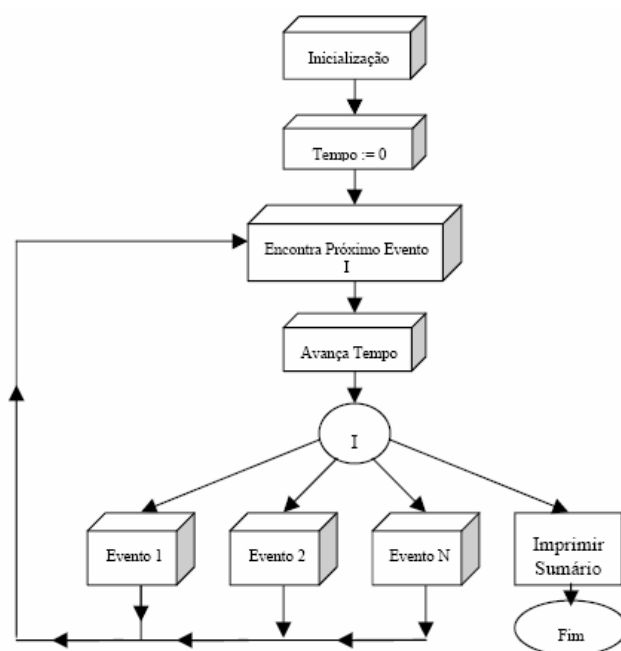


Figura 3.1 – Modelo evento-discreto [Davies 1979].

Rus [Rus et al. 1998] define um modelo evento-discreto, com o objetivo de avaliar a eficiência de políticas de aprimoramento de processos, e seus efeitos na confiabilidade dos produtos de software, construídos através destes processos. Algumas políticas analisadas incluem especificação formal de requisitos, inspeções, reutilização de software, etc. A viabilidade de tais políticas está relacionada com fatores inerentes ao produto e processo de software, e o resultado da combinação dessas políticas pode ser previsto através de simulação, permitindo a detecção do conjunto de políticas, que apresenta maiores vantagens de utilização, com relação ao projeto em questão.

Scacchi [Scacchi 1999] apresenta um modelo de projeto de software evento-discreto, baseado em conhecimento. Os predicados possuem condições dependentes do estado do sistema (fatos da base de conhecimento), e ao serem disparadas, executam procedimentos. A simulação baseada em eventos discretos é utilizada para predição do custo e tempo de realização do projeto, além do registro de histórico de fatos ocorridos durante o processo de desenvolvimento, permitido pela simulação baseada em conhecimento. O ambiente *Articulator* [Mi e Scacchi], [Scacchi 1999] está diretamente relacionado com este modelo e será descrito com maiores detalhes na Seção 3.3.1.

Silva [Silva 2001] também define um modelo de simulação evento-discreto, baseado em conhecimento, através de agentes inteligentes e cooperativos, que realizam tarefas de acordo com seu conhecimento armazenado. O Modelo de Simulação, apresentado nesta dissertação, foi definido com base em algumas características do modelo *AgentProcess* [Silva 2001]. Este modelo será apresentado com maiores detalhes na Seção 3.3.2.

3.1.3.2 Modelo Baseado em Diagrama de Estados

Diagramas de Estado são utilizados para representar todos os possíveis estados e transições, das entidades que compõem um dado sistema, ao longo de sua existência. As transições do diagrama, que representam uma mudança de estado, são disparadas por eventos, que dependem de condições lógicas, definidas pelo estado do sistema, avaliadas durante a simulação. Tais diagramas permitem também a descrição de ciclos de realimentação, permitindo que um estado possa ser alcançado a partir de si próprio.

Humphrey e Kellner [Humphrey e Kellner 1989] definem uma técnica para modelagem de processos de software baseada em diagramas de estado. A técnica consiste na identificação das entidades do processo, cuja dinâmica é representada por um diagrama de estados. Um estado do diagrama é ativo, quando uma entidade sofre uma transformação neste estado. Neste caso, o processo não é descrito por suas tarefas, mas pela dinâmica e evolução de suas entidades. A simulação é utilizada para avaliar a dinâmica das entidades envolvidas no processo.

3.1.3.3 Modelo Baseado em Dinâmica de Sistemas

Técnicas de Dinâmica de Sistemas podem ser aplicadas, para entender e influenciar a forma como os elementos de um sistema variam ao longo do tempo. A Dinâmica de Sistemas foi inicialmente aplicada nos campos da administração e engenharia, e progressivamente sua aplicação se deu na análise de sistemas sociais, econômicos, agrícolas, físicos, químicos, biológicos e ecológicos [Martin 1997]. A aplicação no domínio de desenvolvimento de software se iniciou com o trabalho [Abdel-Hamid e Madnick 1961]. Posteriormente tal trabalho foi estendido por outros ([Lin e Levary 1989]).

Segundo Barros [Barros 2001], a Dinâmica de Sistemas apresenta limitações, como a incapacidade de representação satisfatória de atributos das entidades dos modelos. Neste sentido, em seu trabalho [Barros 2001], Barros define uma abordagem baseada em cenários, que permite uma melhor representação dos componentes que definem o domínio de processos de desenvolvimento de software, além de permitir a representação de incertezas, que determinam a característica estocástica de tais ambientes.

O trabalho de Guedes [Guedes 2006] utiliza o meta-modelo de processo, definido por Barros, na definição de um modelo de simulação baseado em Dinâmica de Sistemas, utilizado na criação de um jogo denominado *Virtual Team*, que visa à capacitação de gerentes de projeto, com foco na gestão de pessoas. O trabalho de Guedes será apresentado com maiores detalhes na Seção 3.3.4.

3.2 Simulação de Processo de Software

Um modelo de processo de software definido pode conter falhas e inconsistências. Falta de recursos, ou existência de agentes com carga alta de trabalho são exemplos típicos de inconsistências relacionadas ao processo, que não são facilmente identificadas por meio

da representação estrutural do mesmo, através de um modelo. Neste caso, é importante que tal modelo possa ser verificado e validado.

Na prática, a maioria das empresas apresenta resistência à implantação ou mudança de seus processos, dificultando a validação de um modelo, a partir de sua execução real em uma organização. Além disso, o tempo levado para se conseguir resultados com relação à performance do processo, quando institucionalizado em uma organização, é geralmente grande, o que acaba inviabilizando essa prática. Neste sentido, a necessidade de se refinar e validar um modelo, antes de ser executado em um projeto real, tem gerado propostas diversas, dentre as quais se destaca a simulação de processo de software [Mi e Scacchi 1990].

Em ambientes de desenvolvimento de software, a simulação pode ser aplicada utilizando-se diferentes abordagens. Uma abordagem bastante conhecida e já mencionada utiliza a simulação para permitir a análise e validação de um modelo de processo, instanciado para um determinado projeto, permitindo que possíveis resultados sejam vistos, antes que o processo seja institucionalizado na organização. Essa validação é importante, porque pode ser que os resultados obtidos com a simulação da execução do modelo, não sejam condizentes com o previsto, e assim, pode-se evitar que projetos mal planejados sejam colocados em prática, evitando prejuízos. Além disso, vários cenários de um mesmo modelo podem ser simulados, possibilitando a execução na prática, da instância do modelo que apresente maior desempenho e qualidade. Esse estudo é denominado Análise de Cenário [Barros 2001].

Neste contexto, a simulação tem como objetivo principal o refinamento do modelo de processo definido, a partir da detecção de possíveis deficiências no mesmo, promovendo sua melhoria contínua. Isso faz com que a organização promova uma grande economia de tempo e recursos financeiros do projeto, bem como ganhos de produtividade e qualidade. Os trabalhos [Silva 2001], [Silva et al. 1999], [Mi e Scacchi 1990] e [Scacchi 1999] utilizam essa abordagem, na definição de modelos e ferramentas para simulação de processo de software. O Modelo de Simulação, detalhado na Seção 5.1, também utiliza essa abordagem.

Uma outra abordagem utiliza simulação como auxílio na comunicação organizacional, mostrando o funcionamento e evolução do processo, bem como o papel dos membros da organização durante a execução do mesmo [Madachy e Boehm 1999]. Outra abordagem relativamente similar utiliza a simulação como ferramenta de treinamento e aprendizagem [Martin 1997], possibilitando ao usuário a vivência de situações de tomada de decisão, próprias do cotidiano de um ambiente de desenvolvimento de software, e sua interação com o mesmo. Essa abordagem é normalmente utilizada em ambientes de jogos de simulação, que visam à capacitação de gerentes de projeto. Os trabalhos [Drappa e Ludewig 2000] e [Guedes 2006] definem modelos de simulação baseados em jogos, com foco na interação com o usuário, e na sua capacitação com relação à definição de modelos de processo e gerência de projetos.

Como já definido, apenas modelos formais são passíveis de serem executados e, portanto, terem seu comportamento simulado. Um modelo de processo de software executável é um modelo descrito formalmente, instanciado para determinado projeto, com seus respectivos desenvolvedores, prazos, orçamentos e recursos, passível de ser interpretado por uma máquina de processos [Silva et al. 1999].

3.3 Trabalhos Relacionados

Esta seção apresenta os trabalhos na área de simulação de processo de software, que foram analisados para a definição do Modelo de Simulação (Capítulo 5), e para o desenvolvimento da ferramenta *ProSimulator* (Capítulo 6), ambos relacionados ao trabalho descrito nesta dissertação.

Os trabalhos *Articulator* ([Mi e Scacchi 1990] e [Scacchi 1999]), e *AgentProcess* ([Silva 2001] e [Silva et al. 1999]) utilizam a abordagem de simulação, no contexto de estudo, validação e melhoria contínua dos processos; os trabalhos *SESAM* [Drappa e Ludewig 2000] e *Virtual Team* [Guedes 2006] utilizam a abordagem de simulação, no contexto do treinamento, aprendizado e capacitação de gerentes, na definição de processos e gerência de projetos.

3.3.1 *Articulator*

Mi e Scacchi ([Mi e Scacchi 1990] e [Scacchi 1999]) definem um ambiente para estudo de processos de software, pioneiro na utilização de agentes autônomos na simulação. Esse ambiente é definido, segundo o modelo evento-discreto, baseado em conhecimento, e provê um meta-modelo de processo de software, uma linguagem baseada em objetos, para especificar modelos de processo, e um mecanismo de simulação automatizado.

O ambiente *Articulator* permite a modelagem de tarefas, de acordo com a estrutura de precedência de sub-tarefas, especificada no modelo de processo instanciado. No modelo de simulação definido, agentes e tarefas consomem tempo e recursos. O comportamento dos agentes durante a simulação, se dá de acordo com alguns atributos, tais como experiência e especialidades, e também de acordo com informações armazenadas na base de conhecimento do simulador.

Sua arquitetura consiste de cinco sub-sistemas: uma base de conhecimento, que implementa um meta-modelo de processo, no qual se baseia a definição dos demais modelos; um mecanismo de consulta, que auxilia os usuários a acessarem informações a cerca do meta-modelo de processo, de maneira eficiente; um gerente de instanciação, responsável por gerenciar o relacionamento entre o meta-modelo, o modelo customizado ou especializado, e suas instâncias; um gerente de aquisição de conhecimento, que realiza a configuração de um dado modelo de processo instanciado, com os agentes e recursos disponíveis; e um simulador comportamental, que realiza a simulação do processo, criando uma trajetória sobre o período de desenvolvimento. O simulador comportamental tem como saída para seus usuários: as regras de produção utilizadas para a simulação, os eventos gerados a partir da execução dessas regras, e as regras que apresentaram conflito no momento do seu disparo, bem como a política para resolução de conflitos utilizada.

Os usuários do ambiente *Articulator* são pesquisadores de processo, que definem, testam e refinam diferentes modelos de processo; gerentes de projeto, que

definem, instanciam e simulam modelos de processo, com o objetivo de criar um plano de desenvolvimento a ser aplicado no ambiente organizacional; e desenvolvedores de software, que interagem com o *Articulator* através de um ambiente CASE.

O *Articulator* é um PSEE orientado à simulação. A ferramenta *ProSimulator*, desenvolvida no contexto deste trabalho, está relacionada com alguns componentes do *Articulator*: gerente de aquisição de conhecimento, responsável pela interação do modelo de processo instanciado, com um plano de execução, que define os agentes e recursos disponíveis para a execução do processo; e simulador comportamental, que realiza a simulação do processo instanciado, identificando falhas, de acordo com métricas de simulação previamente definidas. Os demais componentes do *Articulator* estão relacionados com outros módulos do ambiente ImPProS (Seção 6.1)

3.3.2 AgentProcess

AgentProcess ([Silva 2001] e [Silva et al. 1999]) é um modelo de simulação de processo de software evento-discreto, baseado em conhecimento, através de agentes inteligentes e cooperativos. A abordagem trata de maneira efetiva o caráter cooperativo do processo de desenvolvimento, ou seja, a possibilidade de atividades serem executadas por vários desenvolvedores, interagindo entre si para desempenhar a sua tarefa, caracterizando a arquitetura de sistemas multi agentes.

Dado um modelo de processo instanciado, ou seja, com papéis e recursos alocados e cronograma inicial previsto, o modelo de simulação disponibiliza atividades para os agentes desenvolvedores, através de uma agenda do agente, e estes utilizam seu conhecimento armazenado, para executar as atividades nas quais eles estão envolvidos.

O modelo possui ainda mecanismos, que levam em conta as afinidades e habilidades dos profissionais existentes na organização de desenvolvimento de software, de modo que essas características possam influenciar diretamente no processo de simulação. Por exemplo, na alocação de desenvolvedores em papéis, responsáveis pela execução de atividades, de acordo com suas habilidades, na alocação de

desenvolvedores para execução de atividades cooperativas, com base na afinidade entre os mesmos, dentre outros.

O modelo *AgentProcess* permite a descrição do modelo de processo instanciado a ser simulado, através da definição das atividades que o compõem, o cronograma dessas atividades e os recursos e desenvolvedores alocados para as mesmas.

De acordo com o modelo, cada atividade é executada por um ou mais agentes desenvolvedores, possui um estado (pronta, esperando, ativa e completa), necessita de recursos, possui um cronograma previsto e realizado, possui um conjunto de atividades das quais ela depende, artefatos que produz e um script. O script pode ser a descrição da tarefa a ser realizada, uma atividade automática, ou ainda um novo conjunto de atividades. Para cada atividade existe um conjunto de características, que funcionam como pré-requisitos para sua execução, devendo estar diretamente ligadas às habilidades dos agentes alocados para a mesma.

Existem dois tipos de agentes: Agente Simulador, responsável por coordenar a simulação, alocando as tarefas nas agendas dos agentes; e Agente Desenvolvedor, responsável pela execução das atividades encontradas em sua agenda, e pela tomada de decisões, de acordo com a percepção do seu ambiente, por meio de suas bases de conhecimento.

O Modelo de Simulação, definido no contexto desse trabalho, também utiliza a técnica de modelagem de eventos discretos. Apesar de não contemplar simulação baseada em conhecimento (agentes desenvolvedores e agente simulador não possuem base de conhecimento), o modelo foi baseado em algumas características do modelo *AgentProcess*: a contemplação de atividades cooperativas, executadas por mais de um agente; o tratamento de habilidades e afinidades dos agentes desenvolvedores, relacionado à alocação e tempo de tarefas; e o mecanismo de agenda do agente, que define as atividades em que o mesmo está alocado.

Diferente do modelo *AgentProcess*, o Modelo de Simulação deste trabalho utiliza o modelo evento-discreto, considerando as tarefas dispostas em uma fila por

ordem de data inicial prevista, e não de acordo com seus estados. A abordagem escolhida não permite a ocorrência de problemas de consistência relacionados à linha temporal de simulação, como por exemplo, a possibilidade de execução de atividades paralelas pelo mesmo agente desenvolvedor.

3.3.3 SESAM

O SESAM (*Software Engineering by Simulation of Animated Models*) [Drappa e Ludewig 2000] é um ambiente baseado em conhecimento, que suporta aprendizado colaborativo, através de um jogo entre o usuário, responsável pela criação do modelo, atuando como um gerente de projetos, e um jogador virtual, responsável por se comunicar com o usuário, compartilhando experiências e estudo sobre o modelo sendo criado. O objetivo é testar o usuário e fazê-lo vencer o jogo, caso o projeto seja conduzido com sucesso.

Durante o jogo, o usuário constrói e aperfeiçoa um modelo de projeto de software, em um ciclo de quatro fases, incluindo: a criação do modelo pelo usuário; a interação com o modelo por parte do jogador virtual, durante a simulação do mesmo, procurando verificar dificuldades e realizar questionamentos; a análise individual do resultado da simulação, feita pelo usuário; a análise conjunta do usuário e do jogador virtual, com relação à execução da simulação. O resultado é exibido ao usuário em forma de pontos obtidos no jogo.

O sistema usa três tipos de representações para seu modelo de simulação: o modelo de entidades e relacionamentos, que define as entidades e relacionamentos do ambiente modelado; regras, que descrevem o comportamento dinâmico dos participantes do projeto simulado; e o modelo de situação, que instancia a simulação e representa o contexto para o jogador.

O jogador virtual representa, na verdade, um Agente Simulador, que tem como objetivo principal, auxiliar o usuário na produção da aplicação requisitada, com qualidade, através de sugestões para distribuição de tarefas aos “desenvolvedores

simulados”, aumentando suas respectivas motivações, através do envolvimento dos mesmos com tarefas de suas preferências.

No trabalho descrito nesta dissertação, a simulação de processos é utilizada como uma abordagem para validação e melhoria contínua de modelos de processos de software instanciados, não sendo objetivo deste trabalho o treinamento e aprendizado de usuários, no contexto da execução de processos. Na abordagem de treinamento e aprendizado, o usuário interage com o ambiente, durante a simulação, fazendo parte do mesmo. Por outro lado, na abordagem de validação e melhoria do processo, a interação do usuário com o ambiente de processos simulado, não é geralmente permitida durante a simulação.

3.3.4 *Virtual Team*

O *Virtual Team* [Guedes 2006] é o primeiro protótipo, relacionado a um projeto multidisciplinar, denominado *SmartSim*, cujo objetivo principal é a definição de um *framework* de código aberto, pela licença LGPL, para o desenvolvimento de jogos sérios, baseados em simulação, que utilizam atores sintéticos para simular as personalidades envolvidas no processo simulado.

O *Virtual Team* é um jogo que utiliza simulação, baseada na Dinâmica de Sistemas, e tem como objetivo auxiliar na capacitação de gerentes de projetos, com ênfase em Gestão de Pessoas, promovendo um aprendizado com grande economia de tempo e recursos. O modelo de simulação utilizado no jogo é uma integração do processo de gerência de projetos do PMBOK (*Project Management Body of Knowledge*) [PMI 2004], com uma estrutura de processo instanciado baseada no RUP (*Rational Unified Process*) [Rational 1987].

O jogo *Virtual Team* consiste na simulação de cenários do cotidiano de um ambiente de desenvolvimento de software, em que os atores sintéticos, que são agentes inteligentes com características de personalidade [Silva 2000], são utilizados para modelar os participantes da equipe de desenvolvimento. O jogo proverá ao jogador

(gerente de projetos), a vivência de situações que envolvam decisões, na área de gerência de prazos, custos, riscos e escopo do projeto. Um exemplo de situação a ser vivenciada, seria a alocação de desenvolvedores, levando-se em conta a ausência repentina de membros do projeto, ou humor, nível de stress, motivação e produtividade dos mesmos.

O desafio proposto pelo *Virtual Team*, ao gerente de projetos, é o desenvolvimento do software dentro dos prazos e com os recursos disponíveis. O desempenho do jogador é obtido através da média ponderada de indicadores de dimensão (prazos e custos), utilizados pelo mesmo durante o jogo.

Como já mencionado, a abordagem de treinamento e aprendizado não é objetivo deste trabalho. Por outro lado, a simulação é utilizada para validação, visando a melhoria contínua de processos, definidos no ambiente ImPProS, além de auxiliar na predição de término de execução de projetos de uma organização.

3.4 Considerações Finais

Este capítulo apresentou diversos conceitos, relacionados à modelagem e simulação de processos. A aplicação de tais técnicas permite um melhor entendimento e compartilhamento comum de idéias sobre um processo; além de auxiliar no aprendizado e possibilitar a análise prévia do comportamento do mesmo.

No contexto do ambiente ImPProS, este trabalho utiliza a simulação para auxiliar na validação e refinamento de modelos de processo, definidos e instanciados no ambiente, visando sua melhoria contínua. Tal abordagem está diretamente ligada aos propósitos de modelos de qualidade como CMMI [Paulk et al. 1993] e MPS-BR [Softex 2005], que defendem a evolução e melhoria dos processos em níveis, de maneira contínua e incremental.

Através da análise dos trabalhos relacionados na literatura, percebe-se que grande parte utiliza a simulação como ferramenta de melhoria de processos ou como auxílio no treinamento e aprendizado em gestão de projetos. Muitos destes trabalhos se baseiam em características próprias da organização e de seus membros para realizar a simulação. Por outro lado, muitos não tratam a ocorrência de possíveis eventos, que podem atrasar atividades cruciais do projeto, simulando sua execução de maneira determinística. Um exemplo de evento comum em ambientes de desenvolvimento de software é a ausência de uma pessoa importante em um momento crítico do projeto.

É necessária a preocupação com eventos desta natureza, pois os mesmos estão ligados diretamente a riscos, que devem ser previstos durante o planejamento de projetos. Caso não seja contemplado na simulação, o impacto causado por tais eventos deve ser levado em conta por algum tipo de análise, possibilitando que os resultados fornecidos pela simulação possam ser “calibrados”, a fim de fornecer uma saída mais condizente com a realidade dinâmica dos ambientes de desenvolvimento de software.

No caso deste trabalho, a simulação é utilizada, principalmente, como ferramenta de auxílio na predição de término de projetos. Neste caso, a contemplação do possível atraso, relacionado a riscos de projeto, é feita fora da simulação, através da análise estatística de um histórico de projetos já executados pela organização. Desta forma, espera-se que o intervalo de término estimado para um projeto esteja mais condizente com a realidade organizacional.

4

Predição de Projetos de Software

Este capítulo define conceitos gerais relacionados à predição de projetos, aplicados no contexto de organizações de desenvolvimento de software. A Seção 4.1 define a importância em se prover maior confiabilidade ao processo de predição, evitando a extrapolação de prazos e custos de projeto na prática, funcionando ainda como ferramenta de incentivo à melhoria de processo de software. Na Seção 4.2 são apresentados métodos estatísticos, aplicados à predição de projetos, visando maior segurança no processo de predição. A Seção 4.3 define as vantagens do uso da simulação, no sentido de prover maior aproximação à realidade na predição estatística de projetos. A Seção 4.4 apresenta as considerações finais sobre o capítulo.

4.1 Visão geral sobre Predição de Projetos

Segundo Pressman [Pressman 2002] a imprecisão de estimativas é um dos problemas cruciais enfrentados pela indústria de software. A baixa qualidade e produtividade, o atraso significativo na entrega de produtos (mais de 200%), e a extrapolação de custos previamente definidos (mais de 90%) representam a realidade atual do desenvolvimento de software [Standish 1995].

Paula [Paula 2003] afirma que muitos gerentes não sabem de fato estimar, e realizam “previsões” com base em sua experiência, muitas vezes insuficiente, com relação à equipe de desenvolvimento. Isso pode ocasionar prejuízos financeiros à organização, além da insatisfação dos seus clientes.

Uma das maneiras de resolver este problema é dar condições para que a organização possa prever, de maneira confiável, as chances de um projeto futuro atingir seus objetivos dentro do prazo, custo e qualidade definidos. A partir da percepção da execução do projeto, a negociação de prazo pode ser feita com maior segurança, diminuindo o risco de atrasos.

Visando uma maior aproximação à realidade da organização, uma predição pode combinar a estimativa definida pelo gerente, com a natureza e disponibilidade atual de recursos da organização. Isso pode ser feito através de simulação, que prediz a execução de um dado projeto de maneira mais realista, através da utilização dos recursos humanos e financeiros disponíveis para o projeto, durante a execução do cronograma previsto pelo gerente.

Além disso, devido à natureza dinâmica e estocástica de um ambiente de desenvolvimento de software, a predição de um projeto deve considerar ainda o possível aumento no custo e tempo de execução de tarefas, causado pela ocorrência de eventos, associados aos riscos de projeto.

Caso não seja considerado na simulação, esse aumento de tempo pode ser tratado estatisticamente, pela utilização do histórico de projetos da organização. Neste caso, a predição é realizada com base na capacidade e maturidade da organização com relação ao atraso, identificados durante a análise de seu histórico de projetos. A partir daí, as organizações são incentivadas a aumentar a capacidade e maturidade do processo de estimativas, já identificadas, fornecendo maior confiabilidade na predição de seus projetos.

Segundo Oliveira [Oliveira 2006], a aplicação de práticas, definidas em modelos de qualidade como o CMMI [Paulk et al. 1993], que visam aumentar continuamente a

qualidade dos processos de uma organização, acarreta as seguintes melhorias (ilustradas na Figura 4.1, definida no trabalho [Paulk et al. 1994]):

- Previsibilidade: o amadurecimento da organização ocasiona uma maior previsibilidade, ou seja, menor será a diferença entre resultados esperados e realizados dos projetos, aumentando a validade e eficácia de previsões. Tal característica é comum em organizações no nível de maturidade 2 do modelo CMMI.
- Controle: organizações maduras conseguem um maior controle sobre seus projetos, resultando em uma menor variabilidade dos resultados observados ao redor dos resultados estimados, aumentando a confiabilidade das estimativas, e a eficiência do processo. Tal característica começa a se tornar comum em organizações no nível de maturidade 3 e prossegue no nível 4 do CMMI.
- Efetividade: o amadurecimento da organização aumenta ainda sua efetividade, ou seja, o aumento na qualidade dos resultados estimados dos projetos, o que está ligado ao aumento da capacidade do processo. Tal característica é mais evidente em organizações nos níveis de maturidade 5 do CMMI.

