PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Programa de Pós-Graduação em Informática

Avaliação e Formalização de um Processo para Agrupamento de Solicitações de Manutenção de Software

Gladston Junio Aparecido

Belo Horizonte

2011

Gladston Junio Aparecido

Avaliação e Formalização de um Processo para Agrupamento de Solicitações de Manutenção de Software

Dissertação apresentada ao Programa de Pós-Graduação em Informática como requisito obrigatório para qualificação ao Grau de Mestre em Informática pela Pontifícia Universidade Católica de Minas Gerais.

Orientador: Marco Tulio Oliveira Valente

Co-Orientador: Humberto Torres Marques Neto

Belo Horizonte

2011

PUC Minas
Programa de Pós-graduação em Informática

FOLHA DE APROVAÇÃO

Avaliação e Formalização de um Processo para Agrupamento de Solicitações de Manutenção de Software

GLADSTON JÚNIO APARECIDO

Dissertação defendida e aprovada pela seguinte banca examinadora:

Prof. Marco Túlio de Oliveira Valente - Orientador (UFMG)
Doutor em Ciências da Computação - UFMG

Prof. Humberto Torres Marques Neto (PUC Minas)
Doutor em Ciências da Computação - UFMG

Profa. Ana Regina Cavalcanti da Rocha (UFRJ)
Doutora em Informática - (PUC Rio)

Prof. Gustavo da Gama Torres (PUC Minas)
Doutor em Ciências da Computação - UFMG

Prof. Mark Alan Junho Song (PUC Minas)
Doutor em Ciências da Computação - UFMG

Belo Horizonte, 24 de fevereiro de 2011.

Aos meus pais, pelos valores éticos e morais e pelos incentivos que contribuíram para minha formação e permitiram a realização deste trabalho.

À minha sempre presente namorada, pelo apoio, carinho e compreensão em todos os momentos.

Agradecimentos

Aos meus pais, *Jairo Miguel Aparecido* e *Rosângela Aparecida Fonseca*, por todo apoio e incentivo e à *minha família* por sempre estarem presentes nos momentos mais importantes.

À minha namorada, *Daniela Ferreira Guimarães*, pelo apoio, pela compreensão e pelos incentivos que foram fundamentais para que eu pudesse vencer os obstáculos desta trajetória.

Aos meus orientadores, *Marco Tulio de Oliveira Valente* e *Humberto Torres Marques Neto*, por aceitarem me guiar na realização desse trabalho, pelos sábios ensinamentos, pela paciência e pela confiança que foram essenciais para que pudesse atingir meus objetivos.

Ao *Humberto Mossri* e ao *Marcelo Nassau* pela receptividade e apoio na extração e no entendimento dos dados utilizados na realização deste trabalho.

Aos membros da banca avaliadora, *Ana Regina Cavalcanti da Rocha, Gustavo da Gama Torres e Mark Alan Song*, pelas contribuições e críticas importantes para a conclusão do meu trabalho.

Aos amigos da *DOCUMENTAR*, em especial a *Cláudia Kleinsorge*, *Fátima Garcia* e *Rosália Paraíso*, por todo apoio que foi dado.

E a todos os demais, e não menos importantes, que contribuíram de alguma forma para que eu pudesse atingir meus objetivos.

Resumo

Manutenção de software é caracterizada como uma das fases mais importantes, complexas e onerosas do ciclo de vida de um software, sendo responsável por cerca de 40% a 80% de todo o esforço aplicado no desenvolvimento e na manutenção de um sistema. Os processos de manutenção de software são responsáveis por nortear a execução de todas as demandas registradas nessa fase, o que abrange a operação, a manutenção, a evolução, a substituição e também o descarte seguro do software após o término de sua vida útil. Além do custo elevado e da diversidade de atividades contempladas pelos processos de manutenção, a dificuldade para gerenciar e acompanhar o atendimento das solicitações de manutenção contribuem para a busca contínua do aperfeiçoamento desses processos. Esta dissertação de mestrado formaliza o conjunto de práticas implantado no DATAPUC (Divisão de Tecnologia da Informação da PUC Minas), com o proposto de sistematizar o registro, a priorização e a execução de demandas de manutenção de software, bem como o agrupamento de pequenas solicitações em projetos de manutenção. O resultado dessa formalização foi denominado Processo para Agrupamento de Solicitações de Manutenção de Software (PASM), especificado nesta dissertação utilizando o meta-modelo *Software & Systems Process Engineering Metamodel* (SPEM). Esta dissertação também apresenta uma avaliação dos benefícios do uso desse conjunto de práticas. A avaliação realizada utiliza um algoritmo de *clusterização* e visa auxiliar a representação e a compreensão do fluxo de atendimento das solicitações de manutenção, assim como viabilizar a análise e a comparação dos padrões existentes no atendimento dessas solicitações. A avaliação do atendimento das solicitações foi realizada para evidenciar os benefícios proporcionados pela prática de agrupar solicitações de manutenção em projetos de manutenção. Para isso foram consideradas 3.686 solicitações entre projetos de manutenção e pequenas solicitações atendidas individualmente. Os resultados obtidos indicam que o número de projetos de manutenção criados para tratar solicitações de manutenção aumentou quase três vezes depois da adoção da prática de agrupar pequenas solicitações em projetos de manutenção de software. Além disso, os projetos realizados com base nas diretrizes adotadas apresentaram um melhor equilíbrio de uso das melhores práticas da Engenharia de Software. Por exemplo, observou-se que a equipe de manutenção dedicou mais tempo na execução de atividades de análise e validação das solicitações e menos tempo nas atividades de implementação.

Palavras-chave: engenharia de software, manutenção de software, processos de manutenção de software, avaliação de processos de software, caracterização de solicitações de manutenção.

Abstract

Software maintenance is characterized as one of the most important, complex and costly phases of a software life cycle and accounting for about 40% to 80% of the entire effort applied in the development and maintenance of a software system. The software maintenance processes are responsible for guiding the handling of all demands registred in that phase, which covers operation, maintenance, evolution, replacement and also the safe disposal of the software after the end of its useful life. Besides the high costs and the diversity of activities covered by the maintenance processes, the difficulty to manage and monitor the servicing of maintenance requests contribute to the continuous search for improvements of these processes. This dissertation formalizes the practices deployed in DATAPUC (Information Technology Division of PUC Minas), with the proposal of organizing the registration, prioritization and implementation of software maintenance demands and the grouping of small requests in maintenance projects. The result of this formalization was called Process for Arranging Software Maintenance Requests (PASM), specified in this dissertation using the *Software & Systems Process Engineering Metamodel* (SPEM). This dissertation also presents an evaluation of the benefits of using this set of practices. The evaluation uses a clustering algorithm and aims to assist the representation and understanding of the service flow of maintenance requests, as well as facilitate the analysis and comparison of existing patterns in the handling of these requests. The evaluation of servicing the requests was used to highlight the benefits provided by the practice of grouping maintenance requests in maintenance projects. For that, 3,686 requests were considered among maintenance projects and small requests handled individually. The results indicate that the number of projects created to handle maintenance requests has increased almost three times after the adoption of the practice of grouping small maintenance requests in maintenance projects. Moreover, the projects based on the set of adopted guidelines present a better balance of use of best practices of Software Engineering. For example, we observed that the maintenance team has dedicated more time to perform activities of analysis and validation of requests and less time on implementation activities.

Keywords: software engineering, software maintenance, software process, software process evaluation, maintenance requests characterization.

Lista de Figuras

FIGURA 1 Processo de manutenção de software da norma ISO/IEC 14764 25

FIGURA 2 Calendário Mensal de Atendimento de Solicitações de Manutenção do DATAPUC 38

FIGURA 3 SMRMG dos projetos de manutenção 60

FIGURA 4 SMRMG dos *clusters* dos projetos de manutenção de 2008 64

FIGURA 5 SMRMG dos *clusters* dos projetos de manutenção de 2009 64

FIGURA 6 SMRMG das solicitações urgentes 66

FIGURA 7 SMRMG do *Cluster 1* das solicitações urgentes de 2008 69

FIGURA 8 SMRMG do *Cluster 2* das solicitações urgentes de 2008 70

FIGURA 9 SMRMG do *Cluster 3* das solicitações urgentes de 2008 71

FIGURA 10 SMRMG do *Cluster 4* das solicitações urgentes de 2008 72

FIGURA 11 SMRMG do *Cluster 5* das solicitações urgentes de 2008 73

FIGURA 12 SMRMG do *Cluster 1* das solicitações urgentes de 2009 74

FIGURA 13 SMRMG do *Cluster 2* das solicitações urgentes de 2009 75

FIGURA 14 SMRMG do *Cluster 3* das solicitações urgentes de 2009 76

FIGURA 15 SMRMG do *Cluster 4* das solicitações urgentes de 2009 77

FIGURA 16 SMRMG do *Cluster 5* das solicitações urgentes de 2009 78

FIGURA 17 Fases do ciclo de projetos de manutenção 86

FIGURA 18 Fluxo de trabalho da fase Registro 88

FIGURA 19 Fluxo de trabalho da fase Agrupamento 90

FIGURA 20 Fluxo de trabalho da fase Execução 93

Lista de Tabelas

TABELA 1 Principais sistemas mantidos pelo DATAPUC 51

TABELA 2 Número de solicitações da amostra de dados 54

TABELA 3 Número de horas trabalhadas 55

TABELA 4 *Clusters* dos projetos de manutenção 62

TABELA 5 *Clusters* das solicitações de manutenção urgentes de 2008 68

TABELA 6 *Clusters* das solicitações de manutenção urgentes de 2009 74

TABELA 7 Legenda de estereótipos do SPEM 88

Lista de Siglas

CM^3	Corrective Maintenance Maturity Model
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
COBIT	Control Objectives for Information and related Technology
DATAPUC	Divisão de Tecnologia da Informação da PUC Minas
DEA	Data Envelopment Analysis
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IFPUG	International Function Point Users Group
ISACA	Information Systems Audit and Control Association
ISO	International Organization for Standardization
ITIL	Information Technology Infrastructure Library
MR MPS	Modelo de Referência para Melhoria de Processos do Software
OGC	Office of Government Commerce
OMG	Object Management Group
PASM	Processo para Agrupamento de Solicitações de Manutenção de Software
PDDS	Processo DATAPUC de Desenvolvimento de Software
S3M	Software Maintenance Maturity Model
SEI	Software Engineering Institute
SPEM	Software & Systems Process Engineering Metamodel
SMRMG	Software Maintenance Request Model Graph

Sumário

1 INTRODUÇÃO **14**

1.1 Motivação 14

1.2 Objetivos 17

1.3 Estrutura da Dissertação 18

2 CONTEXTUALIZAÇÃO **19**

2.1 Engenharia de Software 19

2.2 Manutenção de Software 20

2.3 Processos de Manutenção de Software 23

2.3.1 ISO/IEC 14764 24

2.3.2 MANTEMA 26

2.4 Melhorias de Processo de Software 27

2.4.1 Iniciativas de Melhorias de Processos de Software 27

2.4.2 Software Maintenance Maturity Model (S3M) 29

2.4.3 Corrective Maintenance Maturity Model (CM^3) 30

2.4.4 TI Service Capability Maturity Model 31

2.4.5 Comentários sobre Modelos de Maturidade para Manutenção de Software 32

2.5 Planejamento e Gerenciamento da Manutenção de Software 33

2.5.1 Agrupamento de Solicitações de Manutenção de Software 33

2.5.2 Sistematização do Agrupamento de Solicitações de Manutenção de Software 37
2.5.3 Estimativa de Custo da Manutenção de Software 40
2.6 Considerações Finais 46

3 ESTUDO DE CASO **47**
3.1 Introdução 47
3.2 Planejamento do Estudo de Caso 48
3.2.1 Contexto 48
3.2.2 Objetivos 48
3.2.3 Questões de Pesquisa 49
3.2.4 Escopo 49
3.2.5 DATAPUC 50
3.2.6 Cenários Avaliados 51
3.2.7 Amostra de Dados 54
3.2.8 Metodologia de Caracterização 55
3.3 Resultados 59
3.3.1 Projetos de Manutenção 59
3.3.2 Solicitações Urgentes 66
3.3.3 PU e Plantões 81
3.4 Considerações Finais 82

4 PROCESSO PARA AGRUPAMENTO DE SOLICITAÇÕES DE MANUTENÇÃO DE SOFTWARE (PASM) **84**
4.1 Definições 84
4.2 Ciclo de Projetos de Manutenção 85
4.3 Fases do PASM 87
4.3.1 Registro 88

4.3.2 Agrupamento 90
4.3.3 Execução 92
4.4 Considerações Finais 94

5 CONCLUSÕES **96**
5.1 Visão Geral 96
5.2 Contribuições 97
5.3 Trabalhos Futuros 98

REFERÊNCIAS **99**

Capítulo 1

Introdução

1.1 Motivação

A manutenção de software é uma das fases mais importantes, complexas e onerosas do ciclo de vida de um software segundo Mookerjee (2005), Webster, Oliveira e Anquetil (2005) e Antonellis *et al.* (2009). Os processos de software executados ao longo dessa fase são responsáveis por assegurar a operacionalidade de um sistema – que muitas vezes utiliza tecnologias obsoletas – e por garantir sua evolução frente às mudanças do contexto do negócio. Alinhado às responsabilidades, o custo da manutenção também contribui com a sua importância no cenário da Engenharia de Software. Um estudo realizado por Chan, Chung e Ho (1996) aponta que na década de 80 as organizações de todo o mundo gastaram anualmente US 30 bilhões com manutenção de software. Para a década de 90, Goodwin citado por Taylor e Wood-Harper (1996) estimou que a totalidade anual aplicada pelas companhias da Europa Ocidental foi de €26,5 bilhões. Em relação ao mesmo período, Sutherland citado por Tan e Mookerjee (2005), apontou que a soma gasta pelas empresas dos Estados Unidos foi de aproximadamente US 70 bilhões por ano. Avaliações mais recentes como as realizadas por Ware, Wilkie e Shapcott (2007), Sneed (2008) e Aziz, Ahmed e Laghari (2009) indicam que a manutenção de software consome entre 40% a 80% do esforço total despendido durante todo o ciclo de vida de um sistema.

Analisando os estudos recentes descritos na literatura, observa-se que não existem indicações de que os custos da manutenção de software tenham diminuído. Além do fato de a cada dia novos sistemas serem desenvolvidos, e consequentemente mantidos, outros fatores contribuem para o aumento gradativo do custo da fase de manutenção. Por exemplo, o aumento da longevidade do software está diretamente relacionado com o

custo total de sua manutenção. Mookerjee (2005) destaca que nos dias atuais raramente um sistema é concebido para ser utilizado isoladamente. Isso implica no aumento do tempo de vida útil dos sistemas e também na elevação do custo total de manutenção. Também é importante ressaltar que, na medida em que o software se torna mais crítico para as organizações, os procedimentos de governança de tecnologia da informação tendem a elevar o custo total de execução das atividades de manutenção, como observa Chapin (2009).

Entretanto, apesar de sua inegável importância, Mookerjee (2005), Tan e Mookerjee (2005), Webster, Oliveira e Anquetil (2005) e Oliveira, Anquetil e Alonso (2009) destacam que a manutenção de software tem uma imagem negativa entre os profissionais e as organizações. Segundo esses autores, manutenção é considerada uma atividade de pouco prestígio, destinada a pessoas pouco experientes ou pouco criativas e é, na maioria das vezes, classificada apenas como a correção de falhas. Além disso, conforme salientam Kajko-Mattsson, Forssander e Olsson (2001), Mens *et al.* (2005), Webster, Oliveira e Anquetil (2005) e Sneed e Opferkuch (2008) poucas pesquisas são desenvolvidas para investigar os processos de manutenção e suas particularidades.

O desenvolvimento e a evolução de processos constitui uma linha de pesquisa importante para a manutenção de software. Segundo Pressman (2009), os processos de software representam um dos principais pilares da Engenharia de Software. Aos processos de manutenção em especial atribuem-se a responsabilidade de definir o conjunto de atividades necessárias para atender a todos os tipos de solicitações de manutenção. Dessa forma, os processos executados durante essa fase estão estritamente relacionados com o sucesso da manutenção de um sistema e colaboram para satisfazer as particularidades e os principais desafios associados à manutenção de software. Niessink e Vliet (2000), Bhatt, Shroff e Misra (2004), Kung (2004), Mens *et al.* (2005) e Sneed (2008) classificam esses desafios como: a dificuldade de gerenciar a demanda simultânea de serviços, a necessidade de garantir a operacionalidade do sistema, os obstáculos para compor as equipes de manutenção internas e terceirizadas e a necessidade de assegurar a satisfação dos usuários.

Para apoiar a busca pelo aperfeiçoamento dos processos de software e pela melhoria da qualidade dos sistemas, a academia tem proposto e a indústria de software tem adotado diversas iniciativas de melhorias de processos, representadas principalmente por programas e modelos de maturidade. Segundo Sommerville (2007) e Pressman (2009) o objetivo central dessas iniciativas consiste na evolução dos processos de software por meio da análise, da medição e do aperfeiçoamento das práticas de Engenharia de Software. Es-

sas iniciativas também visam determinar a eficácia das práticas executadas e a maturidade de uma organização na execução de um conjunto de processos. Dentre essas iniciativas se destacam os modelos de maturidade CMM (*Capability Maturity Model*)/CMMI (*Capability Maturity Model Integration*) (SEI, 1995) (CHRISSIS; KONRAD; SHRUM, 2006) e o MR MPS (Modelo de Referência para Melhoria de Processos do Software) (SOFTEX, 2009). Segundo Ferreiro *et al.* (2007), Pressman (2009) e Kalinowski *et al.* (2010), tais iniciativas comumentemente focam em metas para reduzir custos, aumentar a qualidade dos produtos e a produtividade das equipes, antecipar a liberação de novas versões e aumentar a satisfação dos usuários.

Embora não sejam modelos destinados exclusivamente à manutenção de software, o CMM/CMMI e o MR MPS contribuem para o aperfeiçoamento dos processos de manutenção. Entretanto, Hall *et al.* (2001), April, Abran e Dumke (2004), April *et al.* (2005) e Jung e Goldenson (2009) ressaltam que esses modelos não tratam adequadamente o conceito de maturidade para a manutenção de software. Durante a fase de manutenção, os processos passam a ser governados pelas solicitações de manutenção. Dessa forma, conforme destacam Mens *et al.* (2005), os usuários e as equipes precisam utilizar mecanismos diferentes para elucidar, planejar, executar e validar as solicitações de manutenção. Além disso, Banker e Slaughter (1997), Niessink e Vliet (2000), Lehman e Ramil (2001), Kung (2004) e Tan e Mookerjee (2005) salientam que a manutenção é caracterizada principalmente pela heterogeneidade e pela simultaneidade na produção e no consumo de serviços, onde a taxa de registro das solicitações apresenta variações ao longo do ciclo de vida do software. Como a qualidade do atendimento está estritamente relacionada com o nível de satisfação dos usuários é importante desenvolver e aperfeiçoar os processos de manutenção para que o atendimento das solicitações de manutenção possa ser realizado dentro das restrições de custo, prazo e qualidade.

Dado o volume e a dinâmica do registro das solicitações de manutenção, é recorrente o enfileiramento dessas solicitações após o registro para que a execução seja realizada após a priorização das demandas (STARK, 1996) (BANKER; SLAUGHTER, 1997) (STARK; OMAN, 1997) (CHAN, 2000) (JI, 2001) (LUCCA; RAGO, 2001) (ANTONIOL *et al.*, 2004). A priorização tende a ser atribuída periodicamente, de acordo com, por exemplo: os contratos estabelecidos para a manutenção do sistema, a prioridade das demandas e as necessidades dos usuários. Em alguns casos, as solicitações enfileiradas são atendidas individualmente. Nesse contexto, os estudos apresentados por Stark (1996), Stark e Oman (1997), Ramaswamy (2000), Ji (2001), Lucca e Rago (2001), Antoniol *et al.* (2004) e Asundi e Sarkar (2005) utilizam conceitos da teoria de filas para tratar o atendimento das solicitações.

Em contrapartida, outros estudos como os realizados por Banker e Slaughter (1997) e Tan e Mookerjee (2005) apontam benefícios proporcionados pelo agrupamento de solicitações em projetos de manutenção. De acordo com os resultados obtidos por Banker e Slaughter (1997) essa prática pode proporcionar uma redução de até 36% do custo de atendimento das solicitações. O trabalho desenvolvido por Malta *et al.* (2009) propõe um conjunto de práticas para sistematizar o registro, a priorização e a execução de solicitações de manutenção, além de estimular o agrupamento de solicitações em projetos de manutenção. Este trabalho também apresenta uma breve avaliação da implantação desse conjunto de práticas. Entretanto, os estudos aqui citados se valem de modelos analíticos ou de observações empíricas preliminares.

Observando as lacunas existentes no conhecimento e na prática dos temas aqui abordados, esta dissertação realizada uma caracterização do atendimento de solicitações de manutenção e investiga os benefícios proporcionados pela prática de agrupar pequenas solicitações em projetos de manutenção de software, elaborados, priorizados e executados sistematicamente.

1.2 Objetivos

Esta dissertação de mestrado tem como objetivo principal *avaliar os benefícios proporcionados pelo conjunto de práticas proposto por Malta et al. (2009) para agrupar solicitações de manutenção em projetos de manutenção de software e formalizar esse conjunto de práticas em um processo denominado Processo para Agrupamento de Solicitações de Manutenção de Software(PASM).* Sendo assim, o objetivo geral desta dissertação decompõe-se nos seguintes objetivos específicos:

- Discutir a importância da manutenção de software e as particularidades dessa área, apontando os principais desafios para a sua evolução.
- Apresentar uma análise das principais propostas de processos de software que endereçam as particularidades e os desafios elucidados e revisar os principais estudos relacionados com a prática de agrupar solicitações de manutenção em projetos de manutenção.
- Avaliar – em um ambiente real – os benefícios advindos do uso do conjunto de práticas proposto por Malta *et al.* (2009) para agrupar solicitações de manutenção em projetos de manutenção utilizando um algoritmo de *clusterização* (BOZDOGAN,

2003) (GOLDSCHMIDT; PASSOS, 2005) (HAN; MICHELINE, 2005) – mais especificamente o algoritmo *k-means* – visando: representar e compreender os padrões existentes no fluxo de atendimento das solicitações e analisar e comparar os padrões identificados, para assim evidenciar as vantagens e eventuais desvantagens da prática de agrupar solicitações de manutenção em projetos de manutenção.

- Formalizar o conjunto de práticas proposto por Malta *et al.* (2009) em um processo denominado Processo para Agrupamento de Solicitações de Manutenção de Software (PASM), de forma a propor um conjunto de recomendações abrangente e fundamentado nas principais referências sobre manutenção de software.

1.3 Estrutura da Dissertação

Além desta Introdução, o presente trabalho é composto por cinco capítulos, organizados conforme descrito a seguir:

- **Capítulo 2 – Contextualização:** realiza uma revisão da literatura diretamente relacionada ao desenvolvimento desta dissertação, incluindo a descrição dos principais conceitos relacionados à manutenção de software, aos processos de manutenção, às iniciativas de melhoria de processos de software, ao agrupamento de solicitações de manutenção e à estimativa de custo para a fase de manutenção de um software.
- **Capítulo 3 – Estudo de Caso:** apresenta uma avaliação da implantação do conjunto de práticas proposto por Malta *et al.* (2009) na Divisão de Tecnologia da Informação da PUC Minas (DATAPUC). Essa avaliação foi desenvolvida utilizando técnicas e algoritmos de *clusterização* de dados (HAN; MICHELINE, 2005).
- **Capítulo 4 – Processo para Agrupamento de Solicitações de Manutenção de Software – (PASM):** apresenta a formalização do conjunto de práticas proposto por Malta *et al.* (2009) no PASM. Esse capítulo também apresenta uma especificação para o PASM, elaborada com base na inspeção das melhores práticas sugeridas pela literatura relacionada à manutenção de software discutidas nesta dissertação. A especificação proposta é representada através do meta-modelo *Software & Systems Process Engineering Metamodel* (SPEM) (OMG, 2002).
- **Capítulo 5 – Conclusão:** apresenta as considerações finais desta dissertação, destacando suas contribuições. Para finalizar, este capítulo apresenta algumas propostas para possíveis trabalhos futuros.

Capítulo 2

Contextualização

2.1 Engenharia de Software

Engenharia de Software é definida como "a aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção de software [...]" (IEEE, 1990). Sommerville (2007) corrobora está definição e constata que os princípios da Engenharia de Software envolvem todos os aspectos da produção de sistemas, o que abrange a concepção, o desenvolvimento, a manutenção e o descarte seguro ao final de sua vida útil. A observância desses princípios é regida pela especificação de processos de software, definidos por Sommerville (2007) como um conjunto de atividades e resultados conexos responsáveis por especificar, produzir, validar e evoluir um sistema. Pressman (2009) ratifica a importância dos processos de software e destaca que a especificação desses é responsável por definir quais atividades devem ser executadas e quem deve realizá-las, porque as atividades a serem desempenhadas são importantes, quais etapas devem ser executadas, qual o resultado esperado e como determinar que o resultado obtido está correto. Dessa maneira, conforme argumenta Pressman (2009), a concepção, a seleção e a adaptação dos processos em conjunto com os métodos, as ferramentas e o foco na qualidade viabilizam a aplicação disciplinada das práticas de Engenharia de Software, além de contribuir para o sucesso dos projetos de desenvolvimento e manutenção de um software.

Dada a ampla variedade de conhecimentos requeridos para a prática da Engenharia de Software, suas competências são geralmente classificadas e divididas em áreas, facilitando seu entendimento, emprego e evolução. O *Guide to the Software Engineering Body of Knowledge* (SWEBOK) (IEEE, 2004) organiza os tópicos relacionados à Engenharia de Software em grupos, conhecidos como áreas de conhecimento. Ao todo o SWEBOK

descreve dez áreas de conhecimento, sendo estas: Requisitos de Software, *Design* de Software, Construção de Software, Teste de Software, Gerência de Configuração de Software, Gerência da Engenharia de Software, Processos de Engenharia de Software, Ferramentas e Métodos da Engenharia de Software, Qualidade de Software e Manutenção de Software. A manutenção de software, no qual se insere esta pesquisa de mestrado, é definida pelo Glossário de Termos de Engenharia de Software como "[...] o processo de modificação de um sistema de software ou componente após a entrega para corrigir falhas, melhorar o desempenho ou outros atributos, ou adaptá-lo a um ambiente modificado. [...]" (IEEE, 1990). Algumas das principais propostas de processos para a manutenção de software disponíveis na literatura serão discutidas nesta dissertação, além de algumas das características que impactam o planejamento, o gerenciamento e a execução da manutenção de um software.

2.2 Manutenção de Software

A definição de manutenção de software discutida na Seção 2.1 retrata a extensão do domínio dessa área, cujos processos precisam sustentar a operacionalidade e a evolução de um sistema durante todo o período após a fase de desenvolvimento inicial. Estudos descritos na literatura revelam estatísticas contundentes sobre a importância da manutenção de software. Chan, Chung e Ho (1996) relataram que nos anos 80 o total gasto com a manutenção em todo o mundo era de aproximadamente US 30 bilhões. Goodwin citado por Taylor e Wood-Harper (1996), apontou que em 1990 foram gastos aproximadamente €26,5 bilhões com a manutenção de sistemas pelas companhias da Europa Ocidental. Sutherland citado por Tan e Mookerjee (2005), estimou que somente nos Estados Unidos, as organizações gastaram aproximadamente US 70 bilhões por ano com manutenção de software. Banker e Slaughter (1997) destacaram o relatório da Forrester Research o qual indica que 75% do orçamento das empresas de sistemas de informação presentes no índice *Fortune 1000* é dedicado apenas a melhoria e a manutenção de sistemas. Em relação ao custo total do ciclo de vida de software, Ware, Wilkie e Shapcott (2007), Sneed (2008) e Aziz, Ahmed e Laghari (2009) afirmam que o custo de manutenção absorve entre 40% e 80% dos gastos das empresas. Bhatt, Shroff e Misra (2004) ratificam esta afirmação através da estatística de que apenas 25% a 33% do esforço despendido durante o ciclo de vida de um software corresponde às atividades de construção.

Como argumentado anteriormente, manutenção de software consome uma parcela significativa do custo total do ciclo de vida de um software. No entanto, além dos sistemas recém implantados, as organizações mantêm sistemas criados há anos, denominados sistemas legados. Sommerville (2007) define sistemas legados como sistemas baseados em computadores e desenvolvidos no passado, que envolvem hardware, software e a automação de processos e procedimentos legados das organizações. Segundo Sommerville (2007), como geralmente esses sistemas são críticos para o negócio e os riscos de substituição representam possibilidades de perdas elevadas, eles continuam sendo mantidos e evoluídos. Esses sistemas introduzem diversos desafios para a manutenção de software. Por exemplo, o ambiente de execução, o paradigma de desenvolvimento utilizado e as linguagens de programação empregadas podem estar obsoletos. Além disso, a composição das equipes de manutenção pode ser comprometida devido a escassez de profissionais aptos a manter a operacionalidade do software e a garantir sua evolução. Outros problemas como os relacionados com a ausência ou a ineficiência da documentação ou o alto índice de degradação da estrutura do software podem elevar o custo e reduzir a qualidade da manutenção. Sommerville (2007) ressalta que o sucesso das operações de diversas organizações está diretamente vinculado com o funcionamento adequado dos sistemas. Portanto, faz-se necessário garantir a disponibilidade dos sistemas e a eficiência do atendimento das solicitações de manutenção. Segundo Niessink e Vliet (2000), esses são os principais critérios adotados pelos usuários ao avaliar a qualidade de um processo de manutenção.

Os esforços consumidos na fase de manutenção originam-se basicamente das solicitações de manutenção. Além disso, conforme postulado pelas leis da evolução de software propostas por Lehman e Belady (1985) e Lehman e Ramil (2001), os sistemas em produção necessitam de mudanças, caso contrário eles serão gradativamente direcionados para o descarte. Esses fatos contribuem para a necessidade de entender os tipos de solicitações de manutenção, suas causas e as características requeridas para o atendimento eficaz das solicitações de manutenção. O padrão IEEE 1219 (IEEE, 1999) classifica as solicitações de manutenção em quatro tipos, descritos a seguir:

- **Manutenções Adaptativas**: representam as manutenções executadas com o objetivo de adaptar o software às mudanças do ambiente, como por exemplo, alterações nas configurações de hardware e de sistema operacional. Segundo Sommerville (2007), esse tipo de solicitação representa em média 18% do esforço total da manutenção de software.

- **Manutenções Corretivas**: contemplam as mudanças acarretadas por falhas no

processamento das funcionalidades ou devido a ausência de atributos especificados, como por exemplo, desempenho ou segurança e que comprometem o uso do sistema. Sommerville (2007) destaca que esse tipo de solicitação representa em média 17% do esforço demandado durante a fase de manutenção.

