FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"
CENTRO UNIVERSITÁRIO "EURÍPIDES DE MARÍLIA" - UNIVEM
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

LUCIA EMI SHIRAISI SARTORI

MELHORIA DO PROCESSO DE TESTE PARA PEQUENAS EMPRESAS

MARÍLIA
2005

LUCIA EMI SHIRAISI SARTORI

MELHORIA DO PROCESSO DE TESTE PARA PEQUENAS EMPRESAS

Dissertação apresentada ao Programa de Mestrado do Centro Universitário Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, para obtenção do Título de Mestre em Ciência da Computação.

Orientador:
Prof. Dr. Auri Marcelo Rizzo Vincenzi

Co-orientador:
Prof. Dr. Edmundo Sergio Spoto

MARÍLIA
2005

SARTORI, Lucia Emi Shiraisi

Melhoria do Processo de Teste para Pequenas Empresas / Lucia Emi Shiraisi Sartori; orientador: Auri Marcelo Rizzo Vincenzi. Marília, SP: [s.n.], 2005.

184 f.

Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília - Fundação de Ensino Eurípides Soares da Rocha.

1. Engenharia de Software 2. Teste de Software 3. Processo de Teste.

CDD: << 005.14 >>

LUCIA EMI SHIRAISI SARTORI

MELHORIA DO PROCESSO DE TESTE PARA PEQUENAS EMPRESAS

Banca examinadora da dissertação apresentada ao Programa de Mestrado da UNIVEM,/F.E.E.S.R., para obtenção do Título de Mestre em Ciência da Computação. Área de Concentração: Engenharia de Software.

Resultado: _____

ORIENTADOR: Prof. Dr. Auri Marcelo Rizzo Vincenzi

1º EXAMINADOR: _____

2º EXAMINADOR: _____

Marília, 21 de Outubro de 2005.

Dedico este trabalho à meu esposo Sergio Sartori pelo seu amor, carinho, incentivo e paciência durante o período ao qual me dediquei para esta realização.
Aos meus pais, Eio e Missao (in memorian), que sempre me incentivaram a estudar e a enfrentar as dificuldades.
Ao meu chefe Edson Lucio Domingues, pelo incentivo e paciência.
A Marisa Silvana de Souza Andrade, minha amiga de todos as horas.

AGRADECIMENTOS

- Sou muito grata ao meu orientador, Prof. Dr. Auri Marcelo Rizzo Vincenzi, pelo seu encorajamento, interesse, estimulo e orientação durante este trabalho. Esta realização só foi possível devido sua iniciativa e valiosa instrução.
- Agradeço à Máquinas Agrícolas Jacto S.A. pela contribuição à este trabalho, fornecendo subsídios para a realização do curso. Sem esta colaboração não seria possível esta realização.
- Agradeço à meu esposo, Sergio, pelo sacrifício e companheirismo. Dividindo meus sucessos e desapontamentos durante estes anos.

SARTORI, Lucia Emi Shiraisi. **Melhoria do Processo de Teste para Pequenas Empresas**. 2005. 184 f. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília/SP, 2005.

RESUMO

A qualidade de um produto de software é dependente da qualidade do processo de desenvolvimento utilizado em sua construção. Nesses últimos anos inúmeros trabalhos têm buscado o desenvolvimento de normas e modelos visando a melhoria na qualidade de produtos de software, tais como o CMM (*Capability Maturity Model*), CMM-I (*Capability Maturity Model Integration*) e ISO-IEC 15504 (*SPICE*). Em paralelo e inspirados nesses modelos e normas também foram desenvolvidos os chamados Modelos de Maturidade de Teste que complementam os modelos de maturidade de processo, facilitando a avaliação e implantação de melhorias no que se refere à atividade de teste de software. Tanto os modelos de maturidade de processo quanto de teste foram conceitualmente desenvolvidos para grandes empresas, sendo difícil utilizá-los em sua totalidade no contexto de micro e pequenas empresas. Para facilitar a aplicação em pequenas empresas que dispõem de pessoal e recursos limitados, foram desenvolvidos modelos simplificados (ou que implantam as melhorias de forma mais cadenciada) de maturidade e avaliação de processo de software, tais como o PSP (*Personal Software Process*), P-CMM (*People CMM*) e MPS-Br (Melhoria de Processo do Software Brasileiro). Embora tais adaptações tenham sido realizadas para os modelos de maturidade de processo, o mesmo ainda não foi feito em relação aos modelos de maturidade de teste. Nesse trabalho é proposto um modelo, denominado Melhoria do Processo de Teste para Pequenas Empresas (MPT-PE), que visa a ser um modelo de referência para facilitar tanto a avaliação quanto a implantação de melhorias na área de teste de software no contexto de micro e pequenas empresas.