A próxima seção apresentará alguns conceitos estatísticos utilizados na predição de projetos de uma organização. Tais conceitos foram utilizados na definição do Modelo de Predição de projetos relacionado com este trabalho, detalhado na Seção 5.2.

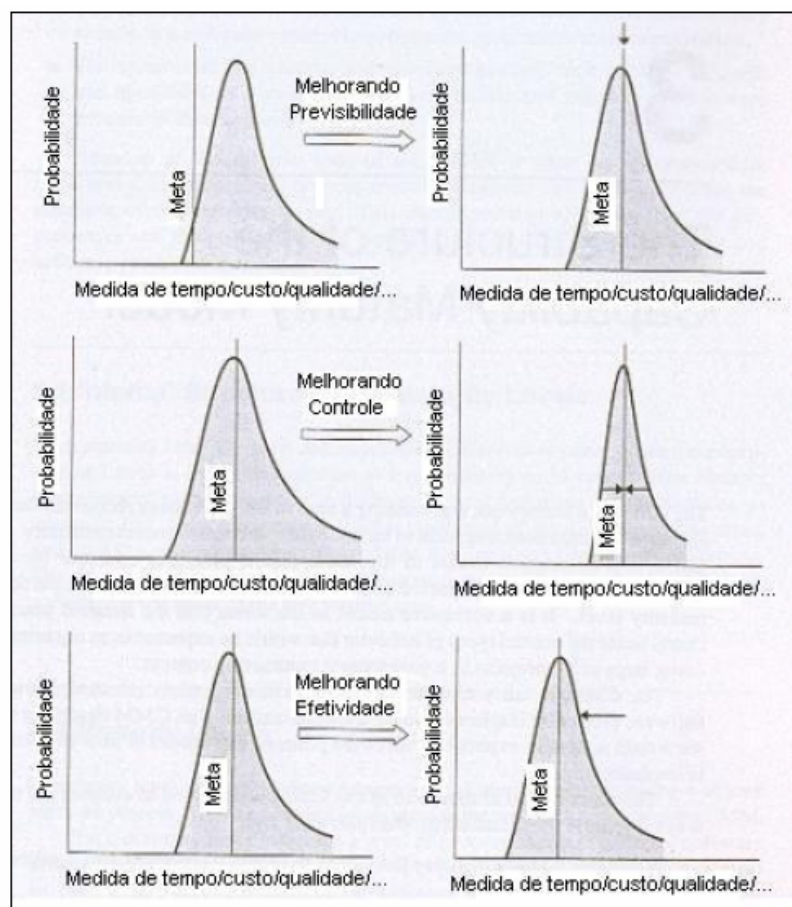


Figura 4.1 – Aumento da maturidade e melhoria na previsibilidade, controle e efetividade [Paulk et al. 1994].

4.2 Conceitos Estatísticos aplicados à Predição de Projetos

A predição de projetos pode ser beneficiada através da utilização de métodos estatísticos. Assim, a predição dos valores das variáveis relevantes que caracterizam um projeto, tais como prazo e custo, pode ser realizada quantitativamente, de maneira objetiva, fazendo com que as decisões organizacionais possam ser tomadas com base não apenas em estimativas imprecisas feitas pelo gerente de projeto, mas em uma predição de maior confiabilidade.

Inicialmente, a análise estatística de um dado sistema ou população pode ser feita utilizando-se uma amostra. A escolha da amostra é geralmente feita de maneira aleatória (técnica de amostragem aleatória), porém, a amostra deve ser representativa, ou seja, caracterizar a população como um todo, pois a mesma é analisada para se tirar conclusões a cerca da população. Recomenda-se que a amostra escolhida tenha no mínimo trinta elementos [Oliveira 2006]. No caso deste trabalho, a amostra é composta de projetos de naturezas semelhantes (projetos de desenvolvimento de software), com atividades de diferentes níveis de detalhe (granularidade).

Escolhida a amostra, é esperado que seu conjunto de dados apresente variação. Apesar dessa variação, tal conjunto tende a apresentar certo padrão de comportamento. A identificação do padrão de comportamento da amostra, que caracteriza a população em análise, é feita através da definição de sua distribuição de probabilidades. Para isso podem ser utilizados histogramas (exemplificado na Figura 4.2, adaptada de [Oliveira 2006]), que representam a distribuição de frequência dos dados de uma amostra, e servem para identificar a forma da distribuição de probabilidades. O ideal é que a predição estatística de uma ou mais variáveis, seja feita com base em uma distribuição de probabilidades, cuja forma se assemelhe a uma curva normal. A verificação da normalidade de uma amostra pode ser feita através do teste de Shapiro-Wilk [Royston 1995], que é estatisticamente garantido com 95% de confiança.

Outra ferramenta utilizada em estatística são os gráficos de dispersão (exemplificado na Figura 4.3), que servem para verificar a posição de cada elemento da amostra com relação à variável estudada. A partir desses gráficos, podem ser identificados elementos anômalos (*outliers*), distantes dos grupos principais [Oliveira 2006]. Geralmente esses pontos são identificados em gráficos de organizações imaturas, que não possuem ainda um grande controle de seus processos. Por outro lado, tais pontos podem ser retirados da amostra, caso sejam frutos de ocorrências esporádicas dentro da organização, por exemplo, um atraso fora do normal causado pela introdução de um novo projeto de maior prioridade.

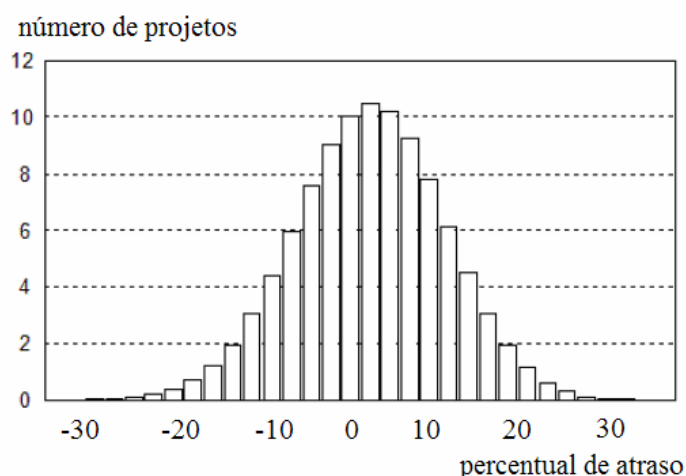


Figura 4.2 – Exemplo de histograma: distribuição normal de frequência do percentual de atraso de projetos de uma organização (adaptado de [Oliveira 2006]).

Para a análise de um conjunto de dados também são utilizadas medidas estatísticas de posição, que determinam a tendência central dos dados, e de dispersão, que analisam como os dados estão concentrados em torno do centro [Oliveira 2006]. A média é uma medida de posição conhecida, cujo valor, na maioria das vezes, se espera alcançar na medição da(s) variável(is) da amostra; O desvio padrão é uma medida de dispersão, também amplamente utilizada, que determina um intervalo de possível variação dos valores da(s) variável(is) medida(s). A Figura 4.4 ilustra os conceitos de média e desvio padrão, utilizados na definição da distribuição normal de probabilidades de uma dada população.

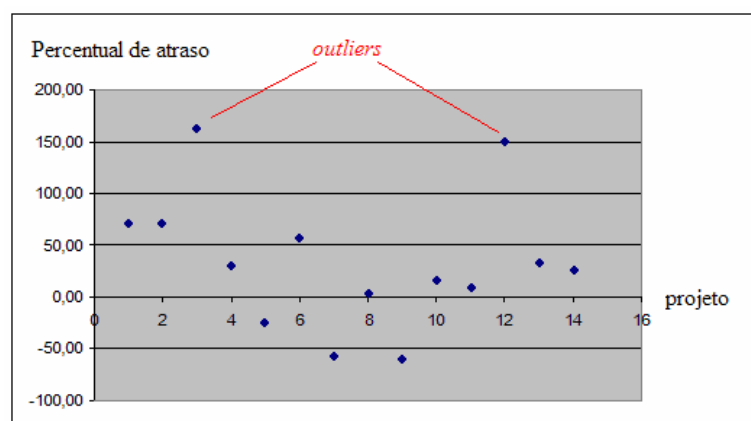


Figura 4.3 – Gráfico de dispersão: percentual de atraso de cada elemento de uma amostra de projetos.

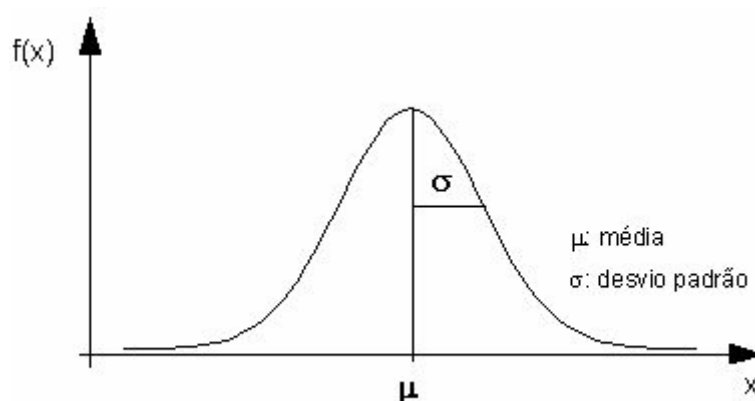


Figura 4.4 – Curva de distribuição normal de probabilidades (adaptado de [Oliveira 2006]).

Na linguagem estatística, a maturidade de um processo é a distribuição de probabilidades conjunta, que um conjunto de variáveis relacionadas a esse processo possui [Oliveira 2006]. Os gráficos ilustrados na Figura 4.1 representam distribuições de probabilidades com relação às possíveis medidas de tempo, custo ou qualidade (variáveis) do processo. A maturidade do processo pode ser medida para uma dada variável, como por exemplo, “percentual de atraso de projeto”, em que se trabalharia com a distribuição de probabilidades somente dessa variável.

Também na linguagem estatística, a capacidade de um processo é o conjunto de intervalos de valores esperados, que tal processo prevê para o conjunto das variáveis medidas. Da mesma forma, a capacidade do processo pode ser analisada para uma dada variável. Além disso, se apenas um valor for necessário, e não um intervalo inteiro, pode-se tomar o centro do intervalo como a capacidade esperada, ou capacidade média do processo. No caso da Figura 4.1 o ponto médio da curva normal representa a capacidade média do processo [Oliveira 2006]. Exemplificando, para a variável “percentual de atraso de projeto”, pode-se determinar o atraso médio na execução dos projetos de uma organização, caracterizando a capacidade média do seu processo com relação à variável definida.

Já a performance de um processo são os valores realizados das variáveis que caracterizam tal processo, para um dado projeto. Igualmente, podemos falar em performance do processo para uma dada variável [Oliveira 2006]. No exemplo já mencionado, a performance de um processo para a variável “percentual de atraso de

projeto” é o percentual de atraso apresentado na execução de um dado projeto que utiliza esse processo na prática. Na Figura 4.3, cada ponto do gráfico representa a performance do processo, com relação ao atraso na execução de projetos.

Estatisticamente, a maturidade é inversamente proporcional ao desvio padrão da variável medida. Já a capacidade e performance estão diretamente relacionadas com a média [Oliveira 2006]. Quanto mais maduro um processo, menos variável em torno do valor médio previsto será a sua performance. O gráfico de melhoria de controle, na Figura 4.1, ilustra a diminuição do desvio padrão ocasionado pelo aumento da maturidade. Quanto mais capaz um processo, maior a qualidade dos resultados previstos para as variáveis. O gráfico de melhoria de efetividade, na Figura 4.1, ilustra a melhoria da capacidade.

Existem 68,26% de chances de um dado elemento pertencer ao intervalo de valores, fornecido pelo cálculo do desvio padrão de uma distribuição normal ($[\mu - \sigma \text{ e } \mu + \sigma]$, onde μ e σ representam respectivamente a média e o desvio padrão populacionais, como mostra a Figura 4.5). Porém, sabe-se que existem 95% de probabilidade de um elemento pertencer ao intervalo: $[\mu - 1,96\sigma \text{ e } \mu + 1,96\sigma]$. A escolha do intervalo em que se deve basear na predição, depende do grau de confiança que se deseja obter na mesma. O Modelo de Predição, definido neste trabalho, utilizou o valor padrão 95%. Uma vez que um processo é considerado uma população infinita [Oliveira 2006], a distribuição de probabilidades a ser definida é relativa à amostra representativa analisada. Assim, no caso deste trabalho, a terminologia do intervalo a ser utilizada diz respeito à média e ao desvio padrão amostral, respectivamente definidos como \bar{x} e s , ou seja, $[\bar{x} - 1,96s \text{ e } \bar{x} + 1,96s]$.

O Modelo de Predição de projetos deste trabalho se baseia na aplicação dos métodos estatísticos apresentados na presente seção. Tal modelo procura predizer um intervalo de término de execução de um dado projeto, baseando-se em uma amostra de projetos da organização, considerando seu atraso. Tal amostra deve obedecer a uma distribuição normal, e a partir daí, é fornecido um intervalo ao qual, com 95% de

chances, o término da execução do projeto em análise poderá pertencer. Maiores detalhes sobre o Modelo de Predição são apresentados na Seção 5.2.

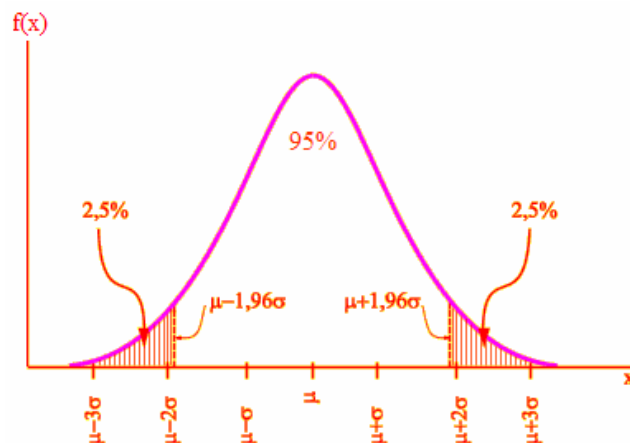


Figura 4.5 – Distribuição normal: probabilidade de ocorrência de elementos (adaptado de [Oliveira 2006]).

4.3 Simulação aplicada à Predição de Projetos

Como visto na seção anterior, a predição de projetos, realizada geralmente com base na experiência do gerente, pode se tornar mais confiável através da utilização de métodos estatísticos. Desta forma, para prever o término de um projeto, os dados históricos de uma organização, relacionados ao atraso na execução de seus projetos, podem ser analisados através de uma distribuição de probabilidades. Tal análise permite ainda a determinação da capacidade e maturidade do processo de estimativas da organização, através dos conceitos estatísticos de média e desvio padrão.

A princípio, para determinar o intervalo de término de um dado projeto, a predição estatística pode utilizar como base o histórico de atrasos dos projetos da empresa, e a estimativa de prazo do gerente, com relação ao projeto em análise. Porém, neste caso, o mecanismo de predição pode estar prejudicado, uma vez que a estimativa do gerente pode não considerar a real disponibilidade dos recursos, fornecendo uma predição não condizente com a realidade. Um exemplo de tal situação seria a existência

de apenas um recurso humano, disponível para alocação em um projeto, em que o gerente estimou a execução de atividades em paralelo. Neste caso, o mecanismo de predição forneceria um intervalo não condizente com a realidade, uma vez que o projeto na prática iria atrasar drasticamente com relação à estimativa do gerente (Figura 4.6).

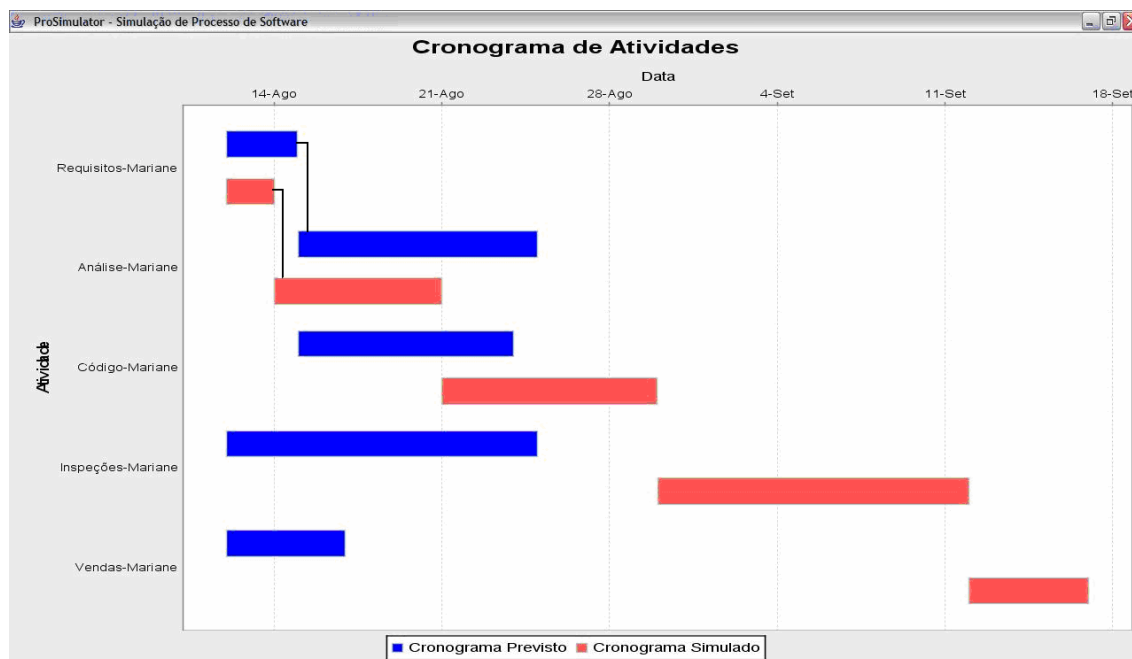


Figura 4.6 – Predição de projetos baseada em simulação: detecção de falhas de estimativas

A simulação pode ser utilizada neste caso, para predizer, de maneira mais realística, o término de um projeto, através da representação da execução do cronograma estimado, de acordo com a realidade dos recursos disponíveis. Com a simulação, torna-se possível ainda a detecção de possíveis falhas de estimativas, como ilustra o gráfico de *gantt*, na Figura 4.6, onde o cronograma previsto pelo gerente é representado pelas barras azuis, e o cronograma simulado é representado pelas barras vermelhas. A descrição das atividades e dos agentes alocados para execução das mesmas é mostrada no lado esquerdo do gráfico, na forma “atividade – agente”.

Os projetos a serem simulados podem ser de diferentes granularidades. A Figura 4.6 ilustra um exemplo de cronograma com atividades de maior granularidade de um projeto, tais como análise, desenvolvimento, dentre outras. Projetos com atividades de menor granularidade, podem ser definidos para uma análise mais detalhada e pontual de

um determinado ponto do desenvolvimento. Por exemplo, a fim de verificar de forma mais detalhada a execução da atividade de maior granularidade “Levantamento de Requisitos”, a mesma pode ser decomposta em um conjunto de atividades de menor granularidade a serem simuladas, tais como “entrevistar usuário”, “preencher documento de requisitos”, dentre outras.

O trabalho descrito nesta dissertação define uma Metodologia de Predição de Projetos, baseada em simulação, e utilização de métodos estatísticos. A abordagem utiliza como base a estimativa do gerente, que serve como entrada para a simulação. Esta, por sua vez, antevê a execução do cronograma previsto, de acordo com a disponibilidade e natureza dos recursos existentes para o projeto em análise. O histórico de projetos já executados pela organização é então analisado, utilizando-se métodos estatísticos. A distribuição normal de probabilidades, definida através da análise estatística, permite a identificação da capacidade e maturidade da organização, com relação ao atraso na execução dos projetos.

Desta maneira, baseando-se na estimativa pontual de término de execução do projeto, fornecida pela simulação, e na distribuição normal, relacionada ao histórico de atraso dos projetos da organização, o mecanismo de predição é capaz de fornecer um intervalo de término para o projeto em análise.

4.4 Considerações Finais

Este capítulo apresentou conceitos relacionados à predição de projetos de software, destacando a importância em garantir maior confiabilidade às estimativas feitas pelo gerente de projetos, através da utilização de simulação e métodos estatísticos. Tal abordagem foi utilizada na definição da Metodologia para Predição de Projetos (Capítulo 5), que prediz um intervalo de término de um projeto, estatisticamente garantido com 95% de confiança.

Para implantação da melhoria do processo organizacional, é necessário, inicialmente, determinar o grau de maturidade que o processo tem, para em seguida definir o conjunto de ações para aumentar essa maturidade. Estabilizado e tornado mais maduro o processo, ações podem ser realizadas visando o aumento de sua capacidade média [Oliveira 2006]. A identificação da maturidade e capacidade dos processos organizacionais pode ser feita através da análise estatística do histórico de projetos, incentivando organizações imaturas na aplicação de práticas de qualidade, visando à melhoria contínua de seus processos, alcançando maior previsibilidade, controle e efetividade.

A análise estatística do histórico dos projetos organizacionais permite que se conheça o comportamento do processo daquela organização, como um todo, com relação à variável em estudo (no caso deste trabalho, atraso). Isso é possível identificando-se a distribuição de probabilidades da amostra em estudo, bem como a média e desvio padrão relacionados às variáveis consideradas. Através da utilização de tais métodos, pode-se prever o comportamento de um projeto futuro com maior confiabilidade.

A predição estatística pode estar comprometida quando baseada somente em estimativas do gerente de projetos. Neste sentido, a predição de projetos de software baseada em simulação provê maior aproximação à realidade, através da execução do cronograma previsto pelo gerente de acordo com as características e disponibilidade real dos recursos organizacionais.

5

Uma Metodologia para Predição de Projetos

Este capítulo detalha a Metodologia para Predição de Projetos, que representa a principal contribuição deste trabalho. Inicialmente, a Seção 5.1 descreve o Modelo de Simulação, definido no contexto da validação de modelos de processo de software, utilizado como base para a predição. Na Seção 5.2 é descrito o Modelo de Predição, utilizado na determinação de um intervalo de término de um dado projeto futuro da organização. A predição do intervalo é baseada na predição pontual, fornecida pelo Modelo de Simulação, e no histórico de atrasos na execução de projetos da organização, analisado através da utilização de métodos estatísticos. A Seção 5.3 apresenta uma possível validação do intervalo previsto pelo Modelo de Predição. A Seção 5.4 apresenta as considerações finais sobre o capítulo.

5.1 Modelo de Simulação do Processo de Software Instanciado

A tecnologia de simulação de processos de software pode ser de grande utilidade na validação de modelos de processo [Silva 2001]. Dado um modelo de processo de software, instanciado para um dado projeto da organização, a simulação permite antever sua execução, utilizando-se os recursos definidos para aquele projeto. A simulação

permite ainda a detecção de falhas no modelo, auxiliando no refinamento e melhoria contínua do processo.

Este trabalho define um modelo de simulação de processo de software evento-discreto, que permite a validação e auxilia no refinamento de modelos de processo, no contexto de um ambiente de implementação progressiva de processo de software (ambiente ImPProS [Oliveira et al. 2005], detalhado na Seção 6.1). Isso é permitido através da simulação da execução de atividades do processo, por agentes desenvolvedores com características específicas. Além disso, o modelo de simulação definido é utilizado como auxílio na predição do término de execução de projetos futuros de uma organização (Modelo de Predição, definido na Seção 5.2). A Figura 5.1 ilustra a arquitetura do modelo de simulação definido.

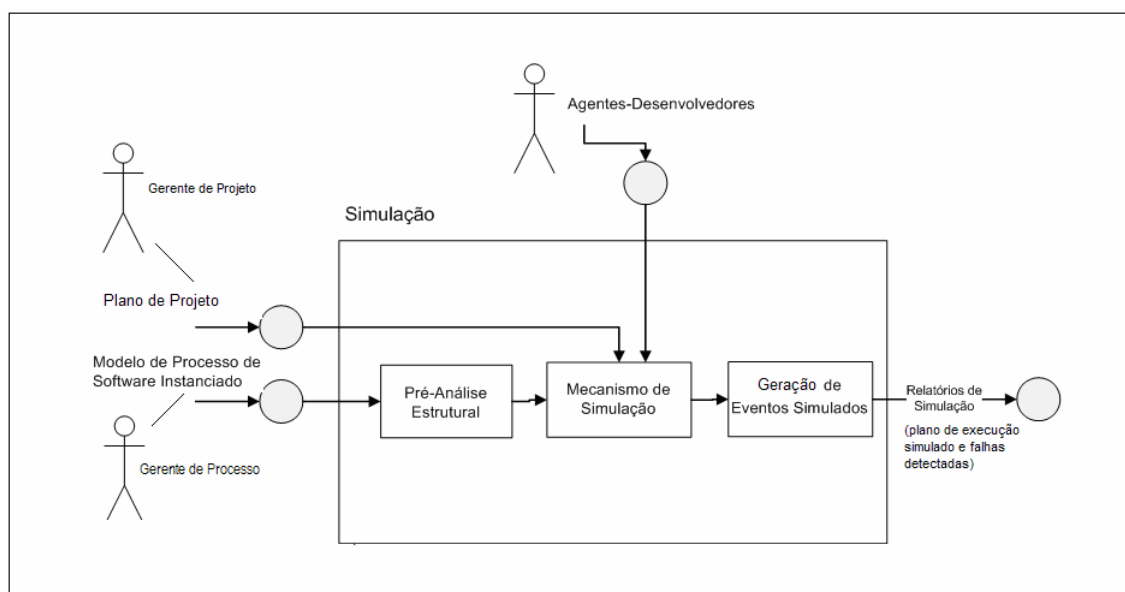


Figura 5.1 – Arquitetura do Modelo de Simulação de Processo de Software.