- **Manutenções Perfectivas**: envolvem as manutenções responsáveis por inserir ou modificar os requisitos funcionais e não funcionais do software, assim como as manutenções executadas para melhorar a manutenibilidade do sistema, reduzindo o esforço necessário para realização das manutenções futuras. Nessa categoria incluem-se as alterações que almejam melhorar a compreensão sobre o software, reduzir a complexidade do código ou eliminar a degradação inserida por outras atividades de manutenção. De acordo com Sommerville (2007), as solicitações perfectivas representam em média 65% do esforço total da fase de manutenção.

- **Manutenções Preventivas**: abrangem as correções diagnosticadas e concluídas antes que se tornem um problema real. Esse tipo de manutenção é parte de uma política de gerenciamento pró-ativo da fase de manutenção. Entretanto, Chapin (2000) e Vehvilainen (2000) questionam a inclusão das manutenções preventivas na classificação de tipos de solicitações de manutenção, embora destaquem a importância dessas solicitações. Sommerville (2007) não apresenta uma estatística sobre o percentual do esforço demandado pela execução das solicitações preventivas.

Esta dissertação adota a classificação de solicitações de manutenção sugerida pelo padrão IEEE 1219 (IEEE, 1999). Entretanto, é importante ressaltar que, conforme destaca Sommerville (2007), algumas referências descritas na literatura apresentam classificações divergentes. A principal diferença encontrada envolve a inversão do significado das solicitações adaptativas e perfectivas. Além disso, embora as classificações encontradas nas principais normas e pesquisas sobre Engenharia e Manutenção de Software sejam semelhantes, existem definições alternativas citadas na literatura e utilizadas pelas organizações. Por exemplo, Kung (2004) adotou uma classificação composta apenas por solicitações corretivas e melhorias. A norma ISO/IEC 14764 (ISO/IEC, 2006) e a especificação da metodologia MANTEMA (POLO *et al.*, 2001) dividem as solicitações corretivas em solicitações corretivas urgentes e solicitações corretivas não urgentes. A separação das solicitações corretivas por prioridade possibilita determinar ações diferenciadas para solucionar os incidentes reportados pelos usuários sem comprometer o orçamento da fase de manutenção e a satisfação dos usuários.

2.3 Processos de Manutenção de Software

O modelo de processo de manutenção proposto pela norma ISO/IEC 12207 (ISO/IEC, 2008) delimita o escopo da fase de manutenção como o período após a primeira liberação funcional do software aprovada pelos usuários. Entretanto, a transição entre a fase de desenvolvimento inicial e a fase de manutenção impõe mudanças substanciais nos processos de software, principalmente no que tange a perspectiva da execução desses processos. Conforme mencionam Webster, Oliveira e Anquetil (2005), durante a fase de desenvolvimento inicial os processos são governados pelos requisitos elucidados e na fase de manutenção os processos são dirigidos por eventos, representados pelas solicitações de manutenção registradas pelos usuários. Consequentemente, a equipe de manutenção e os usuários passam a usufruir de outros mecanismos para elucidar, negociar, planejar, executar e validar as solicitações de manutenção. Mens *et al.* (2005) apontam o gerenciamento e a evolução desse conjunto de mecanismos como um dos principais desafios para o desenvolvimento da manutenção de software. Kajko-Mattsson, Forssander e Olsson (2001), Mens *et al.* (2005), Webster, Oliveira e Anquetil (2005) e Sneed e Opferkuch (2008) fortalecem esse argumento, salientando a distribuição desproporcional entre os investimentos na pesquisa de metodologias para o desenvolvimento inicial e para a manutenção. Essa divergência é refletida na prática, onde segundo as observações apresentadas por Taylor e Wood-Harper (1996) e Polo *et al.* (1999a), uma parcela significativa das organizações não aplicam um processo de software com consistência durante a fase de manutenção.

A fase de manutenção possui outras características que representam riscos aos projetos e influenciam diretamente a especificação dos processos executados nessa fase do ciclo de vida de software. Conforme relatam Basili *et al.* (1996), durante a fase de manutenção os colaboradores gastam mais tempo isolando e analisando as solicitações do que efetivamente implementando as modificações. Além disso, os processos de manutenção são diversas vezes influenciados pelo planejamento estratégico das organizações e por suas estratégias de marketing. Nesse cenário, Rajlich e Bennett (2000) destacam a política de manutenção adotada por grandes organizações como a Microsoft, que começam a desenvolver uma nova versão de seus produtos quando esses encontram-se no momento de maior sucesso de vendas, evitando que o plano de evolução do produto seja concluído durante o declínio das vendas. Os produtos de software oferecidos a uma grande quantidade de clientes também tendem a ter ciclos de manutenção específicos. Rajlich e Bennett (2000) salientam que nesse contexto, a manutenção rotineira dos sistemas é muitas vezes inexequível, devido à ampla base de clientes.

A norma ISO/IEC 12207 (ISO/IEC, 2008) define um conjunto de processos que abrange todo o ciclo de vida de um sistema. Entretanto, como salientam Polo *et al.* (1999b), sua especificação representa apenas um modelo de referência abstrato, sem mencionar como implementar esses processos. Essa característica contribuiu para o desenvolvimento de diversas normas internacionais e estudos que almejam propor processos de software que complementam a norma ISO/IEC 12207. As subseções a seguir descrevem algumas propostas de processos que estendem ou customizam especificamente o processo de manutenção dessa norma.

2.3.1 ISO/IEC 14764

A norma ISO/IEC 14764 (ISO/IEC, 2006) descreve um *framework* baseado no processo de manutenção da versão de 1995 da norma ISO/IEC 12207 (ISO/IEC, 2008), cujo objetivo é apoiar o planejamento, o gerenciamento e a execução das atividades de manutenção, independentemente do tamanho e da complexidade do software, da criticidade das modificações e do domínio do negócio. As diretrizes apresentadas por essa norma englobam desde o planejamento da fase de manutenção e a definição da estratégia a ser adotada até a retirada do software do ambiente de produção ao final de sua vida útil. A especificação dessa norma estende a norma ISO/IEC 12207 descrevendo detalhadamente as atividades do processo de manutenção, seus objetivos, as entradas e saídas esperadas, as tarefas a serem realizadas em cada atividade e os principais envolvidos. Além disso, a norma ISO/IEC 14764 (ISO/IEC, 2006) propõe orientações para que as metas estabelecidas sejam atingidas.

A Figura 1 ilustra o processo de manutenção proposto pela norma ISO/IEC 14764. Esse processo é dividido em duas fases. A primeira é responsável pelas atividades executadas durante o desenvolvimento inicial e visam, por exemplo, auxiliar a transferência do conhecimento entre a equipe de desenvolvimento e a equipe de manutenção. Essas atividades também viabilizam o planejamento da execução do processo de manutenção após a entrega do software, mitigando antecipadamente os riscos de insucesso. A segunda fase está relacionada ao período após o desenvolvimento inicial e almeja garantir a operacionalidade e a evolução do software para que esse acompanhe as necessidades dos usuários. A seguir resumem-se os objetivos das atividades do processo proposto pela norma ISO/IEC 14764 (ISO/IEC, 2006):

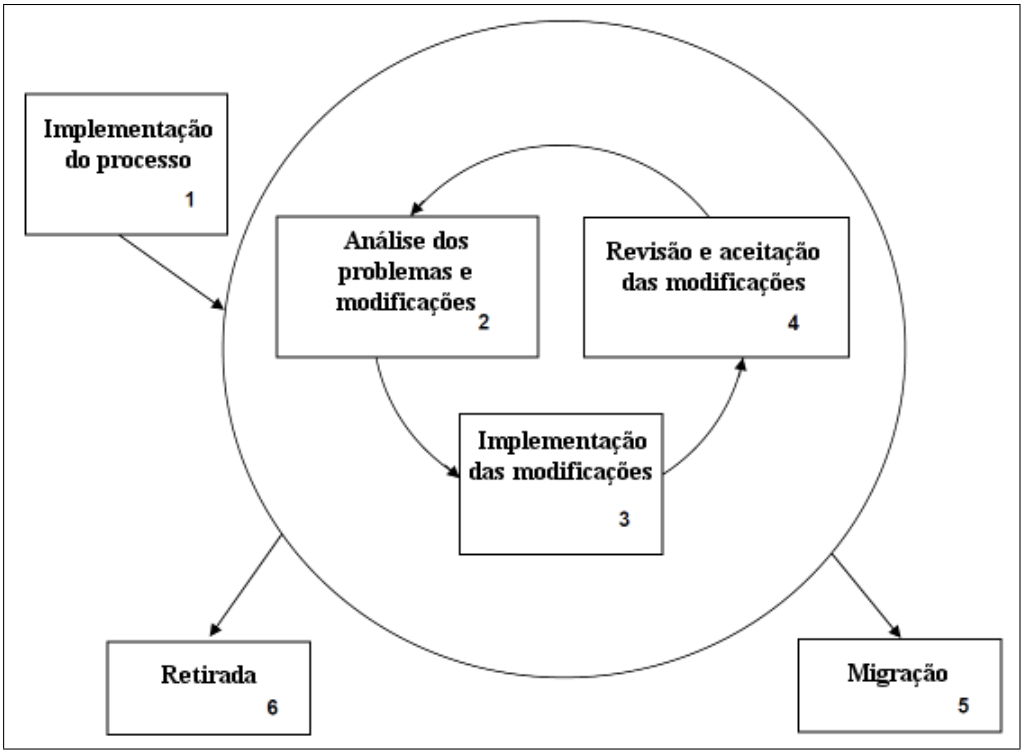

Figura 1: Processo de manutenção de software da norma ISO/IEC 14764

Fonte: Norma ISO/IEC 14764 (ISO/IEC, 2006)

1. **Implementação do processo**: os planos e procedimentos a serem executados durante a fase de manutenção devem ser estabelecidos e aprovados.

2. **Análise dos problemas e modificações**: as solicitações de manutenção devem ser analisadas e as alternativas para a execução das modificações devem ser propostas, documentadas e aprovadas.

3. **Implementação das modificações**: as modificações aprovadas devem ser desenvolvidas e testadas pelos mantenedores.

4. **Revisão e aceitação das modificações**: as modificações realizadas devem ser avaliadas para garantir a conformidade com o escopo aprovado e para determinar que estão corretas.

5. **Migração**: os passos necessários para atender as solicitações de migração do software para uma nova plataforma de execução, sistema de banco de dados e demais necessidades vinculadas às solicitações adaptativas devem ser definidos, documentados, aprovados e executados.

6. **Retirada**: analisar e executar a retirada segura do software do ambiente de produção ao final de sua vida útil. Deve-se também avaliar a necessidade de aquisição ou

de desenvolvimento de um novo sistema motivado por questões econômicas.

2.3.2 MANTEMA

A metodologia MANTEMA (POLO *et al.*, 1999a) (POLO *et al.*, 1999b) foi proposta para nortear a aplicação do processo de manutenção da versão de 1995 da norma ISO/IEC 12207 (ISO/IEC, 2008). De acordo com Polo *et al.* (1999b), o principal objetivo dessa metodologia é descrever um processo detalhado, resultante da consolidação das melhores práticas de todos os processos da norma ISO/IEC 12207 que impactam direta ou indiretamente a manutenção de software. O fluxo de trabalho definido para o atendimento de cada tipo de solicitação de manutenção foi posteriormente incluído na ferramenta MANTOOL (POLO *et al.*, 2001). As atividades dessa metodologia são distribuídas em grupos, conforme descreve-se a seguir:

- **Atividades Iniciais**: conjunto de atividades preliminares ao atendimento das solicitações de manutenção e comuns a todo tipo de solicitação. Envolve as atividades de entendimento e de planejamento da execução das solicitações de manutenção.
- **Atividades de Atendimento**: conjunto de atividades executadas para construir as mudanças e as correções solicitadas pelos usuários. As atividades desse grupo são definidas com base no tipo de solicitação requisitada. Para as manutenções corretivas urgentes Polo *et al.* (1999b) propõem a execução da seguinte sequência de atividades: investigação e análise das causas, implementação das correções, complementação da documentação, verificação da correção, elaboração da documentação correlacionada e implantação da nova versão do software no ambiente de produção.
- **Atividades Finais**: define as atividades de encerramento do atendimento da solicitação. A especificação da metodologia MANTEMA (POLO *et al.*, 1999a) (POLO *et al.*, 1999b) recomenda a avaliação da necessidade de migração ou de retirada segura do software ao término do atendimento de cada solicitação. Esse grupo de atividades é comum a todos os tipos de solicitações de manutenção.

2.4 Melhorias de Processo de Software

2.4.1 Iniciativas de Melhorias de Processos de Software

Na visão de Pressman (2009) e Sommerville (2007) as iniciativas de melhoria de processos representam o conjunto de esforços aplicados na evolução dos processos de software por meio da análise, da medição e do aperfeiçoamento das práticas de Engenharia de Software executadas ao longo do ciclo de vida de um sistema. De acordo com os relatos apresentados por Huang *et al.* (2005), Ferreiro *et al.* (2007), Pressman (2009), Santos, Oliveira e Silva (2009) e Kalinowski *et al.* (2010), as metas comumente estabelecidas para a melhoria desses processos envolvem: a redução dos custos de desenvolvimento e de manutenção, o aperfeiçoamento da qualidade dos produtos entregues, o aumento da produtividade das equipes, a redução do índice de retrabalho e do *time to market* para o lançamento dos produtos e de suas evoluções e o aumento da satisfação dos usuários.

Segundo Pressman (2009), os modelos de maturidade são classificados como abordagens de melhoria de processos de nível organizacional. Modelos de maturidade como o CMM (*Capability Maturity Model*)/CMMI (*Capability Maturity Model Integration*) (SEI, 1995) (CHRISSIS; KONRAD; SHRUM, 2006) e o MR MPS (Modelo de Referência para Melhoria de Processos do Software) (SOFTEX, 2009), estabelecem meta-modelos compostos por práticas genéricas e específicas para, por exemplo, desenvolver e avaliar a maturidade das organizações nos diversos processos executados ao longo do ciclo de vida de um software. Pressman (2009) destaca que os modelos de maturidade provêem indicadores sobre a maturidade dos processos, representada pela qualidade do processo, pelo entendimento da equipe sobre as práticas utilizadas e pelo nível das práticas de Engenharia de Software adotadas.

No CMM/CMMI (SEI, 1995) (CHRISSIS; KONRAD; SHRUM, 2006) as metas e práticas estabelecidas para a avaliação da maturidade das organizações são divididas em cinco níveis de maturidade, sendo eles: Inicial (Nível 1), Gerenciado (Nível 2), Definido (Nível 3), Quantitativamente Gerenciado (Nível 4) e Otimizado (Nível 5). O MR MPS (SOFTEX, 2009) é uma iniciativa semelhante e segundo SOFTEX (2009), Weber *et al.* (2008), Travassos e Kalinowski (2009) e Kalinowski *et al.* (2010), visa definir, estabelecer, disseminar e aprimorar um modelo de avaliação e melhoria de processos de software, focado nas necessidades e nas restrições das micros, pequenas e médias empresas brasileiras. Conforme destacam Weber *et al.* (2008) e Kalinowski *et al.* (2010) o MR MPS possibilita que essas empresas se beneficiem da implementação e da avaliação dos processos de software a um

custo razoável. O MR MPS (SOFTEX, 2009) adota sete níveis de maturidade, sendo eles: Parcialmente Gerenciado (Nível G), Gerenciado (Nível F), Parcialmente Definido (Nível E), Largamente Definido (Nível D), Definido (Nível C), Gerenciado Quantitativamente (Nível B) e Em Otimização (Nível A). De acordo com SOFTEX (2009) a divisão das metas e práticas em sete níveis de maturidade tem como objetivo facilitar a adoção do MR MPS pelas micros, pequenas e médias empresas e reduzir o tempo necessário para obtenção dos resultados das melhorias. É importante salientar que uma discussão mais aprofundada sobre modelos de maturidade está fora do escopo desta pesquisa de mestrado.

De acordo com Rocha, Maldonado e Weber (2001), Travassos e Kalinowski (2009) e Santos, Oliveira e Silva (2009) os modelos de maturidade têm obtido um amplo índice de adoção pela indústria de software. Segundo Solingen (2004), Sommerville (2007), Weber *et al.* (2008) e Santos, Oliveira e Silva (2009) as principais motivações para adoção dos modelos de maturidade são o aumento da competitividade entre as empresas do setor, o aumento da qualidade dos produtos obtidos por meio da melhoria dos processos de software e as possibilidades favoráveis de retorno do investimento. O relatório técnico publicado pelo *Software Engineering Institute* (SEI) e citado por Santos, Oliveira e Silva (2009) destaca que desde 2002, mais de três mil avaliações do CMMI foram realizadas. O estudo empírico realizado por Solingen (2004) destaca um retorno de investimento entre 8 e 13 dólares para cada dólar investido na adoção do CMMI. O relatório técnico apresentado por Gibson, Goldenson e Kost (2006) ratifica o retorno de investimento e outros benefícios proporcionados pelo CMMI. Segundo esse relatório, as organizações avaliadas obtiveram um retorno de investimento entre 7 a 27 dólares para cada dólar investido, além da redução de custo entre 3% a 87%, do aumento da qualidade dos produtos entre 2% a 132% e do aumento da produtividade das equipes entre 2% a 95%. O relatório técnico apresentado por Travassos e Kalinowski (2009) apresenta resultados semelhantes obtidos pelas empresas que adotaram o MR MPS.

Embora o foco principal dos modelos de maturidade como o CMM/CMMI (SEI, 1995) (CHRISSIS; KONRAD; SHRUM, 2006) e o MR MPS (SOFTEX, 2009) seja a fase de desenvolvimento inicial, a manutenção também usufrui de alguns dos benefícios obtidos pela adoção desses modelos. Por exemplo, as orientações propostas podem melhorar a manutenibilidade dos sistemas, garantir a evolução tecnológica e minimizar os problemas gerenciais encontrados durante a manutenção. Entretanto, conforme destacam April, Abran e Dumke (2004), April *et al.* (2005), Hall *et al.* (2001) e Jung e Goldenson (2009), existem dificuldades para aplicá-los durante a fase de manutenção. Entre as causas apontadas para essas dificuldades encontram-se a ausência do tratamento do conceito de maturidade

durante a manutenção, o tratamento superficial das práticas de melhoria de processos para a manutenção e o atendimento insatisfatório dos planos de rejuvenescimento, de reengenharia e de migração do software. Consequentemente, conforme observam Hall *et al.* (2001), as organizações não têm aplicado efetivamente os modelos de maturidade para evoluir os processos de manutenção.

Para estabelecer mecanismos para desenvolver e avaliar o nível de maturidade das organizações em relação a manutenção e orientar a evolução dos processos dessa fase do ciclo de vida, alguns estudos têm proposto modelos de maturidade específicos para manutenção de software, conforme discutido a seguir.

2.4.2 *Software Maintenance Maturity Model* (S3M)

April, Abran e Dumke (2004), April *et al.* (2005) e April e Abran (2009) desenvolveram um modelo de maturidade para a fase de manutenção denominado S3M ou SM^{mm}. Esse modelo é resultante da revisão das melhores práticas de normas e modelos de diversas áreas. Entre as referências utilizadas para concepção do S3M incluem-se as práticas sugeridas pela versão de 2004 da norma ISO/IEC 12207 (ISO/IEC, 2008), pela norma ISO/IEC 14764 (ISO/IEC, 2006), pelo padrão IEEE 1219 (IEEE, 1999) e pelos modelos CMMI (CHRISSIS; KONRAD; SHRUM, 2006), SWEBOK (IEEE, 2004), *Control Objectives for Information and related Technology* (COBIT) (ISACA, 2010) e *Information Technology Infrastructure Library* (ITIL) (OGC, 2011). O modelo S3M foi concebido com base nas necessidades dos diversos envolvidos nas atividades de manutenção, abrangendo os usuários, as equipes de infraestrutura, os desenvolvedores, os fornecedores, os gestores e as equipes de *help desk*. Ao todo são definidos três grupos de processos (processos primários, de apoio e organizacionais), similares aos encontrados na norma ISO/IEC 12207 (ISO/IEC, 2008). Segundo April e Abran (2009) os objetivos dos processos propostos resumem-se na garantia da disponibilidade das aplicações, na reação rápida em caso de falhas, na garantia dos limites de atendimento impostos pelos contratos de nível de serviço, na manutenção da confiança dos usuários sobre a competência e disponibilidade da equipe, além da execução dos serviços requisitados dentro do orçamento destinado à manutenção. Para garantir esses objetivos a especificação do S3M institui 18 *Key Process Areas* (KPA), 74 *roadmaps* e 443 práticas recomendadas, divididos em seis níveis de maturidade definidos como:

- **Nível 0 - Incompleto**: o processo não é executado na organização ou não existem evidências sobre a existência do processo.
- **Nível 1 - Executado**: as práticas são executadas de forma informal e não são documentadas. O sucesso da execução das atividades depende do conhecimento e da experiência individual dos membros da equipe de manutenção.
- **Nível 2 - Gerenciado**: existe consciência sobre a existência e a importância do processo e suas práticas se encontram disseminadas entre alguns membros das equipes de manutenção. Eventualmente são executadas atividades similares às definidas pelo processo.
- **Nível 3 - Estabelecido**: o processo e suas práticas são documentados, entendidos e executados por toda a organização de acordo com o especificado.
- **Nível 4 - Previsível**: as práticas estabelecidas são formalmente executadas e gerenciadas quantitativamente, conforme definido pelas metas acordadas para a avaliação do processo de manutenção.
- **Nível 5 - Otimizado**: as práticas e processos têm metas quantitativas de melhoria e são submetidos a revisões contínuas para adaptação às mudanças de objetivos do negócio e dos requisitos de software.

O modelo S3M endereça as necessidades dos diversos envolvidos na execução de todos os tipos de solicitações de manutenção. O modelo de referência proposto contém práticas relevantes para garantir a satisfação dos envolvidos e o sucesso dos projetos de manutenção. Dentre essas práticas destacam-se as práticas avançadas de gerenciamento quantitativo do processo e de melhoria contínua com base em soluções inovadoras, semelhantes às executadas pelas organizações avaliadas nos níveis de maturidade mais altos do CMM/CMMI (SEI, 1995) (CHRISSIS; KONRAD; SHRUM, 2006) e do MPS.BR (SOFTEX, 2009).

2.4.3 *Corrective Maintenance Maturity Model* (CM^3)

O modelo CM^3 proposto e avaliado por Kajko-Mattsson, Forssander e Olsson (2001), Kajko-Mattsson (2001), Kajko-Mattsson (2002a), Kajko-Mattsson (2002b), Kajko-Mattsson e Meyer (2005), Kajko-Mattsson (2006) e Kajko-Mattsson (2007) propõe uma plataforma para avaliação da maturidade de organizações no atendimento de solicitações

corretivas. A proposta apresentada é composta pelo modelo conceitual e seus elementos – responsáveis por definir os processos, suas atividades, os conceitos relacionados e as explicações e motivações para execução de cada atividade dos processos – pelos níveis de maturidade e pelos *roadmaps* para execução das atividades. Os processos propostos pelo CM^3 endereçam, do ponto de vista das manutenções corretivas, aspectos de maturidade relacionados às seguintes atividades: pré-entrega e transição para a equipe de manutenção, suporte aos usuários, atividades de resolução de problemas, testes e documentação das correções, atividades executadas durante a pré-liberação de solicitações adaptativas e perfectivas – que eventualmente podem produzir novos erros – e treinamento das equipes envolvidas no atendimento das solicitações corretivas. A maturidade das organizações na execução do conjunto de processos propostos é avaliada pelo CM^3 em três níveis, conforme descreve-se a seguir:

- **Nível 1 - Inicial**: o processo de execução das solicitações corretivas não é definido e adequadamente documentado. As atividades realizadas dependem das habilidades individuais dos profissionais. As organizações nesse nível de maturidade não possuem controle sobre o *status* do atendimento das solicitações corretivas e geralmente não provêem um *feedback* preciso aos usuários.

- **Nível 2 - Definido**: um modelo simplificado de processo é definido e documentado. As práticas propostas são rudimentares, porém consistentes e as atividades permitem o controle em alto nível do atendimento das solicitações corretivas.

- **Nível 3 - Otimizado**: o processo permite que as organizações tenham uma visão detalhada sobre o atendimento das manutenções corretivas. Toda a organização tem ciência sobre cada atividade do processo. O processo é submetido a iniciativas constantes de melhorias e são adotadas técnicas para prevenção de defeitos e para melhoria da produtividade da equipe.

2.4.4 *TI Service Capability Maturity Model*

Niessink e Vliet (2000) aplicaram uma abordagem distinta para o desenvolvimento, a avaliação e a evolução de processos de manutenção e das técnicas correlatas para coordenar os programas de melhoria contínua desses processos. No trabalho apresentado, Niessink e Vliet (2000) traçaram um paralelo entre manutenção de software e prestação de serviços, cujas características principais são a intangibilidade, a heterogeneidade, a simultaneidade na produção e no consumo dos resultados e a perecibilidade. Essas

características motivaram Niessink e Vliet (2000) a proporem um modelo denominado *IT Service Capability Maturity Model*, cujo objetivo principal é permitir a avaliação do nível de maturidade das organizações no fornecimento de serviços vinculados a manutenção de software. O modelo proposto por Niessink e Vliet (2000) baseia-se nos mecanismos de governança de processos e na possibilidade de melhoria da disponibilidade, da gestão e da eficiência das equipes de manutenção. Esse modelo adota uma estrutura de níveis de maturidade similar aquela apresentada pelo CMM (SEI, 1995), porém, sua aplicação é voltada para processos que apóiam a prestação de serviços na área de tecnologia da informação.

2.4.5 Comentários sobre Modelos de Maturidade para Manutenção de Software

Os três modelos de maturidade apresentados nesta seção têm como objetivo principal desenvolver e avaliar a maturidade das organizações em relação a manutenção de software. Embora a abordagem adotada e o foco de cada modelo sejam distintos, eles contribuem para a melhoria dos processos executados na fase de manutenção e estabelecem contratos para a comunicação entre os envolvidos de forma a melhorar o registro, o entendimento, a execução e o gerenciamento de solicitações de manutenção.

O modelo S3M (APRIL *et al.*, 2005) abrange todos os tipos de solicitações e seu escopo envolve as atividades de manutenção executadas durante o desenvolvimento inicial e as atividades de suporte operacional, correção de falhas e evolução do sistema, executadas após a transição para a fase de manutenção. No modelo *TI Service Capability Maturity Model* (NIESSINK; VLIET, 2000) as práticas recomendadas são definidas com base nos contratos de nível de serviço estabelecidos entre os consumidores e os provedores de serviços de manutenção. Uma preocupação recorrente das práticas desse modelo é alinhar o atendimento dos diversos tipos de solicitações com a capacidade real do provedor do serviço e estabelecer uma comunicação transparente entre as partes.

Apesar do modelo CM^3 (KAJKO-MATTSSON, 2002b) ser específico para as solicitações corretivas, as práticas recomendadas se destacam pela preocupação com a equipe de manutenção, principalmente com a evolução profissional dos colaboradores e com a definição de políticas de treinamento para melhorar eventuais ineficiências. Embora o planejamento da execução das solicitações esteja presente nos três modelos de maturidade discutidos, eles não explicitam práticas específicas para estimativa do custo das atividades

des de manutenção e também não mencionam a possibilidade de se agrupar solicitações durante o planejamento de novas versões dos produtos de software.

2.5 Planejamento e Gerenciamento da Manutenção de Software

Conforme discutido na Seção 2.2, os recursos necessários para atender os serviços solicitados durante a fase de manutenção representam uma parcela significativa do total dos recursos consumidos ao longo do ciclo de vida de um software. Além disso, de acordo com Kung e Hsu (1998) e Kung (2004), os principais problemas associados à manutenção de software estão relacionados aos aspectos gerenciais, derivados basicamente da dificuldade de gerenciar a alocação de recursos frente à demanda simultânea de serviços e das incertezas sobre a execução das solicitações durante o período de planejamento. Dessa maneira, percebe-se que a previsão do custo e de outras variáveis relacionadas, como por exemplo, a composição das equipes de manutenção, são mecanismos importantes para apoiar o planejamento e o gerenciamento da execução das solicitações. As subseções a seguir abordam dois aspectos relacionados ao planejamento e ao gerenciamento das atividades de manutenção de um software. A Subseção 2.5.1 relata a importância e os benefícios proporcionados pelo agrupamento de solicitações de manutenção em projetos de manutenção e a Subseção 2.5.3 apresenta as principais pesquisas sobre modelos de estimativa de custo para manutenção de software.

2.5.1 Agrupamento de Solicitações de Manutenção de Software

Ao longo da fase de manutenção, as mudanças do contexto no qual o software está inserido e os erros reportados pelos usuários contribuem para a necessidade de mudanças contínuas nos produtos de software. Entretanto, segundo Banker e Slaughter (1997) e Tan e Mookerjee (2005), as solicitações de manutenção tendem a ser executadas sob demanda, na medida em que são registradas. Esse cenário acontece tanto para manutenções corretivas – independente da prioridade – quanto para manutenções adaptativas e perfectivas. Do ponto de vista econômico, esse cenário pode representar perdas de produtividade das equipes e consequentemente perdas financeiras para as organizações responsáveis pela manutenção do software. Para aumentar o valor agregado ao sistema em cada liberação e obter a máxima eficiência produtiva, é necessário planejar a liberação de novas versões do

software, alinhando a importância das solicitações com a capacidade produtiva da equipe e com o prazo para liberação das versões.

Conforme destacam Jadallah *et al.* (2009), ao determinar quais solicitações devem compor as liberações dos sistemas é preciso avaliar os aspectos técnicos, os recursos disponíveis e as restrições de orçamento para execução das solicitações. Para viabilizar o planejamento das liberações de novas versões é necessário agrupar as solicitações para que essas sejam executadas em lotes. Porém, na medida em que as solicitações são agrupadas é preciso estabelecer uma política para determinar quando a manutenção do sistema será realizada. Tan e Mookerjee (2005) destacam que as políticas para agrupamento de solicitações de manutenção são baseadas em restrições temporais, podendo ser uniformes ou flexíveis. A política de manutenções uniforme impõe um intervalo de tempo constante para execução das solicitações, enquanto a política de manutenções flexível possibilita uma variação do intervalo de tempo entre a execução de duas atividades de manutenção. De acordo com Tan e Mookerjee (2005), a política de manutenção uniforme é mais fácil de ser gerenciada devido ao maior controle das ações. Porém, a política de manutenção flexível pode prover resultados melhores em relação à eficiência econômica.