Palavras-chave: Engenharia de Software. Teste de Software. Processo de Teste.

SARTORI, Lucia Emi Shiraisi. **Melhoria do Processo de Teste para Pequenas Empresas**. 2005. 184 f. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília/SP, 2005.

ABSTRACT

The quality of a software product is dependent of the quality of its development process. In the last years a lot of work has been doing to development standards and models for improving the quality of software products, such as the CMM (*Capability Maturity Model*), the CMM-I *Capability Maturity Model Integration*) and the ISO-IEC 15504 (*SPICE - Software Process Improvement and Capability dEtertmination*). At same time and inspired by these models and standards, the so-called Testing Maturity Models have also been developed. They complement the process maturity models making easier the assessments and the implementation of improvements relate to software test activities. Both processes and testing maturity models were primary developed considering the context of big organizations what make difficult to use them in the context of small business organizations. To ease the application in small organizations, which, in general, have a limited number of people and resources, more simplified processes maturity models and evaluation methods have been proposed. Such simplified maturity models take in account the essential activities, discarding the others that are out of scope of a small business organization. Although such adaptations have already been done for the process maturity models, there is no research on this direction considering the testing maturity models. In this work it is proposed a model, named *Test Process Improvement for Small Enterprises (MPT-PE)*, as a reference model to turn easy evaluations and to make improvements in the testing area of small enterprises.

Keywords: Software Engineering. Software Testing. Test Process.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ATR	Activities, Tasks and Responsabilities
CAF	Capability Appraisal Framework
CMM	Capability Maturity Model
CMM-I	Capability Maturity Model Integrated
CV	Critical View
IEEE	Institute of Electrical and Electronic Engineers
ISO	International Organization for Standardization
KPA	Key Process Area
LOC	Line of Code
MAPPE	Modelo de Avaliação de Processo de Software para Pequena Empresa
MPSPE	Modelo de Processo de Software para Pequenas Empresas
MPS-Br	Melhoria de Processo do Software Brasileiro
MPT-PE	Melhoria do Processo de Teste para Pequenas Empresas
P-CMM	People - Capability Maturity Model
PSP	Personal Software Process
SEI	Software Engineering Institute
SPICE	Software Process Improvement and Capability dEtermination
STEP	Systematic Testing and Evaluation Process
SW-CMM	Capability Maturity Model for Software
SW-CMM KPA	Capability Maturity Model for Software - Key Process Area
SW-TMM	Software Testing Maturity Model
TAP	Testing Assessment Program
TMM	Testability Maturity Model
TMM-AM	Testability Maturity Model Assessment Model
TOM	Test Organization Maturity
TPI	Test Process Improvement

Capítulo

1

Introdução

A engenharia de software abrange todo o processo de produção e sua integração com a organização e com os clientes. Ela cuida da geração de produtos que devem competir no mercado mundial por sua qualidade, confiabilidade e preço. A qualidade de um produto ou serviço é medida pela satisfação total do consumidor. No seu conceito moderno a qualidade é entendida como a adequação às necessidades do cliente.