O modelo de simulação utiliza o modelo de estrutura de processo de software, definido por Oliveira, no contexto do ambiente ImPProS [Oliveira et al. 2005]. Tal estrutura é utilizada como base para descrição do modelo de processo de software, posteriormente instanciado pelo gerente de processo. O gerente de projetos também se baseia nessa estrutura para definir o plano de projeto a ser simulado, que inclui, dentre

outros, o cronograma de atividades, métricas, estimativas e recursos relacionados. A Seção 5.1.1 descreve, com mais detalhes, a estrutura do plano de projeto.

Além do modelo estrutural, o modelo de simulação utiliza agentes desenvolvedores, que representam as pessoas da organização relacionadas com a fase de execução do processo. Durante a simulação, tais agentes são responsáveis pela execução das atividades definidas para o processo, alocadas em suas agendas, de acordo com suas características, tais como habilidades e afinidades. O mecanismo de agenda dos agentes e o tratamento de suas habilidades e afinidades foram baseados no modelo de simulação *AgentProcess* ([Silva 2001] e [Silva et al. 1999]). A Seção 5.1.2 descreve o tratamento de habilidades e afinidades dos agentes desenvolvedores, e o mecanismo de agenda do agente.

Instanciado o modelo de processo de software, o primeiro passo, anterior à simulação, é realizar a análise estrutural automática de itens pré-definidos, relacionados a processo de software. Tal análise procura detectar falhas estruturais no modelo, tais como atividades desencadeadas, atividades de tipos diferentes fazendo parte de uma mesma composição, incompatibilidade entre tipo de atividade e tipo de ferramenta utilizada, dentre outras. A análise estrutural identifica o problema, mostra as possíveis consequências do mesmo e fornece possíveis soluções. Maiores detalhes sobre a análise estrutural são mostrados no Apêndice B.

Concluída a análise estrutural, caso todos os itens sejam considerados atendidos no modelo de processo instanciado, a simulação pode iniciar. Neste caso, com base no modelo evento-discreto, as atividades do processo instanciado são simuladas, na ordem crescente de data inicial prevista, identificando-se os agentes mais aptos para execução das mesmas. A Seção 5.1.3 fornece informações mais detalhadas sobre cada um dos passos que compõem a simulação do processo.

O resultado da simulação é representado através da ocorrência de eventos, que geram informações relevantes, caracterizando o comportamento do processo. Essas informações são descritas através de relatórios de simulação. As principais informações fornecidas incluem:

- Dados gerais sobre atividades, incluindo descrição, recursos necessários, tempo e custo previstos.
- Agentes desenvolvedores alocados para execução de atividades, segundo suas disponibilidades, habilidades e afinidades, com o respectivo papel assumido por eles.
- Tempo e custo simulados de atividades, calculados de acordo com habilidades e afinidades dos agentes desenvolvedores alocados.
- Recurso financeiro disponível e utilizado durante a simulação (utilização diretamente ligada às características dos agentes desenvolvedores alocados nas atividades).
- Data de término de projeto prevista, de acordo com o cronograma fornecido pelo gerente de projetos, e simulada, de acordo com a disponibilidade e natureza dos recursos atuais.

Outras informações relevantes, fornecidas pela simulação, são as possíveis falhas e incompatibilidades entre estimativas e o realizado (de acordo com a realidade da organização), detectadas através das informações definidas anteriormente:

- Inconsistência entre datas iniciais e finais previstas de uma atividade: data final prevista anterior à data inicial prevista. Esse problema é detectado antes da simulação, não permitindo que a mesma seja executada.
- Inconsistência entre datas iniciais e finais previstas de pré-atividades e pós-atividades: data inicial prevista de uma pós-atividade, anterior ou na mesma data final prevista de sua pré-atividade. A existência desse problema também não permite a execução da simulação.
- Número insuficiente de recursos humanos para execução da simulação: número de agentes existente, inferior ao maior número de agentes

necessário para execução de uma atividade. A detecção desse problema elimina as chances de ocorrência de *starvation*, ou seja, da espera indefinida para execução de um processo devido à falta de recursos. Caso o problema não fosse detectado, a atividade seria realocada em um dia indefinidamente, até que o número de recursos necessários fosse “liberado”, o que nunca iria acontecer. Essa falha também é detectada antes da simulação, não permitindo sua execução.

- Extrapolação de prazos definidos: data final simulada posterior à data final prevista pelo gerente de projetos. Isso pode ocorrer devido à falta de conhecimento sobre a disponibilidade e natureza da equipe atual de desenvolvedores da organização, fazendo com que as estimativas do gerente não estejam de acordo com a provável execução prática do projeto.
- Extrapolação de custos definidos: custos despendidos durante a simulação maiores que o recurso financeiro disponível. Custos relacionados à infra-estrutura (máquinas, licenças de software) são fixos. A variação está diretamente ligada aos agentes alocados, cuja habilidade e afinidade resultarão no aumento ou diminuição na duração de uma dada atividade, impactando no seu custo final.
- Existência de agentes ociosos ou com carga alta de trabalho: agentes alocados em menos de 20% do tempo estimado para o projeto são considerados ociosos, e os que são alocados em mais de 70% do tempo são considerados com carga alta de trabalho. Tais valores podem ser parametrizados de acordo com as necessidades da organização.
- Existência de atividades que extrapolaram, significativamente, prazos e custos com relação ao previsto: atividades cujos tempo e custo simulados correspondem a 70% ou mais dos respectivos valores previstos. Este valor também pode ser parametrizado de acordo com as necessidades da

organização. Caso estejam no caminho crítico de um projeto [PMI 2004], tais atividades são responsáveis pela extrapolação de prazos e custos gerais do projeto e necessitam maior atenção. Existem algoritmos que detectam caminhos críticos em cronogramas de atividades de projeto, porém, tal abordagem não foi utilizada no modelo definido neste trabalho.

5.1.1 Plano de Projeto

O plano de projeto, definido pelo gerente, é a principal entrada da simulação, sendo especificado ao final da fase de instanciação do ciclo de vida do processo de software (Seção 2.2). A estrutura do plano de projeto e o seu relacionamento com as demais entidades estão ilustrados na Figura 5.2.

Dentre outras informações, o plano define o recurso financeiro inicial disponível, que não deve ser extrapolado; os recursos gerais disponibilizados para o projeto; e as métricas e estimativas definidas, relacionadas geralmente com prazos e custos, que servem de guia na análise posterior à simulação. Um plano está associado a, no máximo, um dado modelo de processo instanciado. Porém, um modelo de processo instanciado pode definir vários planos a serem simulados (análise de diferentes cenários).

O plano de projeto está relacionado com um conjunto de atividades do processo instanciado, às quais agrega informações necessárias à simulação. As principais informações agregadas são: cronograma previsto para as atividades, com datas de início e término e duração, e os recursos utilizados para sua execução, incluindo infraestrutura, tais como máquinas e licenças de software, e recursos humanos, sendo, neste caso, definida a habilidade do recurso, que caracteriza seu papel (ex: habilidade “Análise de Sistemas” para o recurso “Analista de Sistemas”). Em ambos os casos, o custo por dia do recurso utilizado é definido. Os artefatos consumidos e produzidos por uma atividade são definidos a priori durante a instanciação do processo.

O estado de recursos/artefatos, bem como de atividades são informações utilizadas no modelo *AgentProcess* [Silva 2001] [Silva et al. 1999], respectivamente para verificar a disponibilidade de um recurso/artefato, e para determinar a ordem de simulação das atividades. Este trabalho procurou focar na alocação de recursos humanos em atividades, analisando a disponibilidade dos mesmos através do mecanismo de agenda do agente. Desta forma, o estado de um recurso não-humano não foi contemplado neste modelo, sendo contabilizado apenas seu custo. Da mesma forma, o estado de atividades não foi contemplado, pois a ordem de execução de atividades é definida de acordo com a data inicial prevista para cada uma delas, inseridas em uma fila. Tal abordagem não permite a existência de casos em que um mesmo agente possa ser alocado para execução de atividades em paralelo.

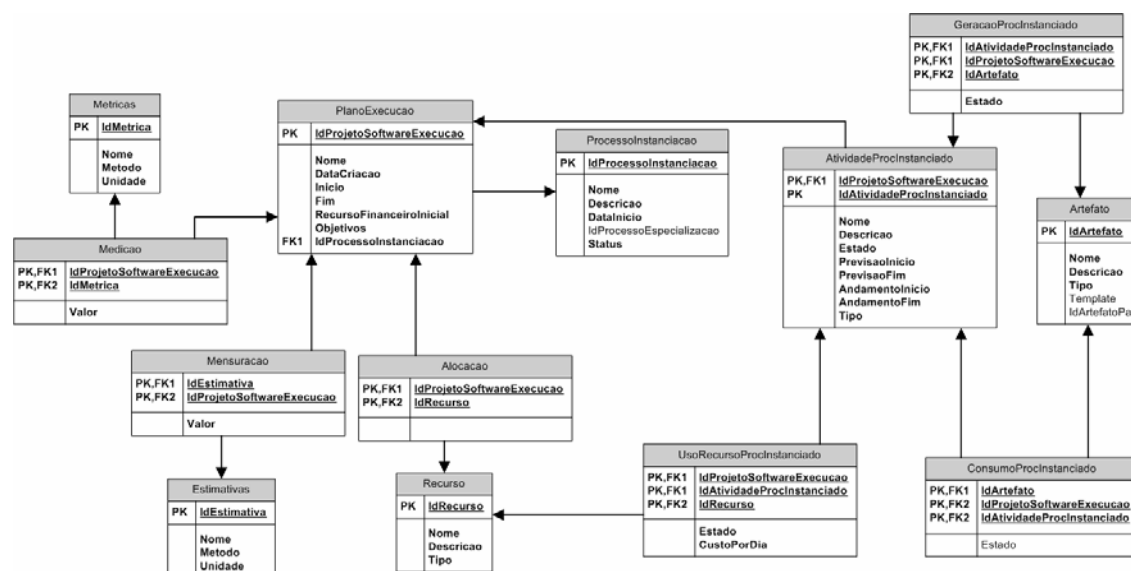


Figura 5.2 – Modelo relacional: estrutura do Plano de Projeto.

5.1.2 Tratamento de Habilidades e Afinidades dos Agentes de Simulação

Os agentes desenvolvedores definidos no Modelo de Simulação (Figura 5.1) representam os colaboradores da organização, relacionados com a fase de execução do processo de desenvolvimento de software. Essas pessoas possuem diversas características particulares, tais como personalidade, humor, habilidades e afinidades,

que podem influenciar diretamente na execução de suas atividades em um ambiente de desenvolvimento de software.

A contemplação de tais características durante a simulação, possibilita a geração de resultados mais realísticos com relação ao comportamento do processo [Silva 2001]. No caso do modelo definido neste trabalho, as características de habilidades e afinidades dos agentes são utilizadas na alocação dos mesmos para execução de atividades, e no cálculo do tempo de uma dada atividade executada por um agente. Tal abordagem foi definida no trabalho de Silva [Silva 2001].

As habilidades e afinidades são definidas em níveis de zero (0) a um (1), representando, respectivamente, o menor e maior nível de habilidade/afinidade. Desta forma, cada agente possui um nível, relacionado à habilidade do(s) recurso(s) a ser(em) utilizado(s) para execução de cada atividade. Por exemplo, determinado agente possui o nível “0,7” para a habilidade “Análise de Sistemas” de um dado recurso humano, relacionado com a atividade “Definição do Diagrama de Classes”. Da mesma forma, cada agente possui um dado nível de afinidade com cada um dos outros agentes definidos para simulação.

A alocação de um agente é dada em função dos seus níveis de habilidade e afinidade, como ilustra a Função 5.1 definida neste trabalho. Tal função procura definir um peso médio para os parâmetros de habilidade e afinidade, para que nenhum deles possa influenciar mais que o outro na decisão de se alocar um dado agente em uma atividade. Desta maneira, para cada agente, é verificado o seu nível com relação à habilidade do recurso definido para a atividade, cujo valor é somado à média das afinidades do agente em questão, com os demais agentes já alocados para a atividade. Caso a atividade não seja cooperativa, a média de atividades não influenciará no cálculo da função de alocação. Assim, o agente que apresentar o maior valor da função de alocação, será considerado o mais apto para executar a tarefa, e caso esteja disponível, será alocado para a mesma.

No caso de uma atividade cooperativa, ou seja, que envolva dois ou mais agentes, a decisão de alocação através da afinidade é determinada pelo primeiro agente

alocado (mais habilidoso). Tal fato pode gerar problemas, caso o primeiro agente alocado (considerando apenas sua habilidade) não tenha grandes afinidades com os demais existentes, acarretando em um aumento significativo no tempo de execução da tarefa. Uma possível solução seria aplicar a *heurística de valor menos restritivo* [Russel e Norvig 2003], pela qual o primeiro agente é alocado, segundo a soma de sua habilidade com a média de afinidades, para com os demais agentes existentes, ou seja, procura-se um agente que menos restrinja a escolha dos demais. Neste caso, nem sempre o primeiro agente alocado será o mais habilidoso, porém, isso poderá ser compensado pela sua maior afinidade. A aplicação dessa heurística no algoritmo de simulação (Seção 5.1.3) é sugerida como trabalho futuro desta pesquisa, apresentado na Seção 8.2.

$$f(x) = habilidade(x) + \frac{\sum afnidade(x, y)}{\# B} \quad \forall x \in A; y \in B; \# B > 0; x \neq y \quad (5.1)$$

Onde :

A : conjunto de agentes

B : conjunto de agentes alocados para execução da atividade

A decisão de alocação depende ainda da disponibilidade do agente. A análise da disponibilidade do agente é feita através de sua agenda. A agenda do agente é o meio de comunicação do simulador com os agentes desenvolvedores, que contém as atividades em que um agente está alocado, bem como o período durante o qual o mesmo estará ocupado. O mecanismo de agenda do agente foi definido com base no trabalho de Silva [Silva 2001]

Alocada a atividade na agenda do agente, o próximo passo é calcular o tempo que o agente levará para executar essa atividade (Função 5.2). Esse cálculo também é baseado nos níveis de habilidade e afinidade do agente. A Tabela 5.1 e a Tabela 5.2 (descritas no trabalho [Silva 2001]), definem o percentual de acréscimo ou decréscimo no tempo previsto, com relação aos níveis de habilidade e afinidade do agente. Por exemplo, um agente com nível de habilidade “0,4”, executa a atividade com 20% de aumento no tempo previsto. Já um agente com a média de afinidades “0,8”, acarreta em

uma diminuição de 10% no tempo previsto para a atividade. Habilidades e afinidades médias (em torno de 0,6 e 0,7, respectivamente) não têm impacto no cálculo do tempo da atividade. Caso a atividade seja cooperativa, o tempo da atividade simulada é o tempo do agente mais lento.

O cálculo do custo da execução de uma atividade é dado em função do tempo de execução dessa atividade, gasto pelo(s) agente(s) alocado(s) para sua execução, durante a simulação. O cálculo é feito somando-se o custo fixo dos recursos não humanos, com o custo de alocação por dia de cada recurso humano, multiplicado pelo tempo de execução da atividade pelo respectivo agente, como mostra a Função 5.3.

$$t(x) = tempo_previsto \times fator(habilidade(x)) \times fator\left(\frac{\sum afinidade(x, y)}{\#B}\right) \quad (5.2)$$

$$\forall x \in B; y \in B; \#B > 0; x \neq y$$

Onde :

B : conjunto de agentes alocados para execução da atividade

Tabela 5.1 – Cálculo da variação do tempo em função da habilidade do agente [Silva 2001].

Habilidade geral do agente	Alteração no tempo previsto
hab ≤ 0,3	Acréscimo de 30%
hab > 0,3 & hab ≤ 0,4	Acréscimo de 20%
hab > 0,4 & hab ≤ 0,5	Acréscimo de 10%
hab > 0,5 & hab ≤ 0,6	Tempo igual ao previsto
hab > 0,6 & hab ≤ 0,7	Decréscimo de 10%
hab > 0,7 & hab ≤ 0,8	Decréscimo de 20%
hab > 0,8 & hab ≤ 1,0	Decréscimo de 30%

Tabela 5.2 – Cálculo da variação do tempo em função da afinidade do agente [Silva 2001].

Afinidade geral do agente	Alteração no tempo previsto
afin ≤ 0,5	Acréscimo de 30%
afin > 0,5 & afin ≤ 0,6	Acréscimo de 10%
afin > 0,6 & afin ≤ 0,7	Tempo igual ao previsto
afin > 0,7	Decréscimo de 10%

$$c(a) = \sum \text{custo}(x) + \sum [t(y) \times \text{custo}(y)] \quad a \in C; \quad x \in D; \quad y \in B; \quad (5.3)$$

Onde :

B : conjunto de agentes alocados para execução da atividade

C : conjunto de atividades

D : conjunto de recursos não humanos da atividade

As habilidades e afinidades dos agentes desenvolvedores devem representar a realidade atual das pessoas da organização, e devem ser atualizadas à medida que mudanças ocorram. Tais informações podem ser obtidas através da análise de dados históricos dos colaboradores, caso existam, ou podem ser definidos pelo gerente de projetos. Porém, aconselha-se a captura automática desses dados através de ferramentas de descoberta de conhecimento, instaladas nos ambientes de desenvolvimento, para se obter informações confiáveis, que representem a realidade organizacional (mecanismo “*knowledge discovery*”, proposto no trabalho [Lima 2001]). Segundo Silva [Silva 2001], tanto a Tabela 5.1 como a Tabela 5.2 não foram definidas através de pesquisas em ambientes reais.

A alocação de recursos para execução de tarefas é um problema clássico da área de Otimização Combinatória [Papadimitriou e Steiglitz 1982]. A alocação de um agente, através da aplicação da heurística de maior habilidade e afinidade, foi definida pelo trabalho [Silva 2001]. Neste caso, a utilização da heurística possibilita, de acordo com a realidade da organização, a execução do cronograma dentro dos prazos e custos definidos, ou até mesmo em prazos e custos menores. Porém, somente pela utilização desta heurística, existem casos em que prazos e custos são significativamente extrapolados. O Apêndice A apresenta uma proposta de melhoria dos valores obtidos com a heurística definida, através de um modelo de otimização.

5.1.3 Algoritmo de Simulação

O algoritmo apresentado na presente seção, se baseia na técnica de simulação evento-discreta (seções 3.1.2.1 e 3.1.3.1). Segundo tal técnica, a simulação ocorre

discretamente, em pontos determinados pela ocorrência de eventos. A ocorrência de eventos provoca alterações no estado do sistema como um todo. Como exemplo de eventos, pode-se citar a execução de uma atividade, que pode resultar no adiantamento ou atraso de atividades dependentes, alterando o estado do cronograma. A Figura 5.3 descreve, em pseudocódigo, o algoritmo de simulação definido neste trabalho.

```
ordenarAtividades();
realizarVerificacoesIniciais();
calcularPrevisaoPrazoCusto();

Enquanto existir atividades a serem simuladas{
    atividade = removerAtividadeDaFila();
    Se (recursosSuficientes(atividade)){
        clock = atividade.retornarDataPrevista();
        Para cada recurso alocado para esta atividade{
            agente = verificarAgenteMaisAptoDisponivel(recurso);
            datas = calcularDataSimulada(agente);
            alocarAtividadeAgendaAgente(agente, datas);
            inserirPossivelDataSimuladaAtividade(datas);
        }
        atividade.definirDataSimulada(dataSimuladaAgenteMaisLento);
        atividade.calcularCustoExecucao();
        atividade.realocarPosAtividades(shift);
    } Senão{
        realocarAtividade(atividade);
        realocarPosAtividades(atividade);
        inserirAtividadeOrdenadaNaFila(atividade);
    }
}

calcularPrazoCustoSimulado();
realizarVerificacoesFinais();
```

Figura 5.3 – Algoritmo de simulação em pseudocódigo detalhado.

Inicialmente, as atividades do processo instanciado são inseridas em uma fila, por ordem crescente de data inicial prevista. No caso de atividades que iniciam na mesma data, a escolha pode ser feita pela atividade de menor duração, ou aquela que tiver algum tipo de prioridade. Antes do início da simulação, existem algumas condições que devem ser atendidas: consistências de datas iniciais e finais previstas, consistência de datas de pré-atividades e pós-atividades, e existência de agentes suficientes para execução de atividades. Cada uma dessas condições foi descrita com mais detalhes no início deste capítulo, na Seção 5.1.

Após a verificação das condições necessárias, são calculados o prazo e custo previstos em linha base. O custo e prazo previstos na linha base de um projeto é o somatório dos prazos e custos previstos para cada atividade do cronograma. Através desse cálculo, estima-se o custo total previsto para o projeto, já verificando se o mesmo extrapola o recurso financeiro inicialmente definido. Também através desse cálculo, define-se o prazo final estimado para término do projeto.

O *clock* de simulação é então adiantado para o tempo do próximo evento, ou seja, para a data de início prevista para execução da primeira atividade da fila. Caso existam recursos humanos suficientes para execução da atividade em questão, para cada recurso humano identifica-se o(s) agente(s) mais apto(s) a ser(em) alocado(s). A análise do agente mais apto é baseada no cálculo da função de alocação (Função 5.1, Seção 5.1.2), dada em função do nível de habilidade e da média de afinidades com os demais agentes já alocados (caso seja uma tarefa cooperativa). Para o agente que possuir a maior função de alocação, será verificada sua disponibilidade no tempo previsto para atividade, por meio de sua agenda. Caso esteja disponível, a atividade será alocada em sua agenda.

Alocada a atividade, é calculado o tempo gasto para cada agente responsável realizar aquela atividade. Esse cálculo (Função 5.2, Seção 5.1.2) é baseado no acréscimo ou decréscimo de tempo, definido de acordo com os níveis de habilidade e média de afinidades. No caso de uma atividade cooperativa, o tempo simulado da atividade é o tempo gasto pelo agente mais lento.

Após a simulação da atividade atual, é calculado o custo simulado da mesma (Função 5.3, Seção 5.1.2). Esse cálculo é dado em função dos custos fixos de recursos não humanos, e do custo variável de recursos humanos, de acordo com o número de dias em que um agente foi alocado.

Calculado o custo simulado da atividade, é verificada a necessidade de realocação de suas pós-atividades. Caso a atividade tenha adiantado ou atrasado com relação ao previsto, as atividades dependentes da mesma também devem ter sua data inicial prevista “ajustada”. Esse ajuste deve ser feito com base na diferença existente

entre as datas finais previstas e simuladas da pré-atividade em questão. A Figura 4.6 (Seção 4.3) mostra a realocação à esquerda de uma pós-atividade, após a simulação de sua pré-atividade.

Caso não existam recursos disponíveis para execução da atividade, ela será realocada em um dia, e reinserida novamente, por ordem de data inicial, na fila de atividades. Neste caso, suas atividades dependentes também serão realocadas em um dia. Isso será feito até que exista o número suficiente de recursos humanos, disponíveis para execução da atividade na data prevista. Suponha uma atividade que inicie no dia 2/5. Caso não existam recursos nesta data, a atividade será realocada para o dia 3/5, e assim por diante, até que existam recursos suficientes.

Os passos definidos acima ocorrem iterativamente, enquanto houver atividades a serem executadas na fila. Após a simulação de todas as atividades definidas no cronograma, algumas verificações finais são realizadas: extrapolação de prazos e custos de projeto, de acordo com o cálculo total do custo e prazo simulados; verificação de agentes ociosos ou com carga alta de trabalho; e verificação de atividades que extrapolaram custos e prazos previstos de maneira significativa, podendo ser responsáveis pela extrapolação geral de prazos e custos de projeto. Essas verificações são definidas com mais detalhes na Seção 5.1.

5.2 Modelo de Predição de Projetos

A predição de projetos, quando realizada de maneira confiável na organização, permite a negociação de prazos, de acordo com a sua realidade. A estimativa baseada em métodos confiáveis pode evitar prejuízos, tais como multas de atrasos de projetos, bem como a insatisfação de funcionários e clientes. Além disso, através da predição de projetos, as organizações podem identificar a maturidade e capacidade de seus processos, sendo então incentivadas a promover a melhoria contínua dos mesmos.

Esta seção apresenta um modelo para predição de término de projetos, baseado em métodos estatísticos. Tal modelo utiliza informações fornecidas pelo Modelo de Simulação, descrito na seção anterior, para conferir maior realidade às estimativas definidas pelo gerente de projetos, enquanto a utilização de métodos estatísticos permite definir, com maior confiabilidade, um intervalo, ao qual, na prática, o término do projeto terá chances de pertencer.

A Figura 5.4 ilustra a arquitetura do Modelo de Predição proposto. A partir da simulação do cronograma do projeto futuro, previsto pelo gerente, uma predição pontual do término do projeto é fornecida, sendo utilizada, junto ao histórico de projetos da organização, para prever o intervalo de término do projeto em análise. As próximas seções irão descrever mais detalhes do Modelo de Predição proposto.

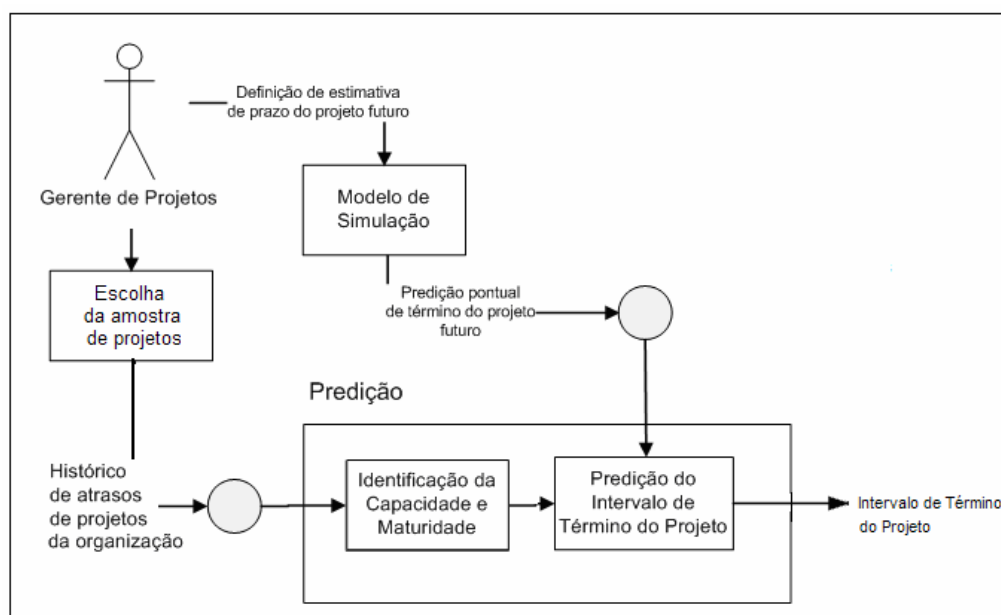


Figura 5.4 – Arquitetura do Modelo de Predição de Projetos.