De acordo com Rocha, Maldonado e Weber (2001), o custo total de manutenção abrange outros custos menos tangíveis do que o custo monetário decorrente da execução das atividades do processo de manutenção. Esses custos representam, por exemplo, a insatisfação do cliente quando sua solicitação não pode ser atendida em tempo hábil. O agrupamento de solicitações de manutenção auxilia no planejamento e no controle da execução dessas demandas. Entretanto, o atraso na entrega das implementações ocasionado pelo enfileiramento e pelo agrupamento dessas representa um custo para os usuários. Banker e Slaughter (1997) e Tan e Mookerjee (2005) classificam esses custos como custo de oportunidade ou custo de espera. Dessa forma, o custo de oportunidade deve ser considerado durante a priorização das solicitações para que a qualidade do atendimento não seja afetada. Todavia, é importante salientar que o agrupamento não deve ser aplicado para todas as solicitações. Para as solicitações onde o custo de oportunidade é demasiadamente elevado, o atendimento individual oferece vantagens em relação ao agrupamento de solicitações. Por exemplo, a prioridade elevada atribuída às solicitações corretivas devido à severidade do erro ou devido a questões definidas por contratos de nível de serviço podem elevar o custo de oportunidade e inviabilizar o agrupamento das solicitações. No caso de solicitações adaptativas e perfectivas, o custo de oportunidade também pode inviabilizar o agrupamento quando são impostas penalizações legais ou devido à incapacidade do software de atender às metas estabelecidas.

Um estudo realizado por Tan e Mookerjee (2005) destaca que o agrupamento de solicitações de manutenção pode auxiliar na composição e no dimensionamento das equipes de manutenção devido à facilidade de previsão da demanda. Outro benefício proporcionado pelo agrupamento de solicitações é o aumento do ganho de escala. Segundo Banker e Slaughter (1997) o ganho de escala representa a redução do custo de execução das solicitações quando essas são agrupadas em projetos de manutenção. O ganho econômico é proveniente de diversos fatores, como por exemplo, a redução do custo fixo ocasionado pela unificação dos esforços relacionados às atividades de gestão de configuração, dos testes das modificações e da implantação da nova versão do sistema. O ganho de escala também é positivamente influenciado pela curva de aprendizagem dos membros da equipe em relação à estrutura e ao comportamento do sistema. Neste caso, conforme defende Chan (2000), a redução do custo da manutenção ocorre devido à relação existente entre a familiaridade com o sistema e a produtividade das equipes.

Um estudo conduzido por Banker e Slaughter (1997) avaliou o ganho de escala em 129 projetos de manutenção compostos por solicitações adaptativas e perfectivas executados em uma grande instituição financeira. A investigação realizada empregou o modelo estatístico não paramétrico chamado *Data Envelopment Analysis* (DEA). Segundo Fonseca (2005), DEA é um modelo utilizado para estimar a eficiência produtiva de unidades com base na relação entre os produtos ou serviços produzidos (*outputs*) e os recursos consumidos para produzi-los (*inputs*). Para determinar a eficiência das unidades avaliadas cria-se uma unidade virtual ótima, cuja eficiência é a representação da eficiência máxima existente no conjunto de unidades avaliadas. Durante a avaliação é projetada no plano uma fronteira, definida pela unidade de eficiência ótima, de forma a compará-la com as unidades avaliadas. Conforme explicam Fonseca (2005) e Castanheira (2008), essa abordagem permite identificar as unidades eficientes e ineficientes, além de possibilitar a realização de *benchmarks* que indiquem a má utilização de recursos ou produtos com baixo rendimento, permitindo aos tomadores de decisões estudar estratégias para reverter o grau de ineficiência de suas unidades. A robustez das análises proporcionadas pelo DEA são destacadas por Banker e Slaughter (1997), onde explanam que o modelo não impõe uma forma específica para a função de cálculo, permitindo a avaliação dos dados considerando poucas suposições e, ao contrário do que acontece com os modelos paramétricos, não requer a aplicação de uma estrutura específica, como a forma linear ou quadrática.

A eficiência dos projetos de manutenção foi determinada por Banker e Slaughter (1997) com base na relação entre a quantidade de Pontos de Função (PF) (IFPUG, 2010) produzidos e o número de horas demandadas para implementar os pontos de função. Em

outras palavras, os projetos eficientes foram aqueles onde foi possível implementar a maior quantidade de pontos de função com o menor número de horas trabalhadas. A função adotada por Banker e Slaughter (1997) para avaliar a eficiência produtiva dos projetos de manutenção é dada por:

$$\chi_i = f(y_i, Z_i) * \theta_i \tag{1}$$

onde χ_i representa a eficiência produtiva, mensurada por meio da quantidade de recursos (horas trabalhadas) necessários para concluir os projetos de manutenção; f denota a função de avaliação da produção do modelo DEA; y_i determina o volume de produção obtido com a execução das modificações realizadas para atender aos requisitos que compõem as solicitações de manutenção – mensurado através da métrica Pontos de Função (PF) –; Z é o vetor de características dos projetos avaliados, composto pelas seguintes variáveis: experiência da equipe, número de plataformas de hardware envolvidas no projeto, número de técnicas de *design* utilizadas durante a execução do projeto, quantidade de ferramentas de desenvolvimento requeridas e número de ferramentas e técnicas de testes requeridas; θ_i representa o índice de ineficiência do projeto i, computado com base no modelo Banker, Charnes, Cooper (BCC), proposto por Banker, Charnes e Cooper (1984). Segundo Banker e Slaughter (1997), este modelo estima a eficiência das unidades identificando possíveis ganhos variáveis de escala.

Banker e Slaughter (1997) aplicaram esse modelo para avaliar a eficiência dos projetos de manutenção considerando as horas trabalhadas, a produção obtida e as variáveis do vetor de características que permitem identificar as similaridades entre os projetos. A avaliação da eficiência produtiva foi complementada com a comparação das perdas proporcionadas pelo custo de oportunidade inerente ao agrupamento das solicitações. Os resultados obtidos apontam uma potencial redução de até 36% no custo dos projetos de manutenção. Assim, tais resultados constituem evidências importantes a favor do agrupamento de solicitações. Todavia, conforme salientam os autores, as solicitações de manutenção geralmente não são agrupadas devido a falta de conhecimento por parte das equipes sobre os benefícios proporcionados por essa prática e pela resistência cultural dos usuários.

Embora exista a preocupação de se determinar o tamanho ótimo para os projetos de manutenção durante o agrupamento das solicitações, após o início da execução diversos eventos – como por exemplo o registro de novas solicitações com custo de oportunidade elevado ou alterações na composição da equipe de manutenção – podem requerer ações

reativas que podem culminar na reestruturação do escopo do projeto. Com base nesse cenário, Feng, Mookerjee e Sethi (2006) propuseram um modelo analítico baseado em programação dinâmica para determinar o tamanho e o tempo ideal para a execução de projetos de manutenção, identificando quando um projeto já em andamento deve ser encerrado ou reavaliado para garantir sua eficiência econômica. As conclusões apresentadas neste estudo apontam a possibilidade da aplicação de critérios de restrições temporais ou baseados em restrições de esforço despendido para identificar, a partir das solicitações que compõem um projeto de manutenção em execução, o subconjunto ótimo de solicitações a serem remanejadas, garantindo a eficiência produtiva e o valor agregado ao software.

2.5.2 Sistematização do Agrupamento de Solicitações de Manutenção de Software

O trabalho realizado por Malta *et al.* (2009) descreve uma iniciativa implantada na Divisão de Tecnologia da Informação da PUC Minas (DATAPUC) desenvolvida com o intuito de melhorar o planejamento e o gerenciamento do atendimento das solicitações de manutenção registradas para os sistemas da instituição. Essa iniciativa foi concebida com base na adaptação das práticas de gestão de requisitos de software do Processo DATAPUC de Desenvolvimento de Software (PDDS). O PDDS é um processo de software derivado do Rational Unified Process (RUP) (IBM CORPORATION, 2005), concebido com o intuito de aplicar as melhores práticas de Engenharia de Software e de Gestão de Projetos, além de integrar e sistematizar as atividades executadas pelas diversas equipes envolvidas no desenvolvimento inicial e na manutenção dos sistemas do DATAPUC, incluindo por exemplo, os analistas de negócio, os gerentes de projetos, os desenvolvedores, a equipe de garantia da qualidade e a equipe de suporte aos usuários.

A iniciativa proposta por Malta *et al.* (2009) e implantada no DATAPUC apresenta um conjunto de práticas que visam sistematizar o registro, o agrupamento e a priorização de solicitações de manutenção, além de recomendar o agrupamento e o atendimento de solicitações em projetos de manutenção. Essa iniciativa está fundamentada nas seguintes premissas: (1) as demandas de manutenção devem estar organizadas em projetos para maximizar a eficiência das atividades de planejamento e controle de execução e (2) o registro de solicitações das áreas usuárias deve ser feito de forma disciplinada e organizada para viabilizar a elaboração de propostas de projetos e, consequentemente, diminuir a quantidade de manutenções classificadas como urgentes.

Segundo o conjunto de práticas proposto por Malta *et al.* (2009), o planejamento de liberações e o atendimento das solicitações de manutenção deve ser regido por um calendário mensal de atendimento de solicitações de manutenção, apresentado na Figura 2. Esse calendário impõe uma política de manutenção uniforme, com intervalo de um mês para elaboração e execução de projetos de manutenção resultantes do agrupamento das solicitações de manutenção. As atividades executadas durante o planejamento e a execução dos projetos de manutenção são descritas a seguir:

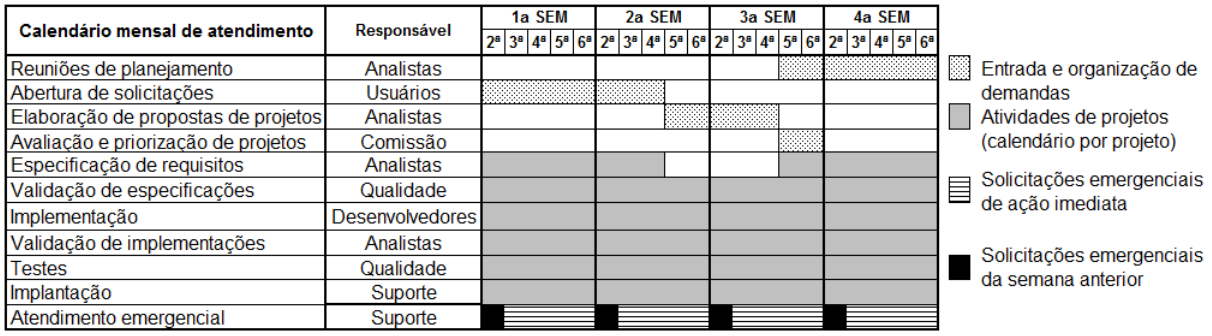

Figura 2: Calendário Mensal de Atendimento de Solicitações de Manutenção do DATAPUC

1. **Reuniões de planejamento:** na última semana de cada mês, os analistas de sistemas e gerentes de projetos devem realizar reuniões com os usuários-chave para apresentar a situação de cada projeto de manutenção que se encontra em execução e discutir as demandas, novas ou antigas, que deverão ser avaliadas no ciclo de priorização de projetos de manutenção de software do mês subseqüente.

2. **Registro de solicitações:** nos dez primeiros dias de um determinado mês, os usuários-chave de cada sistema devem formalizar as manutenções de software definidas na etapa anterior, através do registro de solicitações de manutenção, por exemplo, em um sistema de controle de solicitações de mudanças.

3. **Elaboração de propostas de projetos:** após o registro das solicitações, os analistas de sistemas têm aproximadamente uma semana para elaborar propostas de projetos de manutenção - em conjunto com a equipe de desenvolvedores - para serem apreciados pelos usuários-chave. As estimativas iniciais de esforço e prazo previstos para conclusão das solicitações devem ser mensuradas com base nos dados históricos, considerando as tecnologias envolvidas, a experiência e a disponibilidade da equipe e a complexidade e o tamanho da modificação.

4. **Avaliação e priorização de projetos:** ao final da terceira semana de cada mês, uma comissão de priorização de projetos, formada pelos líderes da equipe de manutenção e usuários-chave, deve se reunir para avaliar e priorizar as propostas de projetos. O painel de projetos deve apresentar a situação atualizada dos projetos em execução. A priorização deve se basear nas informações desse painel e na capacidade produtiva da equipe. As propostas aprovadas serão convertidas em projetos e, para cada projeto, será iniciado um ciclo de execução distinto, baseado na metodologia de desenvolvimento adotada pela instituição.

5. **Especificação de requisitos:** Ao término da aprovação do escopo do projeto, inicia-se a especificação detalhada das necessidades dos usuários, envolvendo a equipe de manutenção e os principais usuários do sistema em questão.

6. **Validação das especificações de requisitos:** Os artefatos construídos na etapa anterior são encaminhados para a área de qualidade para verificação da conformidade com os padrões definidos pelo *Software Engineering Process Group* (SEPG) (FOWLER; RIFKIN, 1990) da instituição. Os profissionais da área de garantia da qualidade produzem um relatório de avaliação com o resultado da verificação dos artefatos e, caso estes estejam aderentes aos propósitos dos usuários, o projeto de manutenção é então encaminhado para a implementação.

7. **Implementação:** Após receber o projeto, os desenvolvedores analisam os artefatos produzidos até então, planejam a execução desta atividade e constroem o desenho para implementação do código e a definição dos testes unitários.

8. **Validação da implementação:** Após concluir as atividades de implementação do projeto de manutenção em questão e realizar os testes funcionais de primeiro nível, a equipe de desenvolvedores libera uma nova versão do software no ambiente de homologação para que esta seja verificada pelo arquiteto de software e pelos analistas de sistemas.

9. **Testes:** Os projetos de manutenção aprovados na validação funcional são encaminhados para a equipe de garantida da qualidade. Nesta etapa são realizados os procedimentos de testes definidos para o projeto. As atividades de planejamento dos testes devem ocorrer em paralelo à implementação do projeto.

10. **Implantação:** Após concluídas as atividades de testes, a implantação dos projetos aprovados é realizada em ambiente de produção, de acordo com os prazos acordados com os usuários.

Conforme ilustrado na Figura 2, quando necessário uma solicitação pode ser atendida de forma individual, entretanto, esse modelo de atendimento deve ser adotado apenas para solicitações urgentes – por exemplo, manutenções corretivas ou evolutivas com datas críticas para atendimento. Essas solicitações são reportadas para o suporte de primeiro nível que, de acordo com procedimentos específicos para caracterização e triagem de solicitações urgentes definidos pela instituição, avalia a demanda e a encaminha para a equipe responsável pela manutenção do sistema. Além disso, a equipe do DATAPUC realizada plantões em todas as segundas-feiras para atender as solicitações da semana anterior que não podem aguardar o início do próximo projeto de manutenção.

2.5.3 Estimativa de Custo da Manutenção de Software

A estimativa de custo tem importância significativa no planejamento e no gerenciamento da manutenção de software. Conforme ressaltam Basili *et al.* (1996), uma das principais preocupações das organizações em relação a manutenção de software é entender e estimar o custo das manutenções realizadas. Além disso, Auer *et al.* (2006) defendem que a estimativa de custo afeta todas as decisões do gerenciamento da manutenção, o que abrange a alocação de recursos, a realização de licitações para terceirização de serviços, a definição do calendário de execução das solicitações, a gestão dos riscos inerentes dessa fase do ciclo de vida de software, etc. Portanto, os modelos de estimativa de custo devem ser submetidos a validações criteriosas para garantir a acurácia das estimativas, independentemente das características da equipe, do sistema e das solicitações registradas.

A fim de apoiar o gerenciamento da manutenção de software diversos modelos de estimativa de custo têm sido desenvolvidos e apresentados na literatura. Segundo Menzies *et al.* (2006) e Auer *et al.* (2006), os modelos de estimativa de custo podem ser baseados em métodos fundamentados na experiência dos profissionais ou em métodos formais. Os modelos baseados na experiência dos profissionais obtêm a estimativa de custo através do conhecimento empírico sobre a estrutura e o comportamento do sistema e do julgamento de *experts* no domínio. Já os modelos formais usufruem da aplicação de técnicas matemáticas e de abordagens algorítmicas para mensurar o custo requerido para atender às solicitações de manutenção. Pesquisas como as realizadas por Granja-Alvarez e Barranco-García (1997), Niessink e Vliet (1997) e Freire e Belchior (2007) apresentam modelos formais de estimativa de custo resultantes da adaptação de modelos originalmente desenvolvidos para a estimativa da fase de desenvolvimento inicial. Outros estudos como os realizados por Sneed (2004), Ahn *et al.* (2003) e Tan e Mookerjee (2005) apresentam modelos formais

concebidos exclusivamente para a fase de manutenção.

Motivados pelo estudo apresentado por Banker e Slaughter (1997), Tan e Mookerjee (2005) desenvolveram um modelo analítico para estimativa do custo de atendimento de solicitações de manutenção evolutivas (adaptativas e perfectivas) executadas como projetos de manutenção. O modelo proposto é destinado a organizações que mantêm seu portifólio de produtos por meio de departamentos de tecnologia da informação internos. Essa limitação ocorre, por exemplo, devido ao elevado controle requerido para estimar as métricas utilizadas pelo modelo e devido à necessidade de ajustar os coeficientes com base na experiência da equipe de manutenção. Além disso, esse modelo se difere dos demais pelo fato de estimar o custo de um conjunto de solicitações de manutenção. A função de estimativa de custo proposta considera os esforços demandados pela equipe desde a análise até a implantação das mudanças. A métrica Pontos de Função (PF) foi utilizada como unidade de tamanho para as modificações. Os coeficientes e termos da função de cálculo prosposta consideram os aspectos funcionais e técnicos que exercem impacto na implementação das mudanças.

Tendo em vista que um projeto de manutenção i é composto por N_i solicitações de manutenção, que por sua vez demandam M_i pontos de função, o custo do projeto i, de acordo com Tan e Mookerjee (2005), é dado por:

$$C_i^M = \gamma_0^i + \gamma_1^i M_i + \frac{1}{2}\gamma_2^i M_i^2 + c_3^i M_i \sum_{j=0}^{i-1} M_j \tag{2}$$

Os termos da função acima são descritos a seguir:

1. γ_0^i denota o custo fixo vinculado à execução do projeto i; Tan e Mookerjee (2005) descrevem a composição deste termo como o conjunto de despesas ocasionadas pela execuçao de atividades cujo custo não está diretamente relacionado com o tamanho da modificação necessária para atender às solicitações agrupadas no projeto de manutenção. Este termo permite que a estimativa de custo contemple, por exemplo, os custos com a aquisição de produtos e serviços necessários para o atendimento das solicitações (consultorias, licenças de software, compra de equipamentos, etc.), os gastos com o planejamento da execução das solicitações, os custos administrativos (impostos, despesas com viagens e transportes, etc.), os gastos com a preparação do ambiente para execução das solicitações (preparação dos ambientes de desenvolvimento, testes e homologação, procedimentos de gestão de configuração, etc.) e as horas dedicadas à implantação da nova versão no ambiente de produção.

2. $\gamma_1^i M_i$ representa o custo de análise e implementação dos M_i PF estimados para as solicitações contempladas no projeto i; neste termo γ_1^i representa o coeficiente de normalização e permite ajustar a estimativa do custo de análise e implementação às características das solicitações, da equipe e do sistema. Por exemplo, é possível tratar o impacto do índice de produtividade da equipe no custo dessas atividades.

3. $\frac{1}{2}\gamma_2^i M_i^2$ simboliza o custo envolvido na integração e teste das novas funcionalidades entre elas próprias; γ_2^i é o coeficiente de normalização e permite determinar o custo dessas atividades com base, por exemplo, na produtividade da equipe, nas características das solicitações e no índice de retrabalho. Conforme esclarecem Tan e Mookerjee (2005), o número de pontos de função é definido na forma quadrática devido à necessidade de integrar o conjunto de n funcionalidades entre elas próprias. Este termo representa o custo de todas as atividades envolvidas na integração e testes das funcionalidades, incluindo a correção das inconformidades identificadas durante os testes.

4. $c_3^i M_i \sum_{j=0}^{i-1}(M_j)$ representa o custo de integração e teste das novas funcionalidades com as funcionalidades já existentes no sistema antes da execução do projeto i; o fator c_3^i representa o nível de degradação do software devido à execução de outras manutenções e por isso, o custo de integração das novas funcionalidades entre elas próprias é calculado individualmente. Segundo Tan e Mookerjee (2005), esse coeficiente deve ser calculado como $\gamma_3^i \exp(\kappa n_M)$, sendo γ_3^i o coeficiente de normalização do custo de integração e teste do primeiro projeto de manutenção executado após a implantação do software ou após a última substituição do sistema (caso exista) e n_M o número de manutenções realizadas desde então. Por último, κ representa o coeficiente médio de degradação inserido no software a cada manutenção.

Em síntese, no modelo proposto, o custo de um projeto de manutenção é composto pelo custo fixo relacionado às solicitações de manutenção, pelo custo de análise, pelo custo de implementação e pelo custo de integração e teste das funcionalidades. Para ilustrar a aplicação desse modelo, imagine um sistema hipotético de *Content Management System* (CMS), baseado na plataforma Web, cujas funcionalidades somam 4.217 Pontos de Função (PF). É importante ressaltar que todos valores desse exemplo são estimativas. Em um determinado projeto de manutenção foram solicitadas as seguintes melhorias:

- Inclusão dos campos *tipo de documento* e *data de criação* no filtro das quatro pesquisas de documentos existentes – 32 PF.

- Inclusão de uma nova funcionalidade que permita alertar os usuários sobre a ocorrência de mudanças nos documentos (alteração, exclusão, inclusão, anotação e relacionamento com outros documentos) – 68 PF.

- Inclusão de um mecanismo para controle do vocabulário utilizado durante o cadastro dos documentos – 76 PF.

- Inclusão de um mecanismo para acesso ao sistema por *Secure Socket Layer* (SSL) – 13 PF.

A análise realizada para mensurar o tamanho das melhorias solicitadas pelos usuários resultou na identificação de 189 PF. Já a análise de impacto realizada identificou a necessidade de se integrar as novas funcionalidades a outras quatro funcionalidades já existentes. Essas funcionalidades têm o tamanho estimado de 226 PF. Com base nos dados históricos, nas características das solicitações, da equipe e do sistema, os coeficientes e parâmetros presentes na função de custo são calculados como:

- γ_0^i: o custo fixo neste exemplo é oriundo de gastos diversos. Segundo a base histórica da organização, a soma da quantidade de horas gastas com as reuniões de planejamento, priorização e gerenciamento das solicitações é de 32 horas. Este valor representa a duração das reuniões multiplicada pela quantidade de profissionais presentes. O custo médio por hora referente ao salário dos profissionais envolvidos é de R$ 65,00. A liberação de modificações semelhantes às solicitadas demandam em média quatro horas para serem implantadas, sendo que o valor hora da equipe de suporte responsável pela implantação é de R$ 25,00. Essas atividades são geralmente realizadas por um único membro da equipe de suporte. A equipe responsável pela garantia da qualidade tem um custo médio de R$ 41,00 por hora e é responsável pelos procedimentos de preparação dos ambientes de desenvolvimento e homologação. Essas atividades demandam em média seis horas e são realizadas por quatro profissionais. Além disso, a solicitação de disponibilização de acesso ao sistema por meio de SSL requer a aquisição de um certificado digital junto às entidades competentes. O custo apurado para aquisição é de R$ 3.500,00. Dessa maneira, o custo fixo desse projeto de manutenção é de R$ 6.664,00, conforme ilustrado a seguir:

$$\begin{aligned} \gamma_0^i &= (32 * 65) + (4 * 25) + (6 * 4 * 41) + (3.500) \\ &= 6.664,00 \end{aligned} \tag{3}$$

- γ_1^i: a base histórica da organização aponta o valor deste coeficiente como 3,73. Esse valor considera algumas características da equipe como o conhecimento sobre o sistema e o índice de produtividade. Também considera-se características das solicitações como, por exemplo, a complexidade das mudanças.

- γ_2^i: o valor deste coeficiente obtido da base histórica da organização é 0,73. O cálculo do coeficiente pode considerar, por exemplo, a produtividade das equipes envolvidas nas atividades de integração e teste, a complexidade das solicitações, o índice médio de retrabalho e as estratégias de testes necessárias para validação das modificações.

- c_3^i: a primeira atividade de manutenção realizada no software após a implantação teve o coeficiente de integração e teste (γ_3^i) computado como 0,07. Desde então foram realizados cinco projetos de manutenção. A organização estima que a cada projeto é inserido um percentual de degradação de 0,02. Dessa forma, ao aplicarmos esses valores na Fórmula 4 obtemos um coeficiente de degradação igual 0,76.

$$\begin{aligned} c_3^i &= \gamma_3^i \exp(\kappa n_M) \\ &= 0,07 \exp(0,02 * 5) \\ &= 0,76 \end{aligned} \tag{4}$$

Ao aplicamos esses valores na função de estimativa proposta por Tan e Mookerjee (2005), obtemos o custo do projeto de manutenção sendo igual a R$ 52.869,78. A equação a seguir apresenta a aplicação dos valores mencionados na função de custo:

$$\begin{aligned} C_i^M &= \gamma_0^i + \gamma_1^i M_i + \frac{1}{2}\gamma_2^i M_i^2 + c_3^i M_i \sum_{j=0}^{i-1} M_j \\ &= \gamma_0^i + (\gamma_1^i * 189) + \frac{1}{2}(\gamma_2^i * 189^2) + (c_3^i * 189 * 226) \\ &= 6.664 + (3,73 * 189) + \frac{1}{2}(0,73 * 189^2) + (0,76 * 189 * 226) \\ &= 6.664 + 704,97 + 13.038,17 + 32.462,64 \\ &= 52.869,78 \end{aligned} \tag{5}$$

Esse modelo foi avaliado por Tan e Mookerjee (2005) por meio de simulações matemáticas e apesar da legitimidade da técnica empregada, é necessário realizar estudos empíricos que reforcem sua eficiência no apoio às atividades cotidianas de estimativa de custo e de planejamento dos projetos de manutenção. A acurácia do modelo pode ser avaliada, por exemplo, através das técnicas *The Mean Magnitude of Relative Error* (MMRE)

e *Prediction at Level* (PRED), similar ao realizado nas investigações apresentadas por Foss *et al.* (2003), Korte e Port (2008) e Menzies *et al.* (2006). As técnicas MMRE e PRED possibilitam avaliar a performance de modelos de estimativa por meio de erros relativos e complementam as simulações matemáticas comumente encontradas nos estudos da área, apontando valores superestimados e reduzindo os falsos positivos.

Observando a aplicação da função de custo do modelo proposto por Tan e Mookerjee (2005), nota-se que para se obter uma estimativa precisa as organizações precisam manter uma base histórica consistente e preparada para mensurar os diversos coeficientes adotados. A ausência de diretrizes para estimativa desses parâmetros também pode representar um obstáculo para aplicação do modelo em ambientes reais de manutenção de software. Além disso, conforme sugerem Tan e Mookerjee (2005) é importante adaptar o modelo proposto de forma a contemplar cenários onde ocorre a terceirização de serviços, incluir o tratamento de modificações realizadas com o intuito de reverter a degradação inserida por outra solicitação de manutenção e evoluir o modelo para que esse permita estimar o custo dos demais tipos de solicitações e não apenas das manutenção evolutivas.

A avaliação do modelo proposto por Tan e Mookerjee (2005) apontou outras limitações além das aduzidas pelos autores. O terceiro termo da função de estimativa de custo determina o custo de integração e teste das novas funcionalidades entre elas próprias. Entretanto, o modelo não considera o impacto das mudanças, visto que as novas funcionalidades podem não requerer a integração e a execução de testes conjuntos com as demais funcionalidades do projeto de manutenção. Por exemplo, um projeto de manutenção pode conter solicitações que afetarão módulos distintos e isolados da aplicação. Esse comportamento também se reflete no custo de integração e teste das novas funcionalidades com as funcionalidades já existentes, representado pelo quarto termo da função sugerida. No modelo proposto, as consequências desses cenários precisam ser tratadas através dos coeficientes de normalização. Contudo, esse tratamento agrega complexidade à aplicação do modelo. Além disso, esses cenários podem ocasionar a superestimação do custo de manutenção e eventualmente produzir falsos positivos sobre a desvantagem do agrupamento de solicitações em projetos de manutenção. Também é preciso investigar quais os impactos dos diversos mecanismos de ajuste da estimativa dos pontos de função no modelo proposto.

2.6 Considerações Finais

A sistematização do desenvolvimento inicial e da manutenção de software obtida através das práticas de Engenharia de Software contribui para que as organizações entreguem sistemas de qualidade sem violar as restrições de prazo e de orçamento estabelecidas. Essas práticas são distribuídas entre as diversas fases do ciclo de vida de um sistema, sendo a manutenção de software uma das mais importantes, devido ao volume de esforço demandado e à responsabilidade de sustentar a operacionalidade e a evolução do software.

As propostas de processos de manutenção são fundamentadas na seleção, revisão e adaptação das melhores práticas sugeridas por referências importantes e algumas representam padrões internacionais – caso das normas ISO/IEC 12207 (ISO/IEC, 2006) e ISO/IEC 14764 (ISO/IEC, 2006). A especificação desses processos mapeiam as diversas particularidades e necessidades da manutenção, como por exemplo, a necessidade de planejamento e controle das liberações, a interação entre consumidores e provedores de serviços de manutenção, o estabelecimento, cumprimento e revisão dos contratos de nível de serviço e a necessidade de estimular o crescimento profissional dos colaboradores, mantendo-os motivados e aptos e garantir a evolução dos produtos de software. Essas características estão alinhadas com as propostas de modelos de maturidade desenvolvidas exclusivamente para a manutenção, os quais diversos estudos têm comprovado o seu papel na melhoria contínua dos processos de manutenção.

Ainda que os processos de software e os modelos de maturidade tratem o planejamento e a composição das liberações, as investigações sobre o agrupamento de solicitações de manutenção ainda são superficiais. Conforme evidencia o estudo apresentado por Banker e Slaughter (1997), o agrupamento de tais solicitações pode contribuir para o aumento da eficiência produtiva das equipes de manutenção. Esse fato é a principal motivação para a proposição de um processo para governar o agrupamento de solicitações em projetos de manutenção, abrangendo desde o registro das solicitações até a execução e implantação da nova versão e que auxilie no controle da expectativa dos usuários inerente à eventual postergação do atendimento de suas solicitações. Também é importante identificar e avaliar os benefícios proporcionados por esse processo que não estejam diretamente vinculados com a redução do custo de execução das solicitações, contribuindo para fundamentar a recomendação do processo proposto como uma iniciativa para a melhoria de processos de manutenção.