A prática tem mostrado que a geração de software de qualidade é fortemente dependente da qualidade do próprio processo de desenvolvimento. Nesse sentido, grande esforço internacional na busca e na padronização de modelos de avaliação de processo, como o CMM (*Capability Maturity Model*), CMM-I (*Capability Maturity Model Integration*), e ISO-IEC 15504 (*SPICE - Software Process Improvement and Capability dEtertmination*), buscam viabilizar um mecanismo de avaliação de processo e sugestão de melhorias. Independente do modelo de maturidade escolhido, as atividades de garantia de qualidade, tais como atividades de verificação, validação e teste (VV&T), estão presentes em todos eles, servindo como um mecanismo para assegurar que o que foi produzido se comporta como o desejado.

Entretanto, como observado por (Staab, 2002) e (Burnstein et al., 1996a), o enfoque dado pelos modelos de maturidade à implementação do processo de teste é vago, o que deu origem a pesquisas visando ao desenvolvimento dos chamados Modelos de Maturidade de Teste (*TMM - Testing Maturity Model*).

Basicamente, da mesma forma como o CMM-I é utilizado para avaliar o grau de maturidade de um processo de desenvolvimento de software de dada empresa e estabelecer metas para melhoria desse processo, os modelos de maturidade de teste visam a avaliar a maturidade do processo de teste empregado, indicando qual a qualidade desse processo e como este pode ser melhorado.

Nesse sentido, os modelos de maturidade de teste visam a complementar os modelos de maturidade de processo, considerando, especificamente, as atividades relacionadas à área de teste.

Os modelos de maturidade de processo e os modelos de maturidade de teste foram desenvolvidos para atender grandes empresas, que desenvolvem software grandes e complexos para diversos domínios de aplicação. De forma geral, tais modelos não são adequados para o contexto de micro e pequenas empresas as quais possuem número reduzido de funcionários e restrições de custos que inviabiliza a implantação de tais modelos de melhoria.

No Brasil, a grande maioria das empresas se enquadra na classificação de micro e pequena empresa, com até 49 pessoas, como pode ser observado na Tabela 1.1. Das 444 empresas avaliadas 273 (61%) delas representam micro e pequenas empresas.

Tabela 1.1: Porte das Organizações, Segundo Força de Trabalho Total e Efetiva - dez/2000. (MCT, 2000)

Porte		**Total**		**Efetivo**	
		Nº	%	Nº	%
Micro	De 1 a 5 pessoas	43	9,7	109	24,5
	De 6 a 9 pessoas	64	14,4	52	11,7
	Resumo	107	24,1	161	36,3
Pequena	De 10 a 49 pessoas	166	37,4	145	32,7
Média	De 50 a 99 pessoas	50	11,3	42	9,5
Grande	De 100 a 499 pessoas	73	16,4	54	12,2
	500 ou mais pessoas	48	10,8	42	9,5
	Resumo	121	27,2	96	21,6
Total das Organizações		444	100	444	100

Visando a atender essas micro e pequenas empresas, algumas iniciativas nacionais de adaptações de modelos de maturidade de processo vêm sendo propostas, tais como o MPSPE (Modelo de Processo de Software para Pequena Empresa) e o MPS-Br (Melhoria de Processo do Software Brasileiro). O MPSPE, é dividido em três etapas, Gerencial, Construção e Organização e a ordem estabelecida para a implantação dessas etapas foi baseada no CMM (Rosa, 1997), considerando as KPAs (áreas chaves de processo) que são aplicadas a micro e pequenas empresas.

Já o MPS-Br (Melhoria de Processo do Software Brasileiro) é uma iniciativa recente desencadeada por parte do Governo Federal em parceria com a Sociedade SOFTEX e

outras instituições visando ao desenvolvimento de um modelo de melhoria de processo voltado para micro e pequenas empresas (SOFTEX et al., 2003). Tal modelo foi desenvolvido de modo a ser compatível com o CMM-I e aderente à norma ISO-IEC 15504 mas adota uma estratégia mais gradativa para sua implantação tornando-se mais adequado para o contexto de micro e pequenas empresas mas não restrito somente a tais tipos de empresas. A meta estipulada para o MPS-Br é ter até o final do ano de 2006, 60 empresas avaliadas Nível G MPS-Br (o que corresponde ao Nível 2 do modelo CMM-I) (Rocha, 2005).