5.2.1 Escolha da Amostra de Projetos da Organização

A utilização de dados históricos da organização permite a representação do seu perfil no processo de predição. No modelo aqui proposto, o histórico de atraso na execução de projetos, é utilizado como base para a predição estatística do término do projeto. Tal

atraso é possivelmente causado por eventos associados a riscos de projeto, não contemplados durante a simulação.

A amostra de projetos da organização pode ser escolhida aleatoriamente, porém, deve ser representativa, ou seja, apresentar características da população como um todo. Desta maneira, devem ser escolhidos projetos de diferentes tipos, tamanhos e complexidades, para que a predição seja feita de maneira não tendenciosa. No caso deste trabalho, foram escolhidos projetos de mesma natureza (atividades relacionadas ao desenvolvimento de software), porém, com atividades de diferentes granularidades.

Como já mencionado, é recomendado que se escolha pelo menos trinta elementos para compor a amostra [Oliveira 2006], porém, no caso deste trabalho, não foi possível a utilização de uma amostra de tal tamanho, devido ao pequeno porte da empresa na qual foi feito o experimento.

Escolhida a amostra, a variável em questão deve ser medida para cada elemento. Assim, é definida a porcentagem de atraso com relação às estimativas para cada um dos projetos escolhidos. A partir daí, podem ser utilizadas ferramentas, tais como histogramas e gráficos de dispersão (Seção 4.2), para melhor visualizar a frequência e dispersão dos dados, bem como identificar o padrão de comportamento da amostra.

Como já definido, é interessante que o conjunto de dados em análise tenha uma distribuição, cuja forma se assemelhe a uma curva normal, devido ao grande estudo realizado sobre a mesma. Neste caso, definido o conjunto de dados relacionados ao atraso na execução de projetos, verifica-se a normalidade do mesmo, através da aplicação do teste de Shapiro-Wilk (Seção 4.2), para que a predição possa se basear nas informações da distribuição normal de probabilidades, referente à amostra.

5.2.2 Definição da Capacidade e Maturidade

O próximo passo, após a escolha da amostra de projetos, é determinar a capacidade e maturidade do processo de estimativas da organização. Como já mencionado, a

capacidade está diretamente ligada à medida de posição, e a maturidade é inversamente proporcional à dispersão dos dados.

Neste caso, é calculado o atraso médio na execução de projetos da organização (Fórmula 5.4), baseado em seu histórico, que representa a sua capacidade média. Depois disso, calcula-se o desvio padrão da amostra fornecida (Fórmula 5.5), definindo assim, a distribuição normal de probabilidades, que representa a maturidade do processo de estimativas da organização, como mostra a Figura 5.5. É importante observar que a organização pode ter tido projetos adiantados com relação ao previsto (deslocamento da curva para esquerda do eixo).

$$\bar{x} = \frac{\sum \text{atraso}(p)}{\#P} \quad p \in P; \#P > 0 \quad (5.4)$$

Onde :

P : amostra de projetos da organização

$$s = \sqrt{\frac{\sum (\text{atraso}(p) - \bar{x})^2}{\#P - 1}} \quad p \in P; \#P > 1 \quad (5.5)$$

Onde :

P : amostra de projetos da organização

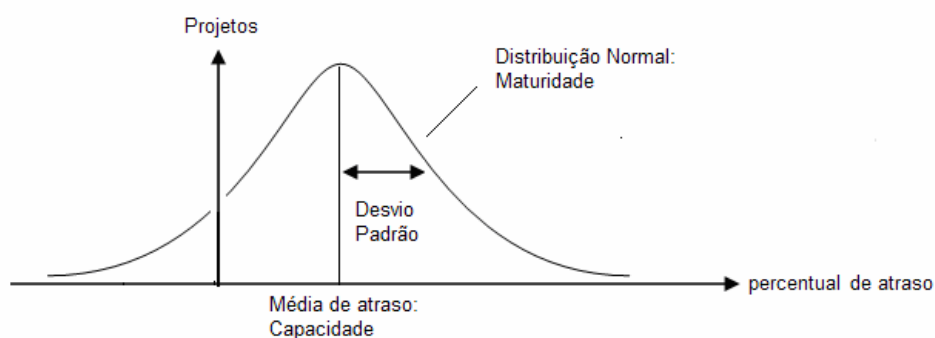


Figura 5.5 – Definição da capacidade e maturidade da organização relacionada ao atraso na execução de projetos.

5.2.3 Simulação: Predição de Término de Execução do Projeto

Este trabalho propõe a utilização de simulação, para fornecer uma predição da data de término do projeto em análise, de acordo com a realidade atual dos recursos organizacionais. Logo, em paralelo à análise estatística, é realizada a simulação do projeto, alvo de predição.

O Modelo de Simulação foi utilizado neste contexto, para simular o plano, relacionado ao projeto em análise, cujas atividades serão executadas pelos agentes desenvolvedores, de acordo com suas características de habilidades e afinidades. Tais características causam impacto direto no tempo de execução das atividades e, possivelmente, na data de término do projeto. Tais características devem sempre refletir a realidade organizacional, para que a predição também o faça.

A Figura 5.6 mostra um exemplo de cronograma, simulado de acordo com a disponibilidade, natureza e realidade dos recursos da organização. Pelo exemplo é possível perceber a diferença entre o prazo de atividades, estimado pelo gerente (em azul), e o tempo de fato realizado na simulação, de acordo com os recursos disponíveis alocados (em vermelho).

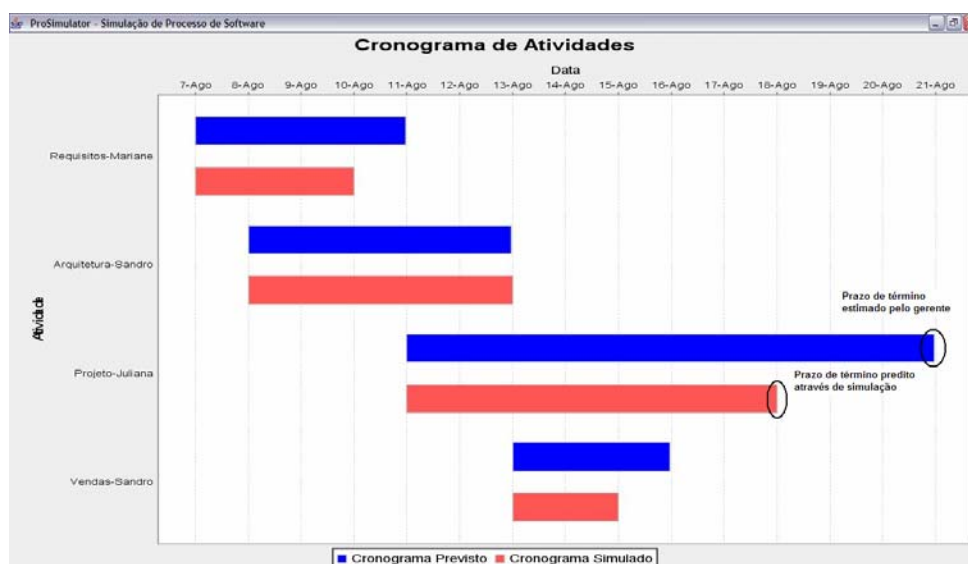


Figura 5.6 – Exemplo de cronograma simulado: diferença entre prazo de projeto estimado e simulado.

5.2.4 Predição do Intervalo de Término do Projeto

A partir da predição do término do projeto, realizada através da simulação, aliada à análise estatística do histórico de atraso de projetos da organização, torna-se possível prever um intervalo de término para o projeto em análise.

Como mostra a Figura 5.7, inicialmente, a data de término do projeto, fornecida pela simulação, é localizada no gráfico, que representa a distribuição normal de probabilidades. Depois disso, adiciona-se à data prevista, a porcentagem média de atrasos de projetos (capacidade), convertida em dias, determinando assim uma nova data de término, de acordo com o histórico da organização.

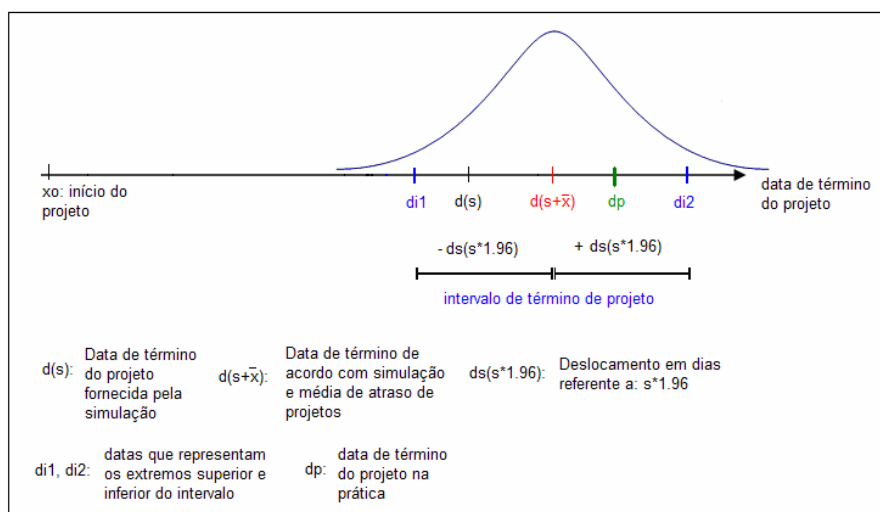


Figura 5.7 – Predição do intervalo de término do projeto.

Definida a nova data de término do projeto, o intervalo é determinado em torno da data prevista, deslocando-se à direita e à esquerda desse valor, o produto do desvio padrão da amostra fornecida, pela constante 1,96, convertido em dias. A constante 1,96 é utilizada para determinar um intervalo de término que seja estatisticamente garantido com 95% de confiança, ou seja, existem 95% de chances, do projeto na prática terminar em uma data pertencente ao intervalo determinado. Ao se utilizar um menor grau de confiança, o tamanho do intervalo tende a diminuir. Por exemplo, caso fosse utilizado o grau de confiança de 68,23%, o deslocamento seria dado apenas pelo valor do desvio convertido em dias.

Vale ressaltar, que o modelo é adaptável ao tamanho do projeto em questão, pois, para se modelar a capacidade e maturidade da organização, é utilizado o percentual de atraso em projetos da mesma, e não o número de dias de atraso. Desta maneira, tanto o tamanho do intervalo, quanto a forma da curva normal irão variar de acordo com o tamanho do projeto.

5.3 Avaliação do Intervalo de Término

A Metodologia de Predição, definida neste capítulo, fornece um intervalo de término para um projeto futuro da organização, com base na realidade atual dos seus recursos, e em uma amostra de projetos, analisada, estatisticamente, segundo o atraso.

Caso a amostra de projetos represente de fato a população, e esta apresente normalidade, existem 95% de chances do término do projeto em questão pertencer ao intervalo previsto. A aceitação do intervalo é determinada pela sua avaliação (Figura 5.8). Essa avaliação pode ser feita através da execução prática do projeto, pela equipe atual de desenvolvimento da organização, obtendo-se o término real do mesmo na prática. Caso a data de término esteja dentro do intervalo previsto, este pode ser considerado aceitável (Figura 5.7 – Seção 5.2.4).

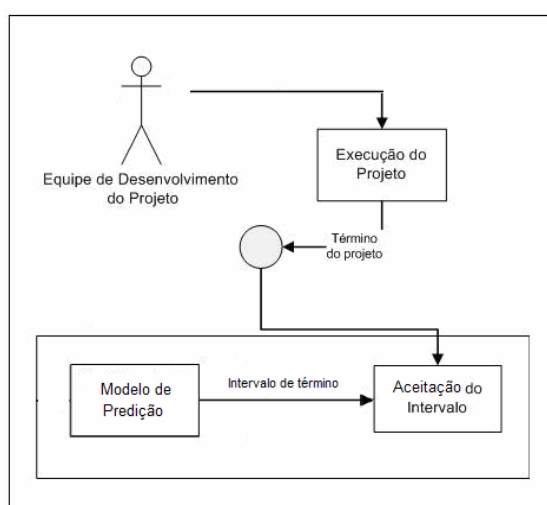


Figura 5.8 – Avaliação do intervalo de término.

5.4 Considerações Finais

Este capítulo apresentou a Metodologia de Predição de Projetos, através da descrição dos modelos que a compõem. O Modelo de Simulação é utilizado na validação de modelos instanciados de processo de software, e na predição pontual do término de um dado projeto, alvo de predição. O Modelo de Predição prediz um intervalo de término do projeto, baseando-se na data de término, fornecida pela aplicação do Modelo de Simulação, e na análise estatística do histórico de atrasos de projetos da organização.

É importante ressaltar a importância das características da organização (habilidades, e afinidades dos membros, atraso), utilizadas em ambos os modelos, no sentido de refletirem a realidade atual da organização. Desta maneira, a metodologia de predição poderá fornecer resultados mais condizentes com a execução prática do projeto.

A Metodologia de Predição de Projetos, definida por este trabalho, foi apresentada no contexto de projetos de software. Apesar disso, a metodologia pode ser aplicada para projetos em geral, como por exemplo, projetos de construção civil.

6

ProSimulator: Ferramenta de Simulação Integrada a um PSEE

Este capítulo apresenta a ferramenta de simulação de processo de software ProSimulator, construída com base no Modelo de Simulação, definido na Seção 5.1. A Seção 6.1 fornece uma visão geral sobre o ambiente ImPProS, no qual a ferramenta ProSimulator se contextualiza. A Seção 6.2 descreve, em linhas gerais, a ferramenta de análise de itens de processo de software, ProAnalyser, responsável pela análise de itens em um modelo de processo de software antes e depois da simulação. A Seção 6.3 detalha o fluxo de atividades da ferramenta ProSimulator, e apresenta os diagramas de caso de uso relacionados à mesma. A Seção 6.4 apresenta as considerações finais sobre o capítulo.

6.1 O Ambiente ImPProS

Os PSEE [Gimenez 1994] constituem uma nova geração de ADS [Reis 2000] que suportam, além da função de desenvolvimento de software, também as funções associadas de gerência e garantia da qualidade durante o ciclo de vida do software. No entanto, em alguns casos, percebe-se ao longo da execução do processo, a partir destes ambientes de desenvolvimento, que sua implementação nem sempre perfaz a realidade

das características da organização, ou do projeto desenvolvido por esta, acarretando em processos não condizentes com a realidade das empresas.

Neste contexto, o ambiente ImPProS (Implementação Progressiva de Processo de Software) [Oliveira et al. 2005], definido como projeto de Tese de Doutorado do CIn-UFPE, representa um PSEE, concebido com o objetivo principal de apoiar a implementação de um processo de software em uma organização, fornecendo apoio automatizado às fases do ciclo de vida do processo de software. O termo “progressiva” decorre do fato de que a implementação do processo é aperfeiçoada, de acordo com as experiências obtidas na sua definição, simulação, execução e avaliação. Neste caso, podem ser caracterizados como seus objetivos específicos:

- Especificar um meta-modelo de processo de software, a fim de definir uma terminologia única entre os vários modelos de qualidade de processo existentes, para uso do ambiente em seus serviços providos;
- Apoiar a definição de um processo de software para organização;
- Permitir a modelagem e instanciação deste processo;
- Permitir a simulação do processo a partir das características instanciadas para um projeto específico (ferramenta *ProSimulator*);
- Dar apoio à execução do processo de software tomando como base uma máquina de inferência;
- Possibilitar a avaliação dos critérios do processo de software;
- Apoiar a melhoria contínua do processo de software e o reuso através da realimentação e coleta das experiências aprendidas.

A ferramenta *ProSimulator*, construída com base no Modelo de Simulação definido na Seção 5.1, representa o módulo de simulação do ambiente ImPProS. A

arquitetura do ambiente é composta pelos seguintes mecanismos (Seção 1.3 - Figura 1.2):

- Mecanismos de Interação com o Usuário: responsável por prover aos usuários diferentes visões da mesma informação, sendo definida e especificada, bem como a interação entre esses usuários;
- Mecanismos para Gerenciamento do Processo no Ambiente: responsável por prover os serviços (definição, simulação, execução e avaliação do processo de software) especificados ao ambiente de forma automatizada;
- Mecanismo para Integração de Ferramentas ao Ambiente: responsável por prover a integração do ambiente com outras ferramentas de apoio ao processo de software, e à execução do projeto de software, possibilitando a automação de atividades definidas no processo e a execução de alguns módulos do ambiente;
- Mecanismo de Repositório do Ambiente: responsável por prover ao ambiente o sistema de gerenciamento dos seus objetos, a partir de bases de dados que provejam o controle de evolução e manutenção dos componentes do processo de software.

Foram definidos dois tipos de interação entre a ferramenta *ProSimulator* e o ambiente ImPProS, de acordo com o serviço requisitado pelo ambiente [Oliveira 2005a]: simulação direta, após a definição do plano de projeto no ambiente ImPProS, ou simulação a partir da seleção do processo instanciado na ferramenta *ProSimulator*.

A Figura 6.1 mostra a interação entre o ambiente ImPProS e a ferramenta *ProSimulator*, quando é feita a simulação direta, após a definição do plano de projeto. Neste caso, após o usuário definir o plano, ele infere a sua aceitação em simular os parâmetros definidos pelo mesmo. A partir daí, o ImPProS gera um arquivo XML contendo informações relevantes, incluindo a função do *ProSimulator* a ser executada (neste caso, “Simulação”), o identificador do projeto de implementação do processo, e o

identificador do plano de execução do processo de software instanciado, como mostra a Figura 6.2. Tais informações são utilizadas para efetuar a parametrização e a configuração dos dados do plano e do processo na ferramenta de simulação. Em seguida o ImPProS ativa a execução do *ProSimulator*, o qual lê e captura as informações mantidas no arquivo XML. O usuário acessa então a ferramenta *ProSimulator*, através do uso de *login* e senha, e a ferramenta, por sua vez, carrega os seus serviços que possibilitam a simulação do plano de execução, referente ao processo instanciado.

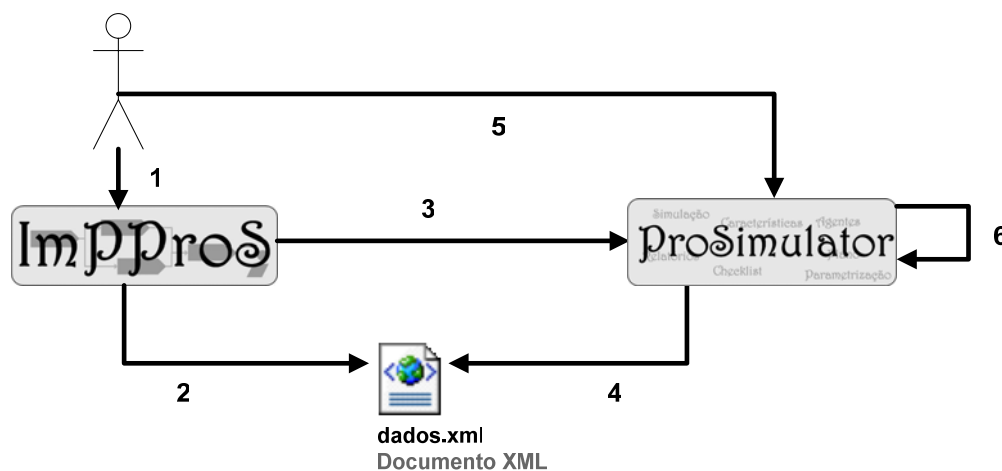


Figura 6.1 – Fluxo de execução da integração entre o ImPProS e o *ProSimulator*: simulação direta do Plano de Projeto [Oliveira 2005a].