Capítulo 3

Estudo de Caso

3.1 Introdução

A Engenharia de Software Experimental – ou Engenharia de Software Baseada em Evidências – (MAFRA; TRAVASSOS, 2006) (MAFRA; BARCELOS; TRAVASSOS, 2006) visa caracterizar uma tecnologia em uso de forma a determinar, com níveis razoáveis de segurança, o que funciona e o que não funciona sob quais circunstâncias. Dessa forma, este capítulo apresenta um estudo de caso realizado com o intuito de conduzir uma avaliação detalhada da implantação do conjunto de práticas para sistematizar o registro, o agrupamento e a priorização de solicitações de manutenção de software. Esse conjunto de práticas foi proposto por Malta *et al.* (2009) – e discutido na Subseção 2.5.2 – e implantado na Divisão de Tecnologia da Informação da PUC Minas (DATAPUC).

Segundo Eisenhardt (1989), Mafra e Travassos (2006) e Conte (2009) estudo de caso é uma estratégia de pesquisa com o propósito de se investigar e entender uma entidade ou fenômeno dentro de um determinado espaço ou intervalo de tempo específico. Segundo esses autores, na Engenharia de Software os estudos de caso podem ser utilizados para monitor e avaliar atributos presentes em projetos, atividades ou atribuições através de análises quantitativas e/ou qualitativas. Além disso, os estudos de caso podem combinar diferentes métodos de coleta de dados como, por exemplo, *surveys*, entrevistas, questionários e observações.

Em síntese, para permitir a avaliação apresentada neste capítulo realizou-se uma caracterização do fluxo de atendimento das solicitações de manutenção, utilizando o algoritmo de *clusterização* denominado *k-means* (HAN; MICHELINE, 2005). A caracterização

foi utilizada como instrumento para viabilizar a análise e a comparação das mudanças ocorridas no atendimento de cada tipo de solicitação de manutenção, executadas antes e após a implantação do conjunto de práticas avaliado.

O restante deste capítulo está organizado conforme descrito a seguir. A Seção 3.2 apresenta o planejamento do estudo de caso. A Seção 3.3 a caracterização das solicitações de manutenção e a avaliação dos benefícios proporcionados pela adoção da prática de agrupar solicitações de manutenção em projetos de manutenção.

3.2 Planejamento do Estudo de Caso

3.2.1 Contexto

Nesta dissertação, com a realização do estudo de caso, pretendeu-se avaliar quais os benefícios e os eventuais pontos negativos proporcionados pela prática de agrupar solicitações de manutenção em projetos de manutenção. Mais especificamente este estudo de caso avaliou o uso do conjunto de práticas para sistematizar o registro, o agrupamento e a priorização das solicitações de manutenção proposto por Malta *et al.* (2009).

3.2.2 Objetivos

Seguindo o paradigma *Goal Question Metrics* (GQM) (BASILI; ROMBACH, 1988), este estudo de caso pode ser definido da seguinte forma:

Analisar	o conjunto de práticas para sistematizar o agrupamento de solicitações de manutenção proposto por Malta *et al.* (2009)
Com o propósito de	caracterizar o fluxo de atendimento das solicitações de manutenção e identificar os benefícios e os eventuais pontos negativos
Com respeito ao	planejamento e atendimento das solicitações de manutenção
Do ponto de vista da	distribuição do tempo de vida e de atendidamento das solicitações entre as atividades de Engenharia de Software

	executadas e os intervalos de inatividade ocorridos ao longo do atendimento
No contexto das	solicitações de manutenção atendidas pelo DATAPUC em 2008 e 2009

3.2.3 Questões de Pesquisa

O propósito deste estudo de caso foi responder as seguintes questões:

- Qual o fluxo de atendimento típico das solicitações agrupadas em projetos antes e após a implantação do conjunto de práticas avaliado?
- Qual o fluxo de atendimento típico das solicitações atendidas individualmente antes e após a implantação do conjunto de práticas avaliado?
- Quais são as variações existentes no fluxo de atendimento das solicitações atendidas individualmente antes e após a implantação do conjunto de práticas avaliado?
- Quais são os pontos negativos da implantação do conjunto de práticas avaliado?

3.2.4 Escopo

Para delinear o escopo desta pesquisa foram estabelecidos critérios para garantir, de forma equilibrada, a imparcialidade, a abrangência e a acurácia das análises e das comparações realizadas. Dessa forma, os seguintes critérios foram adotados para selecionar e avaliar a amostra de dados:

- Quanto à seleção da amostra de dados:
  - Conter solicitações de sistemas com característiscas diversas, incluindo diferentes linguagens de programação e plataformas de execução.
  - Considerar solicitações registradas antes (2008) e após (2009) a implantação do conjunto de práticas avaliado.
  - Desconsiderar as solicitações registradas antes do período avaliado.

  - Desconsiderar as solicitações atendidas em um determinado período a ser considerado como período de adaptação às práticas adotadas.

- Quanto à caracterização e avaliação da amostra de dados:

  - Permitir a caracterização e avaliação do fluxo de atendimento das solicitações sem a necessidade de se conhecer a estrutura e o código fonte dos sistemas.
  - Permitir a avaliação dos diferentes tipos de solicitações de manutenção utilizando um conjunto único de procedimentos.
  - Permitir identificar a correlação existente entre as características analisadas.

As subseções a seguir descrevem a amostra de dados selecionada com base nos critérios definidos para esta pesquisa e a metodologia de avaliação utilizada para conduzir a caracterização, as análises e as comparações necessárias para atender o escopo definido para este estudo de caso.

3.2.5 DATAPUC

O DATAPUC é responsável pela prestação de serviços de tecnologia da informação para a Arquidiocese de Belo Horizonte e para as instituições mantidas pela Sociedade Mineira de Cultura (SMC). É de responsabilidade do DATAPUC gerir as solicitações registradas durante todo o ciclo de vida dos sistemas. As demandas registradas podem envolver a aquisição, o desenvolvimento ou a manutenção de software. À época da realização desta avaliação o portifólio do DATAPUC era composto por 40 sistemas com aproximadamente quatro milhões de linhas de código. A equipe de manutenção do DATAPUC era formada por 34 profissionais responsáveis pela execução das diversas atividades definidas pelos processos de manutenção da organização. A Tabela 1 apresenta os principais sistemas mantidos pelo DATAPUC (em relação ao número de solicitações registradas). Esses sistemas foram desenvolvidos em linguagens diversas e são executados em plataformas também distintas. Por exemplo, existem sistemas cliente-servidor e sistemas Web desenvolvidos nas linguagens Delphi e Java, acessando diferentes sistemas de gerenciamento de banco de dados.

Tabela 1: Principais sistemas mantidos pelo DATAPUC

Sigla	**Descrição**
SGA	Sistema de Gestão Acadêmica
GRA	Sistema de Gestão de Receita Acadêmica
SAL	Sistema Acadêmico de Pós-Graduação *Lato Sensu*
SGC	Sistema de Gestão de Colégios
GPS	Sistema de Gestão de Processos Seletivos
GPJ	Sistema de Gestão de Práticas Jurídicas
GPE	Sistema de Gestão de Pesquisa e Extensão
SGE	Sistema de Gestão de Estágios
SAE	Sistema de Autorização de Pagamento Eletrônico
GPI	Sistema de Gestão de Processos de Inscrição
GPP	Sistema de Gestão de Projetos Pedagógicos
GAC	Sistema de Gestão de Atendimento de Clínicas
SOL	Sistema de Solicitações

3.2.6 Cenários Avaliados

A análise dos benefícios proporcionados pelas práticas propostas por Malta *et al.* (2009) no DATAPUC considerou dois cenários que representam períodos distintos na gestão e no atendimento das solicitações de manutenção. O primeiro cenário representa o período que antecede a implantação das práticas avaliadas. Neste período as solicitações eram enfileiradas e posteriormente priorizadas pela equipe de manutenção, em conjunto com o coordenador responsável pelo sistema. Entretanto, a priorização não era realizada de forma sistemática e também não considerava o contexto da manutenção e da evolução do sistema como um todo. Além disso, caso a nova solicitação fosse julgada como mais importante do que as atividades que estavam sendo realizadas, o serviço corrente era interrompido. Em algumas situações, a prioridade das manutenções poderia mudar mais de uma vez ao longo do dia. Dessa forma, o sucesso do trabalho dependia da capacidade de argumentação da equipe com os usuários e também da habilidade de organizar o que deveria ser feito. Neste primeiro período avaliado, as solicitações eram classificadas em um dos seguintes grupos:

- **Solicitações Urgentes**: envolve basicamente as solicitações com custo de oportunidade elevado, executadas individualmente e que tendem a representar mudanças pontuais nos sistemas. As solicitações corretivas e solicitações para acerto de dados são os principais tipos de solicitações encontrados nesta categoria.

- **Projetos Internos**: antes da adoção das práticas para sistematizar o agrupamento das solicitações de manutenção, algumas demandas já eram agrupadas e tratadas como projetos. Porém, essa prática era uma iniciativa individual, realizada de maneiras diversas, não era sistematizada e não atendia aos requisitos de uma política de manutenção uniforme. Além disso, os projetos continham solicitações registradas pela própria equipe de manutenção.

- **Projetos Urgentes (PU)**: solicitações atendidas individualmente, cuja prioridade e esforço estimado não permitiam sua classificação como urgente. Em outra palavras, os PU abrangiam as solicitações que não requeriam a interrupção imediata das atividades em andamento ou as solicitações que demandavam 40 horas ou mais para serem atendidas. Apesar do fluxo de atendimento desse grupo ser igual ao fluxo de atendimento das solicitações urgentes, as informações sobre a execução das atividades desempenhadas não eram registradas em sua totalidade no sistema de gestão de mudanças.

O segundo cenário avaliado se refere ao período que sucede a implantação das práticas avaliadas. A implantação do conjunto de práticas proposto por Malta *et al.* (2009) foi efetivamente concluída em novembro de 2008. A partir desse momento as diretrizes propostas para sistematizar o registro, a priorização e o agrupamento de solicitações em projetos de manutenção passaram a ser aplicadas no atendimento de todas as solicitações. A equipe do DATAPUC estabeleceu um intervalo de um mês entre a execução dos ciclos de projetos de manutenção. Essa duração foi determinada com base no volume de demanda e na capacidade da equipe interna de absorver o custo de oportunidade vinculado a cada solicitação de manutenção.

No DATAPUC, a elaboração de propostas de projetos se inicia na última semana de cada mês e tem duração fixa de duas semanas. Devido à estrutura organizacional do DATAPUC e ao relacionamento da equipe com os usuários, o registro das solicitações no sistema de gestão de mudanças só pode ser realizado durante esse período e os projetos são registrados como uma única solicitação no sistema de gestão de mudanças. Para as solicitações de caráter urgente, o registro pode ser realizado a qualquer momento, independentemente da situação dos projetos em andamento. Na segunda semana de cada mês – depois do período de registro das solicitações – a equipe de analistas de sistemas dispõe de uma semana para elaborar as propostas de projetos, que posteriormente são submetidas ao comitê de priorização. A execução dos projetos inicia na terceira semana de cada mês. Além disso, recomenda-se a elaboração de projetos que demandem

aproximadamente uma semana para serem implementados. No segundo período avaliado as solicitações passaram a ser classificadas nos seguintes grupos:

- **Solicitações Urgentes**: grupo análogo ao grupo de solicitações urgentes do primeiro cenário. Essas solicitações são caracterizadas basicamente pelo custo de oportunidade elevado e envolvem a correção de falhas que tendem a comprometer a operacionalidade do software e os pedidos de acerto de dados.
- **Projetos de Manutenção**: solicitações executadas como projetos de manutenção, sendo submetidas ao processo sistemático de registro, agrupamento, priorização e execução, conforme proposto pelo conjunto de práticas proposta por Malta *et al.* (2009).
- **Plantões**: para minimizar o impacto inicial do agrupamento das solicitações para os usuários, a equipe do DATAPUC adotou uma classificação intermediária entre os projetos e as solicitações urgentes, denominada *Plantão*. Os plantões correspondem a um dia da semana no qual a equipe reserva parte da sua capacidade produtiva para atender às solicitações que não podem aguardar o inicío do próximo projeto e que não foram inicialmente classificadas como urgentes. O atendimento dessas solicitações era, em geral, realizado às segundas-feiras. Em setembro de 2010, o DATAPUC extinguiu a realização de plantões.

Além da classificação em grupos, as solicitações também eram classificadas por tipo de manutenção. Nos dois cenários avaliados eram utilizados os seguintes tipos de solicitações: (1) Acerto de dados, (2) Execução Manual de Processos, (3) Manutenção em Funções Existentes, (4) Implementação de Novas Funções e (5) Suporte ao Usuário. As solicitações de suporte a usuário podem envolver, por exemplo, o diagnóstico de problemas, o esclarecimento de dúvidas, a execução de provas de conceito e a administração de segurança dos sistemas. Para adequar a classificação utilizada pelo DATAPUC com a classificação proposta pelo padrão IEEE 1219 (IEEE, 1999), as solicitações *Manutenção em Funções Existentes* e *Implementação de Novas Funções* foram tratadas como *solicitações perfectivas*. Além disso, apesar de solicitações de *suporte ao usuário* não existirem no padrão IEEE 1219 (IEEE, 1999), elas foram consideradas neste estudo de caso pois podem ajudar na identificação de novos erros durante o seu atendimento e por compartilharem a capacidade de produção da equipe de manutenção do DATAPUC.

3.2.7 Amostra de Dados

Para realização do presente estudo de caso foram extraídos do sistema de gestão de mudanças diversos dados sobre as solicitações de manutenção de todos os sistemas do DATAPUC. Conforme ilustrado na Tabela 2, a amostra incial continha 4.150 solicitações registradas entre 2008 e 2009. Dentre essas solicitações, 3.870 foram atendidas entre fevereiro a outubro dos dois anos avaliados. Das solicitações atendidas, 3.686 foram registradas neste intervalo de tempo. Este estudo de caso considerou apenas o subconjunto formado pelas solicitações *registradas* e *atendidas* na sequência de meses avaliada (fevereiro a outubro). Essa seleção foi aplicada para evitar que as solicitações registradas antes da implantação do conjunto de práticas avaliado ou durante o período de adaptação influenciassem os resultados do estudo.

Tabela 2: Número de solicitações da amostra de dados

Solicitações	PU (2008)	Plantões (2009)	Projetos		Urgentes		Total		
			2008	2009	2008	2009	2008	2009	Geral
Registradas	986	967	61	93	1.075	968	2.122	2.028	4.150
Atendidas	748	965	49	74	1.069	965	1.866	2.004	3.870
Registradas e atendidas	666	932	22	62	1.051	953	1.739	1.947	3.686

Conforme apresenta a Tabela 2, os cenários avaliados – e discutidos na Seção 3.2.6 – incluem as solicitações atendidas em dois períodos distintos e consecutivos:

- **2008**: este período inclui as 1.739 solicitações (666 PU, 22 projetos e 1.051 solicitações urgentes) registradas e atendidas entre os meses de fevereiro a outubro de 2008.
- **2009**: este período abrange as 1.947 solicitações (932 plantões, 62 projetos e 953 solicitações urgentes) registradas e atendidas entre os meses de fevereiro a outubro de 2009.

A Tabela 3 apresenta o número de horas trabalhadas pela equipe de manutenção do DATAPUC no atendimento das solicitações durante os dois períodos avaliados. No total, o atendimento das solicitações demandou 22.913 homens horas (11.203 em 2008 e 11.710 em 2009). Entretanto, o esforço total aplicado no atendimento das solicitações consideradas por este estudo de caso (solicitações registradas e atendidas nos meses avaliados) foi de 16.774 homens horas (6.932 em 2008 e 9.842 em 2009). É importante salientar que o

número de colaboradores da equipe de manutenção do DATAPUC não apresentou variação entre os dois períodos avaliados. Além disso, ambos os períodos possuem a mesma sequência de meses.

Tabela 3: Número de horas trabalhadas

Ano		Horas Trabalhadas				
		PU	Plantões	Projetos	Urgentes	Total
2008	Todas solicitações	6.078	0	2.578	2.547	11.203
	Solicitações consideradas	3.734	0	741	2.457	6.932
2009	Todas solicitações	0	4.174	4.383	3.153	11.710
	Solicitações consideradas	0	4.027	2.726	3.089	9.842

Para avaliar os benefícios obtidos com o agrupamento das solicitações em projetos de manutenção, várias informações sobre as solicitações foram extraídas do sistema de gestão de mudanças do DATAPUC. Por exemplo, foram coletadas as seguintes informações sobre cada solicitação: data de registro, descrição e prioridade. Também foram extraídas informações detalhadas sobre o fluxo de atendimento de cada solicitação, incluindo as atividades executadas desde o registro até a implantação no ambiente de produção. É importante ressaltar que os projetos de manutenção são registrados como uma única solicitação e os dados históricos disponíveis não permitiram a identificação da quantidade de solicitações agrupadas em cada projeto.

3.2.8 Metodologia de Caracterização

Esta subseção descreve a metodologia utilizada para representar e compreender o fluxo de atendimento das solicitações de manutenção e para realizar a análise das mudanças ocorridas no atendimento dessas solicitações após a adoção do conjunto de práticas proposto por Malta *et al.* (2009) para sistematizar o agrupamento das solicitações em projetos de manutenção. Esta metodologia é uma adaptação da metodologia de caracterização proposta por Menasce e Almeida (2000) para representar e entender o comportamento de usuários de aplicações Web. Posteriormente, ela foi adaptada por Marques-Neto *et al.* (2004) para avaliar o comportamento de consumidores de serviços de internet banda larga.

Em síntese, para representar, entender, analisar e comparar o fluxo de atendimento das solicitações de manutenção executadas nos dois períodos avaliados (2008 e 2009), os seguintes procedimentos foram realizados:

1. Para cada solicitação de manutenção, foi gerada uma instância de um grafo denominado SMRMG (*Software Maintenance Request Model Graph*). O SMRMG é um grafo direcionado, onde os vértices representam os estados do ciclo de vida de uma determinada solicitação. Esses estados correspondem às atividades de Engenharia de Software previstas para serem executadas durante o atendimento da solicitação, ou aos intervalos de tempo em que não houver a dedicação de horas de trabalho. Já as arestas do SMRMG representam as transições entre os estados e devem ser ponderadas com a diferença (*em horas úteis*[1]) entre: (1) o início da primeira atividade ou do intervalo de inatividade do vértice de origem e (2) o início da primeira atividade ou do intervalo de inatividade do vértice de destino. A representação conceitual do fluxo de atendimento das solicitações de manutenção foi disponibilizada pela equipe do DATAPUC. Posteriormente, o fluxo de atendimento foi melhorado e adaptado para a realidade dos dados existentes na amostra de dados avaliada. As adaptações foram realizadas com o objetivo de permitir representar todas as solicitações de um mesmo grupo utilizando o mesmo SMRMG.

   Devido às diferenças entre as atividades definidas para o atendimento dos projetos de manutenção e dos demais grupos de solicitações, o SMRMG dos projetos foi representado de forma distinta (a representação gráfica e a descrição dos vértices de cada SMRMG são apresentados nas Subseções 3.3.1 e 3.3.2).

2. Em seguida, para cada SMRMG foram computadas algumas características de avaliação que representam indicadores importantes sobre o atendimento das solicitações de manutenção. As características definidas são discutidas a seguir:

   - *QueueTime*: tempo decorrido entre o registro da solicitação no sistema de gestão de mudanças até a sua respectiva conclusão ou cancelamento.
   - *WaitTime*: intervalo entre o registro da solicitação no sistema de gestão de mudanças e o início da execução da primeira atividade definida no fluxo de atendimento. Nos projetos, essa atividade é a análise preliminar e nas demais solicitações a triagem da demanda.
   - *ServiceTime*: período referente ao atendimento efetivo da solicitação, desconsiderando o tempo de espera. Em outras palavras, essa característica inclui a duração das atividades de análise, implementação, validação, implantação, etc. e os períodos de inatividade existentes durante o atendimento da solicitação.

[1]Foi considerado o intervalo entre 08:00 e 12:00 e entre 14:00 e 18:00. Nos casos onde as atividades começaram ou se encerram fora dos limites estabelecidos foi considerado, apenas para o dia do início/término da atividade, o número de horas corridas destinadas ao atendimento da solicitação.

- *AnalysisTime*: duração da execução das atividades de análise, agrupamento (quando aplicável) e priorização das solicitações.
- *PlanningTime*: duração das atividades de planejamento da execução das solicitações aprovadas durante a priorização (característica exclusiva para projetos de manutenção).
- *ImplementaionTime*: intervalo referente à implementação das mudanças.
- *StandByTime*: duração do período em que a execução da solicitação permanecer suspensa. A suspensão da execução pode ocorrer, por exemplo, devido ao registro de outras solicitações com custo de oportunidade elevado ou a mudanças inesperadas na composição das equipes, ou ainda devido a ausência de informações necessárias para conclusão do atendimento das solicitações.
- *ValidationTime*: duração das atividades desempenhadas pela equipe de garantia da qualidade, executadas principalmente para assegurar a aderência das mudanças realizadas ao escopo aprovado para as solicitações.
- *DeploymentTime*: período referente a execução das atividades de implantação das mudanças e obtenção da aprovação final junto aos usuários envolvidos.

Considerando as características definidas, as seguintes equações são verdadeiras:

$$QueueTime = WaitTime + ServiceTime \quad (6)$$

$$\begin{aligned} ServiceTime = {} & AnalysisTime + PlanningTime + \\ & ImplementationTime + StandByTime \\ & ValidationTime + DeploymentTime \end{aligned} \quad (7)$$

3. As arestas dos SMRMG e as características de avaliação foram utilizadas para compor o vetor de características ou vetor de atributos de cada solicitação. Posteriormente, cada vetor de características foi classificado de acordo com os critérios a seguir: *por período* (2008 e 2009) e *por grupo* (projetos de manutenção, solicitações urgentes, PU e plantões). Como as solicitações urgentes, PU e plantões utilizam o mesmo fluxo de atendimento, esses grupos foram representados pelo mesmo SMRMG e consequentemente pelo mesmo vetor de características.

4. Para realizar a caracterização do atendimento das solicitações de manutenção os vetores de características foram submetidos a um algoritmo de *clusterização*. Segundo Menasce e Almeida (2000), Bozdogan (2003), Goldschmidt e Passos (2005)

e Han e Micheline (2005), a *clusterização* ou agrupamento é o processo de agrupar objetos físicos ou abstratos e não rotulados em subconjuntos denominados *clusters*. O agrupamento é realizado de tal forma que os objetos de um determinado *cluster* compartilhem um conjunto de propriedades comuns e que os distinguem dos objetos agrupados em outros *clusters*. A similaridade dos objetos é definida pelos atributos (características) que os descrevem. O agrupamento visa maximizar a similaridade *intracluster* e minimizar a similaridade *intercluster*. A *clusterização* se destaca por possibilitar a descoberta de padrões na distribuição dos dados e por apontar correlações interessantes entre as características que descrevem os objetos analisados. Além disso, por possibilitar o agrupamento de elementos sem a necessidade de rótulos predefinidos, essa técnica é classificada como não supervisionada.

A *clusterização* pode ser realizada por diferentes métodos. Conforme destacam Han e Micheline (2005) os principais métodos existentes são: (1) por particionamento, (2) hierárquicos, (3) baseados na densidade, (4) baseados em *grids* e (5) baseados em modelos. Neste estudo de caso foi utilizada a *clusterização* por particionamento, mais especificamente usando o algoritmo *k-means*(HAN; MICHELINE, 2005). Segundo Menasce e Almeida (2000), Bozdogan (2003), Ordonez e Omiecinski (2004) e Han e Micheline (2005), o *k-means* é um algoritmo heurístico, não supervisionado, não probabilístico e não paramétrico. Segundo esses autores, a similaridade dos objetos no *k-means* é determinada com base na média das características dos objetos e a associação desses aos *clusters* é realizada basicamente pela distância euclidiana entre a média de cada objeto com a média computada para o *cluster*. A associação dos objetos ocorre por meio de um processo iterativo, onde a cada iteração a centróide (média) de cada *cluster* é reavaliada com base na média dos valores das características dos objetos agrupados no *cluster*.

De acordo com Bozdogan (2003), Ordonez e Omiecinski (2004) e Han e Micheline (2005), as principais vantagens desse algoritmo são a velocidade de execução, a garantia da exclusividade na associação dos objetos e a qualidade do agrupamento. Como desvantagens são apontadas questões referentes a perda de desempenho ao lidar com grandes amostras de dados, a possibilidade de requerer um número grande de iterações para definição do resultado ótimo e a necessidade de se informar o número de *clusters* desejados.

Para contornar essa última limitação, foi utilizada uma derivação do algoritmo *k-means*, denominada *Xmeans*, disponível na ferramenta WEKA[2] (*Waikato Environ-*

[2]Disponível em `http://www.cs.waikato.ac.nz/ml/weka`

ment for Knowledge Analysis). A implementação desse algoritmo dispensa a necessidade de informar o número desejado de *clusters*.

5. Após a identificação dos *clusters* e de seus valores, para cada atributo do vetor de características foi realizada a interpretação dos dados computados com o objetivo de identificar padrões para cada *cluster*. Em seguida, os *clusters* de 2008 foram comparados com os *clusters* – de mesmo grupo – de 2009 e assim, foram analisadas as mudanças ocorridas no atendimento das solicitações após a adoção do conjunto de práticas proposto por Malta *et al.* (2009).

6. A interpretação dos padrões que caracterizam cada *cluster* foi complementada com a análise da probabilidade de execução de cada transição representada nos SMRMG. O objetivo dessa análise é aprofundar o entendimento sobre o fluxo de atendimento das solicitações agrupadas em cada *cluster*. Em casos específicos, informações complementares sobre o atendimento das solicitações – extraídas do sistema de gestão de mudanças do DATAPUC – foram utilizadas para enfatizar os padrões identificados.

Os resultados obtidos com a aplicação da metodologia descrita são discutidos no restante deste capítulo.

3.3 Resultados

Esta subseção apresenta a caracterização do atendimento das solicitações da amostra de dados descrita na Subseção 3.2.7 e os resultados da avaliação da caracterização. A Subseção 3.3.1 apresenta a caracterização da execução dos projetos de manutenção. A Subseção 3.3.2 apresenta a caracterização do atendimento das solicitações urgentes e por fim, a Subseção 3.3.3 apresenta a caracterização do atendimento dos PU e dos plantões.

3.3.1 Projetos de Manutenção

A Figura 3 apresenta o SMRMG utilizado para representar o ciclo de vida dos projetos de manutenção. O significado dos vértices deste SMRMG são descritos a seguir:

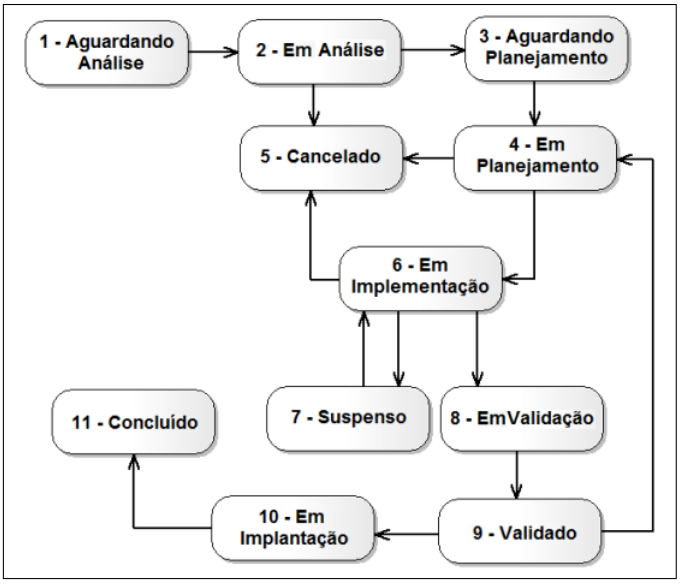

Figura 3: SMRMG dos projetos de manutenção

1. **Aguardando Análise**: quando os usuários definem o escopo desejado para o próximo projeto de manutenção, este é registrado no sistema de gestão de mudanças e permanece em estado de espera até que seja iniciada a análise das solicitações.

2. **Em Análise**: este vértice representa a execução das atividades de análise, agrupamento e priorização das solicitações selecionadas pelos usuários para compor a proposta de projeto de manutenção.

3. **Aguardando Planejamento**: após a priorização os projetos permanecerem em estado de espera até que seja realizado o planejamento da execução.

4. **Em Planejamento**: durante o planejamento da execução os gerentes de projetos devem elaborar o plano de execução detalhado e garantir a disponibilidade dos recursos necessários para realização das atividades definidas.

5. **Cancelado**: um projeto pode ser cancelado em diferentes momentos do fluxo de atendimento representado pelo SMRMG da Figura 3. O cancelamento pode acontecer: (1) durante a análise, o agrupamento ou a priorização da solicitação (transição 2-5), (2) durante o planejamento da execução (transição 4-5) ou (3) após o início da implementação (transição 6-5).

6. **Em Implementação**: representa o início da implementação das solicitações agrupadas no projeto.

7. **Suspenso**: este vértice representa as suspensões ocorridas ao longo da implementação dos projetos.

8. **Em Validação**: representa a execução da validação e das demais atividades de responsabilidade da equipe de garantia da qualidade. A transição entre a validação e a implementação (responsável por determinar quando as modificações foram recusadas e encaminhadas para correção sem a necessidade de replanejamento) não foi representada no SMRMG dos projetos de manutenção. Os dados sobre o atendimento dessas solicitações não permitiram identificar a ocorrência de tais transições.

9. **Validado**: após a validação é atribuído o status de validado para o projeto. Caso eventuais inconsistências sejam identificadas e não possam ser corrigidas de imediato, o projeto será automaticamente encaminhado para replanejamento (Vértice 4) e aguardará a avaliação e a atribuição da correção das falhas aos membros da equipe de desenvolvedores. Por outro lado, caso a implementação seja aprovada, a execução do projeto permanecerá neste vértice até o início da implantação.

10. **Em Implantação**: as implementações aprovadas durante a validação devem ser implantadas no ambiente de produção e a aprovação final dos usuários obtida.

11. **Concluído**: ao término da implantação o projeto é considerado concluído.

A execução do algoritmo de *clusterização* sobre os vetores de características dos projetos de manutenção (projetos internos de 2008 e projetos de manutenção de 2009) resultou na identificação de dois *clusters* para o período que antecede a prática de agrupar as solicitações em projetos de manutenção (2008) e dois *clusters* para o período que sucede a adoção das práticas avaliadas (2009). A Tabela 4 apresenta os valores computados para as características desses *clusters* (*em horas úteis*). Nesta tabela, a coluna Coeficiente de Variação (CV) representa a razão entre o desvio padrão e a respectiva média computada para as características analisadas.

Uma análise dos resultados apresentados na Tabela 4 revela um aumento significativo (281%) no número de projetos de manutenção executados em 2009 (62 projetos), comparado com o número de projetos realizados em 2008 (22 projetos). Este resultado ratifica o comprometimento da equipe do DATAPUC em priorizar a execução dos projetos. Além disso, conforme apresenta a Tabela 3, o número de horas alocadas para a execução de projetos de manutenção (independentemente do mês de registro e de conclusão) aumentou 70% (de 2.578 horas em 2008 para 4.383 horas em 2009).