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
- <prosimulator>
  <funcao>Simulacao</funcao>
  <idprojeto />
  <idplano>5</idplano>
</prosimulator>
  
```

Figura 6.2 – Estrutura do arquivo XML (Simulação direta do Plano de Projeto) [Oliveira 2005a].

No caso da simulação a partir da seleção do processo, inicialmente o usuário necessita ativar a execução do *ProSimulator*, a partir do ImPProS. Isso faz com que o ImPProS gere um arquivo XML, contendo as informações relevantes já definidas. Neste caso o parâmetro função recebe como valor: “Seleção Processo”. O ImPProS então ativa a execução do *ProSimulator*, e este armazena as informações do arquivo XML. O usuário, através de *login* e senha, acessa a ferramenta *ProSimulator*, e esta carrega os

seus serviços que possibilitam a simulação. Por fim, o usuário efetua a seleção do processo instanciado, e do seu respectivo plano, a ser simulado de acordo com a consulta na base de processos do ImPProS.

O arquivo XML deve ser gerado segundo o padrão de diretórios do ImPProS [Oliveira 2005b]. No caso da simulação direta do processo instanciado, a *tag* “idplano” necessita conter o valor do identificador do plano de execução do processo, que servirá como base para a simulação; Neste caso, a *tag* “idprojeto” deve conter valor nulo. Por outro lado, no caso da simulação, após seleção do processo instanciado, a *tag* “idprojeto” necessita conter o valor do identificador do projeto de implementação do processo de software, aberto no ImPProS, que servirá como base para a seleção e posterior simulação do plano de execução; Neste caso, a *tag* “idplano” deve conter valor nulo.

6.2 *ProAnalyser*: Ferramenta de Análise de Itens de Processo de Software

A tendência atual das grandes organizações é direcionar o seu esforço na prevenção de problemas, ao invés de investir na correção dos mesmos. Estima-se que as organizações de bom desempenho gastam cerca de 80% de seu esforço na prevenção de problemas, enquanto que uma organização de baixo desempenho gasta 90% de seu tempo corrigindo sintomas em vez de causas de problemas [Anjos 2001]. A vantagem de se prevenir a ocorrência de problemas, é que quanto mais cedo eles são descobertos, menos esforço será despendido para sua correção, e conseqüentemente menores serão os custos.

As organizações atuais têm ainda a necessidade de gerenciar o conhecimento adquirido no desenvolvimento dos seus projetos, possibilitando que o mesmo seja incorporado à base histórica de conhecimentos da organização, tornando-se assim independente das pessoas envolvidas no projeto em questão. A gestão do conhecimento

implica no armazenamento de informações relevantes, relacionadas ao processo organizacional, e sua disseminação para os membros da organização.

Uma área bastante promissora e crescente na área de Engenharia de Software é a Análise de Problemas de Processo de Software. Tanto o CMMI [Paulk et al. 1993], no nível de maturidade 5, como o MPS/BR [Softex 2005], em seu nível A, possuem áreas de processos (PA - *Process Area*) neste contexto, denominadas *Causal Analysis and Resolution* e Análise e Resolução de Causas, respectivamente, que têm como objetivo descrever as melhores práticas relativas à gestão de problemas.

Neste sentido, a ferramenta *ProAnalyser* visa possibilitar, no contexto da simulação e avaliação de processos no ambiente ImPProS, a análise e avaliação de itens relativos a processos de software, estando alinhada às práticas definidas pelo CMMI e MPS/BR. Os principais objetivos da construção da ferramenta *ProAnalyser* são:

- Prevenir a ocorrência de problemas de processo em ambientes de desenvolvimento de software: a partir do armazenamento de problemas relacionados aos itens, e suas possíveis soluções, uma base histórica de informações será formada, possibilitando que posteriormente sejam realizadas consultas a essa base, diminuindo assim a ocorrência de problemas semelhantes em outros processos da organização;
- Possibilitar a gestão do conhecimento organizacional: a partir da descoberta de um novo item relacionado ao processo, o mesmo será avaliado para certificar sua relevância, antes de ser armazenado na base de conhecimentos do ambiente, e disseminado para todos os membros da organização;
- Auxiliar e automatizar a avaliação de processos: os problemas podem ser descobertos, a partir da análise de itens relativos ao processo, identificando aqueles que não foram atendidos nesse contexto. A ferramenta proposta será responsável por analisar determinados itens automaticamente (análise estrutural antes da simulação). Os itens mais

subjetivos serão analisados pelo usuário (validação do processo, pós-simulação). A partir da descoberta de itens desconformes ao processo, será feita a avaliação dos mesmos, na qual a ferramenta irá inferir as possíveis soluções para essas não conformidades. É importante destacar a limitação que deve haver na quantidade de itens a serem analisados automaticamente pela ferramenta, de modo que a aquisição de conhecimento das pessoas da organização, através do uso da ferramenta, não seja comprometida.

No contexto da ferramenta *ProSimulator*, a ferramenta *ProAnalyser* é utilizada na análise estrutural do modelo de processo, anterior à simulação, e auxilia na validação do comportamento deste modelo, após a simulação. O Apêndice B descreve, com maiores detalhes, as funcionalidades da ferramenta *ProAnalyser*, fornecendo ainda um exemplo de sua utilização.

6.3 *ProSimulator*: Ferramenta de Simulação de Processo de Software

A simulação de processo de software tem se destacado como uma maneira de antever o comportamento de um modelo de processo de software, instanciado para um dado projeto, com um cronograma previsto. Neste caso, a simulação permite a validação de modelos de processos instanciados, impedindo que projetos mal planejados sejam aplicados na prática, evitando prejuízos. Além disso, a simulação pode ser utilizada como auxílio na predição de projetos, fornecendo pontualmente uma data de término para o mesmo.

Neste sentido, a ferramenta *ProSimulator* foi construída, com base no Modelo de Simulação definido na Seção 5.1, com o objetivo de permitir a validação de modelos de processo de software, auxiliando no refinamento e melhoria contínua dos mesmos, no contexto do ambiente ImPProS. Além disso, a ferramenta *ProSimulator* é utilizada para fornecer a predição pontual do término de um projeto, utilizada como base para

predição estatística de um intervalo de término para este projeto (Modelo de Predição - Seção 5.2). A Figura 6.3 ilustra o fluxo de atividades da ferramenta *ProSimulator*, detalhado nas próximas seções .

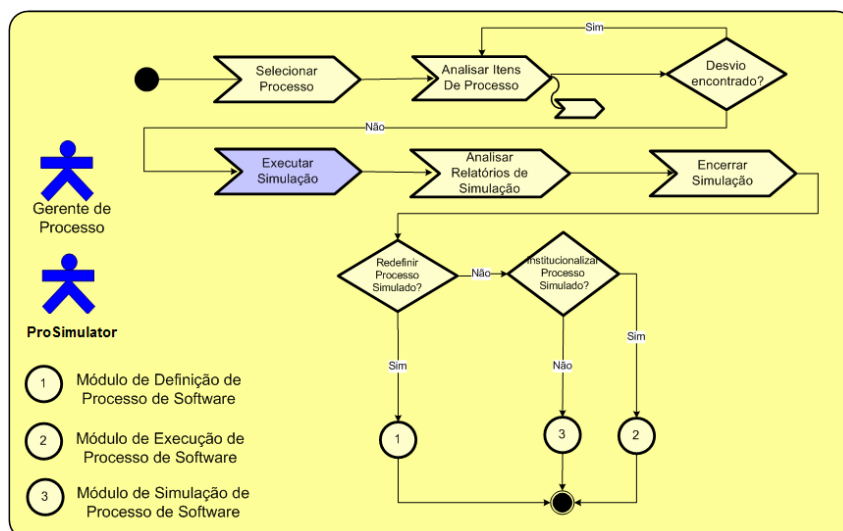


Figura 6.3 – *ProSimulator*: fluxo de atividades.

6.3.1 Seleção do Processo Instanciado

Como já definido, existem duas possibilidades do usuário acessar a ferramenta *ProSimulator*: a partir do ImPProS, após a definição do plano de execução, ou diretamente, através da seleção de um plano a ser simulado.

Neste contexto, caso o acesso seja feito a partir do ImPProS, a ferramenta *ProSimulator* exibirá todas as características específicas da organização, de seus projetos e produtos, bem como toda a estrutura do processo instanciado (atividades, papéis, artefatos...), e ainda os planos de execução associados ao mesmo, com estimativas, métricas, cronograma de atividades, dentre outras informações necessárias para guiar o usuário durante a simulação. Antes do início da simulação, o gerente de processo deve selecionar um plano de execução a ser simulado.

Por outro lado, caso o gerente de processo acesse diretamente a ferramenta *ProSimulator*, ele poderá selecionar algumas características relacionadas ao processo,

que possam ajudar o ambiente a inferir sobre os possíveis processos instanciados que ele queira simular. A partir daí serão listados os processos inferidos, e o gerente deverá escolher um deles, para que sejam listados suas características, estruturas e planos de execução. Em seguida, o gerente deverá escolher um dos planos para ser simulado.

Durante a fase de seleção do processo instanciado, o usuário poderá consultar as lições aprendidas por outros gerentes, com relação a essa atividade de seleção de processo, reutilizando um conhecimento obtido previamente. Essa funcionalidade é permitida através da ferramenta *ProKnowledge*, que possibilita a gestão do conhecimento sobre processos de software, no contexto do ambiente ImPProS.

A Figura 6.4 mostra o diagrama de caso de uso geral, referente à atividade “Selecionar Processo”, em que o gerente de processo seleciona um processo já modelado e instanciado, com um plano de execução associado. A Figura 6.5 mostra o diagrama de caso de uso detalhado da atividade “Selecionar Processo”, em que a partir das informações sobre a organização, seus projetos e produtos, selecionadas pelo gerente, é realizada a inferência de possíveis processos instanciados. O gerente de processo seleciona então um deles, para o qual serão exibidas características próprias e seus planos de execução. Um dos planos será selecionado pelo gerente de processo, no qual a simulação irá se basear.

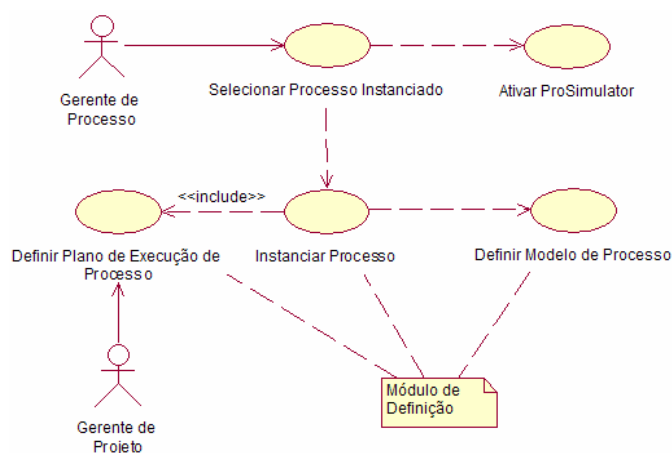


Figura 6.4 – Diagrama de caso de uso geral: seleção do processo instanciado.

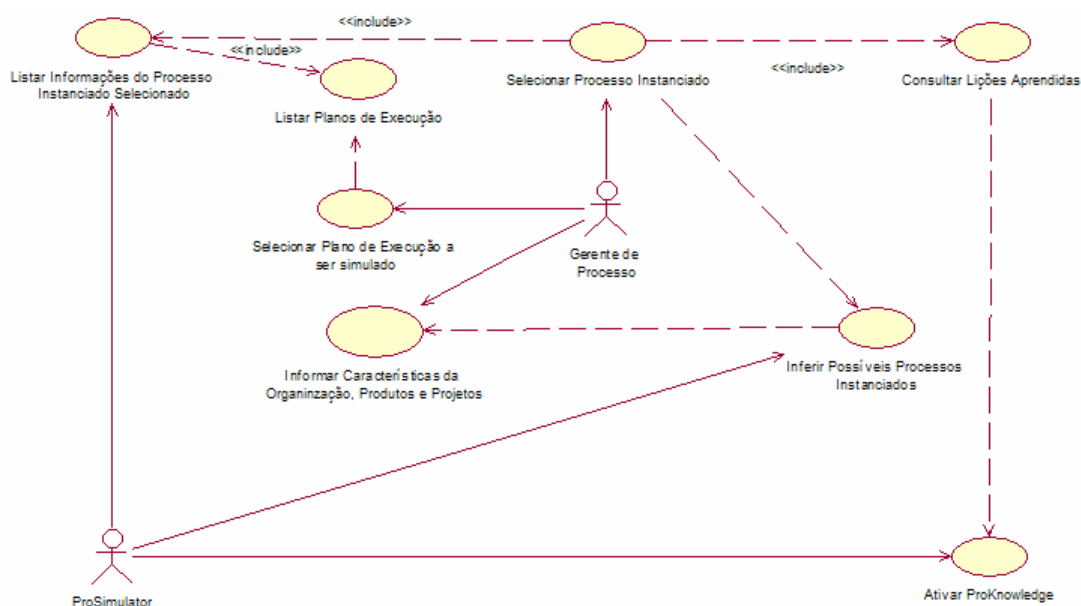


Figura 6.5 – Diagrama de caso de uso detalhado: seleção do processo instanciado.

6.3.2 Execução da Simulação

Antes do início da simulação, é feita uma pré-análise automática do modelo de processo de software instanciado, com relação a um conjunto de itens pré-definidos, relacionados a processo de software. Enquanto houver itens desconformes no modelo de processo, a simulação do respectivo plano de execução não poderá ser realizada. Essa análise de itens é feita pela ferramenta *ProAnalyser*, e está descrita com detalhes no Apêndice B.

Após a pré-análise, algumas verificações iniciais, inerentes à simulação, são feitas automaticamente pela ferramenta *ProSimulator* (consistência de datas, número suficiente de recursos para simulação...). Caso problemas não sejam detectados, a simulação é realizada, com base nas informações contidas no plano de execução definido. A simulação realizada pela ferramenta *ProSimulator*, é feita com base no algoritmo definido no Modelo de Simulação, na Seção 5.1.3. A Figura 6.6 ilustra o diagrama de caso de uso geral, referente à fase de simulação do processo instanciado.

Após a simulação, os seus resultados serão exibidos em forma de relatórios textuais e gráficos, a serem analisados pelo gerente de processo. O relatório textual

apresenta detalhes sobre as atividades executadas, sugestões de alocação de desenvolvedores para execução de tarefas, e os principais desvios relacionados ao plano de execução. O relatório gráfico exibe, na forma de um gráfico de *gantt*, uma comparação entre o cronograma previsto e simulado do projeto. Maiores detalhes sobre os resultados da simulação estão definidos na Seção 5.1. A Seção 6.3.5 mostra um exemplo de funcionamento da ferramenta *ProSimulator*, com os resultados da simulação exibidos graficamente.

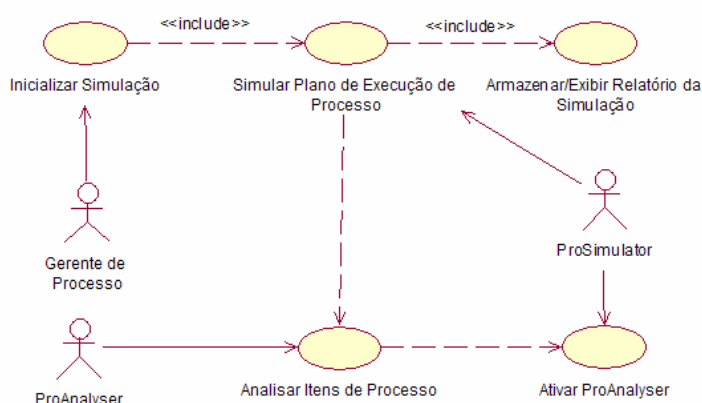


Figura 6.6 – Diagrama de caso de uso geral: simulação do processo de software instanciado.

6.3.3 Análise e Validação do Modelo de Processo

Finalizada a simulação, o próximo passo consiste na análise dos resultados da mesma, para que o modelo de processo de software, definido e instanciado, possa ser validado e institucionalizado na organização, ou refinado, a partir das deficiências encontradas.

A validação do modelo de processo poderá ser feita através da análise dos relatórios de simulação, guiada por um *check-list* de itens relacionados ao comportamento do processo instanciado. Esses itens (Tabela 6.1) complementam o conjunto de itens analisados pelo gerente (Tabela – Apêndice B), e estão relacionados aos possíveis desvios, detectados durante a simulação (Seção 5.1). Assim, o gerente de processo irá preencher o *check-list*, definindo cada um dos itens como atendido, não-atendido, ou parcialmente atendido, de acordo com os relatórios fornecidos. A avaliação

de cada item, analisado manualmente pelo gerente, é feita pela ferramenta *ProAnalyser*, que irá informar as possíveis conseqüências e soluções para os problemas detectados.

Caso não sejam observadas falhas no modelo de processo simulado, o mesmo estará validado e pronto para ser institucionalizado na organização. Caso contrário, os problemas detectados e soluções a serem adotadas, serão registradas na base de dados do ImPProS, sendo utilizados no refinamento do modelo de processo de software em questão.

Se for necessário, nesta fase, o usuário também poderá consultar as lições aprendidas, relacionadas à atividade de análise de relatórios de simulação, o que é permitido através da execução da ferramenta *ProKnowledge*, pertencente ao ambiente ImPProS. A Figura 6.7 ilustra o diagrama de caso de uso relacionado à análise e validação do modelo de processo simulado.

Tabela 6.1 – Conjunto de itens relacionados ao comportamento do processo.

Nº	Itens	Categoria
3	Conformidade entre prazo e custo estimados e simulados de atividades	Atividade
4	Carga de trabalho de agentes-desenvolvedores	Recursos
5	Conformidade entre prazo estimado e simulado de projeto	Projeto
7	Conformidade entre custo estimado e simulado para projeto	

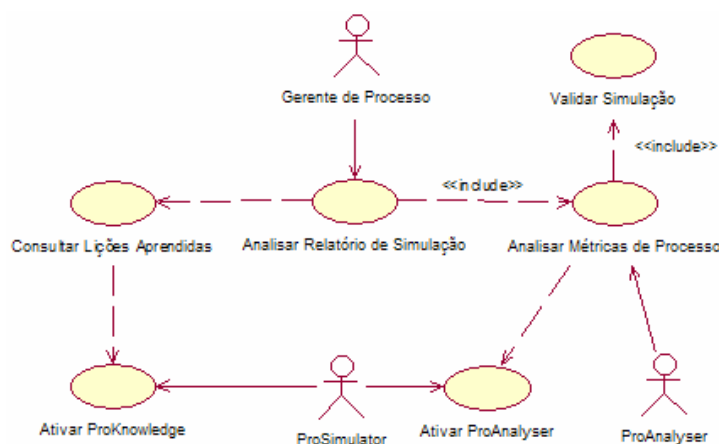


Figura 6.7 – Diagrama de caso de uso: análise e validação da simulação do modelo de processo.

6.3.4 Registro de Lições Aprendidas

Após a análise e validação do modelo de processo instanciado, a fase de simulação estará finalizada. Neste caso, serão registradas experiências e lições aprendidas durante a simulação do processo, que são na verdade fatos positivos e/ou negativos, com suas respectivas causas e consequências, que o gerente de processo julgar relevantes para a reutilização em outros processos. Essas informações serão coletadas de maneira parametrizada, por meio da ferramenta de coleta de experiências (*ProKnowledge*), que realiza o armazenamento dessas informações, formando uma base histórica de conhecimento de processos de software. Posteriormente, tais informações poderão ser consultadas por gerentes de processo, com o objetivo de reutilização do conhecimento, adquirido durante a simulação de um processo, em outros projetos.

Como cada item é avaliado separadamente (consequências e soluções próprias do item), pode ocorrer o fato do gerente de processo não ter terminado a avaliação de todos os itens considerados não-atendidos, ou parcialmente atendidos. Neste caso, a ferramenta deve informar a existência de itens ainda não avaliados, antes do encerramento da simulação. Caso se escolha continuar a análise dos itens, a ferramenta é direcionada novamente para a atividade de Analisar Métricas de Processo. Caso contrário, o *ProSimulator* será desativado, e os itens ainda não avaliados permanecerão em aberto para uma nova avaliação. A Figura 6.8 mostra o diagrama de caso de uso relacionado ao registro de lições aprendidas durante a simulação.

É importante observar que caso existam itens não atendidos, pendentes em avaliação, o processo alvo da simulação não poderá ainda ser considerado como válido para se colocar em execução na organização. Neste caso, o gerente de processo tem a opção de acessar o Módulo de Definição de Processo de Software para que seja refinado o processo de acordo com os desvios detectados na simulação, ou acessar novamente o Módulo de Simulação, caso deseje simular um novo processo instanciado. Caso não haja avaliações pendentes e todos os itens sejam atendidos, o processo é considerado válido, podendo ser institucionalizado na organização (Módulo de Execução do Processo).

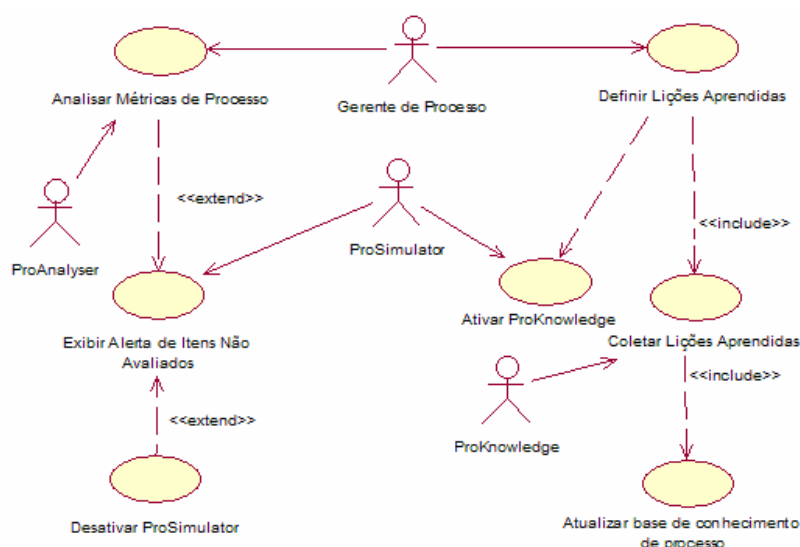


Figura 6.8 – Diagrama de caso de uso: registro de lições aprendidas e encerramento da simulação.

6.3.5 ProSimulator: Exemplo de Aplicação

A ferramenta *ProSimulator* foi desenvolvida utilizando-se a linguagem Java, no padrão J2SE (Java 2 *Standard Edition*) [Sun 2005] para aplicativos *desktop*, e o banco de dados MySQL [MySQL 2005]. Para exibição dos gráficos de *gantt* foi utilizada a *api* gráfica *JFreeChart* [JFree 2005].

Selecionado um dado modelo de processo, é feita a análise estrutural do mesmo através da ferramenta *ProAnalyser*. O Apêndice B mostra um exemplo da análise automática de itens, realizada antes da simulação. Finalizada a análise, é executada a simulação do processo, de acordo com o plano de execução selecionado.

A Figura 6.9 mostra um exemplo de gráfico de *gantt*, gerado pela ferramenta *ProSimulator*, onde as barras em azul representam o cronograma previsto pelo gerente de projetos, e as barras em vermelho, o cronograma simulado de acordo com os agentes desenvolvedores, alocados para execução das atividades. Neste caso específico, os agentes desenvolvedores foram alocados de maneira otimizada, diminuindo significativamente o prazo de projeto previsto pelo gerente. As demais informações sobre custo e desvios do processo, são fornecidas através de relatórios textuais. A

Figura 6.10 mostra um exemplo simplificado de um relatório textual, fornecido pela ferramenta *ProSimulator*.

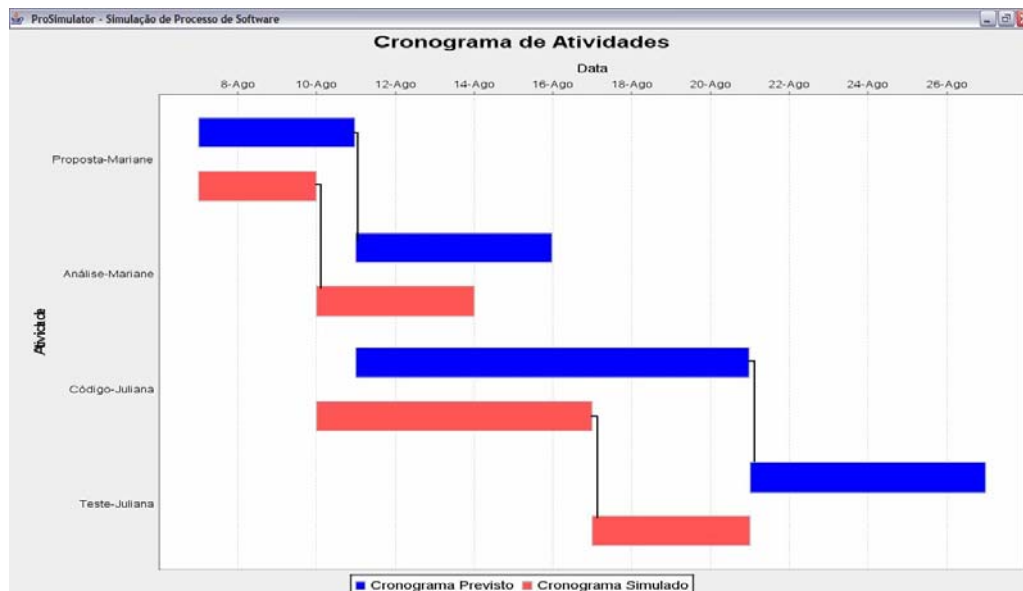


Figura 6.9 – *ProSimulator*: exemplo de gráfico de gantt .

```

ProSimulator - Relatório de Simulação

Recurso inicial disponível: 10000.0
Custo previsto em linha base: 2120.0

*****
Atividade: Elaborar Documento de Requisitos
Número de recursos necessários: 1
Custo previsto para atividade: 200.0

Alocação de Agentes Desenvolvedores:

Recurso: 1

Agente João - Valor (segundo habilidades e afinidades): 0.5
Agente Maria - Valor (segundo habilidades e afinidades): 0.5
Agente José - Valor (segundo habilidades e afinidades): 0.8

Agente mais apto disponível: José
Papel assumido: Engenheiro de Requisitos
*****
Data inicial prevista: 11/4
Data final prevista: 16/4
Tempo previsto em dias: 6
Tempo previsto de acordo com habilidades e afinidades do agente: 5
Datas da tarefa recalculadas de acordo com habilidades e afinidades: 11/4 -- 15/4

Data final simulada de acordo com agente mais lento: 15/4
*****
Custo simulado para atividade: 200.0

Pós-atividades a serem realocadas:
Atividade: Elaborar diagrama de caso de uso
Data inicial antes da realocação: 17/4
Data final antes da realocação: 17/4
Data inicial depois da realocação: 16/4
Data final depois da realocação: 16/4
.....
Custo total simulado: 1770.0

Desvios:

Atenção! Existem agentes ociosos! Verifique distribuição de atividades do cronograma definidas no plano de execução
ou aloque um número maior de recursos humanos

```

Figura 6.10 – *ProSimulator*: exemplo de relatório textual de simulação.

No contexto do ambiente ImPProS, baseando-se nos relatórios gráficos e textuais fornecidos, o gerente de processo deve preencher o *check-list* de itens, assinalando cada um como atendido, não atendido ou parcialmente atendido. A Figura 6.11 mostra um exemplo de preenchimento do *check-list* de itens, pelo gerente de processo, utilizando-se a ferramenta *ProAnalyser*. A Figura 6.12 mostra um possível exemplo do resultado da avaliação de um dado item (item 4 – Figura 6.11), feita pela ferramenta *ProAnalyser*, após a simulação.

Seleção de item para ser avaliado	Atendido	Parcialmente Atendido	Não Atendido
1 - Conformidade entre tipo de papel e tipo de atividade desenvolvida	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
2 - Correspondência entre a ordem das atividades planejadas x executadas	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
3 - Conformidade entre prazo e custo estimados e simulados de atividades	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
4 - Carga de trabalho de agentes-desenvolvedores	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
5 - Conformidade entre prazo estimado e simulado de projeto	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
6 - Adequação das habilidades dos membros em relação aos perfis definidos para as atividades do processo	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
7 - Conformidade entre custo estimado e simulado para projeto	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

Buttons: Avaliar, Analisar Itens

Figura 6.11 – *ProSimulator*: análise do processo instanciado no *ProAnalyser* após a simulação.

Avaliação de Itens

Item: Carga de trabalho de agentes-desenvolvedores

Resultado: Existem agentes ociosos ou com carga alta de trabalho

Consequência: Fluxo do processo não-contínuo

Soluções

Redefinir cronograma de atividades ou alocar mais recursos humanos para execução de atividades

Buttons: Fechar, Escolher Solução

Figura 6.12 – *ProSimulator*: avaliação do *ProAnalyser* de item não-atendido na simulação do processo.

Finalizada a validação do processo, o gerente de processo registra as lições aprendidas durante a simulação, coletadas de maneira parametrizada pela ferramenta *ProKnowledge*. Após isso, o gerente poderá redefinir o processo (Módulo de Definição), simular um novo processo (Módulo de Simulação) ou institucionalizar o processo simulado, validado (Módulo de Execução).

É importante deixar claro, o tipo de interação existente entre a ferramenta *ProSimulator* e o ambiente ImPProS: o ambiente fornece como entrada para a ferramenta *ProSimulator*, um modelo de processo de software instanciado e o(s) plano(s) de execução associado(s) ao mesmo. Por sua vez, o *ProSimulator* utiliza essas informações para fornecer ao ambiente uma avaliação prévia sobre o comportamento do processo, com os principais problemas detectados, e possíveis soluções a serem utilizadas no refinamento desse processo, no próprio ambiente ImPProS.

6.4 Considerações Finais

Este capítulo descreveu as funcionalidades da ferramenta *ProSimulator*, utilizada na validação de modelos de processo instanciados no ambiente ImPProS, e na predição pontual do término de projetos.

Cada vez mais é percebida a importância em se prevenir possíveis problemas dentro da organização, ao invés de somente corrigi-los, posteriormente à sua descoberta, resultando em um menor esforço de correção, e, conseqüentemente, em menores custos para a organização. Da mesma forma, é importante a realização da gestão do conhecimento, obtido com a detecção e correção desses problemas, de maneira que o mesmo possa ser disseminado para todos os membros da organização. A ferramenta *ProAnalyser* foi definida neste contexto, auxiliando na análise e avaliação de possíveis problemas em modelos de processo de software, antes de serem colocados em prática na organização.

O uso da ferramenta *ProSimulator* pode gerar grandes benefícios para a organização, dentre os quais se destacam o refinamento de processos, e a diminuição de prejuízos na prática, relacionados a projetos mal planejados. Quando utilizada para auxiliar no processo de predição, a ferramenta *ProSimulator* permite ainda a predição da data de término do projeto, de acordo com a realidade atual dos recursos organizacionais.

7

Estudo de Caso: Aplicação da Metodologia de Predição no domínio de Projetos de Software

Este capítulo apresenta o estudo de caso realizado em uma pequena organização, procurando avaliar os resultados da metodologia de predição de projetos, no contexto de ambientes de desenvolvimento de software. A Seção 7.1 define o objetivo do estudo de caso realizado. A Seção 7.2 descreve a organização na qual foi realizado o estudo de caso. A Seção 7.3 apresenta o estudo de caso realizado, de acordo com as etapas do Modelo de Predição (Seção 5.2). A Seção 7.5 apresenta as considerações finais sobre o capítulo.

7.1 Objetivo do Estudo de Caso

A Metodologia de Predição de Projetos, definida no Capítulo 5, tem como objetivo prever um intervalo de término para projetos futuros de uma organização, utilizando-se simulação e métodos estatísticos. A partir da aplicação desta metodologia, espera-se que a organização possa negociar prazos de projetos, com um nível maior de confiabilidade, de acordo com a sua realidade.

Como a tomada de decisão organizacional pode se basear nos resultados fornecidos pela predição, torna-se necessária a avaliação da metodologia definida. Neste sentido, o estudo de caso, descrito na presente seção, tem como objetivo verificar o potencial da Metodologia de Predição de Projetos, a partir de sua aplicação em uma empresa de desenvolvimento de software de pequeno porte.

7.2 A Organização *SwFactory*

A *SwFactory* é uma organização de pequeno porte, do setor de software, localizada na cidade de Lavras, ao Sul de Minas Gerais. A micro-empresa possui um setor de desenvolvimento de software (fábrica de software), que desenvolve produtos terceirizados em diversas áreas, além da prestação de serviços de consultoria.

Do quadro atual de funcionários e estagiários da organização, o setor de desenvolvimento de software conta com cerca de vinte colaboradores: cinco profissionais formados em Ciência da Computação e quinze cursando o curso de graduação em Ciência da Computação.

O setor de desenvolvimento de software da organização *SwFactory* possui mais de três anos de atuação, e atualmente possui projetos e produtos nas áreas de gerência de documentos fiscais, gestão imobiliária, administração escolar, automação comercial, customização de ferramentas para ensino à distância e controle cafeeiro. As plataformas de desenvolvimento predominantes, em que são desenvolvidos os produtos de software incluem a linguagem de programação *Visual Basic* e o banco de dados *Postgree-SQL*.

7.3 Aplicação da Metodologia de Predição

O estudo de caso, definido para avaliar o potencial da Metodologia de Predição de Projetos, foi realizado no setor de desenvolvimento de software da organização

SwFactory. As próximas seções detalham cada uma das etapas realizadas durante o estudo de caso.

7.3.1 Escolha dos Projetos

Inicialmente, o histórico da organização foi analisado, com o objetivo de selecionar os projetos a serem utilizados no experimento, tanto para compor a amostra, como para compor a base de projetos-piloto. Neste caso, algumas dificuldades foram encontradas, impossibilitando a escolha de um número suficiente de projetos para compor uma amostra:

- Pequeno número de projetos: devido à existência de apenas dois projetos, a análise do histórico teve que ser baseada em projetos de curta duração, que os compõem. Para cada um desses projetos de curta duração (cerca de dois a três meses), está associado um cronograma com as atividades distribuídas em fases (análise, desenvolvimento, teste...).
- Falta de informações: muitas informações, necessárias para a análise estatística (datas iniciais e finais de atividades), não estavam definidas em determinadas fases de cada projeto de curta duração. Devido a isso, fez-se necessário considerar como projeto uma dada fase (com informações completas) referente a um projeto de curta duração. Por exemplo, considerou-se como um projeto, a fase de desenvolvimento, referente ao projeto “Customização do Banco de Dados”.

Após a análise, de acordo com as limitações encontradas, foram selecionados ao todo, vinte projetos. Dos vinte, 14 foram selecionados para compor uma amostra de projetos da organização, e seis foram escolhidos para compor a base de projetos-piloto. O tamanho dos projetos-piloto escolhidos é relativamente pequeno, variando de 10 a 15 dias. O volume de tais projetos também é pequeno, sendo composto, em média, de 15 atividades. Os projetos da amostra foram utilizados como base para o cálculo da capacidade e maturidade da organização, com relação ao atraso. Os projetos-piloto

foram utilizados para avaliação dos resultados fornecidos pela Metodologia de Predição, comparados com os resultados obtidos na prática, registrados na base de dados da organização.

7.3.2 Cálculo do Atraso dos Projetos da Amostra

Uma vez definidos os projetos para compor a amostra, calculou-se a porcentagem de atraso para cada um deles. O atraso se dá pela diferença em dias entre a data de término do projeto estimada e realizada na prática. A Tabela 7.1 ilustra a porcentagem de atraso para cada projeto pertencente à amostra definida. Para melhor visualização da dispersão dos dados da amostra, a Figura 7.1 mostra o gráfico de dispersão, gerado a partir da Tabela 7.1. Os nomes dos projetos também foram ocultados, pois podem levar ao conhecimento da identidade da organização.

Tabela 7.1 – Percentual de atraso dos projetos da organização.

Histórico de projetos da organização	% atraso
1	70,59
2	71,43
3	162,50
4	30,00
5	-24,00
6	57,14
7	-56,00
8	3,45
9	-60,00
10	16,13
11	8,82
12	66,67
13	33,33
14	26,67

Uma vez definido o conjunto de dados a ser analisado, é necessário verificar se essa distribuição é normal, ou seja, semelhante a uma curva gaussiana. A Figura 7.2 ilustra o teste de Shapiro-Wilk (Seção 4.2), realizado sobre os dados de atraso de projetos da amostra, obtendo $p\text{-value} = 0.4924$, atestando, portanto, a normalidade dos dados utilizados (um valor de $p\text{-value}$ igual ou superior a 0,05, garante, com 95% de

confiança, a normalidade da amostra). O teste foi realizado utilizando-se o software estatístico *R*.

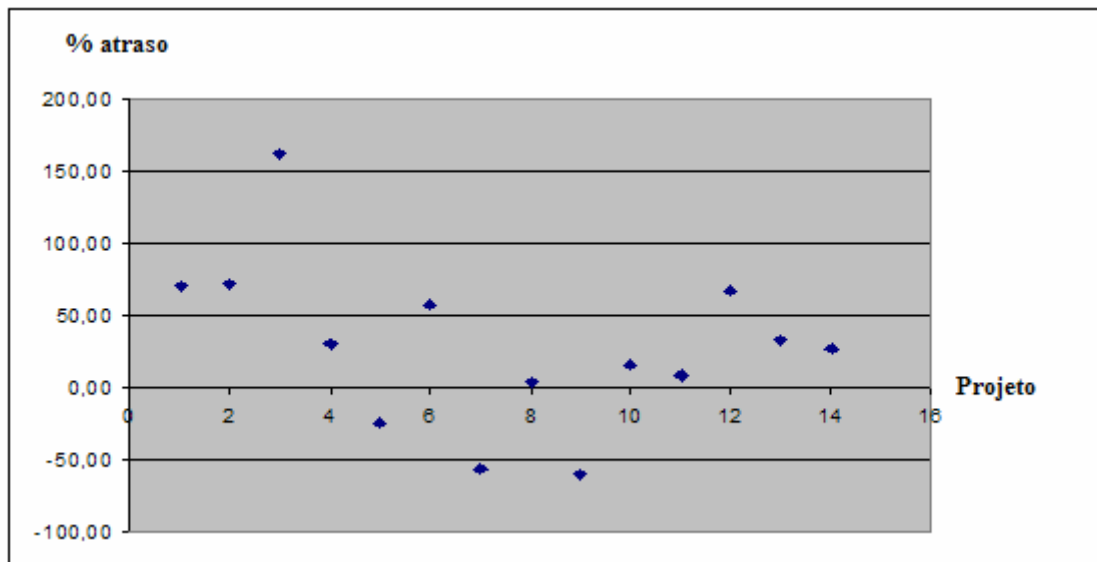


Figura 7.1 – Gráfico de dispersão: percentual de atraso dos projetos da organização.

```
> percentual<-read.table('d:/amostra.txt')
> shapiro.test(percentual[,1])