Tabela 4: *Clusters* dos projetos de manutenção

Características	2008				2009			
	Cluster 1 (19 projetos)		*Cluster 2* (3 projetos)		*Cluster 1* (52 projetos)		*Cluster 2* (10 projetos)	
	Média	*CV*	*Média*	*CV*	*Média*	*CV*	*Média*	*CV*
QueueTime	185,00	0,57	771,87	0,21	245,04	0,66	694,91	0,51
WaitTime	12,16	1,01	17,50	0,85	26,85	1,97	11,95	1,11
ServiceTime	172,84	0,61	754,37	0,20	218,19	0,68	682,97	0,51
AnalysisTime	31,51	1,26	7,48	1,60	48,71	1,05	141,61	1,70
PlanningTime	6,34	1,32	5,02	1,65	3,66	2,94	64,61	0,79
ImplementationTime	74,63	1,30	348,27	0,73	50,39	0,80	122,56	0,92
StandByTime	0,00	0,00	0,00	0,00	0,20	7,10	0,00	0,00
ValidationTime	39,34	1,28	377,46	0,83	94,19	1,29	249,70	1,10
DeploymentTime	21,01	0,85	16,14	0,84	21,04	1,10	104,49	1,38

Os *clusters* gerados para os projetos executados em 2008 são discutidos a seguir:

- **Cluster 1 – 2008**: este *cluster* contém 19 projetos executados entre fevereiro a outubro de 2008. Analisando os resultados da Tabela 4 observa-se que a característica *WaitTime* corresponde a 6,6% do tempo total de vida dos projetos (12,16 / 185), representado pela característica *QueueTime*. A característica *AnalysisTime* representa 18,23% do tempo de atendimento (31,51 / 172,84), definido pela característica *ServiceTime*. O planejamento da execução (característica *PlanningTime*) representa 2,88% do tempo de atendimento (4,98 / 172,84) e a característica *ImplementationTime* corresponde a 43,2% (74,63 / 172,84). A duração das atividades de validação (característica *ValidationTime*) representa 22,76% do tempo de atendimento (39,34 / 172,84) e as atividades de implantação correspondem a 12,15% (21,01 / 172,84). Em síntese, este *cluster* se caracteriza como o *cluster* típico dos projetos realizados antes do DATAPUC adotar as diretrizes para sistematizar o agrupamento de solicitações em projetos de manutenção. Apenas um projeto deste *cluster* foi cancelado e a suspensão da implementação não foi registrada.

- **Cluster 2 – 2008**: este *cluster* contém apenas três projetos que são caracterizados pelo longo período de atendimento (característica *ServiceTime*). Para este *cluster* foi identificado um tempo de atendimento superior a 754 horas (contra 172,84 horas computadas para o *Cluster 1* de 2008). Um desses projetos envolveu a unificação dos objetos de banco de dados do sistema SAL para melhorar sua manutenibilidade. O escopo dos demais projetos visava o desenvolvimento de funcionalidades que permitissem que os próprios usuários dos sistemas GRA e GCM pudessem emitir arquivos em formato texto, utilizados para integração contábil. A execução desses projetos não apresentou o cancelamento ou a sinalização da suspensão da implementação.

As características dos *clusters* gerados para os projetos executados em 2009 – apresentadas na Tabela 4 – são discutidas a seguir:

- **Cluster 1 – 2009**: este *cluster* contém 52 projetos. Dado o volume de projetos agrupados ele é considerado o *cluster* típico dos projetos executados em 2009. Observa-se por meio dos resultados apresentados na Tabela 4 que o tempo total de vida desses projetos foi de 245,04 horas. A característica *WaitTime* corresponde a, respectivamente, 10,96% do tempo de vida (26,85 / 245,04). As características *AnalysisTime*, *PlanningTime*, *ImplementationTime* e *ValidationTime* correspondem a 22,32% (48,71 / 218,19), 1,67% (3,66 / 218,19), 23,19% (50,59 / 218,19) e 43,17% (94,19 / 218,19) da característica *ServiceTime*. Durante a execução dos projetos o tempo médio de suspensão foi de 0,20 horas. Entretanto, o coeficiente de variação (coluna CV) foi elevado (7,1). Finalmente, a implantação dos projetos no ambiente de produção (característica *DeploymentTime*) teve uma duração média de 21,04 horas, o que representa 9,64% do tempo de atendimento (21,04 / 218,19).

- **Cluster 2 – 2009**: neste *cluster* foram agrupados 10 projetos. O tempo total de vida dos projetos deste *cluster* foi de 694,91 horas (característica *QueueTime*) e foi distribuído em 1,72% para o tempo de espera (11,95 / 694,91) e 98,28% para o tempo de atendimento (682,97 / 694,91). A característica *AnalysisTime* corresponde a 20,73% da característica *ServiceTime* (141,61 / 682,97) e o planejamento da execução (característica *PlanningTime*) representa 9,46% (64,61 / 682,97). A implementação demandou 17,95% do tempo de atendimento (122,56 / 682,97) e a validação foi responsável por 36,56% (249,70 / 682,97). Finalmente, a implantação no ambiente de produção teve uma participação de 15,30% no tempo de atendimento (104,49 / 682,97). Foi observado que os dez projetos propunham a renovação completa de componentes do sistema em questão. Em resumo, os valores elevados atribuídos a praticamente todas as características denotam um alto nível de complexidade ou uma baixa prioridade atribuída aos projetos desse *cluster*.

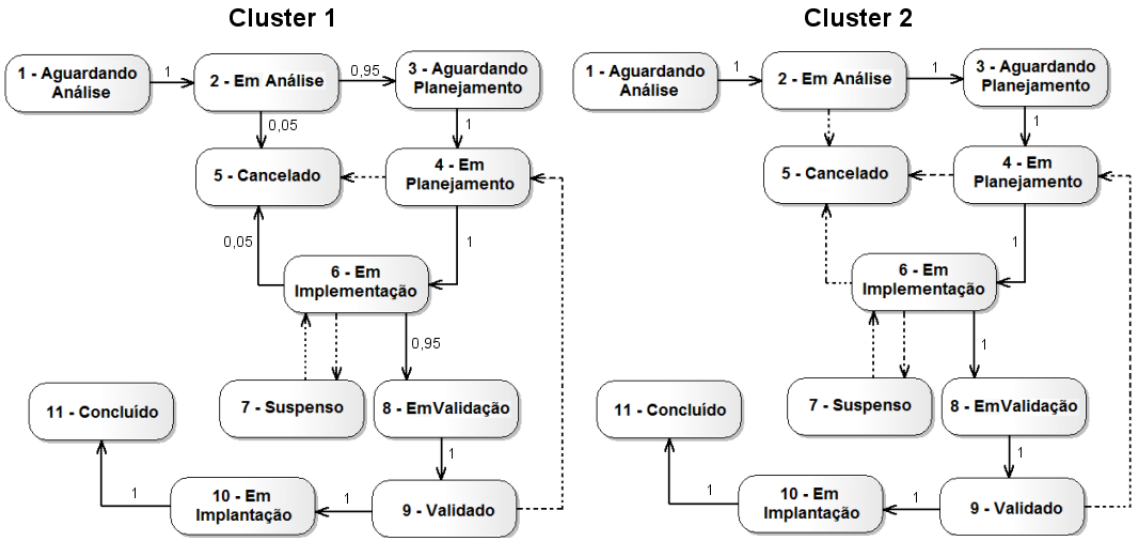

Figura 4: SMRMG dos *clusters* dos projetos de manutenção de 2008

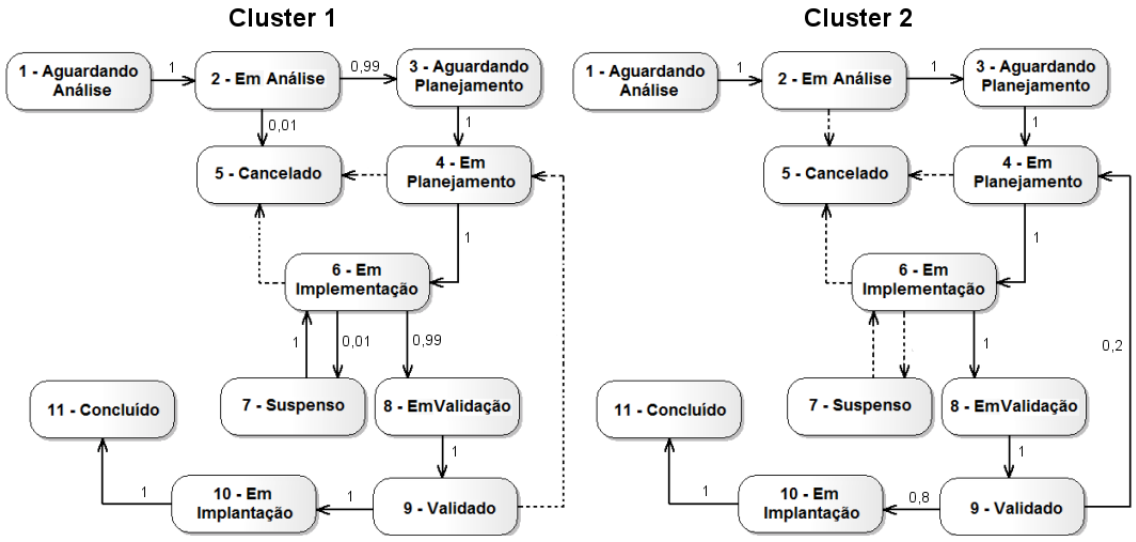

Figura 5: SMRMG dos *clusters* dos projetos de manutenção de 2009

A Figura 4 e a Figura 5 apresentam os SMRMG associados aos *clusters* identificados para os projetos executados em 2008 e em 2009. Basicamente, esses grafos reproduzem o SMRMG do fluxo de atendimento proposto para os projetos de manutenção (Figura 3). Porém, as transições que não ocorreram durante o atendimento foram representadas com linhas pontilhadas. As demais transições foram representadas com linhas sólidas e ponderadas com suas probabilidades de execução. Comparando os SMRMG ilustrados, observa-se as seguintes diferenças:

- Primeiramente, analisando os *clusters* típicos (*Cluster 1* de 2008 e *Cluster 1* de 2009), percebe-se que a probabilidade de cancelamento dos projetos durante a exe-

cução das atividades de análise, agrupamento e priorização (transição 2-5) era de 0,05 em 2008 e passou para 0,01 em 2009. Contudo, o número absoluto de projetos cancelados nessa fase nos dois anos foi de apenas um projeto.

- A segunda diferença apresentada está relacionada com a probabilidade de cancelamento dos projetos típicos após o início da implementação (transição 6-5). No *Cluster 1* de 2008 a probabilidade foi de 0,05 e o *Cluster 1* de 2009 não apresentou esse comportamento.

- A terceira diferença encontrada foi identificada na probabilidade de suspensão dos projetos típicos. No *Cluster 1* de 2009 a probabilidade foi computada como 0,02 (em 2008 ela foi zero). Porém, apenas um projeto foi suspenso em 2009. Portanto, esse único projeto não possui relevância estatística e pode ser considerado um *outlier*.

- Finalmente, comparando os *clusters* atípicos dos dois anos (*Cluster 2* de 2008 e *Cluster 2* de 2009), foi identificada uma divergência na probabilidade de execução da transição 9-4. Essa transição representa os casos onde as atividades de validação encontraram erros que não puderam ser resolvidos de imediato pela equipe de desenvolvimento e com isso, o projeto foi encaminhado para replanejamento. Essa transição ocorreu em apenas um dos projetos agrupados no *Cluster 2* de 2009 e indica que a equipe de garantia da qualidade conseguiu identificar problemas que poderiam comprometer a operacionalidade do software no ambiente de produção.

Os resultados discutidos nesta subseção apontam melhorias na execução dos projetos de manutenção proporcionadas pelas práticas avaliadas. Primeiramente, os projetos que antes continham basicamente solicitações registradas pela equipe de manutenção passaram a contemplar os requisitos de software elucidados junto aos usuários. O aumento do número de projetos realizados é um indício de que o período de um mês permitiu enfileirar e agrupar as solicitações sem comprometer a satisfação dos usuários em relação ao tempo de atendimento. Ao comparar os *clusters* típicos dos dois períodos percebe-se que, embora o tempo total de vida dos projetos tenha sido superior em 2009, a equipe do DATAPUC dedicou mais tempo às atividades de análise das solicitações e de validação das mudanças. Essas atividades são importantes por contribuírem para o entendimento do comportamento e da estrutura do software, para o entendimento do escopo das solicitações, para a mitigação de riscos e para a garantia da qualidade geral dos produtos de software. Por outro lado, em 2009 o tempo de espera foi superior (12,16 em 2008 contra 26,85 em 2009). Porém, o aumento da característica *WaitTime* era esperado, visto que a

execução dos projetos passou a ser regida por uma política de manutenção uniforme, e no caso do DATAPUC o período de registro de solicitações tinha duração predeterminada de duas semanas, o que, consequentemente, implica em um aumento do tempo de espera.

3.3.2 Solicitações Urgentes

A Figura 6 ilustra o SMRMG utilizado para representar o atendimento das solicitações de manutenção urgentes. Os vértices deste SMRMG são descritos a seguir:

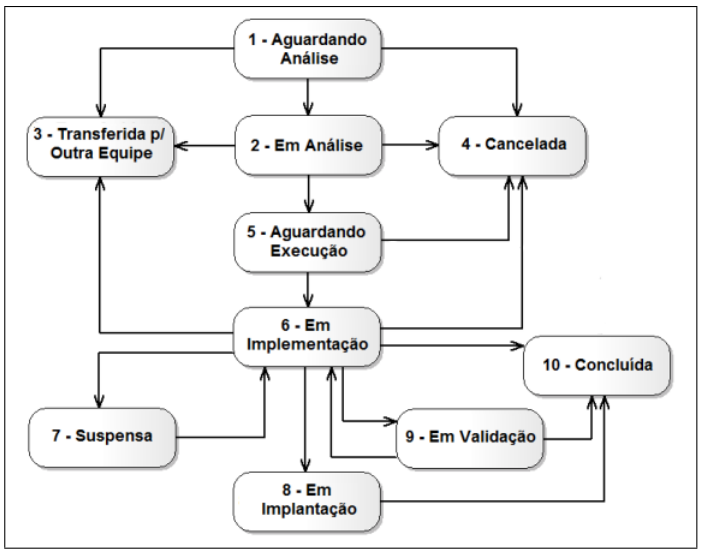

Figura 6: SMRMG das solicitações urgentes

1. **Aguardando Análise**: após o registro, a solicitação permanece em estado de espera, até que seja triada pela equipe de suporte de primeiro nível.

2. **Em Análise**: caso a solicitação seja aprovada durante a triagem a equipe de manutenção deverá analisar a mudança solicitada e determinar os recursos necessários para implementá-la.

3. **Transferida p/ Outra Equipe**: uma solicitação urgente pode ser transferida para outra equipe em três pontos distintos do fluxo de trabalho: (1) quando a triagem inicial da solicitação detectar problemas em seu registro (transição 1-3), (2) após o diagnóstico realizado durante a análise da solicitação (transição 2-3) ou (3) quando a implementação da solicitação envolver a alteração de dois ou mais sistemas (transição 6-3). Quando uma solicitação é transferida para outra equipe, o sistema de gestão de mudanças do DATAPUC gera uma nova solicitação. As

novas solicitações geradas com a transferência das demandas entre as equipes foram consideradas como solicitações independentes.

4. **Cancelada**: durante o atendimento a solicitação pode ser cancelada. O cancelamento pode acontecer quando: (1) a triagem inicial identificar problemas no registro da solicitação (transição 1-4) como, por exemplo, uma solicitação registrada indevidamente como urgente ou o registro duplicado de uma solicitação, (2) quando a análise determinar que a mudança solicitada é inviável (transição 2-4) ou (3) quando a implementação da solicitação não puder ser concluída (transição 6-4).

5. **Aguardando Execução**: após a análise, a solicitação permanece em estado de espera até que sua execução seja atribuída a um dos membros da equipe de desenvolvedores.

6. **Em Implementação**: este estado representa a implementação das mudanças solicitadas. As solicitações urgentes podem envolver diferentes tipos de demandas que variam da implementação de correções e melhorias até a extração de informações e geração de relatórios. Dessa forma, diferentes situações podem ocorrer após a implementação. Por exemplo, as demandas de extração de informação do sistema podem ser concluídas logo após a implementação (transição 6-10). Quando o atendimento da solicitação envolver alguma mudança no código fonte ou em outro artefato do sistema, as alterações precisam ser avaliadas pela equipe de garantia da qualidade e/ou pelos usuários envolvidos antes de serem implantadas no ambiente de produção (transição 6-9). Algumas solicitações podem requerer a realização de atividades específicas durante a implantação, como o treinamento dos usuários ou a implantação da nova versão do sistema em locais geograficamente distribuídos. Nesse caso o atendimento da solicitação será encaminhado para o vértice de Em Implantação (transição 6-8).

7. **Suspensa**: ao longo da implementação das solicitações o atendimento pode ser suspenso.

8. **Em Implantação**: as implementações aprovadas devem ser implantadas no ambiente de produção e a aprovação final junto aos usuários envolvidos obtida.

9. **Em Validação**: após a implementação as mudanças realizadas nos diversos artefatos do sistema devem ser validadas. No atendimento das solicitações urgentes a validação pode ser realizada em alguns casos pela equipe de garantia da qualidade e

em outros pelos próprios usuários envolvidos. Dependendo do tipo da demanda registrado, as solicitações aprovadas podem ser automaticamente consideradas como concluídas (transição 9-10). Solicitações para extração de dados são um exemplo desse cenário. Já a transição 9-6 (Em Validação e Em Implementação) ocorre quando uma modificação é rejeitada e precisa ser corrigida, ou quando as alterações forem aprovadas e a nova versão do sistema requer a execução de atividades específicas para implantação, ou necessita de ações da equipe de desenvolvimento para ser efetivamente concluída.

10. **Concluída**: o atendimento da solicitação é considerado concluído.

A análise de *clusters* foi executada sobre os vetores de características das solicitações urgentes com o objetivo de avaliar como o processo para agrupar solicitações em projetos de manutenção influenciou o atendimento das solicitações desse grupo. Para cada período avaliado foram identificados cinco *clusters*. A Tabela 5 apresenta os *clusters* identificados para as solicitações urgentes atendidas em 2008. Os padrões reconhecidos para os *clusters* de 2008 são discutidos a seguir:

Tabela 5: *Clusters* das solicitações de manutenção urgentes de 2008

Características	**Solicitações Urgentes Atendidas em 2008**									
	Cluster 1 (36 solicitações)		*Cluster 2* (651 solicitações)		*Cluster 3* (30 solicitações)		*Cluster 4* (112 solicitações)		*Cluster 5* (222 solicitações)	
	Média	*CV*	*Média*	*CV*	*Média*	*CV*	*Média*	*CV*	*Média*	*CV*
QueueTime	31,73	1,98	16,79	3,67	29,42	2,06	16,92	1,03	25,82	2,35
WaitTime	2,56	1,72	2,80	1,89	4,08	1,24	2,25	1,76	3,56	1,97
ServiceTime	29,16	2,11	13,98	4,38	25,34	2,33	14,67	1,14	22,26	2,61
AnalysisTime	1,53	2,71	4,46	9,30	11,79	4,41	2,60	1,88	8,33	6,56
ImplementationTime	4,74	1,57	7,02	5,17	12,10	2,73	1,64	2,34	2,28	2,57
StandByTime	15,87	56,92	2,50	27,79	1,45	5,57	0,12	1,25	1,02	4,91
ValidationTime	7,02	2,57	0,00	0,00	0,00	0,00	0,49	8,30	10,63	2,24
DeploymentTime	0,00	0,00	0,00	0,00	0,00	0,00	9,82	1,34	0,00	0,00

- **Cluster 1 – 2008**: *cluster* composto por 36 solicitações sendo: 4 solicitações perfectivas, 5 manutenções corretivas, 24 solicitações de acerto de dados e 3 pedidos de diagnóstico de problemas. Como esperado para as solicitações urgentes, o tempo de espera (característica *WaitTime*) foi relativamente curto (em média 2,56 horas) e corresponde a 8,07% do tempo de vida das solicitações (2,56 / 31,73). As características *AnalysisTime*, *ImplementationTime*, *ValidationTime* foram responsáveis por, respectivamente, 5,25% (1,53 / 29,16), 16,26% (4,74 / 29,16) e 24,07% (7,02 / 29,16) do tempo de atendimento (característica *ServiceTime*). Em nenhuma das

solicitações ocorreu a execução de atividades específicas de implantação e as atividades de validação ocorreram durante o atendimento de todas as solicitações. Além disso, a característica *StandByTime* representa 50,02% do tempo total de vida das solicitações (15,87 / 31,73) e 54,42% do tempo de atendimento (15,87 / 29,16). Dentre as 36 solicitações, 29 tiveram sua implementação suspensa. Analisando os demais *clusters* de 2008 percebe-se que apenas o *Cluster 5* teve a característica *ValidationTime* superior a zero. Porém, a característica *AnalysisTime* do *Cluster 5* foi significativamente superior àquela do *Cluster 1* e por isso as sete solicitações em que não ocorreu a suspensão da implementação foram agrupadas neste *cluster*.

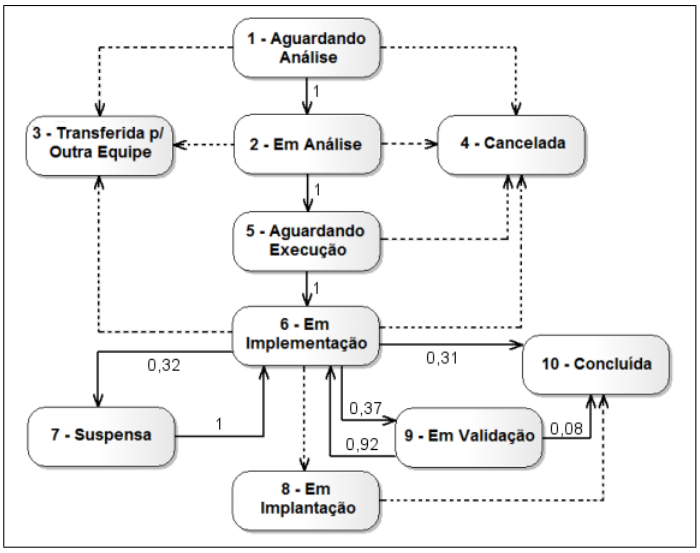

Figura 7: SMRMG do *Cluster 1* das solicitações urgentes de 2008

A Figura 7 apresenta o SMRMG do *Cluster 1* de 2008. Observa-se através desta figura que o atendimento de todas as solicitações atingiu o vértice 6 (Em Implementação). A probabilidade de suspensão da implementação foi de 0,32 e a probabilidade da transição entre a implementação e a validação (transição 6-9) foi de 0,37. Conforme mencionado, todas as solicitações foram validadas. 92% das ocorrências da atividade de validação foram reencaminhadas para a implementação (o retorno não necessariamente significa a identificação de erros) e três solicitações tiveram duas ocorrências de validação. Além disso, a probabilidade de conclusão das solicitações a partir da validação foi de 0,08, ocorrendo em três solicitações. Por fim, a probabilidade de ocorrência da transição entre a implementação e a conclusão (transição 6-10) foi de 0,31, ocorrendo no atendimento de 33 solicitações.

- **Cluster 2 – 2008**: este *cluster* contém 651 solicitações sendo: 106 solicitações perfectivas, 172 manutenções corretivas, 281 solicitações de acerto de dados, 71 pedidos

de diagnóstico de problemas e 21 requisições de execução manual de processos. O tempo médio de vida das solicitações foi de 16,79 horas (o menor entre os *clusters* de 2008). O tempo de espera corresponde a 16,68% do tempo de vida (2,80 / 16,79) e o atendimento foi distribuído na seguinte proporção: 31,90% para a característica *AnalysisTime* (4,46 / 13,98), 50,21% para a característica *ImplementationTime* (7,02 / 13,98) e 17,88% para a característica *StandByTimeTime* (2,50 / 13,98). A suspensão da execução ocorreu em 18 solicitações. Conforme destaca o coeficiente de variação dessa característica, o tempo de suspensão apresentou uma variação elevada, tendo como mínimo e máximo 0,02 e 681,88 horas, respectivamente. Porém, essas solicitações representam apenas 2,76% do conjunto de registros agrupados neste *cluster* (18 / 651). As características *ValidationTime* e *DeploymentTime* foram computadas como zero. Provavelmente, devido à urgência do atendimento (custo de oportunidade elevado), essas solicitações foram implantadas diretamente no ambiente de produção logo após a implementação ou os artefatos construídos não foram incluídos na versão de distribuição do sistema. Em resumo, este *cluster* representa o fluxo de atendimento típico das solicitações urgentes de 2008.

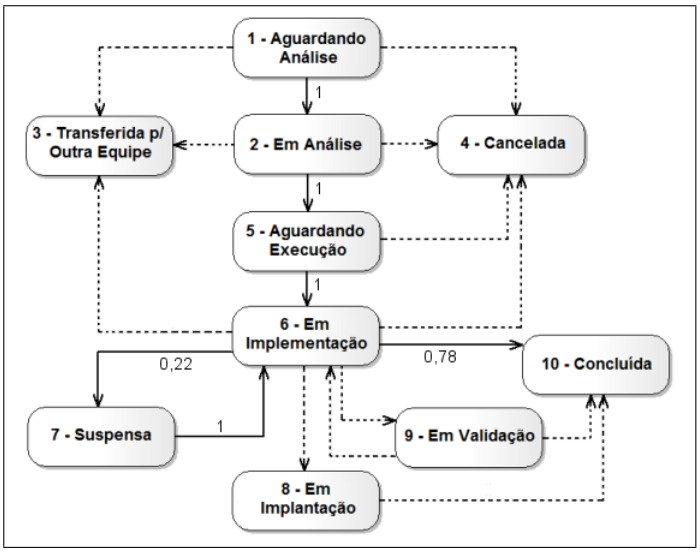

Figura 8: SMRMG do *Cluster 2* das solicitações urgentes de 2008

A Figura 8 ilustra o SMRMG deste *cluster*. Conforme mostra a figura, o atendimento de todas as solicitações atingiram o vértice 6 (Em Implementação). A probabilidade de suspensão do atendimento foi calculada como 0,22 e a probabilidade de conclusão foi computada como 0,78.

- **Cluster 3 – 2008**: *cluster* formado por 30 solicitações sendo 2 solicitações perfectivas, 4 manutenções corretivas, 19 solicitações de acerto de dados e 5 pedidos

de diagnóstico de problemas. A característica *WaitTime* representa 13,87% da característica *QueueTime* (4,08 / 29,42). A característica *AnalysisTime* corresponde a 46,53% do tempo de atendimento (11,79 / 25,34). A duração da implementação representa 47,75% da característica *ServiceTime* (12,10 / 25,34) e a suspensão da implementação corresponde a 5,72% (1, 45 / 25,34). As características *ValidationTime* e *DeploymentTime* foram computadas como zero.

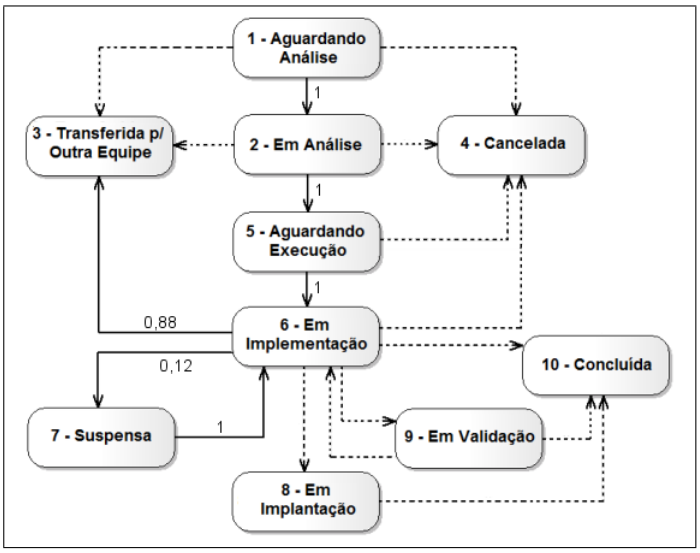

Figura 9: SMRMG do *Cluster 3* das solicitações urgentes de 2008

O SMRMG da Figura 9 apresenta a probabilidade de ocorrência das transições presentes no atendimento das solicitações deste *cluster*. Observa-se que, assim como nos *clusters* anteriores, o atendimento de todas as solicitações atingiu o vértice 6 (Em Implementação). Todas as solicitações envolveram a manutenção de mais de um sistema e por isso, todas foram transferidas para outras equipes.

- **Cluster 4 – 2008**: *cluster* composto por 112 solicitações sendo 34 solicitações perfectivas, 67 manutenções corretivas, 6 solicitações de acerto de dados e 5 pedidos de diagnóstico de problemas. O tempo de espera corresponde a 13,30% do tempo de vida das solicitações (2,25 / 16,92). A característica *AnalysisTime* corresponde a 17,72% do tempo de atendimento (2,60 / 14,67) e a implementação das mudanças representa 11,18% (1,64 / 14,67). A duração média das atividades de validação foi de 0,49 horas. Considerando as oito solicitações validadas, foram identificados como valores mínimo e máximo 0,02 e 41,95 horas, respectivamente. A principal diferença deste *cluster* para os demais está na característica *DeploymentTime*, responsável por 66,94% do tempo de atendimento das solicitações (9,82 / 14,67).

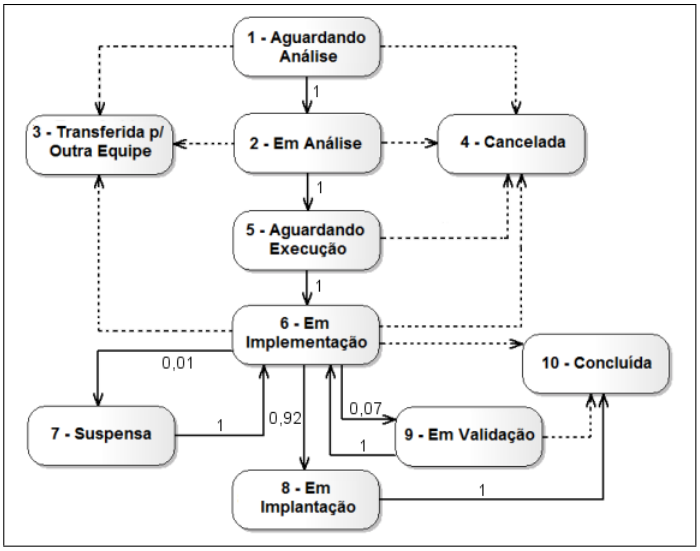

Figura 10: SMRMG do *Cluster 4* das solicitações urgentes de 2008

A Figura 10 apresenta o SMRMG deste *cluster*. Todas as solicitações tiveram a implementação iniciada e a probabilidade de suspensão foi de apenas 0,01. A probabilidade de ocorrência de validação foi de 0,07 (essa transição esteve presente no atendimento de oito solicitações) e em todas o atendimento foi encaminhado para a equipe responsável pela implementação. Conforme mencionado, a principal diferença desse *cluster* para os demais é a ocorrência da transição entre a implementação e a implantação. Essa transição ocorreu durante o atendimento de todas as solicitações agrupadas e sua probabilidade foi computada como 0,92.

- **Cluster 5 – 2008**: este *cluster* contém 222 solicitações sendo 42 solicitações perfectivas, 66 manutenções corretivas, 66 solicitações de acerto de dados, 39 pedidos de diagnóstico de problemas e 9 requisições de execução manual de processos. O tempo médio de vida dessas solicitações foi superior a todos os demais *clusters* identificados para as solicitações urgentes atendidas em 2008 (25,82 horas). A característica *WaitTime* representa 13,79% do tempo de vida das solicitações (3,566 / 25,82). O tempo de atendimento foi distribuído entre as características *AnalysisTime*, *ImplementationTime* e *ValidationTime* nas seguintes proporções: 37,42% (8,33 / 22,26), 10,24% (2,28 / 22,26) e 47,75% (10,63 / 22,26). Além disso, 159 solicitações (71%) tiveram a implementação iniciada. O aumento no tempo de vida foi decorrente do tempo de análise e de validação relativamente elevados. Uma possível justificativa para os valores dessas características é a probabilidade potencial de comprometer o funcionamento do sistema. Dessa forma, as solicitações desse *cluster* são classificadas como solicitações urgentes propensas a erro.

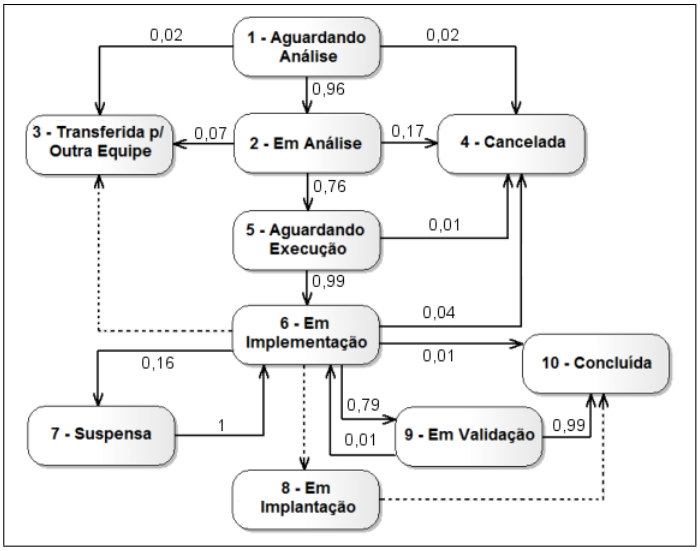

Figura 11: SMRMG do *Cluster 5* das solicitações urgentes de 2008

O SMRMG desse *cluster* é mostrado na Figura 11. Este foi o único *cluster* de 2008 onde ocorreu o cancelamento de solicitações antes e depois do início da execução. A triagem das solicitações identificou problemas no registro de 4% das solicitações (dez registros). Dentre essas solicitações, cinco foram transferidas para outra equipe e cinco foram canceladas. Considerando os 96% das solicitações que foram encaminhadas para a análise (212 solicitações), 7% ou quinze solicitações foram transferidas para outra equipe, 17% (36 solicitações) foram canceladas – provavelmente devido à inviabilidade de implementação ou problemas no registro que não foram identificados durante a triagem – e 76% (161 solicitações) foram analisadas e encaminhadas para implementação. Entretanto, enquanto as solicitações aguardavam o início da implementação, 1% (duas solicitações) foram canceladas. Após o início da execução, a probabilidade de suspensão foi de 0,16 e a probabilidade de cancelamento foi de 0,04 (sete solicitações). A probabilidade de conclusão a partir da implementacão foi de 0,01 (duas solicitações). A transição entre a implementação e a validação teve probabilidade computada como 0,79 (152 ocorrências). Por fim, apenas 1% das ocorrências de validação retornaram para a implementação (duas solicitações) e 150 solicitações foram concluídas após a validação.

A Tabela 6 descreve os *clusters* das solicitações urgentes atendidas em 2009. Os padrões reconhecidos para cada *cluster* são discutidos a seguir:

Tabela 6: *Clusters* das solicitações de manutenção urgentes de 2009

Características	Solicitações Urgentes Atendidas em 2009									
	Cluster 1 (201 solicitações)		*Cluster 2* (212 solicitações)		*Cluster 3* (96 solicitações)		*Cluster 4* (61 solicitações)		*Cluster 5* (383 solicitações)	
	Média	*CV*	*Média*	*CV*	*Média*	*CV*	*Média*	*CV*	*Média*	*CV*
QueueTime	20,87	1,23	26,51	1,73	10,44	1,84	22,15	1,02	12,82	1,95
WaitTime	3,28	1,48	3,05	1,69	3,06	1,55	3,43	1,28	3,22	1,72
ServiceTime	17,59	1,43	23,46	1,91	7,38	2,54	18,72	1,14	9,60	2,48
AnalysisTime	2,54	2,65	3,23	3,79	4,72	2,85	3,07	1,80	2,45	3,66
ImplementationTime	4,05	2,41	4,27	4,15	0,98	6,06	3,80	2,12	3,94	3,64
StandByTime	0,24	2,06	2,07	12,72	1,52	11,92	6,04	11,67	3,21	15,17
ValidationTime	3,98	3,47	13,89	1,51	0,16	9,80	5,81	1,55	0,00	0,00
DeploymentTime	6,78	1,30	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

- **Cluster 1 – 2009**: este *cluster* contém 201 solicitações sendo 81 solicitações perfectivas, 78 manutenções corretivas, 20 solicitações de acerto de dados, 17 pedidos de diagnóstico de problemas e 5 requisições de execução manual de processos. A característica *WaitTime* representa 15,72% do tempo de vida das solicitações (3,28 / 20,87). As características *AnalysisTime*, *ImplementationTime* e *ValidationTime* correspondem a respectivamente 14,44% (2,57 / 17,59), 23,02% (4,05 / 17,59) e 22,63% (3,98 / 17,59) da característica *ServiceTime*. A característica *ValidationTime* apresentou valores entre 0,02 e 117,80 horas, considerando as 63 solicitações validadas. A característica *DeploymentTime* foi computada como 6,78 horas ou 38,54% do tempo de atendimento (6,78 / 17,59).

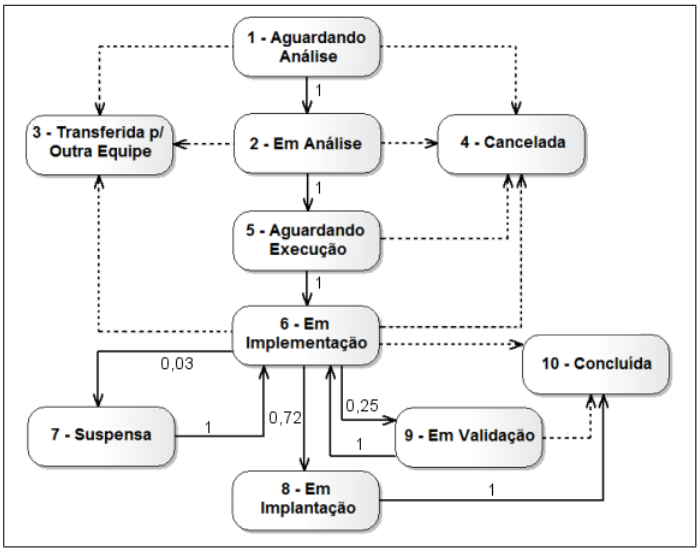

Figura 12: SMRMG do *Cluster 1* das solicitações urgentes de 2009

A Figura 12 ilustra o SMRMG deste *cluster*. Observa-se que o atendimento de todas as solicitações atingiu o vértice 6 (Em Implementação). A probabilidade de

suspensão do atendimento foi de 0,03 (ocorreu em seis solicitações sendo que três foram suspensas duas vezes). A probabilidade de execução das atividades de validação foi de 0,25, ocorrendo 69 vezes para 63 solicitações distintas (seis solicitações foram validadas duas vezes). Além disso, todas ocorrências de validação retornaram para implementação. A probabilidade de ocorrência da transição entre a equipe responsável pela implementação e a implantação (transição 6-8) foi de 0,72. Essa transição ocorreu durante o atendimento das 201 solicitações desse *cluster*.

- **Cluster 2 – 2009**: *cluster* formado por 212 solicitações. Foram agrupadas neste *cluster* 65 solicitações perfectivas, 80 manutenções corretivas, 50 solicitações de acerto de dados, 13 pedidos de diagnóstico de problemas e 4 requisições de execução manual de processos. A característica *QueueTime* foi a maior entre os *clusters* definidos para as solicitações urgentes atendidas em 2009 (26,51 horas). O tempo de espera das solicitações representa 11,51% do tempo de vida (3,05 / 26,51). A análise das solicitações e a implementação das mudanças correspondem a respectivamente 13,77% (3,23 / 23,46) e 18,20% (4,27 / 23,46) do tempo de atendimento. As características das solicitações deste *cluster* denotam que as mudanças realizadas eram suscetíveis a erro. A característica *ValidationTime* foi a maior computada para todos os *clusters* das solicitações urgentes (13,89 horas ou 59,21%). Essa comparação é válida tanto para os *clusters* de 2008 quanto para os *clusters* de 2009. Além disso, 32 solicitações tiveram sua implementação suspensa e o tempo médio de suspensão foi de 2,07 horas. Considerando essas 32 solicitações, os valores mínimo e máximo da característica *StandByTime* foram respectivamente 0,05 e 155,32 horas.

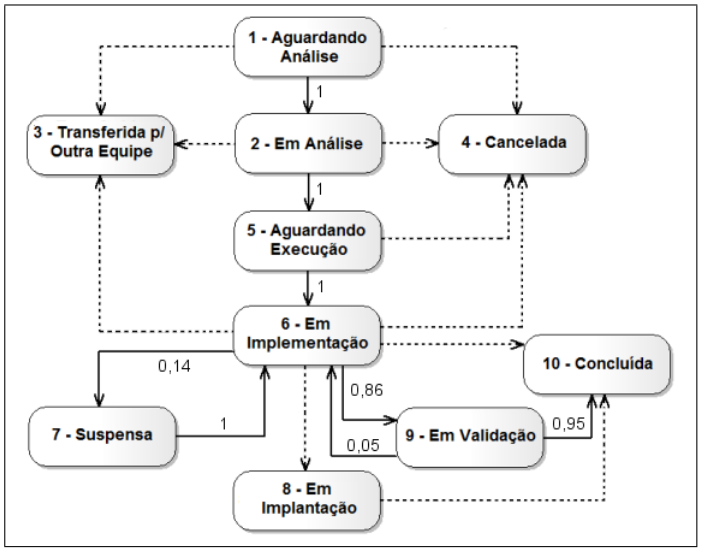

Figura 13: SMRMG do *Cluster 2* das solicitações urgentes de 2009

A Figura 13 apresenta o SMRMG deste *cluster*. Similar ao identificado no *Cluster 1 de 2009*, todas as solicitações foram encaminhadas para implementação (vértice 6). A probabilidade de suspensão da implementação foi de 0,14. Não foram executadas atividades específicas de implantação durante o atendimento dessas solicitações. Além disso, todas foram encaminhadas para validação e 5% das ocorrências (11 solicitações) retornaram para serem corrigidas pela equipe de desenvolvimento.

- **Cluster 3 – 2009**: *cluster* composto por 96 solicitações. Dentre as solicitações deste *cluster* existem 5 solicitações perfectivas, 8 manutenções corretivas, 31 solicitações de acerto de dados, 51 pedidos de diagnóstico de problemas e uma requisição de execução manual de processo. O tempo de vida das solicitações foi o menor identificado entre todos os *clusters* das solicitações urgentes de 2008 e de 2009 (10,44 horas). O tempo de espera dessas solicitações foi computado como 3,06 horas ou 29,31% do tempo total de vida (3,06 / 10,44). A característica *AnalysisTime* corresponde a 63,97% do tempo de atendimento (4,73 / 7,38), a característica *ImplementationTime* representa 13,29% (0,98 / 7,38) e a característica *ValidationTime* corresponde a 2,2% (0,16 / 7,38). As suspensões ocorreram em seis solicitações, e a duração média foi de 1,52 horas ou 20,53% do tempo de atendimento (1,52 / 7,38).

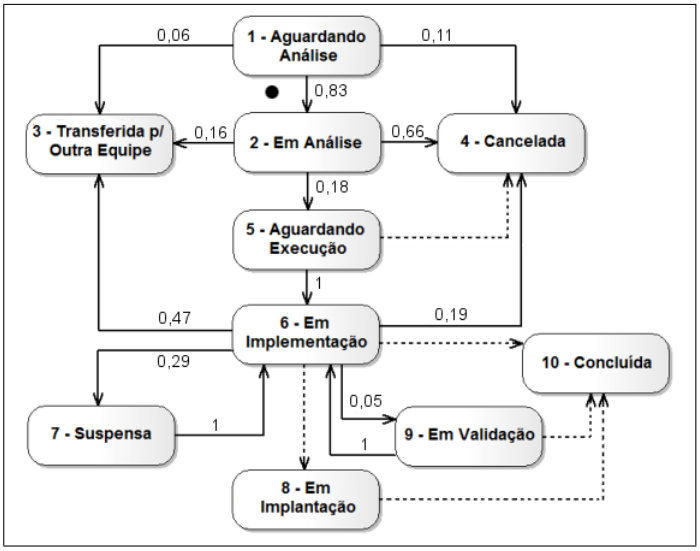

Figura 14: SMRMG do *Cluster 3* das solicitações urgentes de 2009

O SMRMG deste *cluster* é ilustrado na Figura 14. A triagem inicial resultou no cancelamento de 11% das solicitações (dez solicitações) e na transferência de 6% das solicitações para outras equipes (seis solicitações). Dos 83% das solicitações encaminhadas para análise (80 solicitações), 16% (13 solicitações) foram transferidas para outras equipes, 66% foram canceladas (53 solicitações) e 18% (14 solicitações) foram

encaminhadas para implementação. A probabilidade de suspensão das solicitações foi de 0,29 (30 solicitações). A probabilidade de encaminhamento para validação foi de 0,05 (uma solicitação). As solicitações deste *cluster* que tiveram a implementação iniciada foram canceladas – probabilidade de 0,19 (quatro solicitações) – ou transferidas para outras equipes – probabilidade de 0,47 (dez solicitações).

- **Cluster 4 – 2009**: *cluster* menos representativo de 2009 em termos de quantidade de solicitações. Contém 61 solicitações sendo 3 solicitações perfectivas, 5 manutenções corretivas, 47 solicitações de acerto de dados, 5 pedidos de diagnóstico de problemas e uma requisição de execução manual de processo. Este *cluster* apresentou o maior tempo de vida (22,15 horas). As características *AnalysisTime* e *ImplementationTime* apresentaram duração semelhante, correspondendo a, respectivamente, 16,40% (3,07 / 18,72) e 20,30% (3,8 / 18,72) do tempo de atendimento. Já a característica *ValidationTime* corresponde a 32,26% da característica *ServiceTime* (5,81 / 18,72). As suspensões ocorreram em 52 solicitações. O tempo médio determinado para a característica *StandByTime* foi de 6,04 horas ou 31,04% do tempo de atendimento. Considerando apenas as solicitações em que a suspensão ocorreu, essa característica apresentou como valores mínimo e máximo 0,02 e 61,98 horas, respectivamente.

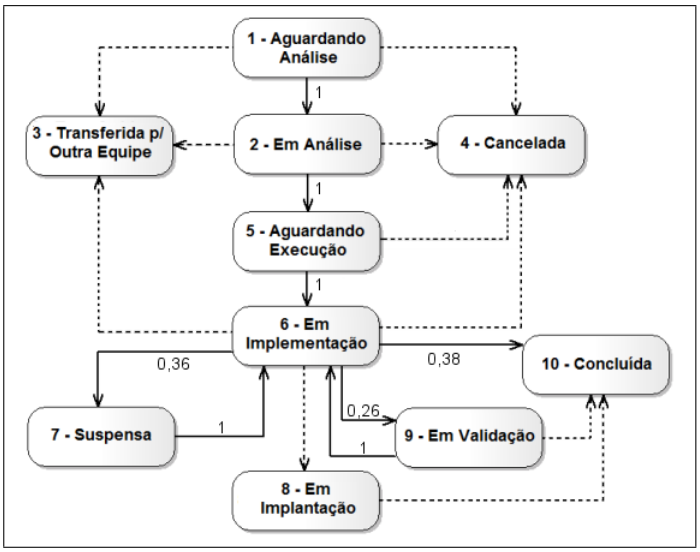

Figura 15: SMRMG do *Cluster 4* das solicitações urgentes de 2009

A Figura 15 mostra o SMRMG deste *cluster*. Observa-se que todas as solicitações foram encaminhadas para implementação. A probabilidade de suspensão da execução foi de 0,36 (ocorreram 66 suspensões, sendo que 52 solicitações foram suspensas). A probabilidade de incidência das atividades de validação foi de 0,26 – 53 solicitações,

sendo que uma teve duas validações. Por fim, todas solicitações foram concluídas a partir a implementação.

- **Cluster 5 – 2009**: *cluster* resultante do agrupamento de 383 solicitações. Das solicitações agrupadas neste *cluster*, 43 são solicitações perfectivas, 67 manutenções corretivas, 184 solicitações de acerto de dados, 60 pedidos de diagnóstico de problemas e 29 requisições de execução manual de processos. Este *cluster* é considerado o *cluster* típico das solicitações urgentes atendidas em 2009. A característica *WaitTime* representa 25,12% do tempo total de vida das solicitações (3,22 / 12,82). A característica *AnalysisTime* representa 25,52% do tempo de atendimento (2,45 / 9,60) e a característica *ImplementationTime* corresponde a 41,04% (3,94 / 9,60). As suspensões da implementação ocorreram em 133 solicitações e representam 33,44% do tempo de atendimento (3,21 / 9,60). Considerando as 133 solicitações em que ocorreu uma suspensão, foram idenficados como valores mínimo e máximo 0,02 e 181,63 horas, respectivamente. Este *cluster* foi o único das solicitações urgentes de 2009 que tanto a característica *ValidationTime* quanto a característica *DeploymentTime* foram computadas como zero. Isso significa que, possivelmente, as demandas solicitadas possuíam um custo de oportunidade elevado (o que implicou na implantação direta no ambiente de produção), ou as solicitações envolveram a construção de artefatos que não foram inseridos na liberação da nova versão do software, tais como relatórios ou scripts de correção de dados.

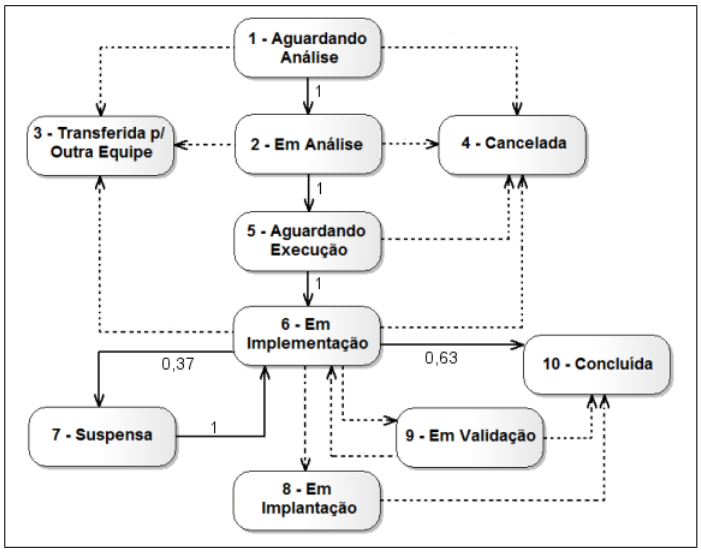

Figura 16: SMRMG do *Cluster 5* das solicitações urgentes de 2009

O SMRMG mostrado na Figura 16 apresenta a probabilidade de ocorrência das transições durante o atendimento das solicitações deste *cluster*. Todas as solicita-

ções foram encaminhadas para implementação e a probabilidade de suspensão do atendimento foi de 0,37. Corroborando o padrão apontado pelas características deste *cluster*, o fluxo de atendimento das solicitações não apresentou a execução das atividades de validação ou de implantação e todas as solicitações foram concluídas logo após a implementação.

Comparando as características e os SMRMG dos *clusters* de 2008 com as características e os SMRMG dos *clusters* de 2009 observa-se que:

- Os *clusters* 1 de 2008 e 4 de 2009 representam basicamente as solicitações que permaneceram suspensas por um longo tempo e apresentaram atividades de validação. As solicitações foram encaminhadas para serem validadas pela equipe de garantia da qualidade e/ou pelos usuários envolvidos, devolvidas para a equipe responsável pela implementação e posteriormente concluídas a partir da implementação. Excepcionalmente, 8% das ocorrências de validação do *Cluster 1* de 2008 (três solicitações) resultaram no encerramento do atendimento sem retornar para a implementação.

- O *Cluster 2* de 2008 apresentou o mesmo fluxo de atendimento do *Cluster 5* de 2009. Embora a probabilidade de suspensão da implementação tenha sido menor no *cluster* de 2008, ambos representam o *cluster* típico de cada ano. Dessa forma, é possível afirmar que a iniciativa de agrupar as solicitações em projetos de manutenção não alterou o comportamento típico da execução das solicitações urgentes. Contudo, é importante ressaltar que em 2009 o *cluster típico passou a ser menos representativo (em relação ao número de solicitações agrupadas)*.

- O SMRMG do *Cluster 3* de 2008 evidencia que todas as solicitações agrupadas nesse *cluster* foram transferidas para outras equipes após o início da implementação. Dentre os *clusters* de 2009 não foi identificado nenhum *cluster* onde todas as solicitações se caracterizam exclusivamente por esse comportamento. Entretanto, no *Cluster 3* de 2009 foram identificadas 13 solicitações encaminhadas para outras equipes após a análise e nove solicitações transferidas após o início da implementação. O *Cluster 3* de 2009 representa as solicitações encerradas sem serem concluídas e abrange o padrão reconhecido para o *Cluster 3* de 2008.

- O *Cluster 4* de 2008 se caracteriza pela incidência da execução de atividades específicas de implantação. Esse padrão também foi identificado no *Cluster 1* de 2009. Todavia, existem diferenças entre os dois *clusters*. A probabilidade da transição

entre a implementação e a validação (transição 6-9) é a principal delas. No *cluster* de 2008 a probabilidade dessa transição foi de 0,07 (oito solicitações) e no *cluster* de 2009 foi de 0,21 (63 solicitações). Além disso, os valores mais elevados para as características *ImplementationTime* e *ValidationTime* do *cluster* de 2009 permitem induzir que as solicitações desse *cluster* envolveram mudanças mais complexas.

- O padrão reconhecido para o *Cluster 5* de 2008 não foi encontrado em sua totalidade em um único *cluster* de 2009. Apesar das transições 1-3 (Aguardando Análise e Transferida p/ Outra Equipe), 1-4 (Aguardando Análise e Cancelada), 2-3 (Em Análise e Transferida p/ Outra Equipe) e 2-4 (Em Análise e Cancelada) existirem no *Cluster 3* de 2009, elas apresentaram probabilidades relativamente inferiores em 2008, o que inverte o significado da presença dessas transições. Além disso, o *cluster* de 2009 não representa solicitações suscetíveis a erro (característica atribuída às solicitações do *Cluster 2* de 2009).

- Não foi identificado nenhum *cluster* de 2008 com o padrão reconhecido para o *Cluster 2* de 2009. As solicitações agrupadas neste *cluster* são caracterizadas como mudanças suscetíveis a erro, que foram implementadas, validadas e posteriormente concluídas sem a incidência de atividades específicas de implantação. Percebe-se que a equipe de garantia da qualidade e/ou os usuários identificaram problemas em 5% das implementações submetidas a validação (onze solicitações).

Considerando a média das características dos cinco *clusters* de cada período, observa-se que em 2009 houve uma redução de 5,58 horas no tempo de vida (24,14 - 18,56) e uma redução de 5,73 horas no tempo de atendimento (21,08 - 15,35). Além disso, em 2009 a duração média das atividades de implementação manteve-se sempre próxima a quatro horas. Provavelmente, esses ganhos ocorreram devido ao fato de o agrupamento das solicitações em projetos de manutenção ter contribuído para a otimização da triagem e da análise das solicitações urgentes. Finalmente, foi observado um pequeno aumento no tempo dedicado às mudanças mais complexas ou suscetíveis a erro. No entanto, esse aumento foi devido ao maior tempo destinado às atividades de validação das mudanças, o que pode ser considerando um ponto positivo (visto que atividades de validação contribuem para melhorar a qualidade final dos produtos de software).

3.3.3 PU e Plantões

As solicitações dos grupos PU e plantões eram executadas com o mesmo fluxo de atendimento das solicitações urgentes. Entretanto, as premissas de execução dessas solicitações eram distintas, conforme discutido na Seção 3.2.6. Além disso, os dados referentes à execução dos PU não eram registrados adequadamente no sistema de gestão de mudanças do DATAPUC. A sinalização das transições existentes no atendimento dos plantões também apresentaram irregularidades, colocando em dúvida a acurácia dos dados. Dessa forma, devido à inviabilidade de se comparar os PU com os plantões ou com as solicitações urgentes, não foi realizada a *clusterização* desses dois grupos de solicitações de manutenção. A seguir descreve-se a amostra de dados dos PU e dos plantões:

- **2008 – PU**: grupo formado por 666 solicitações sendo 343 solicitações perfectivas, 109 manutenções corretivas, 74 solicitações de acerto de dados, 138 pedidos de diagnóstico de problemas e duas requisições de execução manual de processos. As solicitações perfectivas são predominantes neste grupo, sendo responsáveis por 51,50% das demandas (343 / 666).

- **2009 – Plantões**: grupo composto por 932 solicitações. Dentre as solicitações deste grupo existem 517 solicitações perfectivas, 142 manutenções corretivas, 147 solicitações de acerto de dados, 114 pedidos de diagnóstico de problemas e 12 requisições de execução manual de processos. Assim como no grupo PU, as solicitações perfectivas representam a maior parte das demandas, sendo responsáveis por 55,47% das solicitações (517 / 932). Analisando a data de início e de término da implementação e da conclusão do atendimento, foi identificado que 87,23% das solicitações (813 / 932) tiveram a implementação concluída no dia de realização do plantão. Entretanto, apenas 16,09% dessas solicitações (150 / 932) foram efetivamente concluídas no mesmo dia.

A predominância de solicitações perfectivas nos grupos PU e plantões indica que as horas dedicadas ao atendimento das solicitações desses grupos eram destinadas principalmente às demandas relacionadas com a mudança de requisitos funcionais e não funcionais dos sistemas e – no caso dos plantões – que não podiam aguardar o início do próximo projeto de manutenção. Os dados das solicitações do grupo plantões mostram que apenas 16,09% das solicitações foram concluídas no dia específico destinado ao atendimento dessas solicitações. O número de solicitações perfectivas atendidas pode ser uma das causas desse comportamento, visto que as atividades de análise, implementação e principalmente

validação dessas solicitações tendem a demandar um cuidado maior e consequentemente um volume maior de horas de trabalho.

3.4 Considerações Finais

Com base nos resultados discutidos neste capítulo é possível afirmar que o processo para agrupar solicitações de manutenção em projetos de manutenção está direta ou indiretamente relacionado com os seguintes benefícios:

- A equipe do DATAPUC conseguiu incorporar com sucesso o agrupamento de solicitações em projetos de manutenção. Os projetos realizados passaram a contemplar solicitações que representam as necessidades dos usuários e não apenas solicitações internas motivadas por necessidades da própria equipe de manutenção. A liberação de novas versões dos sistemas do DATAPUC passou a ser realizada em geral (considerando o *cluster* típico), a cada 30 dias úteis. Além disso, os projetos incorporaram a atividade de planejamento da execução, a análise das solicitações passou a representar 22% do tempo de atendimento (contra 4,20% em 2008) e a duração das atividades de validação teve um aumento na sua participação sobre o tempo de atendimento de 17% em 2008 para 31% em 2009.

- Após a implantação da iniciativa avaliada houve uma redução do número de registros de solicitações urgentes com custo de oportunidade elevado (prioridade alta) e concluídas logo após a implementação. Em 2008 essas solicitações representaram 61,94% da amostra (651 / 1051) e em 2009 apenas 40,19% (383 / 953). A redução do número dessas solicitações é importante uma vez que as atividades de validação não foram executadas, aumentando a possibilidade de gerar novos erros.

- A equipe de garantia da qualidade dedicou mais horas nas atividades de validação das solicitações urgentes. Dentre os *clusters* de 2008, os *clusters* 1, 4 e 5 apresentaram a ocorrência dessas atividades. Foram agrupadas nesses *clusters* 370 solicitações, sendo que 196 dessas solicitações foram submetidas à validação. Já em 2009, apenas no *Cluster 5* não ocorreu a validação das solicitações. Nos demais *clusters* de 2009 foram agrupadas de 570 solicitações, sendo que 339 foram validadas. Isso representa um aumento de 72,95% (339 / 196) no número de solicitações urgentes atendidas com a ocorrência das atividades de validação. O *Cluster 2* de 2009 – *cluster* exclusivo de 2009 que representa as solicitações propensas a erro que fo-

ram em sua totalidade submetidas à validação – também representa outra evidência da contribuição do agrupamento das solicitações para a melhoria da qualidade das liberações. Em termos gerais, a duração das atividades de validação das solicitações urgentes foi de 3,63 horas em 2008 e 4,77 horas em 2009. Isso representa um aumento de 31,5%.

- A média das características dos cinco *clusters* das solicitações urgentes de 2008 e de 2009 aponta uma redução de 5,58 horas (23,11%) no tempo de vida dessas solicitações e uma redução de 5,73 horas ou 27,19% no tempo de atendimento.

Os pontos negativos ocasionados direta ou indiretamente pela implantação das práticas avaliadas se resumem em:

- O número absoluto de solicitações registradas como urgentes apresentou uma redução de 9,95% (968 / 1075). Embora essa redução não seja um ponto negativo, houve um aumento de 39,93% no número de horas trabalhadas (3.513 / 2.547). Considerando apenas as solicitações corretivas, o número absoluto de solicitações registradas apresentou uma redução de 25,23% (240 / 321). Entretanto, o número de horas trabalhadas aumentou 22,92% (980,26 / 797,47).