      shapiro-wilk normality test

data:  percentual[, 1]
W = 0.9454, p-value = 0.4924
```

Figura 7.2 – Relatório de teste de Shapiro-Wilk: verificação da normalidade da amostra de projetos.

7.3.3 Definição da Capacidade e Maturidade

Verificada a normalidade da distribuição da amostra, tais dados podem ser utilizados como base para predição. O próximo passo foi identificar a maturidade do processo de estimativas da organização. Essa maturidade foi verificada pela distribuição normal de probabilidades, identificada através do cálculo da média (capacidade) e o desvio padrão da amostra (Tabela 7.2). A média e o desvio calculados são relativos ao percentual de atraso da amostra de projetos.

Tabela 7.2 – Média e desvio padrão da amostra de projetos da organização.

Média (%)	29,05
Desvio padrão (%)	57,36

7.3.4 Simulação dos Projetos-Piloto

Depois da análise do histórico de projetos da organização, foram coletados dados (habilidades e afinidades) da equipe do setor de desenvolvimento da organização *SwFactory*. A coleta de dados foi realizada a partir de duas tabelas de níveis de habilidade e afinidade, preenchidas pelo gerente de projetos da organização, de acordo com a realidade e disponibilidade da equipe, na época da execução de cada projeto-piloto. A Tabela 7.3 e a Tabela 7.4 ilustram, respectivamente, as habilidades e afinidades, relacionadas aos membros da equipe do projeto-piloto dois, definidas pelo gerente da organização *SwFactory*. Os nomes dos membros da organização foram ocultados, a pedido dos sócios.

Tabela 7.3 – Projeto-piloto 2: habilidades dos membros da equipe.

	Gerência de Requisitos	Gerência de Projetos	Arquitetura	Banco de Dados	Teste	Implantação
João	0,7	0,8	0,6	0,5	0,6	0,6
José	0,7	0,8	0,7	0,6	0,8	0,6
Maria	0,6	0,7	0,8	0,5	0,8	0,6

Tabela 7.4 – Projeto-piloto 2: afinidade entre membros da equipe.

	João	José	Maria
João	-	0,8	0,8
José	0,8	-	0,9
Maria	0,8	0,9	-

É bom ressaltar que, para a aplicação prática do modelo, com o intuito de prever um projeto futuro, sugere-se a utilização de valores de habilidades e afinidades, baseados no histórico dos membros dentro do ambiente de desenvolvimento. Tais dados

também podem ser coletados através de ferramentas automáticas, instaladas no próprio ambiente de desenvolvimento (Seção 5.1.2).

Baseando-se nos parâmetros coletados, os cronogramas de cada projeto-piloto, previstos pelo gerente, foram simulados pela ferramenta *ProSimulator*. Deve-se frisar que os projetos-piloto não pertencem à amostra, sendo utilizados apenas para avaliação da metodologia. A Figura 7.3 ilustra o gráfico de *gantt*, referente à simulação do cronograma do projeto-piloto dois. A Tabela 7.5 ilustra a predição pontual de término para todos os projetos-piloto, feita pela ferramenta *ProSimulator*.

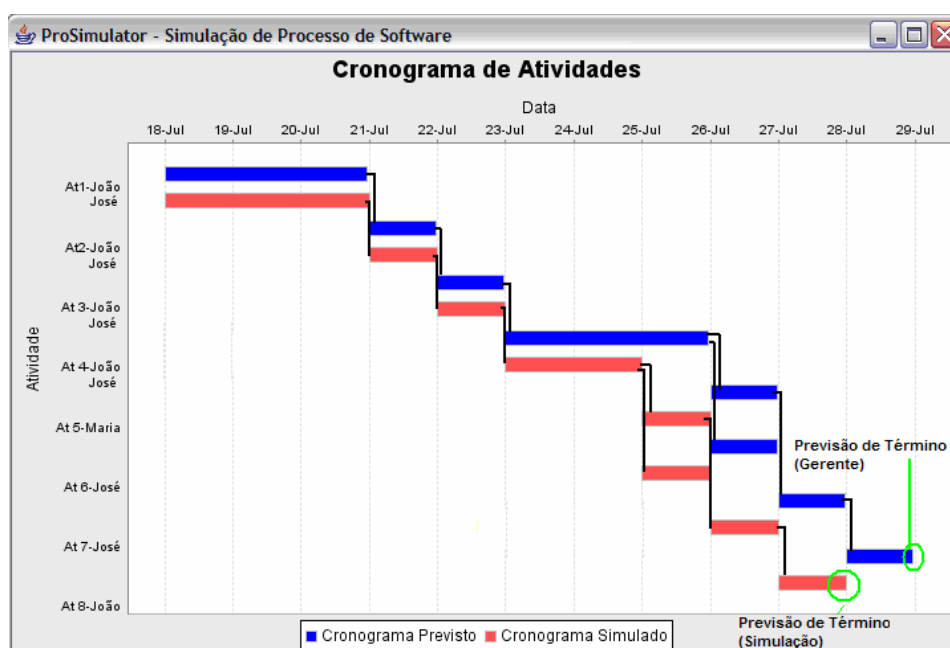


Figura 7.3 – Projeto-piloto 2: predição pontual de término na ferramenta *ProSimulator*.

Tabela 7.5 – Projetos-piloto: predição de término pela simulação.

Projetos	Predição de término (gerente)	Predição de término (simulação)	Data de término (real)
1	23/4/2006	21/4/2006	25/4/2006
2	28/7/2005	27/7/2005	8/8/2005
3	5/10/2005	1/10/2005	6/10/2005
4	29/9/2006	4/10/2006	5/10/2005
5	27/11/2005	23/11/2005	6/12/2005
6	2/7/2006	27/6/2006	23/5/2006

Pelos dados apresentados na Tabela 7.5, verifica-se que, neste caso, as estimativas definidas pelo gerente, baseadas em habilidades médias, se aproximaram de fato do que aconteceu na prática. Isso pode ocorrer em projetos, nos quais todos os desenvolvedores envolvidos possuem habilidades médias ou equivalentes, e o número de recursos disponível é suficiente e compatível com o esperado pelo gerente. Tais casos representam situações ideais de realização de um projeto, e, em tais casos, a previsão do gerente pode ser considerada válida.

Porém, existem muitos casos em que a situação ideal não ocorre (Seção 4.3 - Figura 4.6). Existem casos em que pessoas muito habilidosas, ou de muita afinidade, ou pouco habilidosas e de não muita afinidade, são alocadas nos projetos, fazendo com que haja uma diferença drástica entre a duração de atividades, prevista pelo gerente com base em habilidades ou afinidades médias, e o ocorrido na realidade. Nestes casos o uso da simulação é extremamente vantajoso, predizendo uma data de término mais próxima do que ocorreria na prática.

7.3.5 Predição do Intervalo de Término dos Projetos-Piloto

Com base na análise estatística do histórico de projetos, e na predição pontual através da simulação, o intervalo de término dos projetos analisados pôde ser determinado: Inicialmente, foi feita a predição pontual de término dos projetos-piloto, de acordo com a capacidade média de atraso da organização (Tabela 7.6).

Tabela 7.6 – Projetos-piloto: predição de término pela capacidade.

Projetos	Término (simulação)	Capacidade (dias)	Término (capacidade)
1	21/4/2006	3	24/4/2006
2	27/7/2005	3	30/7/2005
3	1/10/2005	3	4/10/2005
4	4/10/2006	4	8/10/2006
5	23/11/2005	3	26/11/2005
6	27/6/2006	12	9/7/2006

Com base na nova data de término, calculada de acordo com a capacidade, o intervalo é determinado pelo cálculo do desvio padrão multiplicado pela constante 1,96,

para o grau de confiança de 95%. A Tabela 7.7 mostra o intervalo de término previsto para cada projeto-piloto, onde x_0 e x_1 são as datas de término, que representam, respectivamente, o extremo inferior e superior do intervalo.

Tabela 7.7 – Projetos-piloto: predição do intervalo de término.

Projetos	Término (capacidade)	Deslocamento (dias)*	Término (intervalo x_0)	Término (intervalo x_1)
1	24/4/2006	12	12/4/2006	6/5/2006
2	30/7/2005	12	18/7/2005	11/8/2005
3	4/10/2005	12	22/9/2005	16/10/2005
4	8/10/2006	14	24/9/2006	22/10/2006
5	26/11/2005	10	16/11/2005	6/12/2005
6	9/7/2006	47	23/5/2006	25/8/2006

* Deslocamento correspondente a: desvio padrão x 1,96, convertido em dias.

7.4 Avaliação da Metodologia de Predição na Organização *SwFactory*

Uma vez aplicada a Metodologia de Predição, no contexto da Organização *SwFactory*, foi feita uma avaliação da mesma, com o objetivo de verificar o seu potencial. Isso foi feito verificando-se a aceitação dos intervalos de término, relacionados aos projetos-piloto previstos (Seção 5.3).

Os projetos-piloto utilizados na pesquisa têm sua execução registrada na base de dados da organização *SwFactory*. A aceitação dos intervalos foi feita, verificando se a data real de término de cada projeto, registrada na base histórica da organização, pertencia ou não ao respectivo intervalo de término, fornecido pela Metodologia de Predição (Tabela 7.8).

Tabela 7.8 – Projetos-piloto: avaliação do intervalo de término.

Projetos	Término (intervalo x0)	Término (intervalo x1)	Término (real)
1	12/4/2006	6/5/2006	25/4/2006
2	18/7/2005	11/8/2005	8/8/2005
3	22/9/2005	16/10/2005	6/10/2005
4	24/9/2006	22/10/2006	5/10/2006
5	16/11/2005	6/12/2005	6/12/2005
6	23/5/2006	25/8/2006	23/5/2006

Pelos dados apresentados na Tabela 7.8, verificou-se o potencial da Metodologia de Predição definida neste trabalho, uma vez que, para todos os seis projetos, alvo de predição, o término real dos mesmos pertence ao intervalo previsto. Esse resultado foi obtido com base no grau de confiança de 95%, considerado razoável para testes estatísticos.

É importante frisar que o grau de confiança, em que se baseia a predição, pode ser definido segundo as necessidades da organização. É notável que, quanto menor o grau de confiança, mais restrito é o intervalo previsto, e, conseqüentemente, mais difícil é do projeto, na prática, pertencer ao intervalo. Para uma predição com menor grau de confiança, também realizada por este trabalho, dos seis projetos analisados, três tiveram o término real não pertencente ao intervalo previsto.

No caso da aplicação prática da metodologia, com o intervalo de término previsto em mãos, a negociação de prazo do futuro projeto com o cliente, pode ser feita com maior confiabilidade, baseada em uma predição que considera a realidade da organização, com relação a seus atrasos e recursos atuais.

7.5 Considerações Finais

Este capítulo apresentou um estudo de caso, realizado no setor de desenvolvimento de software da organização *SwFactory*, com o objetivo de avaliar a aplicação da Metodologia de Predição.

O estudo de caso realizado permitiu a visualização prática de casos ideais em que as estimativas definidas pelo gerente de projetos estavam condizentes com a realidade. Apesar disso, o uso da simulação é muito importante na maioria dos casos, em que os projetos não executam sobre situações ideais, enfrentando situações como a falta de recursos, ou a existência de equipes grandes de desenvolvimento, em que fica difícil prever a alocação de atividades, dentre outros problemas.

Houve grande dificuldade em encontrar projetos para compor o histórico da empresa, devido ao pequeno porte e pouca maturidade da mesma. Apesar disso, o estudo de caso pôde ser realizado, utilizando-se seis projetos-piloto, para os quais foi determinado o intervalo de término, através da metodologia definida. Neste caso, o estudo de caso foi de grande valia para a empresa, que pôde verificar o problema da falta de informações importantes no histórico de seus projetos, sendo incentivada a corrigir tal desvio, e promover iniciativas para melhoria de seus processos.

Ao final do estudo de caso, foi realizada uma avaliação dos resultados previstos, demonstrando o potencial da metodologia proposta, com intervalos válidos utilizando-se o grau de confiança de 95%.

É importante ressaltar que, como a metodologia é baseada em informações próprias da organização, a aceitação de seus resultados também depende das características de habilidades e afinidades refletirem a realidade da empresa. Neste caso, ao aplicar a metodologia na prática, uma organização pode utilizar mecanismos automáticos de coleta de tais informações, para que a simulação forneça uma predição mais realística.

Além disso, a partir do incentivo às práticas de melhoria, a maturidade da empresa tende a mudar. Neste caso, o histórico de atraso de projetos deve ser constantemente atualizado, para que isso possa ser refletido na predição dos projetos futuros.

8

Considerações Finais

Este capítulo apresenta as considerações finais sobre o trabalho desenvolvido nesta dissertação. A Seção 8.1 descreve as contribuições da pesquisa realizada. Na Seção 8.2 é apresentada uma proposta de possíveis trabalhos futuros.

8.1 Contribuições do Trabalho

O trabalho descrito nesta dissertação teve por objetivo a definição de uma metodologia de predição de término de projetos, que possibilitasse uma negociação mais segura de prazos com o cliente, quando comparada às estimativas de projeto, definidas com base apenas na experiência do gerente. Com base no estudo de caso realizado, pode-se verificar o potencial da metodologia desenvolvida, para predições de intervalos de termos com grau de confiança de 95%. Neste sentido, as principais contribuições deste trabalho são:

- No contexto do ambiente ImPProS, o desenvolvimento da ferramenta *ProAnalyser*, que permite a verificação da estrutura de um modelo de processo, a partir da análise automática e manual de itens relacionados a processo.
- Também no contexto do ambiente ImPProS, o desenvolvimento da ferramenta *ProSimulator*, que permite a validação de um modelo de

processo de software definido e instanciado no ambiente, a partir da simulação de um plano de projeto associado ao mesmo.

- No contexto da Gerência de Projetos, uma metodologia de predição estatística, baseada em simulação, utilizada para predizer um intervalo de término de um projeto, no contexto de diferentes áreas, além da Engenharia de Software.

É importante frisar que o potencial da metodologia definida depende diretamente das informações relacionadas à organização, utilizadas na predição. Tais informações devem sempre ser atualizadas, com o intuito de refletir a realidade atual da organização, tanto com relação a seus recursos, quanto com relação à sua maturidade. Desta maneira, a predição fornecida terá grandes chances de condizer com os resultados futuros, a serem observados na prática.

8.2 Proposta de Trabalhos Futuros

O trabalho descrito nesta dissertação pode ser facilmente estendido. Neste caso, os seguintes trabalhos futuros são propostos:

- Coleta automática de níveis de habilidade e afinidade: no contexto do ambiente ImPProS, para melhor refletir a realidade organizacional, sugere-se como trabalho futuro, a incorporação de uma ferramenta de descoberta de conhecimento, para que os níveis de habilidade e afinidade dos agentes possam ser determinados automaticamente, com base em dados históricos, coletados por essa ferramenta, instalada no ambiente de desenvolvimento.
- Implementação da Heurística do Agente Menos Restritivo: no contexto do Modelo de Simulação, a alocação do agente-desenvolvedor, utilizando-se a heurística do agente mais apto, pode ser incrementada.

Caso a atividade envolva mais de um agente, o primeiro a ser alocado deve possuir a maior média de afinidades com os demais, ao invés de ser analisada apenas a sua habilidade, ou seja, deve ser alocado, o agente que menos restrinja a escolha dos demais (Seção 5.1.2).

- Estudo do Modelo de Otimização de Alocação de Desenvolvedores, aplicado à predição: este trabalho descreve, no Apêndice A, um modelo de otimização, definido para promover a melhoria dos resultados (prazo e custo) obtidos com a heurística de maior habilidade (Modelo de Simulação). O trabalho futuro a ser desenvolvido consiste no estudo do impacto do modelo na Metodologia de Predição, quando o mesmo for aplicado ao Modelo de Simulação.
- Automatização da Metodologia de Predição: a metodologia pode ser automatizada, a partir da incorporação das etapas definidas no Modelo de Predição, à ferramenta *ProSimulator*.

Referências Bibliográficas

- [Abdel-Hamid e Madnick 1961] ABDEL-HAMID, T.; MADNICK, S. E. Software Project Dynamics: An Integrate Approach. 1961. Prentice-Hall, Englewood Cliffs, NJ, EUA.
- [Acuña e Ferré 2001] ACUÑA, S. T.; FERRÉ, X. Software Process Modelling. In: WORLD MULTICONFERENCE ON SYSTEMICS, CIBERNETICS AND INFORMATICS, 5, 2001, Orlando, EUA. **Proceedings...** p. 1-6.
- [Anjos 2001] ANJOS, R. Implantação de Processo de Software, 2001. Disponível em:<<ftp.mct.gov.br/temas/info/Dsi/PBQP/Reuniao%20BSB/ApresLG.pdf>>. Acesso em: 15 Set. 2005.
- [Armenise et al. 1992] ARMENISE, P.; BANDINELLI, S; GHEZZI, C. MORZENTI, A. Software Processes Representation Languages: Survey and Assessment. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, 4, 1992, Capri, Italy. **Proceedings...** p. 455-462.
- [Barros 2001] BARROS, M. **Gerenciamento de Projetos Baseados em Cenários**. 2001. 249 p. Tese (Doutorado em Engenharia de Sistemas e Computação), Universidade Federal do Rio de Janeiro, Rio de Janeiro.
- [Curtis et al. 1992] CURTIS, B; KELLNER, M. I; OVER, J. Process Modeling. **Communications of the ACM**, v. 35, n. 9, 1992 New York, EUA. p. 75-90.
- [Davies 1979] DAVIES, N.R. Interactive Simulation Program Generation. Methodology in Systems Modeling and Simulation. North-Kolland Publishing Company, 1979.

- [Drappa e Ludewig 2000] DRAPPA, A.; LUDEWIG, J. Simulation in Software Engineering Training. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE '00), 22, 2000, Limerick, Ireland. **Proceedings...** p. 199.
- [Figueira e Ramalho 2000] FIGUEIRA, R; RAMALHO, G. L. JEOPS - The Java Embedded Object Production System. **Lecture Notes in Computer Science** In: ADVANCES IN ARTIFICIAL INTELLIGENCE: INTERNATIONAL JOINT CONFERENCE; 7th IBERO-AMERICAN CONFERENCE ON AI; IBERAMIA-SBIA 2000, Atibaia, SP, Brazil. **Proceedings...** p. 53.
- [Gimenez 1994] GIMENEZ, I. M. S. O Processo de Engenharia de Software: Ambientes e Formalismos. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA, 13, 1994, Caxambu, MG, Brazil. **Anais...**
- [Guedes 2006] GUEDES, M. S. **Um Modelo Integrado para Construção de Jogos de Computador Aplicado à Capacitação em Gerência de Projetos**. 2006. 156 p. Dissertação (Mestrado em Ciência da Computação), Universidade Federal de Pernambuco, Recife.
- [Hoover e Perry 1989] HOOVER, S. V.; PERRY, R. F. Simulation: a problem-solving approach. Reading, Massachusetts: Addison-Wesley Longman Publishing Co., Inc. Boston, EUA, 1989.
- [Humphrey e Kellner 1989] HUMPHREY, W. S.; KELLNER, M. I. Software process modeling: principles of entity process models. In ICSE '89: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE '89), 11, 1989, New York, EUA. **Proceedings...** p. 331–342.
- [Humphrey 1989] HUMPHREY, W. S. Managing the Software Process. **SEI Series in Software Engineering**, Massachusetts: Addison-Wesley Longman Publishing Co., Inc. Boston, EUA, 1989.
- [JFree 2005] JFree. *JFreeChart Project*. 2005. Disponível em: <<http://www.jfree.org/jfreechart/>>. Acesso em: 20 Set. 2006.
- [Jones 1999] Jones, C. Software Project Management in the 21st Century. American Programmer, Volume XI, N. 2, fevereiro 1999.
- [Lima 2001] LIMA, C. A. R. **Instanciação, Validação e Execução de processos de software baseados em conhecimento**. 2001. Proposta de Tese (Doutorado em Informática), Universidade Federal do Rio Grande do Sul, Porto Alegre.

- [Lin e Levary 1989] LIN, C. Y.; LEVARY, R. R. 1989, Computer-Aided Softwar Development Process Design. **IEEE Transactions on Software Engineering**, v. 1, 15, pp. 1025 – 1037.
- [Madachy e Boehm 1999] MADACHY, R. J.; BOEHM, B.W. Software Process Dynamics. Early Draft v. 4, 1999, **IEEE Computer Society Press**, Los Alamitos, CA, EUA. Disponível em: <<http://www-rcf.usc.edu/~madachy/spd>>. Acesso em: 20 Out. 2006.
- [Martin 1997] MARTIN, L. A. **The First Step**. 1997, Relatório Técnico, D-4694, MIT System Dynamics Group, Cambridge, MA.
- [Mi e Scacchi 1990] MI, P.; SCACCHI, W. A Knowledge-Based Environment for Modeling and Simulating Software Engineering Process. **IEEE Transactions on Knowledge and Data Engineering**, v.2, n. 3, Los Alamitos, CA, EUA, 1990. p. 283-294.
- [Murta 2002] MURTA, L. G. P. **Charon: Uma Máquina de Processos Extensível Baseada em Agentes Inteligentes**. 2002. 118 p. Dissertação (Mestrado em Engenharia de Sistemas e Computação), Universidade Federal do Rio de Janeiro, Rio de Janeiro.
- [MySQL 2005] MySQL. *MySQL Documentation*. 2005. Disponível em: <<http://mysql.com/doc/>> Acesso em: 20 Dez. 2005.
- [Oliveira 2006] OLIVEIRA, M. S. de. Qualidade de Processo de Software: Medição e Análise. 2006 (Livro) Lavras/MG: FAEPE – Pós-Graduação Lato Sensu Universidade Federal de Lavras.
- [Oliveira et al. 2005] OLIVEIRA S. R. B; VASCONCELOS, A. M. L.; ROUILLER, A. C. Uma Proposta de um Ambiente de Implementação de Processo de Software. **INFOCOMP Journal of Computer Science**, v. 4, n. 1, Lavras, MG, Brazil, 2005. p. 70-77.
- [Oliveira 2005a] OLIVEIRA S. R. B. Integração ImPProS. v. 1.1, 2005. Documentação interna do projeto ImPProS.
- [Oliveira 2005b] OLIVEIRA S. R. B. Padrão de Diretórios das Ferramentas. v. 1.1, 2005. Documentação interna do projeto ImPProS.
- [OMG 2005a] *Object Management Group (OMG). UML – Unified Modeling Language, Version 2.0*, 2005.
- [OMG 2005b] *Object Management Group (OMG). SPem – Software Process Engineering Metamodel Specification, V.1.1*, 2005.

- [Paula 2003] Paula, W. P. Engenharia de software : fundamentos, métodos e padrões. Rio de Janeiro: LTC, 2003. 584p.
- [Papadimitriou e Steiglitz 1982] PAPADIMITRIOU, C. H; STEIGLITZ, K. Combinatorial Optimization - Algorithms and Complexity. 1982. Dover Publications INC.
- [Paulk et al. 1993] PAULK, M.; CURTIS, B., CHRISSIS, M. B., WEBER, C. V. Capability Maturity Model for Software, Version.1.1. **IEEE Software**, Reading: IEEE Computer Society Press, Los Alamitos, CA, 1993.
- [Paulk et al. 1994] PAULK, M.C., CURTIS, B., CHRISSIS, M. B., WEBER, C. V. The Capability Maturity Model: guidelines for improving software process. The SEI Series in Software Engineering, Addison-Wesley, 1994.
- [PMI 2004] *Project Management Institute (PMI) editor. A guide to the Project Management body of knowledge: PMBOK Guide. 3th Edition: Project Management Institute, Inc, 2004.*
- [Pressman 2002] PRESSMAN, R. Engenharia de Software, 2002. São Paulo: Makron Books.
- [Rational 1987] *Rational. RUP – The Rational Unified Process, 1987.*
- [Reis 1998] REIS, C. A. L. **Um Gerenciador de Processos de Software para o Ambiente PROSOFT**. 1998. 197 p. Dissertação (Mestrado em Informática), Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [Reis 2000] REIS, R. Q. **Reutilização de Processos de Software**. 2000. Exame de Qualificação de Tese (Doutorado em Informática), Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [Reis 2003] REIS, C. A. L. **Uma Abordagem Flexível para Execução de Processos de Software Evolutivos**. 2003. 267 p. Tese (Doutorado em Informática), Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [Royston 1995] ROYSTON P. A Remark on Algorithm AS 181: The W Test for Normality. **Applied Statistics**, 44, p. 547-551, 1995.
- [Rus et al. 1998] RUS, I.; COLLOFELLO, J.; LAKEY, P. Software Process Simulation for Reliability Strategy Assessment. **Journal of System and Software**, 46, p. 173-182, 1999.

- [Russel e Norvig 2003] RUSSELL, S.; NORVIG, P. Artificial Intelligence: A Modern Approach. 2003 2th Edition: Prentice-Hall, Englewood Cliffs, NJ, EUA.
- [Scacchi 1999] SCACCHI, W. Experience with Software Process Simulation and Modeling. **Journal of System and Software**, 46, p. 183-192, 1999.
- [Sendall e Kuster 2004] SENDALL, S.; KUSTER, J. Taming model round-trip engineering. In: WORKSHOP ON BEST PRACTICES FOR MODEL-DRIVEN SOFTWARE DEVELOPMENT, 2004, Vancouver, Canada. **Proceedings...**
- [Shimizu 1975] SHIMIZU, T. Simulação em Computador Digital. 1975. São Paulo: Edgard Blucher.
- [Silva 2000] SILVA, D. R. D da. **Atores Sintéticos em Jogos de Aventura Interativos: O Projeto Enigmas no Campus**. 2000. 165 p. Dissertação (Mestrado em Ciência da Computação), Universidade Federal de Pernambuco, Recife.
- [Silva et al. 1999] SILVA, F. A. das D.; REIS, R. Q.; REIS C. A. L.; NUNES, D. J. Um Modelo de Simulação de Processos de Software Baseado em Agentes Cooperativos. In: SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE, 13, 1999, Florianópolis, Brasil. **Anais...** p. 163-178.
- [Silva 2001] SILVA, F. A. das D. **Um Modelo de Simulação de Processos de Software Baseado em Conhecimento para o Ambiente PROSOFT**. 2001. 124 p. Dissertação (Mestrado em Informática) - Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [Softex 2005] SOFTEX - Associação para Promoção da Excelência do Software Brasileiro: MPS/BR – Melhoria de Processo de Software Brasileiro; 2005. Disponível em: <http://www.softex.br/mpsbr/> Acesso em: 20 Out. 2005.
- [Sun 2005] SUN MICROSYSTEMS. J2SE – Java 2 *Standard Edition* – Disponível em: <http://java.sun.com/j2se/> Acesso em 20 Dez. 2005.
- [Standish 1995] The Standish Group, “Chaos”, Disponível em: <http://www.standishgroup.com/chaos.htm>. 1995. Acessado em abril de 2005.

Apêndice A

Modelo de Otimização aplicado à Alocação de Desenvolvedores

Alocação de Desenvolvedores com Heurística de Maior Habilidade

A alocação de desenvolvedores para execução de atividades é uma tarefa a ser realizada pela ferramenta *ProSimulator*, durante a simulação do plano de execução do processo. Segundo Silva [Silva 2001], uma das maneiras de se realizar a alocação, visando melhores soluções, e uma simulação que represente fielmente a execução do processo, é escolher para ser alocado, aquele desenvolvedor que possuir a maior habilidade para execução da tarefa em questão.

Porém, essa decisão por si só, não reflete, necessariamente, em uma otimização no tempo total de execução do processo, com relação ao estimado. Ao contrário, existem casos, em que a utilização dessa heurística da maior habilidade, acaba por atrasar o processo de desenvolvimento. A Figura A.1 e a Figura A.2 mostram um exemplo em que isso acontece.

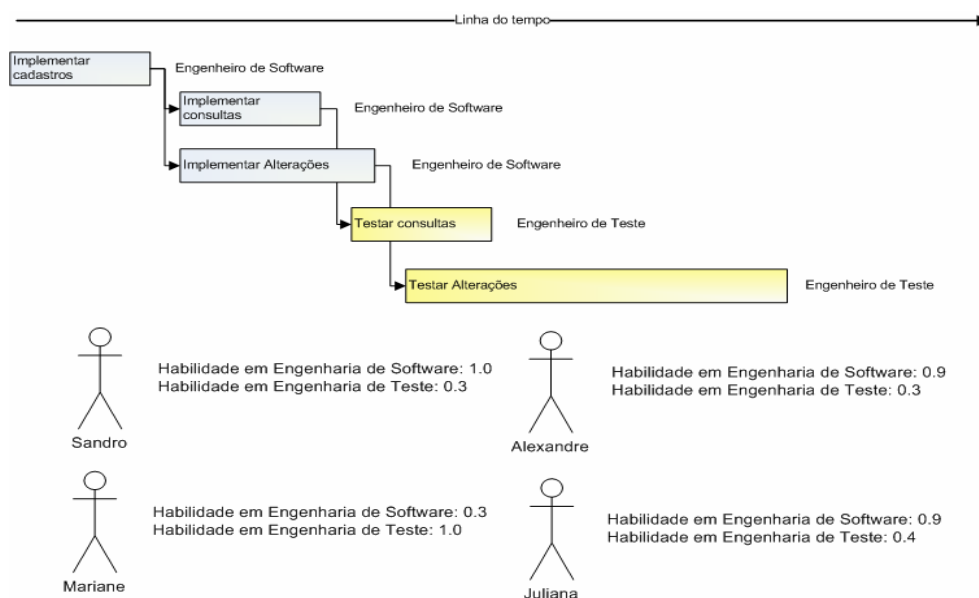


Figura A.1 – Cronograma de atividades e habilidades dos desenvolvedores.

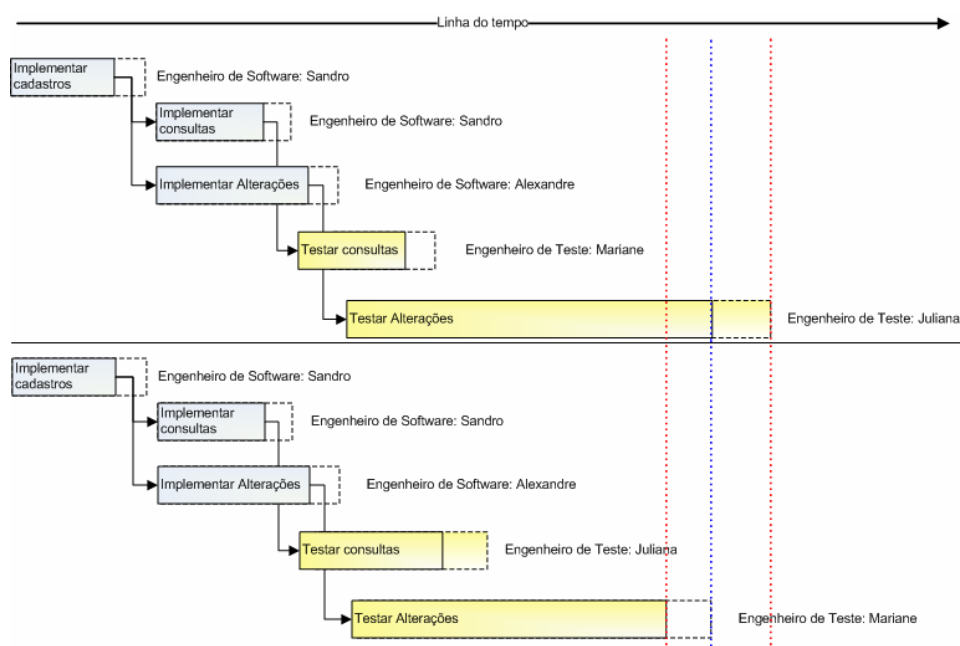


Figura A.2 – Execução do cronograma: heurística de maior habilidade x busca local.

A Figura A.1 mostra um cronograma de atividades, com os respectivos papéis definidos para execução das mesmas. São mostrados ainda alguns desenvolvedores com seus respectivos níveis de habilidades para determinados tipos de atividades.

A Figura A.2 mostra, em sua primeira parte, a alocação dos desenvolvedores de acordo com a heurística de maior habilidade, baseando-se nessas características para determinar o tempo de execução de cada atividade. A Seção 5.1.2 mostra o cálculo do tempo de acordo com habilidades e afinidades.

Percebe-se que, devido à utilização da heurística de maior habilidade, na alocação da atividade “Testar consultas”, a atividade ‘Testar alterações’ teve um aumento no seu tempo de execução, pois o desenvolvedor de menor habilidade para tal atividade, teve que ser alocado, por ser o único disponível no momento. Esse aumento resultou em um tempo total de execução maior do que o estimado (linha mediana pontilhada).

De outra maneira, a Figura A.2 mostra, em sua segunda parte, uma outra possível configuração de alocação. Desta vez, é feita a alocação de um desenvolvedor com menor habilidade para a tarefa “Testar consultas”, o que fez a última opção de desenvolvedor, a ser alocado para a tarefa “Testar alterações”, ser o de maior habilidade para a mesma, diminuindo significativamente seu tempo de execução. Essa decisão fez com que o tempo total de execução do processo fosse menor que o estimado.

A próxima seção descreve, em linhas gerais, um modelo de otimização do tempo total de execução, obtido com a heurística do agente mais apto, a partir da identificação de pontos, considerados focos de melhoria (tal como o par de atividades “Testar consultas” e “Testar alterações”, cuja troca de alocação resulta em uma otimização no tempo final de execução).

Modelo de Otimização: Melhoria dos Resultados obtidos com a Heurística do Agente Mais Apto

A alocação de desenvolvedores, segundo a heurística de maior habilidade, nem sempre resulta em bons resultados com relação ao tempo final de execução do processo. Neste

sentido, este trabalho propõe a utilização de um Modelo de Otimização, que procura melhorar os resultados obtidos com a heurística do agente mais apto (Figura A.3).

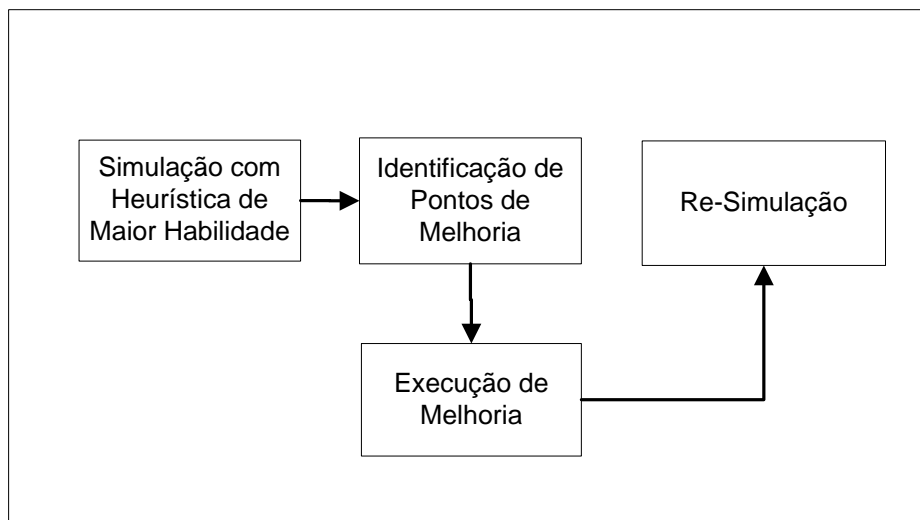


Figura A.3 – Modelo de Otimização de Alocação de Desenvolvedores.

Inicialmente a simulação é realizada, com base na alocação pela heurística do agente mais apto. Posteriormente à simulação, são identificados pontos, considerados focos de melhoria. Neste caso, pontos de melhoria são os possíveis pares de atividades, cuja troca de desenvolvedores alocados promove uma melhoria no tempo total de execução do processo. Para isso devem ser obedecidos alguns critérios:

- Intersecção das atividades: Para que o par de atividades seja considerado um ponto de melhoria, deve haver intersecção entre os períodos de execução de cada atividade. Além disso, o período de execução da segunda atividade deve ser maior que o da primeira (isso aumenta as chances da melhoria realizada se refletir no tempo total de execução).
- Habilidade do desenvolvedor: Para que o par de atividades seja foco de melhoria, a habilidade do desenvolvedor, alocado para a primeira tarefa, com relação à segunda tarefa, deve ser maior que a habilidade do desenvolvedor já alocado para a segunda tarefa, com relação à mesma.

- Disponibilidade do desenvolvedor: O desenvolvedor alocado previamente na segunda tarefa, não deve estar ocupado entre o intervalo das datas iniciais das atividades envolvidas no par de melhoria. Caso esteja, ele não poderá ser realocado (execução da melhoria), e, portanto, este não será considerado um ponto de melhoria.

Após a identificação de um ponto de melhoria, a otimização é realizada neste ponto, ou seja, é realizada a troca dos desenvolvedores alocados em cada atividade, ocasionando a diminuição do tempo total de execução do cronograma, após sua re-simulação. A Figura A.4 mostra um exemplo da aplicação do Modelo de Otimização, na ferramenta *ProSimulator*.

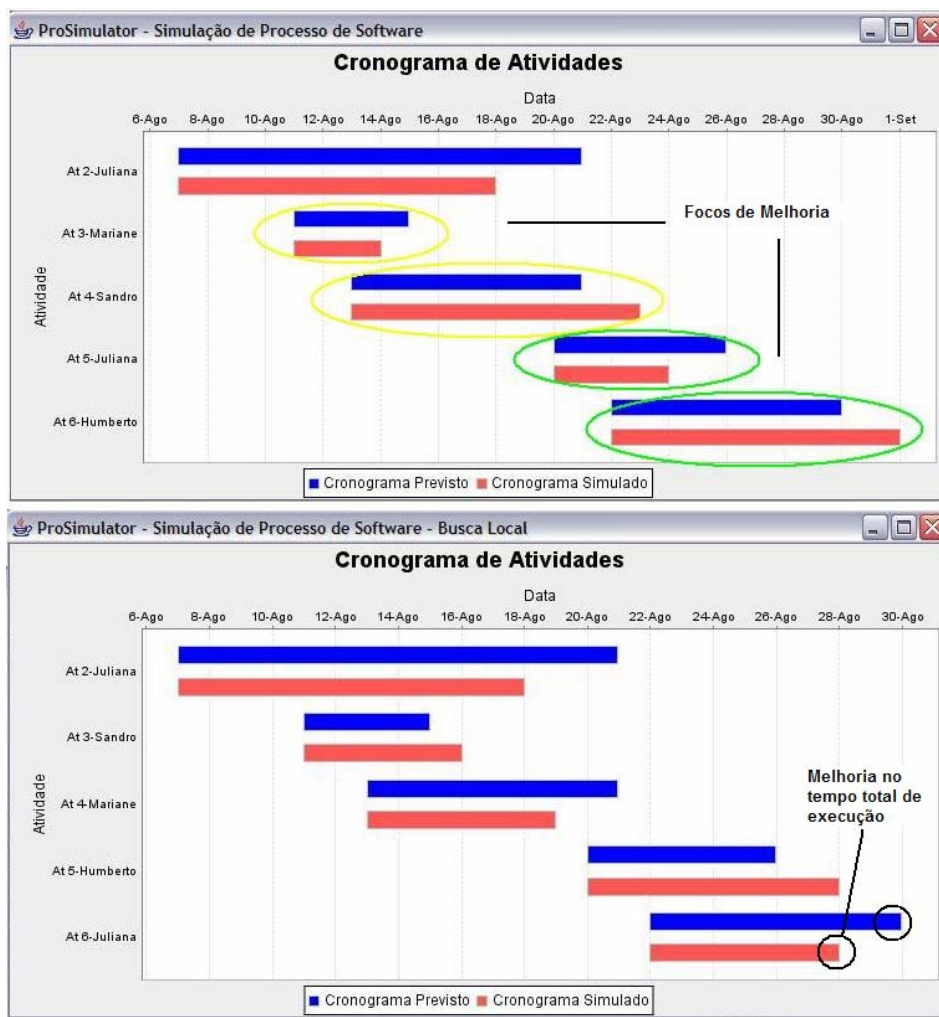


Figura A.4 – ProSimulator: aplicação do modelo de otimização.

É importante ressaltar, que o objetivo final deste trabalho não está no contexto da otimização da alocação de desenvolvedores, sendo o modelo definido na Figura A.3, uma possível extensão do Modelo de Simulação aqui proposto. O objetivo principal deste trabalho está relacionado à Metodologia de Predição, definida no Capítulo 5. Apesar disso, é proposto como trabalho futuro, o estudo do impacto da aplicação do Modelo de Otimização, definido na presente seção, como base para a Metodologia de Predição já definida.

Apêndice B

ProAnalyser: Ferramenta de Análise de Itens de Processo de Software

Diagrama de Caso de Uso e Fluxo de Atividades

A ferramenta *ProAnalyser* é responsável pela análise de itens de processo de software, a ser realizada antes da simulação, em uma fase de pré-verificação estrutural do modelo de processo, e após a simulação, em uma fase de validação do modelo instanciado e simulado.

A modelagem por diagramas de caso de uso da ferramenta *ProAnalyser* (usando a notação UML), é ilustrada na Figura B.1. A Figura B.2, usando a notação de modelagem de processo de software, definida pelo SPEM, descreve o fluxo de atividades da Avaliação da Relevância de um Item, antes que o mesmo possa ser de fato considerado válido, e armazenado na base de dados. Da mesma forma, a Figura B.3 descreve o fluxo de atividades da Análise e Avaliação de Itens de Processo de Software, referente à ferramenta *ProAnalyser*. As próximas seções detalham os fluxos da B.2 e

B.3, e apresentam um mapeamento entre as áreas de processo definidas no CMMI e no MPS/BR, e as funcionalidades da ferramenta *ProAnalyser*, no contexto do ambiente ImPProS.

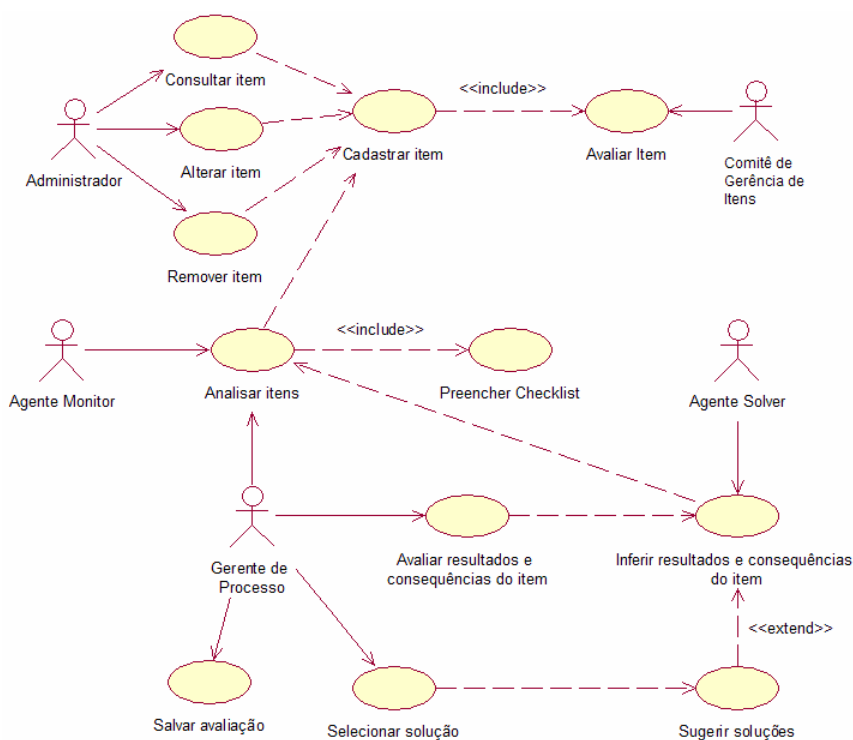


Figura B.1 – Diagrama de caso de uso da ferramenta *ProAnalyser*.

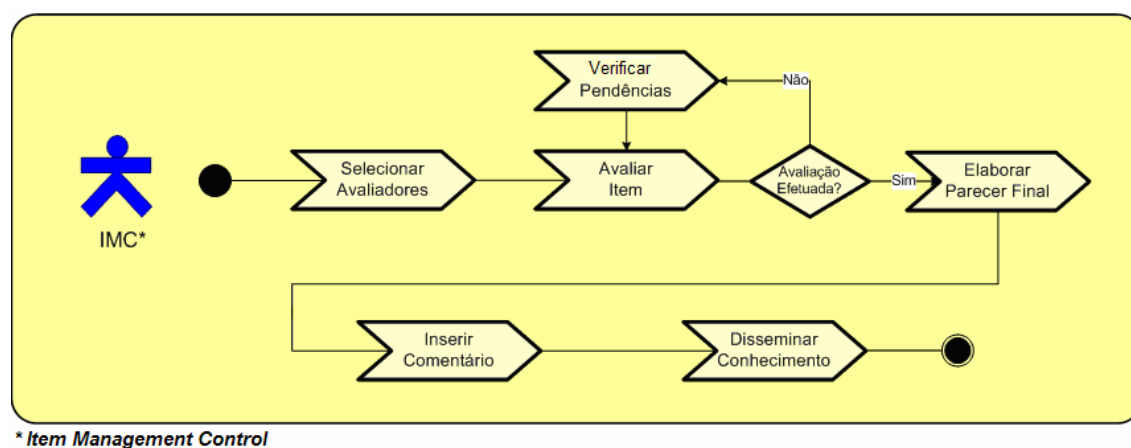


Figura B.2 – Fluxo de atividades: avaliação da relevância de itens de processo de software.

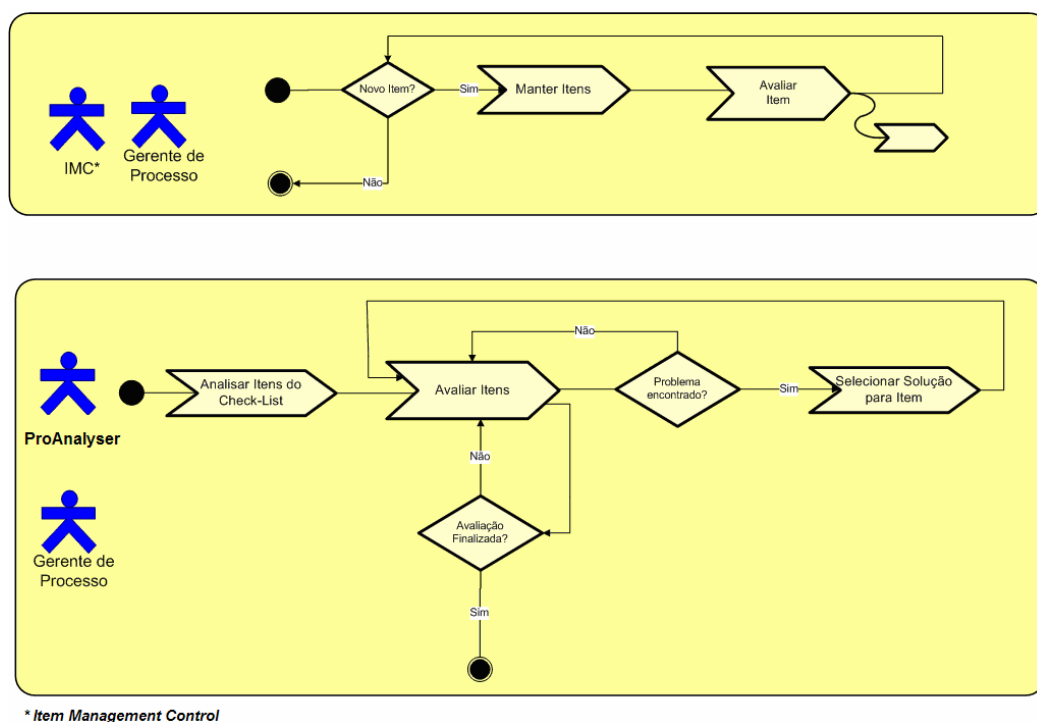


Figura B.3 – Fluxo de atividades: análise e avaliação de itens de processo de software.

Avaliação da Relevância do Item

Para cada item cadastrado, deverá ser feita sua avaliação, antes de ser efetivamente registrado na base de dados. Para cada item a ser avaliado, será escolhido um grupo de avaliadores que pertencem ao IMC (*Item Management Control*), que é um comitê de gerenciamento de itens. Os avaliadores serão selecionados, de acordo com o grau de conhecimento em relação ao item (atividade “Selecionar Avaliadores” – Figura B.2).

O comitê é responsável por avaliar a relevância e consistência do item, e ponderar o impacto do mesmo (atividade “Avaliar Item” - Figura B.2). Cada um dos avaliadores selecionados é responsável por realizar uma avaliação individual com relação ao item.

O controle das avaliações individuais dos itens (atividade “Verificar Pendências” - Figura B.2) será feito da seguinte maneira: inicialmente, cada membro

escolhido receberá uma notificação de que existe uma nova avaliação de item a ser feita, com prazo de três dias. Se o membro escolhido não realizar a avaliação, será concedido mais um dia para que ele a realize. Se ainda assim a avaliação não for feita, ele será substituído por outro membro do comitê, ao qual será enviada a notificação de avaliação.

O avaliador que não tiver participado da primeira avaliação será responsável pelo parecer final, baseado nos resultados das avaliações individuais (atividade “Elaborar Parecer Final” - Figura B.2). Um novo avaliador é escolhido, para que o resultado final não seja condicionado a uma pré-avaliação individual, mas seja uma representação da opinião de todos os demais avaliadores. Os avaliadores podem ainda fazer comentários a respeito do item analisado, sendo este aprovado ou reprovado na avaliação (atividade “Inserir Comentário” - Figura B.2).

Os itens aprovados pelo grupo de avaliadores serão notificados a todas as pessoas da organização, com o objetivo de disseminação do conhecimento (atividade “Disseminar Conhecimento” - Figura B.2). Caso algum item não seja aprovado, ele será excluído do sistema, e a pessoa que o cadastrou será informada.

Análise e Avaliação de Itens

Em qualquer momento, podem ser cadastrados os itens que o gerente de processo julgar relevantes para a avaliação do processo (atividade “Manter Itens” - Figura B.3), que serão avaliados pelo IMC (atividade “Avaliar Item” - Figura B.3). Por padrão, os itens possuem uma descrição, os valores que ele pode assumir e a categoria a qual eles pertencem.

Os valores que o item pode assumir são “Atende”, “Não Atende” ou “Atende Parcialmente”. Se o gerente julgar que o item não tem problemas, ele assume o valor “Atende”, caso contrário assume o valor “Não Atende”. Se o item foi atendido de maneira parcial, ele assume o valor “Atende Parcialmente”. Um exemplo de item a ser

analisado é “Os artefatos definidos pelo processo foram produzidos”. Se todos os artefatos do processo foram produzidos, este item está atendido; se apenas uma porcentagem dos artefatos do processo foram produzidos, o item está parcialmente atendido; se nenhum artefato do processo foi produzido, o item não foi atendido.

Um agrupamento de itens relacionados forma uma categoria. Ex: Poderíamos agrupar os itens “Checar relacionamento entre as atividades” e “Checar conformidade do tipo de atividade” na categoria Atividade. Um item pode pertencer a mais de uma categoria. Ex: o item “Analisar conformidade do tipo de procedimento com a granularidade da atividade” está relacionado à categoria Atividade e Procedimento.

A ferramenta irá analisar automaticamente alguns itens pré-definidos, que são passíveis de serem analisados pela mesma. Os itens restantes serão analisados pelo gerente de processo, a partir do *check-list* exibido pela ferramenta, atribuindo-lhes “Atende”, “Não Atende” ou “Atende Parcialmente”, de acordo com a sua conformidade ao processo (atividade “Analisar Itens do *Check-List*” - Figura B.3). Esse *check-list* é formado pelos itens cadastrados durante a manutenção dos itens.

A Tabela B.1 e a Tabela B.2 mostram os itens definidos para serem analisados respectivamente pela ferramenta *ProAnalyser* (antes da simulação), e pelo gerente de processo (depois da simulação, durante validação do processo). No contexto da simulação, alguns itens definidos não serão analisados: item 3 (Tabela B.1), e itens 3, 4, 5, 7 e 8 (Tabela B.2). Tais itens foram definidos no contexto da avaliação, após a execução real do processo em um ambiente de desenvolvimento de software, e foram substituídos por itens (Tabela 6.1) relacionados aos possíveis desvios encontrados na simulação (Seção 5.1).

Se não foram atendidos todos os objetivos definidos para o item, ele tem um problema associado e deve ser avaliado para que se descubra uma possível solução para o mesmo. À medida que os problemas forem sendo descobertos, eles serão armazenados em uma base de conhecimento.

Tabela B.1 – Relação de itens analisados pela ferramenta *ProAnalyser*.

Nº	Itens	Categoria
1	Relacionamento entre as atividades	Atividade
2	Conformidade do tipo de atividade	
3	Quantidade de artefatos planejados x produzidos	Artefato
4	Execução do modelo de ciclo de vida condizente com sua composição	Modelo de Ciclo de Vida
5	Conformidade entre tipo de ferramenta e tipo de atividade	Recursos
6	Conformidade da automação do tipo de ferramenta com tipo de método/técnica	
7	Conformidade do tipo de procedimento (método/técnica) com granularidade da atividade (macro/folha)	Procedimento
8	Conformidade do tipo de procedimento (método/técnica) com tipo de atividade (construção/gerência/qualidade)	

Tabela B.2 – Relação de itens analisados pelo usuário.

Nº	Itens	Categoria
1	Conformidade entre tipo de papel e tipo de atividade desenvolvida	Atividade
2	Correspondência entre a ordem das atividades planejadas x executadas	
3	Corretude no uso/manipulação dos artefatos	Artefato
4	Produção dos artefatos de acordo com o <i>template</i>	
5	Adequação da representação da estrutura do tipo de modelo de ciclo de vida no processo	Modelo de Ciclo de Vida
6	Adequação das habilidades dos membros em relação aos perfis definidos para as atividades do processo	Recursos
7	Adequação da ferramenta ao tipo de software definido ao processo	
8	Conformidade do procedimento definido em relação ao paradigma e à tecnologia de desenvolvimento utilizados	Procedimento

Assim, após a conclusão da análise dos itens, feita pela ferramenta e pelo gerente de processo, este poderá visualizar os respectivos resultados e conseqüências associados a esses itens, selecionando um item por vez para ser avaliado pelo sistema (atividade “Avaliar Itens” - Figura B.3). Caso o item seja considerado desconforme, serão exibidos os resultados, conseqüências e as possíveis soluções para o problema associado a esse item. Se o gerente de processo escolher uma solução sugerida pelo sistema (atividade “Selecionar Solução para Item” - Figura B.3), essa escolha será armazenada na base de conhecimento, formando uma base histórica de soluções de

problemas, para um determinado processo. A Tabela B.3 exibe exemplos de possíveis resultados, consequências e soluções de um dado item.

Tabela B.3 – Exemplo de resultados, consequências e soluções de um item.

Item 6 - Adequação das habilidades dos membros em relação aos perfis definidos para as atividades do processo	
Resultados	Consequências
(Atendido) Os membros estão executando corretamente os seus papéis	Composição da equipe bem planejada
(Parcialmente Atendido) Há membros que não possuem todas as habilidades necessárias para executar as atividades onde estão alocados	Alguns membros da equipe não foram bem alocados
(Não atendido) Os membros não possuem as habilidades necessárias para executar as atividades do processo	Alocação da equipe foi mal planejada
Soluções	
1) Analisar e redefinir composição da equipe em função de suas habilidades no contexto das atividades do processo	

Esse processo continuará acontecendo até que todos os itens sejam avaliados. Quando isso acontecer, a análise de itens de processo de software estará finalizada. Essa finalização deve ser notificada ao mecanismo de comunicação do ambiente, responsável por gerenciar a comunicação entre as ferramentas que formam o mesmo.

Mapeamento entre CMMI, MPS/BR e *ProAnalyser* no Ambiente ImPProS

Como mencionado na Seção 6.2, o ambiente ImPProS, através das funcionalidades do *ProAnalyser*, contempla os requisitos da área de processo do CMMI “*Causal Analysis and Resolution*”, que faz parte do nível de maturidade 5 do modelo. O ambiente também está alinhado aos requisitos da área de processo “Análise e Resolução de Causas”, que está no nível A do MPS/BR.

Desta maneira, a Tabela B.4 e a Tabela B.5 mostram o mapeamento entre as práticas de gerência de problemas, definidas respectivamente no CMMI e MPS/BR, e as funcionalidades da ferramenta *ProAnalyser*, no contexto do ambiente ImPProS.

Tabela B.4 – Gestão de problemas: mapeamento CMMI e *ProAnalyser*/ImPProS.

Soft Goal CMMI	Sub-Practice CMMI	ImPProS
SG 1 – <i>Determine Causes of Defects</i>	SP 1.1 <i>Select Defect Data for Analysis</i>	<i>ProSimulator</i> : Ativação da ferramenta <i>ProAnalyser</i> para fazer a análise de itens desconformes ao processo, anterior à simulação.
	SP 1.2 <i>Analyze Causes</i>	<i>ProAnalyser</i> : Detecção de um novo item ou problema, julgado pelo IMC antes de armazená-lo na base de dados.
SG 2 – <i>Address Causes of Defects</i>	SP 2.1 <i>Implement the Action Proposal</i>	<i>ProDefiner</i> ¹ : Verificação de sugestões fornecidas pelo <i>ProAnalyser</i> para problema encontrado, decidindo fazer ou não o refinamento do processo de acordo com as mesmas.
	SP 2.2 <i>Evaluate the Effect of Changes</i>	<i>ProSimulator/ProImprove</i> ² : Re-simulação ou Re-execução do processo após refinamento, verificando a persistência do problema e quais os impactos causados pelas mudanças no processo sugeridas pelo <i>ProAnalyser</i> .
	SP 2.3 <i>Record Data</i>	<i>ProKnowledge</i> ³ : Armazenamento sobre problemas detectados, soluções sugeridas e lições aprendidas sobre processos após a simulação.

Tabela B.5 – Gestão de problemas: mapeamento MPS/BR e *ProAnalyser/ImPProS*.

<i>MPS/BR</i>	<i>ImPProS</i>
ARC1. Dados de defeitos e de outros problemas são selecionados para análise	<i>ProSimulator</i> : ativação da ferramenta <i>ProAnalyser</i> para fazer a análise de itens desconformes ao processo.
ARC2. Análise da causa dos defeitos e de outros problemas selecionados é realizada com as pessoas responsáveis por realizar a tarefa para identificar sua raiz	<i>ProAnalyser</i> : Detecção de um novo item ou problema, julgado pelo IMC antes de armazená-lo na base de dados.
ARC3. Ações necessárias para evitar a ocorrência futura de defeitos e outros problemas similares aos selecionados são propostas e documentadas	<i>ProAnalyser</i> : Fornecimento de feedback para <i>ProDefiner</i> ¹ , para que sejam verificadas as sugestões fornecidas pelo <i>ProAnalyser</i> para problema encontrado.
ARC4. Ações propostas e documentadas que foram desenvolvidas durante a análise de causa e que possuem uma relação custo/benefício satisfatória são implementadas para remover as causas dos defeitos e problemas analisados e evitar recorrências	<i>ProDefiner</i> ¹ : Realização do refinamento do processo de acordo com as sugestões feitas pelo <i>ProAnalyser</i> , para simulá-lo ou executá-lo novamente. <i>ProAnalyser</i> : Verificação da similaridade das características de outros processos com o processo que está sendo analisado. Comitê Gerenciador de Itens: Análise do ranking dos processos parecidos e verificação da ocorrência dos mesmos erros, a partir dos problemas identificados pelo <i>ProAnalyser</i> .
ARC5. Defeitos e outros problemas similares que existem em outros processos ou produtos de trabalho são identificados e removidos	<i>ProAnalyser</i> : Fornecimento de feedback para <i>ProDefiner</i> ¹ , para que sejam verificadas as sugestões fornecidas pelo <i>ProAnalyser</i> para problema encontrado antes de serem utilizadas para refinar o processo.
ARC6. Propostas de melhorias para o conjunto de processos-padrão da organização são identificadas e documentadas	<i>ProSimulator/ProImprove</i> ² : Re-simulação ou Re-execução do processo após refinamento, verificando a persistência do problema e quais os impactos causados pelas mudanças no processo sugeridas pelo <i>ProAnalyser</i> .
ARC7. Os efeitos das mudanças no desempenho dos processos é medido e avaliado para obter evidência de que as mudanças nos processos corrigiram o problema e melhoraram o desempenho	<i>ProKnowledge</i> ³ : Armazenamento sobre problemas detectados, soluções sugeridas e lições aprendidas sobre processos durante a análise de relatórios de simulação e de relatório de métricas coletadas durante a execução do processo.
ARC8. Dados da análise e resolução de causas são armazenados para que outros projetos e organizações possam realizar mudanças apropriadas nos processos e alcançar resultados similares	

¹ *ProDefiner*: Ferramenta de Definição de Processo de Software.² *ProImprove*: Ferramenta de Melhoria de Processo de Software.³ *ProKnowledge*: Ferramenta de Gestão do Conhecimento sobre Processos de Software.

ProAnalyser: Tecnologias e Funcionalidades

O *ProAnalyser* é uma ferramenta baseada em conhecimento, desenvolvida utilizando a linguagem Java, no padrão J2SE (Java 2 *Standard Edition*) [Sun 2005] para aplicativos *desktop*, o banco de dados MySQL [MySQL 2005], e a máquina de inferência JEOPS (Java *Embedded Object Production System*) [Figueira e Ramalho 2000]. A máquina de inferência foi utilizada na análise automática de itens feita pelo Agente Monitor, e na avaliação dos itens feita pelo Agente *Solver* (Figura B.1).

A Figura B.4 ilustra a tela inicial da ferramenta *ProAnalyser*, que apresenta: informações sobre a organização e sobre o processo instanciado, ao qual se relaciona; o contexto pelo qual a ferramenta *ProAnalyser* foi ativada, que no caso é a ferramenta *ProSimulator*, e um conjunto de funcionalidades, incluindo a análise automática, análise manual, geração de relatórios e consultar soluções.

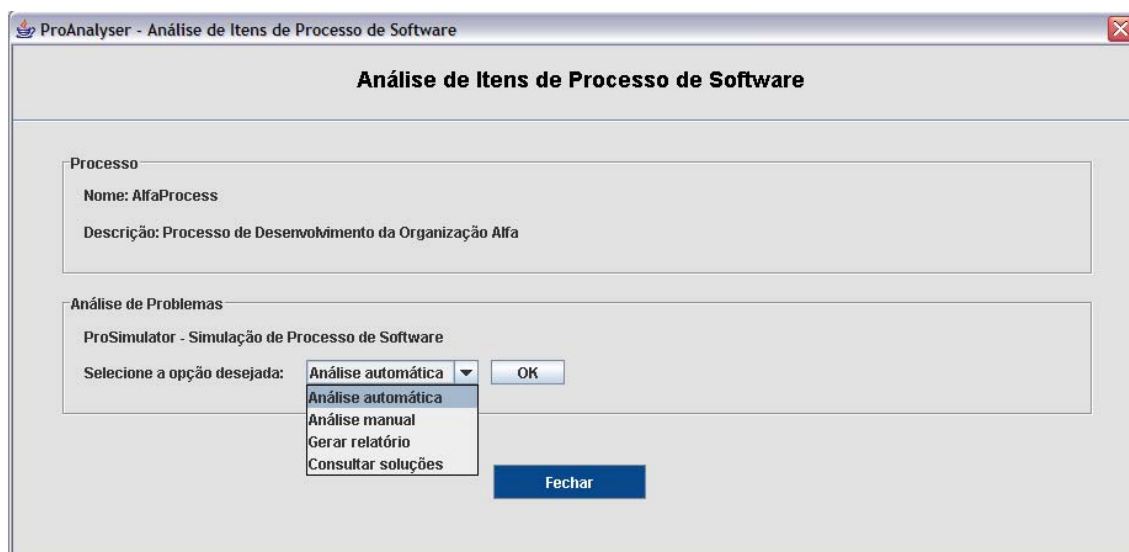


Figura B.4 – *ProAnalyser*: apresentação.

Inicialmente, o gerente de processo poderá realizar a análise automática ou manual de itens. Caso decida pela análise automática, basta que o gerente inicie a análise clicando no botão “Analisar Itens” (Figura B.5). A partir daí, é feita a análise automática pela ferramenta *ProAnalyser*, que irá preencher o *check-list*, de acordo com

o atendimento dos itens no modelo de processo instanciado (Figura B.6). Caso seja escolhida a análise manual, o próprio gerente preencherá o *check-list* de itens.

	Atendido	Parcialmente	Não Atendido
1 - Relacionamento entre as atividades	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
2 - Conformidade do tipo de atividade	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
3 - Quantidade de artefatos planejados x produzidos	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
4 - Execução do modelo de ciclo de vida condizente com sua composição	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
5 - Conformidade entre tipo de ferramenta e tipo de atividade	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
6 - Conformidade da automação do tipo de ferramenta com tipo de método/técnica	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
7 - Conformidade do tipo de procedimento com granularidade da atividade	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
8 - Conformidade do tipo de procedimento com tipo de atividade	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

Avaliar **Analisar Itens**

Figura B.5 – ProAnalyser: análise automática de itens.

Selecione um item para ser avaliado

	Atendido	Parcialmente	Não Atendido
<input type="radio"/> 1 - Relacionamento entre as atividades	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
<input type="radio"/> 2 - Conformidade do tipo de atividade	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
<input type="radio"/> 3 - Quantidade de artefatos planejados x produzidos	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<input type="radio"/> 4 - Execução do modelo de ciclo de vida condizente com sua composição	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
<input type="radio"/> 5 - Conformidade entre tipo de ferramenta e tipo de atividade	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
<input type="radio"/> 6 - Conformidade da automação do tipo de ferramenta com tipo de método/técnica	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
<input type="radio"/> 7 - Conformidade do tipo de procedimento com granularidade da atividade	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
<input type="radio"/> 8 - Conformidade do tipo de procedimento com tipo de atividade	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>

Avaliar **Analisar Itens**

Figura B.6 – ProAnalyser: resultado da análise automática de itens.

Em seguida, o gerente de processo poderá selecionar um item de cada vez para ser avaliado pela ferramenta, clicando em seguida no botão “Avaliar”. A Figura B.7 mostra o resultado da avaliação de um item possivelmente escolhido pelo gerente (item 1, considerado “Não Atendido”). Neste caso, existe a opção de se escolher a solução sugerida para resolver o problema detectado, clicando no botão “Escolher Solução”. Essa escolha será registrada e utilizada posteriormente, quando o usuário escolher a funcionalidade “Consultar Soluções”, no *menu* principal (Figura B.4), que mostra as soluções mais utilizadas para resolver problemas em um dado processo.

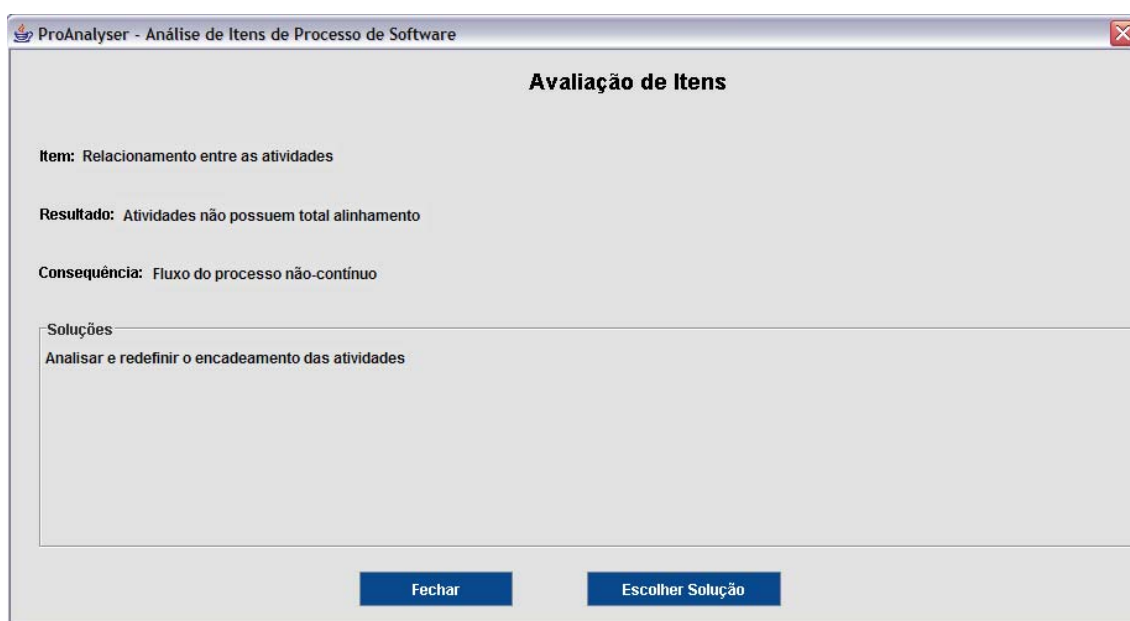


Figura B.7 – ProAnalyser: resultado da avaliação do item “relacionamento entre as atividades”.

No *menu* principal da Figura B.4, caso o gerente escolha a opção de gerar relatórios, será exibido uma caixa de diálogo, em que deverá ser definido o caminho e o nome do arquivo, segundo os quais se deseja que o relatório seja salvo. Esse relatório contém um resumo sobre toda a análise automática e manual dos itens, bem como a porcentagem de itens atendidos, parcialmente atendidos e não atendidos, para que o gerente possa ter uma noção sobre a qualidade do modelo de processo, de acordo com os itens analisados.