- O tempo de espera das solicitações urgentes teve um aumento médio de 1,93 hora ou 68,19% (4,76 / 2,83). Entretanto, é importante salientar que a capacidade produtiva da equipe de manutenção em 2009 foi compartilhada com a execução dos projetos de manutenção e dos plantões.

- A suspensão da implementação ocorreu com maior frequência em 2009. A probabilidade de ocorrência da transição entre a implementação e a suspensão em 2009 foi, em todos os *clusters*, superior à probabilidade computada para o *cluster* similar de 2008. Entretanto, embora o número de suspensões tenha sido superior, a duração das suspensões foi em média 1,58 hora inferior em 2009 ou 62,53% (2,62 / 4,19).

Capítulo 4

Processo para Agrupamento de Solicitações de Manutenção de Software (PASM)

Este capítulo apresenta a formalização do conjunto de práticas proposto por Malta *et al.* (2009) – e discutido na Subseção 2.5.2 – no processo denominado Processo para Agrupamento de Solicitações de Manutenção de Software (PASM). Além disso, este capítulo apresenta uma especificação detalhada do processo proposto, elaborada com o intuito de complementar as diretrizes e práticas originalmente propostas por Malta *et al.* (2009). O restante deste capítulo está organizado conforme descrito a seguir. A Seção 4.1 define o PASM, suas premissas e seus objetivos. A Seção 4.2 discute o ciclo de projetos de manutenção proposto pelo PASM. A Seção 4.3 apresenta as fases do PASM e, por fim, a Seção 4.4 apresenta as considerações finais deste capítulo.

4.1 Definições

O Processo para Agrupamento de Solicitações de Manutenção de Software (PASM) propõe um conjunto de diretrizes com o objetivo de: (1) apoiar a organização das solicitações de manutenção durante o planejamento da liberação de novas versões, (2) contribuir para a execução das melhores práticas de Engenharia de Software ao longo do atendimento das solicitações de manutenção e (3) possibilitar que as empresas consumidoras e prestadoras de serviços de manutenção de software possam se beneficiar da economia de escala que se manifesta quando pequenas solicitações são agrupadas em projetos maio-

res. O PASM está fundamentado em duas premissas: (1) o registro das solicitações pelos usuários deve ser feito de forma disciplinada e organizada para viabilizar a elaboração de propostas de projetos de manutenção e (2) as solicitações devem ser agrupadas em projetos para melhorar a eficiência das atividades de planejamento e de controle da execução, para reduzir o custo da manutenção e para melhorar a qualidade das liberações. Ao fomentar a realização de projetos, o PASM contribui para reduzir a quantidade de manutenções classificadas como urgentes (incluindo as manutenções corretivas) e para assegurar a execução das atividades de análise, especificação e garantia da qualidade, comumente não executadas de forma eficiente quando uma solicitação de manutenção é atendida individualmente.

A proposta de processo formalizada nesta dissertação foi concebida utilizando as idéias centrais do conjunto de práticas proposto por Malta *et al.* (2009). Isso envolve, as premissas, os objetivos, o foco na priorização das solicitações do ponto de vista de todos os envolvidos, a preocupação em balancear o retorno de investimento de cada solicitação para os usuários com a capacidade produtiva da equipe e as atividades realizadas durante a elaboração e aprovação das propostas de projetos de manutenção.

As melhorias realizadas durante a elaboração da especificação proposta nesta dissertação permitiram estruturar o PASM como um processo simples, porém abrangente e capaz de ser utilizado como referência para a gestão e o atendimento eficiente dos diversos tipos de solicitações de manutenção de software. O conjunto de diretrizes e melhores práticas definido na revisão da especificação do PASM foi fundamentado na avaliação de importantes trabalhos descritos na literatura. Dentre as contribuições utilizadas encontram-se a especificação do processo de manutenção das normas ISO/IEC 12207 (ISO/IEC, 2008) e ISO/IEC 14764 (ISO/IEC, 2006), as práticas de tratamento da documentação e de rejuvenescimento de software propostas pela metodologia MANTEMA (POLO *et al.*, 1999a), as práticas propostas pelos modelos de maturidade S3M (APRIL *et al.*, 2005) e CM^3 (KAJKO-MATTSSON, 2002b) e as evidências sobre os benefícios proporcionados pelo agrupamento de solicitações de manutenção, derivadas das simulações realizadas por Banker e Slaughter (1997) e Tan e Mookerjee (2005).

4.2 Ciclo de Projetos de Manutenção

No PASM, um projeto representa a execução de uma ou mais solicitações de manutenção utilizando um conjunto de atividades bem definidas, tais como análise, desenho,

planejamento, implementação, testes, implantação, etc. O ciclo de vida de um projeto é composto por três fases: (1) **Registro**, (2) **Agrupamento** e (3) **Execução**, apresentadas na Figura 17 e discutidas na Seção 4.3. Como premissa do processo, todas as solicitações de manutenção devem ser registradas, enfileiradas e submetidas ao ciclo de projetos. Esse ciclo é regido por uma política de manutenção uniforme, responsável por pré-fixar o intervalo entre os ciclos de projetos e a duração de suas fases. A política de manutenção uniforme contribui para disciplinar os usuários e as equipes na definição do planejamento da manutenção e da evolução do software. Conforme observam Tan e Mookerjee (2005), uma política de manutenção uniforme facilita o planejamento e a gestão das atividades de manutenção, visto que proporciona um melhor controle sobre a execução dos processos.

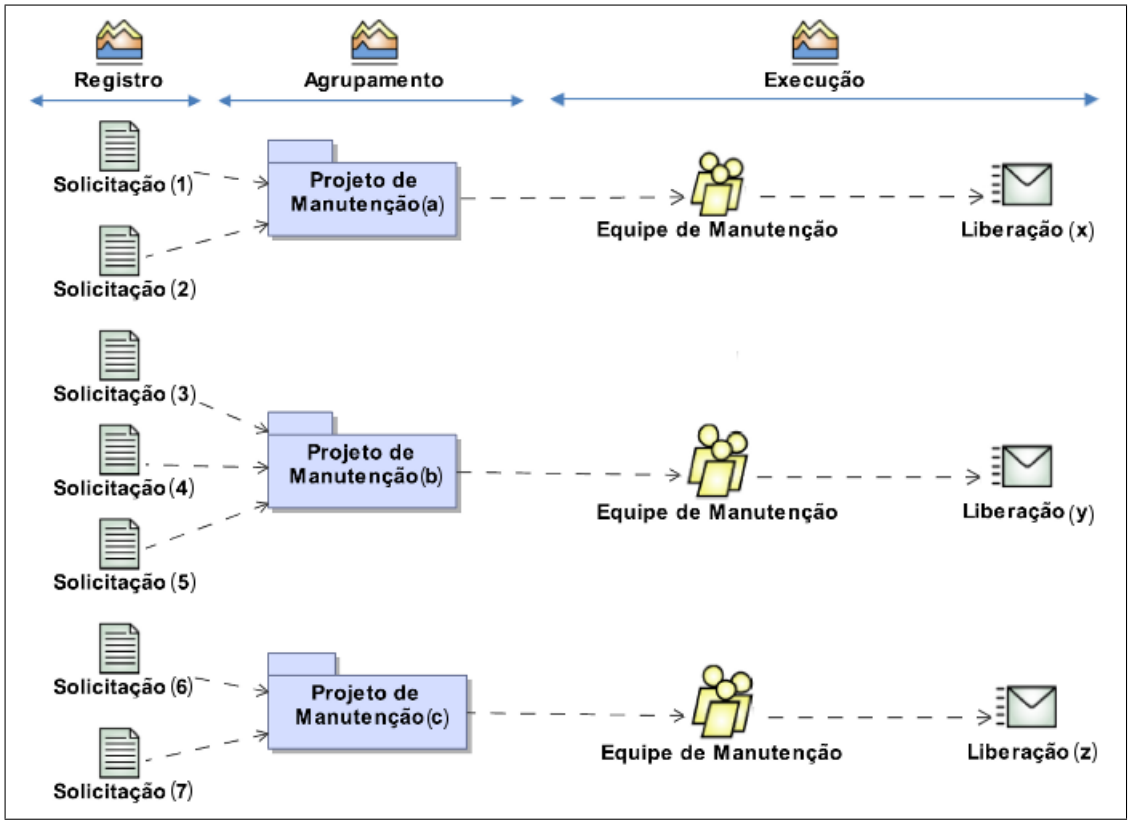

Figura 17: Fases do ciclo de projetos de manutenção

O PASM não estabelece o intervalo entre os ciclos de projetos nem a duração de suas fases, visto que cada equipe de manutenção, para cada sistema, pode adotar valores distintos. Entretanto, a duração dos ciclos de projetos deve ser curta, como por exemplo um mês, para amenizar o impacto do enfileiramento das solicitações para os usuários e justificar o agrupamento das solicitações. Além disso, é necessário manter a regularidade da execução dos projetos para que a manutenção e a evolução do software acompanhem as necessidades dos usuários. Ao determinar a frequência de realização dos projetos e a dura-

ção das fases do PASM, é importante considerar as características dos sistemas e da equipe de manutenção, o perfil e o volume de registro das solicitações, além das expectativas dos usuários. Ciclos longos podem implicar na necessidade frequente de replanejamento e inviabilizar o agrupamento das solicitações. O agrupamento das solicitações corretivas também pode ser comprometido devido à uma eventual demora no início da execução do próximo projeto. Conforme descrevem Banker e Slaughter (1997), Lehman e Ramil (2001), Kung (2004) e Tan e Mookerjee (2005) os padrões de registro de solicitações apresentam variações ao longo do tempo de vida de um software. Portanto, a periodicidade de execução dos projetos deve ser reavaliada junto aos usuários sempre que necessário e de forma pró-ativa.

O atendimento individual de uma solicitação deve ser realizado apenas quando o custo de oportunidade for elevado, a ponto de inviabilizar o agrupamento da solicitação. As solicitações atendidas individualmente são classificadas pelo PASM como urgentes e não necessariamente são solicitações corretivas. Nos casos em que for necessário realizar o atendimento individual de uma solicitação, os usuários devem sinalizar essa necessidade no ato do registro da demanda. As solicitações classificadas pelos usuários como urgentes devem ser triadas e caso a prioridade atribuída seja pertinente o atendimento deve ter seu início planejado para o quanto antes. Esse fluxo de exceção é importante para, por exemplo, garantir a operacionalidade do software em caso de falhas graves que eventualmente possam comprometer a imagem da organização ou proporcionar perdas financeiras.

4.3 Fases do PASM

Esta seção descreve as fases e atividades do ciclo de projetos de manutenção. O meta-modelo *Software & Systems Process Engineering Metamodel* (SPEM) (OMG, 2002) foi utilizado para modelar o PASM e seus componentes. Segundo (OMG, 2002), Henderson-Sellers e Gonzalez-Perez (2005) e Nardini *et al.* (2008), o SPEM é um meta-modelo que usa os perfis da *Unified Modeling Language* (UML) para descrever um processo de software e seus componentes ou uma família de processos relacionados. Particularmente, foram utilizados os elementos que representam as fases e as atividades de um processo de software. A relação dos elementos do SPEM utilizados neste trabalho é apresentada na Tabela 7.

Tabela 7: Legenda de estereótipos do SPEM

Termo	Ícone
Artefato	
Atividade	
Fase	
Liberação	
Marco	
Processo	
Time	

4.3.1 Registro

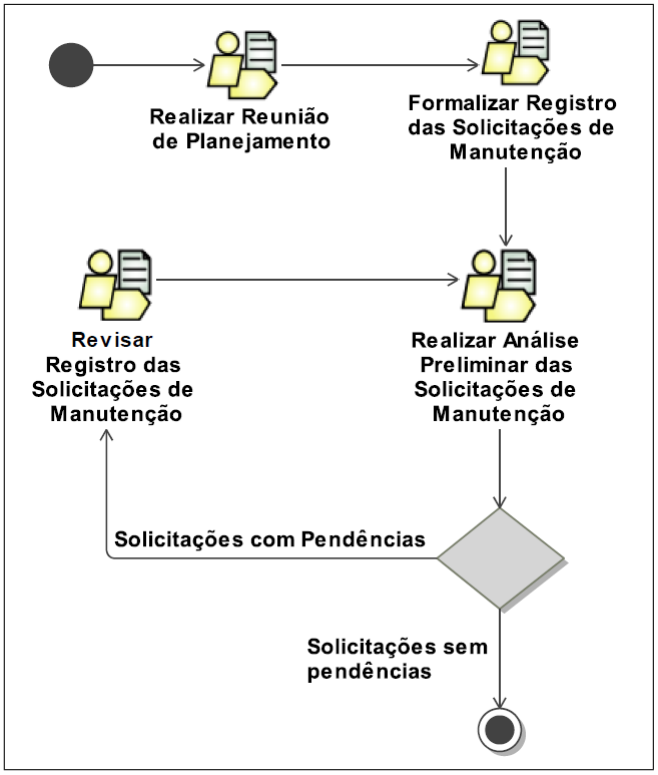

Figura 18: Fluxo de trabalho da fase Registro

A primeira fase do ciclo de projetos, denominada **Registro**, tem como propósito sistematizar o processo de elucidação e formalização dos requisitos de software que devem compor o próximo projeto de manutenção. O conjunto de atividades desta fase – ilustrado na Figura 18 – propõe apoiar os usuários na identificação e na priorização dos

requisitos funcionais e não funcionais, assim como avaliar, discutir e priorizar as falhas eventualmente encontradas no sistema. O auxílio aos usuários nesta fase contribui para melhorar o processo de estruturação dos requisitos, reduzindo o esforço necessário para análise e especificação das solicitações. A aproximação entre os usuários e a equipe de manutenção contribui também para: desenvolver a maturidade dos usuários em relação ao software e ao processo de manutenção, reduzir falsas expectativas e permitir o controle do registro de demandas com base na capacidade produtiva da equipe e nos contratos de manutenção estabelecidos. O acompanhamento dos usuários na definição do próximo projeto é mapeado pela atividade *Realizar Reunião de Planejamento*. Similar ao proposto pelo conjunto de práticas apresentado por Malta *et al.* (2009), a reunião de planejamento dá início ao planejamento do projeto de manutenção.

O planejamento do próximo projeto deve contemplar tanto as solicitações enfileiradas – solicitações já registradas no sistema de gestão de mudanças – quanto as novas solicitações que venham a ser identificadas através do *feedback* dos usuários (obtido durante a *reunião de planejamento*). Os diversos tipos de solicitações de manutenção (IEEE, 1999) requerem habilidades distintas. Lehman, Kahen e Ramil (2002) ratificam essa característica e destacam que a capacidade de produção da equipe geralmente é compartilhada com a execução de diferentes tipos de demandas que envolvem, por exemplo, as solicitações que propõem alinhar as funcionalidades do software com as mudanças do negócio ou as solicitações realizadas para melhorar a manutenibilidade do sistema. Entretanto, conforme observam Souza, Anquetil e Oliveira (2005), esse segundo conjunto envolve atividades onerosas e muitas vezes difícil de se justificar aos usuários. Portanto, durante a fase **Registro** é importante discutir com os usuários a necessidade de melhorar a estrutura do software, sua documentação ou qualquer outro artefato que possa contribuir com a melhoria da qualidade das liberações e para manter o custo da manutenção sob controle.

A atividade *Formalizar Registro das Solicitações de Manutenção* atribui aos usuários a responsabilidade de formalizar o escopo desejado para o próximo projeto de manutenção. As solicitações selecionadas serão analisadas pela equipe de manutenção – responsabilidade mapeada pela atividade *Realizar Análise Preliminar das Solicitações de Manutenção* – com o objetivo de identificar se as informações necessárias para a análise das solicitações foram devidamente registradas e se não existe duplicidade de demandas. Segundo Kajko-Mattsson (2002b), a análise das solicitações deve considerar a exatidão, a consistência e a integridade dos dados informados no registro das solicitações, para assim melhorar o entendimento das mesmas e possibilitar o *feedback* correto. Durante a execução da atividade *Revisar Registro das Solicitações de Manutenção*, as solicitações com

inconsistências devem ser reavaliadas pelos usuários.

O conjunto de atividades desta fase utiliza como principal artefato o *registro das solicitações de manutenção*. O principal artefato produzido é o *escopo preliminar da proposta de projeto de manutenção*.

4.3.2 Agrupamento

A fase **Agrupamento** tem como objetivo principal a elaboração e a aprovação da proposta de projeto de manutenção, concebida com base nas solicitações selecionadas pelos usuários (solicitações que compõem a proposta preliminar do projeto de manutenção). A Figura 19 apresenta o fluxo de trabalho proposto para essa fase.

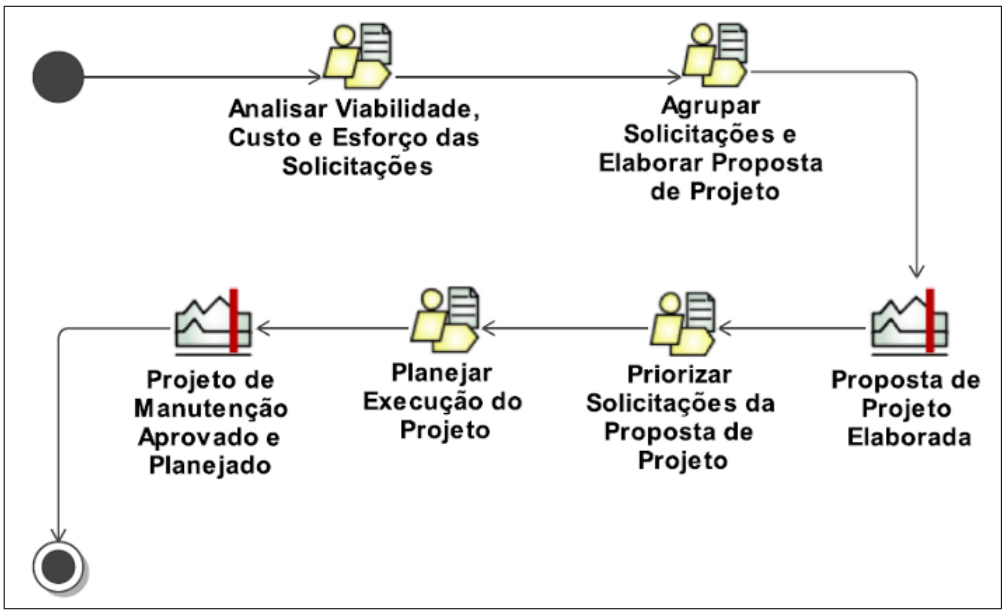

Figura 19: Fluxo de trabalho da fase Agrupamento

A primeira atividade proposta para a fase **Agrupamento** – *Analisar Viabilidade, Custo e Esforço das Solicitações* – tem por objetivo garantir o entendimento das mudanças e elucidar as informações necessárias para o agrupamento e a execução das solicitações e, consequentemente, a elaboração das propostas de projetos. Conforme sugerem Kajko-Mattsson (2002b) e April *et al.* (2005), as principais tarefas que devem ser realizadas durante a análise das mudanças são:

- **Análise da Viabilidade**: a equipe deve entender a mudança solicitada e identificar os recursos necessários para sua implementação. Dessa forma será possível averiguar a viabilidade da execução da solicitação antes da construção do projeto. Conforme

sugerem Polo *et al.* (1999a) e a norma ISO/IEC 14764 (ISO/IEC, 2006), a análise das solicitações deve contemplar as mudanças necessárias em toda a estrutura do software, incluindo a documentação e os sistemas relacionados.

De acordo com Antoniol *et al.* (2000), Antoniol *et al.* (2002), Lucia *et al.* (2007), Hill (2008) e Oliveira-Filho (2010) a análise de impacto auxilia a compreensão das solicitações, contribui para a definição do escopo dos projetos e para o gerenciamento de riscos. Dessa forma, é importante que a análise de impacto deve ser realizada durante a avaliação da viabilidade das solicitações de modo a apoiar a determinação do relacionamento entre as regras de negócio, os requisitos e os demais artefatos que compõem o software e assim apoiar a equipe na identificação das mudanças necessárias para atender a cada solicitação. Kajko-Mattsson (2002b) recomenda que a análise de impacto seja utilizada como suporte para a execução dos testes de regressão.

- **Estimativa de Custo e Esforço**: a análise das solicitações deve abranger a estimativa do custo e do esforço necessário para o atendimento das mesmas. Chan, Chung e Ho (1996), Granja-Alvarez e Barranco-García (1997), Kung (2004), Menzies *et al.* (2006), Sneed (2004) e Tan e Mookerjee (2005) destacam que a estimativa de custo deve contemplar, por exemplo, o tipo de solicitação, as características técnicas do sistema, a produtividade da equipe, a fase do ciclo de vida no qual o sistema se encontra, o nível de degradação da estrutura do sistema, o percentual médio de reúso, etc. O custo e o esforço estimados possibilitam a priorização adequada das solicitações e apoiam o planejamento da execução das solicitações a serem incluídas no escopo final do projeto de manutenção.

Após a análise das solicitações a equipe de manutenção – responsável pela atividade *Agrupar Solicitações e Elaborar Proposta de Projeto* – deve produzir a *proposta de projeto de manutenção*. A elaboração da proposta deve levar em consideração os eventuais projetos em execução e – conforme observam Ruhe e Saliu (2005) – as solicitações agrupadas devem prover o maior retorno de investimento possível, satisfazer os principais *stakeholders*, serem viáveis e devem considerar os relacionamentos e as dependências entre as funcionalidades e os módulos do sistema. A inclusão de solicitações correlacionadas possibilita a redução do esforço necessário para especificar, implementar e testar as mudanças. Solicitações cujo impacto se concentra no mesmo módulo do sistema também podem reduzir o custo da manutenção, uma vez que o agrupamento dessas solicitações reduz, por exemplo, o esforço das atividades de gestão de configuração e de implantação

das mudanças. Dessa forma, a equipe deverá avaliar a possibilidade de incluir outras solicitações na proposta de projeto que não tenham sido selecionadas pelos usuários na fase **Registro**. O marco *Proposta de Projeto Elaborada* representa a conclusão da elaboração da proposta de projeto. A proposta deve especificar todas as atividades requeridas para execução das solicitações, incluindo as ações preventivas necessárias para garantir a manutenibilidade do sistema e o custo/esforço total estimado para a execução do projeto de manutenção.

A próxima atividade do fluxo de trabalho da fase **Agrupamento** – *Priorizar Solicitações da Proposta de Projeto* – visa priorizar as propostas de forma a determinar o escopo final do projeto. A priorização deve considerar o retorno de investimento das solicitações e a capacidade de produção da equipe. Kajko-Mattsson (2002b) ressalta que a priorização deve ser realizada considerando o ambiente corporativo como um todo. Svahnberg *et al.* (2010) ratificam essa argumentação e destacam que as liberações devem ser definidas com base nos benefícios de curto, médio e longo prazo, atendendo às solicitações mais importantes antecipadamente. A priorização é de responsabilidade do *comitê de priorização das propostas de projetos*. Esse comitê deve ser formado pelos líderes das equipes envolvidas na manutenção dos sistemas, pelos gerentes e eventualmente pelos usuários chave. O principal objetivo da priorização das propostas é avaliar a possibilidade de execução dos projetos diante da capacidade produtiva de cada equipe e da demanda atual de serviços. A priorização deve considerar também o impacto nos projetos em execução e no desenvolvimento e na manutenção de outros sistemas da organização.

Os impasses na priorização das propostas devem ser negociados com os usuários. As organizações que trabalham com terceirização de serviços devem considerar a terceirização da execução dos projetos quando a capacidade da equipe interna não permitir a inclusão de uma solicitação importante no escopo final do projeto. Durante a atividade *Planejar Execução do Projeto* os gerentes de projeto devem elaborar o plano de execução e submetê-lo a aprovação dos usuários. O marco *Projeto de Manutenção Aprovado e Planejado* representa a aprovação do escopo e do plano de execução do projeto de manutenção.

4.3.3 Execução

A Figura 20 apresenta o fluxo de trabalho da fase **Execução** do ciclo de projetos do PASM. Ao contrário do proposto pelo conjunto de práticas apresentado por Malta *et al.* (2009), o PASM não enumera as atividades de Engenharia de Software a serem executadas para atender cada solicitação agrupada no projeto de manutenção. Isso se dá

devido ao fato dessas atividades serem de responsabilidade do processo de manutenção de cada organização.

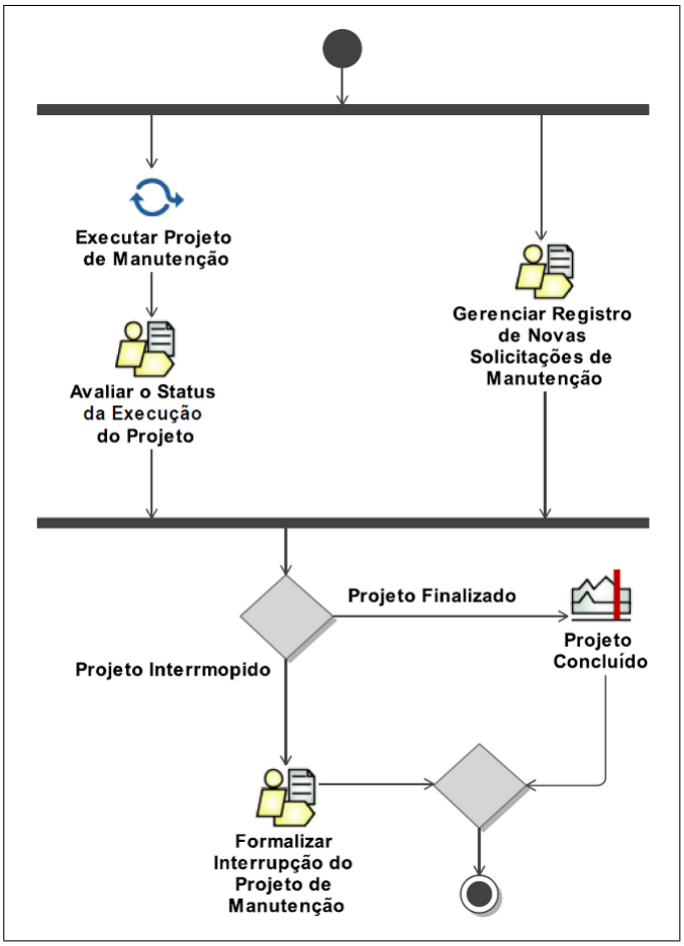

Figura 20: Fluxo de trabalho da fase Execução

O fluxo de trabalho da fase execução contém a atividade *Executar Projeto de Manutenção* – do tipo `processo` – que representa uma interface entre o PASM e o processo de manutenção da organização. Essa atividade determina que os projetos aprovados devem ser executados de acordo com o processo de manutenção da organização. Conforme recomendam Polo *et al.* (1999a), a norma ISO/IEC 14764 (ISO/IEC, 2006) e Tan e Mookerjee (2005), durante a execução das solicitações de manutenção – tanto evolutivas quanto corretivas – a documentação do sistema deve ser atualizada e o nível de degradação da estrutura do software controlado para evitar o comprometimento de sua manutenibilidade. Kajko-Mattsson (2002b) e April *et al.* (2005) destacam que a equipe de manutenção deve manter os usuários atualizados sobre a execução das solicitações e que a comunicação deve ser realizada de forma pró-ativa, incluindo o relato de problemas encontrados ou qualquer desvio no planejamento estabelecido. Em paralelo à execução dos projetos, a atividade *Gerenciar Registro de Novas Solicitações* delega à equipe de

manutenção a responsabilidade de monitorar o registro de novas solicitações de forma a determinar uma eventual necessidade de interrupção do projeto corrente.

A equipe de manutenção também deve avaliar a necessidade de interrupção do projeto em caso de desvios nos planos estabelecidos ou por mudanças provocadas por fatores externos. Segundo Feng, Mookerjee e Sethi (2006) e Svahnberg *et al.* (2010), fatores como defeitos encontrados, novas demandas registradas, *feedback* obtido dos usuários e eventos relacionados ao negócio podem acarretar na necessidade de replanejamento dos projetos. Durante a execução da atividade *Avaliar o Status da Execução do Projeto* a equipe deve revisar o histórico de informações registradas durante a execução do projeto, avaliar as medidas e metas estabelecidas, avaliar as lições aprendidas e caso o projeto tenha sido cancelado ou interrompido, sinalizar o percentual de conclusão de cada solicitação integrante do mesmo. O marco *Projeto Concluído* representa que todas as solicitações do projeto foram devidamente executadas e a nova versão do sistema está disponível para uso pelos usuários.

A especificação das fases do ciclo de projetos menciona a necessidade de se avaliar os projetos já em execução. Dependendo da quantidade, do tamanho e do custo de oportunidade das solicitações aprovadas, a fase de execução de um projeto pode ultrapassar a duração recomendada. Nesses casos, a equipe deve iniciar um novo ciclo de projeto mesmo com um ou vários projetos em andamento. Entretanto, o planejamento dos novos projetos deve se preocupar em não prejudicar a execução dos projetos já aprovados. Além disso, algumas situações podem demandar a execução simultânea de mais de um projeto. Por exemplo, conforme observa Kung (2004), alguns sistemas oferecidos a vários usuários podem ter diferentes versões implantadas, incluindo versões para plataformas distintas. Nesses casos, cada versão pode ter seu próprio ciclo de projetos. Solicitações perfectivas que envolvem a criação de um novo módulo também podem ser executadas em paralelo com outras solicitações. A execução de projetos paralelos também é encorajada nos cenários de terceirização da manutenção, onde diferentes fornecedores e a equipe interna podem realizar diferentes projetos e melhorar o tempo de resposta do atendimento das solicitações de manutenção.

4.4 Considerações Finais

As liberações de novas versões de um software durante a fase de manutenção precisam atender aos diferentes tipos de demandas registradas pelos usuários e garantir a

operacionalidade e a evolução do sistema. Porém, o atendimento individual das solicitações tende a comprometer o planejamento, o custo e a qualidade das liberações. A prática de agrupamento das solicitações em projetos de manutenção, em conjunto com a sistematização do registro, da priorização e da execução das solicitações apóiam o planejamento das liberações ao longo da fase de manutenção. O conjunto de atividades proposto pelo PASM objetiva reduzir o custo das liberações, reduzir o índice de retrabalho através da melhoria da análise das solicitações, melhorar a produtividade das equipes e aumentar a qualidade das liberações. Segundo Solingen (2004), Sommerville (2007), Kalinowski *et al.* (2010) e Weber *et al.* (2008) esses são alguns dos principais objetivos das iniciativas de melhoria de processos, o que contribui para caracterizar o PASM como uma iniciativa para a melhoria de processos de manutenção.

As diretrizes do PASM se aplicam a diferentes modelos de manutenção. Organizações nas quais a manutenção de sistemas é de responsabilidade dos departamentos internos de Tecnologia da Informação se beneficiam diretamente do agrupamento das solicitações em projetos de manutenção. Nesse caso, as interferências políticas e a incoerência da prioridade atribuída às solicitações podem ser responsáveis por prejudicar o agrupamento e comprometer o planejamento das liberações. Nos cenários onde a manutenção é terceirizada, o agrupamento pode auxiliar as organizações a reduzir os custos e a embasar as negociações entre consumidores e prestadores de serviços de manutenção. Entretanto, as diretrizes propostas pelo PASM para fomentar o agrupamento das solicitações podem apresentar algumas limitações em determinados contextos. Os sistemas de prateleira e os produtos de software utilizados por uma grande base de usuários costumam ter modelos de manutenção específicos. Nesses casos, o planejamento das liberações está diretamente relacionado com as decisões estratégicas das organizações e os consumidores não têm influência direta na evolução do software, embora possam sugerir melhorias. Além disso, a manutenção de alguns sistemas é regida por contratos de nível de serviço que impõem restrições, por exemplo, para o início e para a conclusão do atendimento das solicitações. Perante essas restrições, o agrupamento pode implicar na violação das cláusulas dos contratos estabelecidos.

Capítulo 5

Conclusões

5.1 Visão Geral

A solução para agrupar solicitações de manutenção apresentada nesta dissertação se baseia na premissa de que as iniciativas de melhorias de processos contribuem de forma efetiva para a evolução das práticas de Engenharia de Software. Diante disso, esta dissertação realizou a caracterização do atendimento das solicitações de manutenção utilizando um algoritmo de *clusterização*, visando representar, compreender, analisar e comparar os padrões existentes no atendimento das solicitações de manutenção e, assim, auxiliar o estabelecimento de *benchmarks* e fundamentar a adoção de iniciativas de melhorias de processos de manutenção de software. A abordagem utilizada é particularmente interessante, uma vez que: permite a avaliação de atividades já concluídas ou que ainda estejam em execução, não impõe a necessidade de se conhecer a estrutura e o comportamento dos sistemas e possibilita a avaliação de um processo por diferentes perspectivas.

A *clusterização* foi utilizada para agrupar as solicitações de manutenção similares levando em consideração um conjunto de características que representam: (1) indicadores importantes sobre o atendimento das solicitações (tempo total de vida, tempo de espera e tempo de atendimento), (2) as principais atividades desempenhadas (análise, planejamento, implementação, validação e implantação) e (3) as suspensões da implementação ocorridas durante o atendimento.

Para demonstrar a aplicabilidade da solução proposta, esta dissertação avaliou a implantação do conjunto de práticas para sistematizar o registro, o agrupamento e a priorização das solicitações de manutenção, proposto por Malta *et al.* (2009) e implantado

no DATAPUC. Além disso, as práticas propostas foram formalizadas nesta dissertação no processo denominado Processo para Agrupamento de Solicitações de Manutenção de Software (PASM). Na avaliação realizada foram consideradas 3.686 solicitações atendidas pelo DATAPUC entre fevereiro e outubro de 2008 e de 2009.

A avaliação realizada apontou melhorias no atendimento das solicitações após a adoção das práticas avaliadas. Os principais benefícios identificados foram uma redução média de 5,73 horas no tempo de atendimento das solicitações urgentes (21,08 - 15,35), uma redução de 56,24% (383 / 681) no número de solicitações urgentes concluídas sem a ocorrência de atividades de validação e um aumento de 31,5% na duração das atividades de validação (4,77 / 3,63). Em contrapartida, houve um aumento no número de horas dedicadas ao atendimento das solicitações urgentes e a frequência de suspensão da implementação foi superior nos *clusters* identificados para as solicitações urgentes de 2009.

5.2 Contribuições

As principais contribuições desta dissertação são descritas a seguir:

- Apresentação de evidências sobre os principais benefícios e os eventuais pontos negativos da prática de agrupar solicitações de manutenção em projetos de manutenção. Foi realizada uma caracterização do atendimento de 3.686 solicitações de manutenção registradas e atendidas entre fevereiro e outubro de 2008 e de 2009 pelo DATAPUC. A caracterização usou o algoritmo de *clusterização* denominado *k-means* (HAN; MICHELINE, 2005) e o resultado dessa caracterização foi avaliado com o objetivo de identificar as mudanças ocorridas no atendimento das solicitações de manutenção depois da adoção do conjunto de práticas proposto por Malta *et al.* (2009) e assim, produzir evidências sobre os benefícios da prática de agrupar solicitações de manutenção em projetos de manutenção.

- Formalização do PASM: esta dissertação apresentou uma formalização do PASM, acompanhada de uma especificação detalhada elaborada usando o meta-modelo *Software & Systems Process Engineering Metamodel* (SPEM) (OMG, 2002). A nova versão foi concebida com o objetivo de tornar o PASM um processo simples, porém abrangente, fundamentado nas principais referências da área de manutenção de software e que possa ser utilizado como ponto de partida para que outras organiza-

ções adotem as diretrizes propostas para apoiar o agrupamento de solicitações de manutenção em projetos de manutenção.

Publicações: Os resultados desta dissertação deram origem às seguintes publicações:

- APARECIDO, Gladston *et al.* Proposta e avaliação de um processo para agrupamento de solicitações de manutenções. In: **IX Simpósio Brasileiro de Qualidade de Software (SBQS)**. Belém: UFPA, 2010.
- APARECIDO, Gladston *et al.* On the benefits of planning and grouping software maintenance requests. In: **15th IEEE European Conference on Software Maintenance and Reengineering (CSMR)**. Oldemburgo, Alemanha: IEEE Computer Society, 2011.

5.3 Trabalhos Futuros

Com o objetivo de aprofundar as investigações sobre o planejamento de liberações durante a fase de manutenção, sobre o agrupamento de solicitações de manutenção e sobre a avaliação de processos de manutenção de software, sugere-se as seguintes linhas de pesquisas:

- Investigar como a avaliação apresentada nesta dissertação pode contribuir para comprovar os resultados obtidos – por meio de modelos analíticos – por Banker e Slaughter (1997) e Tan e Mookerjee (2005) sobre o ganho de escala proveniente do agrupamento de solicitações em projetos de manutenção. Pretende-se também complementar a avaliação da prática de agrupar solicitações de manutenção através da análise da satisfação dos usuários.
- Conceber um modelo para a gestão do agrupamento de solicitações que permita sugerir o tamanho e a composição ideal de cada liberação, levando em consideração o custo, o prazo e as particularidades do ambiente organizacional e do sistema em manutenção.
- Integrar a caracterização do atendimento de solicitações de manutenção realizada nesta dissertação com as práticas propostas por modelos de maturidade genéricos e específicos para a manutenção de software.

Referências

AHN, Yunsik *et al.* The software maintenance project effort estimation model based on function points. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 15, n. 2, p. 71–85, mar. 2003.

ANTONELLIS, Panagiotis *et al.* Clustering for monitoring software systems maintainability evolution. **Electronic Notes in Theoretical Computer Science**, Amsterdã, Holanda, v. 233, p. 43–57, mar. 2009.

ANTONIOL, Giuliano *et al.* Identifying the starting impact set of a maintenance request: A case study. In: **4th IEEE European Conference on Software Maintenance and Reengineering (CSMR)**. Zurique, Suíça: IEEE Computer Society, 2000. p. 227–230.

ANTONIOL, Giuliano *et al.* Recovering traceability links between code and documentation. **IEEE Transactions Software Engineering**, Piscataway, EUA, v. 28, n. 10, p. 970–983, out. 2002.

ANTONIOL, Giuliano *et al.* Assessing staffing needs for a software maintenance project through queuing simulation. **IEEE Transactions on Software Engineering**, Washington, EUA, v. 30, n. 1, p. 43–58, jan. 2004.

APRIL, Alain; ABRAN, Alain. A software maintenance maturity model (S3M): Measurement practices at maturity levels 3 and 4. **Electronic Notes Theoretical Computer Science**, Amsterdã, Holanda, v. 233, p. 73–87, mar. 2009.

APRIL, Alain; ABRAN, Alain; DUMKE, Reiner R. SM^{CMM} model to evaluate and improve the quality of the software maintenance process. In: **8th IEEE European Conference on Software Maintenance and Reengineering (CSMR)**. Tampere, Finlândia: IEEE Computer Society, 2004. p. 243–248.

APRIL, Alain *et al.* Software maintenance maturity model (SM^{MM}): the software maintenance process model. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 17, n. 3, p. 197–223, mai. 2005.

ASUNDI, Jai; SARKAR, Sumit. Staffing software maintenance and support projects. In: **38th Hawaii International Conference on System Sciences (HICSS)**. Hilton Waikoloa Village, Hawaii: IEEE Computer Society, 2005. p. 314.2.

AUER, Martin *et al.* Optimal project feature weights in analogy-based cost estimation: Improvement and limitations. **IEEE Transactions on Software Engineering**, IEEE Computer Society, Piscataway, EUA, v. 32, n. 2, p. 83–92, 2006.

AZIZ, Junaid; AHMED, Faheem; LAGHARI, Mohammad Shakeel. Empirical analysis of team and application size on software maintenance and support activities. In: **1st IEEE International Conference on Information Management and Engineering (ICIME)**. Washington, EUA: IEEE Computer Society, 2009. p. 47–51.

BANKER, Rajiv D.; CHARNES, Abraham; COOPER, William W. Some models for estimating technical scale inefficiencies in Data Envelopment Analysis. **Management Science**, Linthicum, EUA, v. 30, n. 9, p. 1078–1092, set. 1984.

BANKER, Rajiv D.; SLAUGHTER, Sandra A. A field study of scale economies in software maintenance. **Management Science**, Linthicum, EUA, v. 43, n. 12, p. 1709–1725, dez. 1997.

BASILI, Victor *et al.* Understanding and predicting the process of software maintenance release. In: **8th IEEE International Conference on Software Engineering (ICSE)**. Berlim, Alemanha: IEEE Computer Society, 1996. p. 464–474.

BASILI, Victor R.; ROMBACH, Dieter. The tame project: Towards improvement-oriented software enviromnents. **IEEE Transactions on Software Engineering**, Piscataway, EUA, v. 14, n. 6, p. 758–773, jun. 1988.

BHATT, Pankaj; SHROFF, Gautam; MISRA, Arun K. Dynamics of software maintenance. **ACM SIGSOFT Software Engineering Notes**, New York, v. 29, n. 5, p. 1–5, set. 2004.

BOZDOGAN, Hamparsum. **Statistical Data Mining and Knowledge Discovery**. 1. ed. Boca Raton, EUA: CRC Press, Inc., 2003.

CASTANHEIRA, Eunice T. J. **UM MODELO DE ANÁLISE POR ENVOLTÓRIA DE DADOS PARA O SEGUNDO PARADOXO TOYOTA DE DESENVOLVIMENTO DE NOVOS PRODUTOS**. 100 f. Dissertação (Mestrado em Ciência em Engenharia Oceânica) — Instituto Alberto Luiz Coimbra de Pós-graduação e Pesquisa de Engenharia (COPPE), Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2008.

CHAN, Taizan. Beyond productivity in software maintenance: Factors affecting lead time in servicing users' requests. In: **16th IEEE International Conference on Software Maintenance (ICSM)**. San Jose, EUA: IEEE Computer Society, 2000. p. 228–235.

CHAN, Taizan; CHUNG, Siu-Leung; HO, Teck Hua. An economic model to estimate software rewriting and replacement times. **IEEE Transactions on Software Engineering**, Piscataway, EUA, v. 22, n. 8, p. 580–598, ago. 1996.

CHAPIN, Ned. Do we know what preventive maintenance is? In: **16th IEEE International Conference on Software Maintenance (ICSM)**. San Jose, EUA: IEEE Computer Society, 2000. p. 15–17.

CHAPIN, Ned. Software maintenance in complying with it governance: A report from the field. In: **25th IEEE International Conference on Software Maintenance (ICSM)**. Edmonton, Canadá: IEEE Computer Society, 2009. p. 499–502.

CHRISSIS, Mary Beth; KONRAD, Mike; SHRUM, Sandy. **Guidelines for Process Integration and Product Improvement**. 2. ed. Reading, EUA: Addison-Wesley Professional, 2006.

CONTE, Tayana Uchôa. **TÉCNICA DE INSPEÇÃO DE USABILIDADE BASEADA EM PERSPECTIVAS DE PROJETO WEB**. 206 f. Tese (Doutorado em Engenharia de Sistemas e Computação) — Instituto Alberto Luiz Coimbra de Pós-graduação e Pesquisa de Engenharia (COPPE), Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2009.

EISENHARDT, Kathleen M. Building Theories from Case Study Research. **The Academy of Management Review**, Academy of Management, v. 14, n. 4, p. 532–550, 1989.

FENG, Qi; MOOKERJEE, Vijay S.; SETHI, Suresh. P. Optimal policies for the sizing and timing of software maintenance projects. **European Journal of Operational Research**, v. 173, n. 3, p. 1047–1066, set. 2006.

FERREIRO, Analia Irigoyen *et al.* Applying ISO 9001:2000, MPS.BR and CMMI to achieve software process maturity: BL informatica's pathway. In: **29th IEEE International Conference on Software Engineering (ICSE)**. Minneapolis, EUA: IEEE Computer Society, 2007. p. 642–651.

FONSECA, Aline Bandeira de Mello. **REVISÃO DAS TARIFAS AEROPORTUÁRIAS COM MODELOS DEA-GSZ NÃO RADIAIS**. 87 f. Dissertação (Mestrado em Engenharia de Produção) — Universidade Federal Fluminense, Centro Tecnológico, Niterói, 2005.

FOSS, Tron *et al.* A simulation study of the model evaluation criterion MMRE. **IEEE Transactions on Software Engineering**, v. 29, n. 11, p. 985–995, nov. 2003.

FOWLER, Priscilla; RIFKIN, Stan. Software engineering process group guide. In: **Technical Report CMU/SEI-90-TR-024 ESD-90-TR-225**. Pittsburgh, EUA: Software Engineering Institute (SEI), 1990. Disponível em: `http://www.sei.cmu.edu/reports/90tr024.pdf`. Acesso em: 15 jan. 2010.

FREIRE, Yara Maria Almeida; BELCHIOR, Arnaldo Dias. Pontos de casos de uso técnicos para manutenção de software. In: **XXI Simpósio Brasileiro de Engenharia de Software (SBES)**. João Pessoa: UFPB, 2007. p. 273–288.

GIBSON, Diane L.; GOLDENSON, Dennis R.; KOST, Keith. **Performance Results of CMMI-Based Process Improvement**. Burgh, EUA, 2006.

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel. **Data Mining: Uma Guia Prático**. 1. ed. Rio de Janeiro: Editora Campus, 2005.

GRANJA-ALVAREZ, Juan Carlos; BARRANCO-GARCÍA, Manuel José. A method for estimating maintenance cost in a software project: a case study. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 9, n. 3, p. 161–175, mai. 1997.

HALL, Tracy *et al.* An empirical study of maintenance issues within process improvement programmes in the software industry. In: **17th IEEE International Conference on Software Maintenance (ICSM)**. Florença, Itália: IEEE Computer Society, 2001. p. 422–428.

HAN, Jiawei; MICHELINE, Kamber. **Data Mining: Concepts and Techniques**. 2. ed. San Francisco, EUA: Morgan Kaufmann Publishers Inc., 2005.

HENDERSON-SELLERS, Brian; GONZALEZ-PEREZ, Cesar. A comparison of four process metamodels and the creation of a new generic standard. **Information and Software Technology**, Newton, EUA, v. 47, n. 1, p. 49–65, jan. 2005.

HILL, Emily. Developing natural language-based program analyses and tools to expedite software maintenance. In: **30th IEEE International Conference on Software Engineering (ICSE)**. Leipzig, Alemanha: IEEE Computer Society, 2008. p. 1015–1018.

HUANG, Shihong *et al.* Adoption-centric software maintenance process improvement via information integration. In: **13th IEEE International Workshop on Software Technology and Engineering Practice (STEP)**. Budapeste, Hungria: IEEE Computer Society, 2005. p. 25–34.

IBM CORPORATION. **Rational Unified Process**. 2005.

IEEE. **IEEE Standard Glossary of Software Engineering Terminology: IEEE Standard 610.121990**. New York, 1990.

IEEE. IEEE standard for software maintenance, IEEE std 1219-1998. In: **IEEE Standards Software Engineering, Volume Two: Process Standards**. New York: IEEE Press, 1999. v. 2.

IEEE. **Guide to the Software Engineering Body of Knowledge (SWEBOK)**. Washington, EUA: Angela Burgess, 2004. Disponível em: `http://www.swebok.org`. Acesso em: 1 mai. 2010.

IFPUG. **Function Point Counting Practices Manual - Release 4.3.1**. Princeton, EUA, 2010.

ISACA. **COBIT Framework for IT Governance and Control**. 2010. Disponível em: `http://www.isaca.org/Knowledge-Center/COBIT/Pages/Overview.aspx`. Acesso em: 26 jul. 2010.

ISO/IEC. **International Standard ISO/IEC/IEEE 14764 Software Engineering - Software Life Cycle Processes - Maintenance**. Piscataway, EUA, 2006.

ISO/IEC. **International Standard ISO/IEC 12207 Software Life Cycle Processes**. Genebra - Suíca, 2008.

JADALLAH, Anas *et al.* Balancing value and modifiability when planning for the next release. In: **25th IEEE International Conference on Software Maintenance (ICSM)**. Edmonton, Canadá: IEEE Computer Society, 2009. p. 495–498.

JI, He. Dynamic and static views of software evolution. In: **17th IEEE International Conference on Software Maintenance (ICSM)**. Florença, Itália: IEEE Computer Society, 2001. p. 592–601.

JUNG, Ho-Won; GOLDENSON, Dennis R. Evaluating the relationship between process improvement and schedule deviation in software maintenance. **Information and Software Technology**, Newton, EUA, v. 51, n. 2, p. 351–361, fev. 2009.

KAJKO-MATTSSON, Mira. Towards a business maintenance model. In: **17th IEEE International Conference on Software Maintenance (ICSM)**. Florença, Itália: IEEE Computer Society, 2001. p. 500–509.

KAJKO-MATTSSON, Mira. Corrective maintenance maturity model: Problem management. In: **18th IEEE International Conference on Software Maintenance (ICSM)**. Montreal, Canadá: IEEE Computer Society, 2002a. p. 486–490.

KAJKO-MATTSSON, Mira. Problem management maturity within corrective maintenance. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 14, n. 3, p. 197–227, mai. 2002b.

KAJKO-MATTSSON, Mira. Evaluation of CM^3: Front-end problem management within industry. In: **10th IEEE European Conference on Software Maintenance and Reengineering (CSMR)**. Bari, Itália: IEEE Computer Society, 2006. p. 367–368.

KAJKO-MATTSSON, Mira. Maturity status within front-end support organisations. In: **29th IEEE International Conference on Software Engineering (ICSE)**. Minneapolis, EUA: IEEE Computer Society, 2007. p. 652–663.

KAJKO-MATTSSON, Mira; FORSSANDER, Stefan; OLSSON, Ulf. Corrective maintenance maturity model (CM^3): maintainer's education and training. In: **23rd IEEE International Conference on Software Engineering (ICSE)**. Toronto, Canadá: IEEE Computer Society, 2001. p. 610–619.

KAJKO-MATTSSON, Mira; MEYER, Pia. Evaluating the acceptor side of EM^3: release management at SAS. In: **4th IEEE International Symposium on Empirical Software Engineering (ISESE)**. Noosa Heads, Austrália: IEEE Computer Society, 2005. p. 315–324.

KALINOWSKI, Marcos *et al.* MPS.BR: Promovendo a adoção de boas práticas de engenharia de software pela indústria brasileira. In: **XIII Congreso Iberoamericano en Software Engineering (CIbSE)**. Cuenca, Equador: Universidad del Azuay, 2010.

KORTE, Marcel; PORT, Dan. Confidence in software cost estimation results based on MMRE and PRED. In: **4th ACM International Workshop on Predictor Models in Software Engineering (PROMISE)**. Leipzig, Alemanha: ACM, 2008. p. 63–70.

KUNG, Hsiang-Jui. Quantitative method to determine software maintenance life cycle. In: **20th IEEE International Conference on Software Maintenance (ICSM)**. Chicago, EUA: IEEE Computer Society, 2004. p. 232–241.

KUNG, Hsiang-Jui; HSU, Cheng. Software maintenance life cycle model. In: **14th IEEE International Conference on Software Maintenance (ICSM)**. Bethesda, EUA: IEEE Computer Society, 1998. p. 113–121.

LEHMAN, Meir M.; BELADY, Laszlo A. (Ed.). **Program evolution: processes of software change**. 1. ed. San Diego, EUA: Academic Press Professional Inc., 1985.

LEHMAN, Meir M.; KAHEN, Goel; RAMIL, Juan F. Behavioural modelling of long-lived evolution processes: some issues and an example. **Journal of Software Maintenance**, New York, v. 14, n. 5, p. 335–351, set. 2002.

LEHMAN, Meir M.; RAMIL, Juan F. Rules and tools for software evolution planning and management. **Annals of Software Engineering**, Red Bank, EUA, v. 11, n. 1, p. 15–44, jun. 2001.

LUCCA, Giuseppe A. di; RAGO, Francesco. A queue theory-based approach to staff software maintenance centers. In: **17th IEEE International Conference on Software Maintenance (ICSM)**. Florença, Itália: IEEE Computer Society, 2001. p. 510.

LUCIA, Andrea De *et al.* Recovering traceability links in software artifact management systems using information retrieval methods. **ACM Transactions Software Engineering and Methodology**, New York, v. 16, n. 4, p. 13–63, set. 2007.

MAFRA, Sômulo Nogueira; BARCELOS, Rafael Ferreira; TRAVASSOS, Guilherme Horta. Aplicando uma metodologia baseada em evidência na definição de novas tecnologias de software. In: **XX Simpósio Brasileiro de Engenharia de Software (SBES)**. Florianópolis: UFSC, 2006. p. 239–254.

MAFRA, Sômulo Nogueira; TRAVASSOS, Guilherme Horta. Estudos primários e secundários apoiando a busca por evidência em engenharia de software. In: **Relatório Técnico RT – ES 687/06**. Rio de Janeiro: Programa de Engenharia de Sistemas e Computação, 2006.

MALTA, Marcelo Nassau *et al.* Modelo de gestão de demandas de manutenção de software: a experiência da PUC Minas. In: **VI Workshop de Manutenção de Software Moderna (WMSWM)**. Ouro Preto: UFOP, 2009. p. 1–4.

MARQUES-NETO, Humberto T. *et al.* A characterization of broadband user behavior and their e-business activities. **ACM SIGMETRICS Performance Evaluation Review**, New York, v. 32, n. 3, p. 3–13, dez. 2004.

MENASCE, Daniel A.; ALMEIDA, Virgilio A. F. **Scaling for E Business: Technologies, Models, Performance, and Capacity Planning**. 1. ed. Upper Saddle River, EUA: Prentice Hall PTR, 2000.

MENS, Tom *et al.* Challenges in software evolution. In: **8th IEEE International Workshop on Principles of Software Evolution (IWPSE)**. Lisboa, Portugal: IEEE Computer Society, 2005. p. 13–22.

MENZIES, Tim *et al.* Selecting best practices for effort estimation. **IEEE Transactions on Software Engineering**, Piscataway, EUA, v. 32, n. 11, p. 883–895, nov. 2006.

MOOKERJEE, Radha. Maintaining enterprise software applications. **Communications of the ACM**, New York, v. 48, n. 11, p. 75–79, nov. 2005.

NARDINI, Elena *et al.* Spem on test: the soda case study. In: **23rd ACM Symposium on Applied Computing (SAC)**. New York: ACM, 2008. p. 700–706.

NIESSINK, Frank; VLIET, Hans van. Predicting maintenance effort with function points. In: **13th IEEE International Conference on Software Maintenance (ICSM)**. Bari, Itália: IEEE Computer Society, 1997. p. 32–39.

NIESSINK, Frank; VLIET, Hans van. Software maintenance from a service perspective. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 12, n. 2, p. 103–120, mar. 2000.

OGC. **Official ITIL Website**. 2011. Disponível em: `http://www.itil-officialsite.com/home/home.asp`. Acesso em: 24 jan. 2011.

OLIVEIRA-FILHO, Antonio. Change impact analysis from business rules. In: **32nd IEEE International Conference on Software Engineering (ICSE)**. Cape Town, África do Sul: IEEE Computer Society, 2010. p. 353–354.

OLIVEIRA, Tito Marcelo; ANQUETIL, Nicolas; ALONSO, Luiza. Os sentidos atribuídos à manutenção de software segundo a teoria das representações sociais. In: **V Workshop Um Olhar Sociotécnico sobre a Engenharia de Software (WOSES)**. Ouro Preto: UFOP, 2009. p. 453–461.

OMG. **Software & Systems Process Engineering Metamodel**. 2002. Disponível em: `http://www.omg.org/spec/SPEM/2.0`. Acesso em: 17 out. 2010.

ORDONEZ, Carlos; OMIECINSKI, Edward. Efficient disk-based k-means clustering for relational databases. **IEEE Transactions on Knowledge and Data Engineering**, Piscataway, EUA, v. 16, n. 8, p. 909–921, ago. 2004.

POLO, Macario *et al.* Mantool: a tool for supporting the software maintenance process. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 13, n. 2, p. 77–95, abr. 2001.

POLO, Macario *et al.* Mantema: A complete rigorous methodology for supporting maintenance based on the ISO/IEC 12207 standard. In: **3rd IEEE European Conference on Software Maintenance and Reengineering (CSMR)**. Amsterdã, Holanda: IEEE Computer Society, 1999a. p. 178–181.

POLO, Macario *et al.* Mantema: A software maintenance methodology based on the ISO/IEC 12207 standard. In: **4th IEEE International Symposium and Forum on Software Engineering Standards (ISESS)**. Otago, Nova Zelândia: IEEE Computer Society, 1999b. p. 76–81.

PRESSMAN, Roger. **Software Engineering: A Practitioner's Approach**. 7. ed. New York: McGraw-Hill Science/Engineering/Math, 2009.

RAJLICH, Václav T.; BENNETT, Keith H. A staged model for the software life cycle. **COMPUTER**, Los Alamitos, EUA, v. 33, n. 7, p. 66–71, jul. 2000.

RAMASWAMY, Ramkumar. How to staff business-critical maintenance projects. **IEEE Software**, Los Alamitos, EUA, v. 17, n. 3, p. 90–94, mai. 2000.

ROCHA, Ana Regina Cavalcanti da; MALDONADO, José Carlos; WEBER, Kival Chaves. **Qualidade de Software: Teoria e Prática**. 1. ed. São Paulo: Makron Books, 2001.

RUHE, Gunther; SALIU, Moshood Omolade. The art and science of software release planning. **IEEE Software**, Los Alamitos, EUA, v. 22, n. 6, p. 47–53, nov. 2005.

SANTOS, Rodrigo Pinheiro dos; OLIVEIRA, Kathia Marçal de; SILVA, Wander Pereira da. Evaluating the service quality of software providers appraised in CMM/CMMI. **Software Quality Journal**, v. 17, n. 3, p. 283–301, set. 2009.

SEI. **The Capability Maturity Model: Guidelines for Improving the Software Process**. Software Engineering Institute, Carnegie Mellon University, Reading, EUA: Addison-Wesley Publishing Company, 1995. 441 p. (SEI Series in Software Engineering).

SNEED, Harry M. A cost model for software maintenance & evolution. In: **20th IEEE International Conference on Software Maintenance (ICSM)**. Chicago, EUA: IEEE Computer Society, 2004. p. 264–273.

SNEED, Harry M. Offering software maintenance as an offshore service. In: **24th IEEE International Conference on Software Maintenance (ICSM)**. Beijing, China: IEEE Computer Society, 2008. p. 1–5.

SNEED, Harry M.; OPFERKUCH, Stefan. Training and certifying software maintainers. In: **12th IEEE European Conference on Software Maintenance and Reengineering (CSMR)**. Washington, EUA: IEEE Computer Society, 2008. p. 113–122.

SOFTEX. **MPS.BR - Melhoria de Processo do Software Brasileiro, Guia Geral**. 2009. Disponível em: `http://www.softex.br/mpsbr`. Acesso em: 15 abr. 2010.

SOLINGEN, Rini van. Measuring the ROI of software process improvement. **IEEE Software**, Los Alamitos, EUA, v. 21, n. 3, p. 32–38, mai. 2004.

SOMMERVILLE, Ian. **Software Engineering**. 8. ed. Reading, EUA: Addison Wesley, 2007.

SOUZA, Sergio Cozzetti B. de; ANQUETIL, Nicolas; OLIVEIRA, Káthia M. de. A study of the documentation essential to software maintenance. In: **23rd ACM International Conference on Design of Communication (SIGDOC)**. Coventry, Reino Unido: ACM, 2005. p. 68–75.

STARK, George E. Measurements for managing software maintenance. In: **8th IEEE International Conference on Software Maintenance (ICSM)**. Monterey, EUA: IEEE Computer Society, 1996. p. 152–161.

STARK, George E.; OMAN, Paul W. Software maintenance management strategies: observations from the field. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 9, p. 365–378, nov. 1997.

SVAHNBERG, Mikael *et al.* A systematic review on strategic release planning models. **Information and Software Technology**, Newton, EUA, v. 52, n. 3, p. 237–248, mar. 2010.

TAN, Yong; MOOKERJEE, Vijay S. Comparing uniform and flexible policies for software maintenance and replacement. **IEEE Transactions on Software Engineering**, Piscataway, EUA, v. 31, n. 3, p. 238–255, mar. 2005.

TAYLOR, M. J.; WOOD-HARPER, A. T. Methodologies and software maintenance. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 8, n. 5, p. 295–308, set. 1996.

TRAVASSOS, Guilherme Horta; KALINOWSKI, Marcos. **iMPS 2009 : caracterização e variação de desempenho de organizações que adotaram o modelo MPS**. Campinas, 2009.

VEHVILAINEN, Risto. What is preventive software maintenance? In: **16th IEEE International Conference on Software Maintenance (ICSM)**. San Jose, EUA: IEEE Computer Society, 2000. p. 18–19.

WARE, M. P.; WILKIE, F. G.; SHAPCOTT, M. The application of product measures in directing software maintenance activity. **Journal of Software Maintenance and Evolution: Research and Practice**, New York, v. 19, n. 2, p. 133–154, mar. 2007.

WEBER, Kival Chaves *et al.* **MPS.BR - Melhoria de Processo do Software Brasileiro: resultados alcançados e lições aprendidas (2004-2008)**. Campinas, 2008.

WEBSTER, Kenia P. Batista; OLIVEIRA, Kathia M. de; ANQUETIL, Nicolas. A risk taxonomy proposal for software maintenance. In: **21th IEEE International Conference on Software Maintenance (ICSM)**. Budapeste, Hungria: IEEE Computer Society, 2005. p. 453–461.