Entretanto, tal adaptação para micro e pequena empresas ainda não foi observada no que se refere aos modelos de maturidade de teste, motivando a realização deste trabalho. Na busca por melhoria de qualidade e aumento de sua competitividade no mercado interno e externo, as micro e pequenas empresas demandam modelos de processo de teste menos complexos, que contenham as práticas mais importantes e adequadas ao seu ambiente.

1.1 Objetivo

O objetivo deste trabalho é definir um modelo de Melhoria do Processo de Teste para Pequenas Empresas (MPT-PE) tendo como base o modelo TMM. Para isso, tal modelo deve levar em consideração as diretrizes essenciais do TMM, e ainda contar com algum mecanismo de avaliação que permita identificar o nível de maturidade em que dada empresa desenvolvedora de software se encontra, fornecendo subsídios para auxiliar a implantação de melhorias na área de teste.

Além disso, uma análise de ferramentas que apóiem os testes será realizada visando o aumento da qualidade e produtividade da atividade de teste.

1.2 Organização do Trabalho

No Capítulo 2, apresenta-se uma visão geral sobre Qualidade, Qualidade de Software e Qualidade de Processo. Essa visão geral é apresentada juntamente com os modelos de maturidade de processo tais como, CMM (*Capability Maturity Model*), CMM-I (*Capability Maturity Model Integration*), ISO-IEC 15504 (*Software Process Improvement and Capability dEtermination*), MPSPE (Modelo de Processo de Software para Pequena Empresa) e o MPS-Br (Melhoria de Processo do Software Brasileiro).

No Capítulo 3, é apresentada a definição de modelos de maturidade do processo de teste de software. Um histórico dos modelos existentes é apresentado e o mais consagrado deles,

o TMM (*Testability Maturity Model*), é descrito detalhadamente devido a sua importância no contexto deste trabalho.

No Capítulo 4, é apresentada a proposta de um modelo de maturidade de teste desenvolvido para pequenas empresas, denominado MPT-PE. São apresentadas as respectivas atividades, tarefas e responsabilidades (ATRs), ferramentas de teste, questionário e check-up de avaliação de processo de teste.

O Capítulo 5 apresenta as conclusões do trabalho desenvolvido e possíveis desdobramentos para trabalhos futuros.

Visando a tornar o trabalho autocontido, são fornecidos três apêndices. No Apêndice A são apresentadas as atividades, tarefas e responsabilidades (ATRs) do TMM. No Apêndice B ilustra o questionário de avaliação utilizado para identificar o nível de maturidade de teste de determinada empresa. Finalmente, o Apêndice C são dados exemplos de possíveis ferramentas que auxiliam na automatização de partes das atividades que devem ser realizadas durante a implementação do TMM.

Capítulo

2

Revisão Bibliográfica

Neste capítulo é dada uma visão geral da terminologia e conceitos básicos na área de qualidade de produto e processo de software. Alguns modelos de maturidade de processo tanto para grandes quanto para pequenas empresas são descritos brevemente visando a mostrar as principais iniciativas internacionais e nacionais nesse contexto.

2.1 Terminologia e Conceitos Básicos

Cada pessoa tem seu próprio conceito de qualidade (Grifo, 1994). Nesse sentido, qualidade é uma determinação do cliente, e não uma determinação da engenharia, do marketing ou do gerente geral. É baseada na experiência real do cliente com produtos ou serviços, medidos contra ou a partir dos requisitos dele ou dela, declarada ou não declarada, convencida ou compreendida, e representa o alvo em um mercado competitivo.

De um modo geral, qualidade de produtos e serviços pode ser definida como: "A combinação total das características de marketing, engenharia, produção, e manutenção através da qual o produto e serviço em uso devem atender as expectativas de um cliente" (Feigenbaum, 1983). A qualidade de um produto ou serviço é medida pela satisfação total do consumidor. O conceito de qualidade está sempre num equilíbrio entre os fatores: qualidade do próprio produto ou serviço, custo e atendimento (quantidade certa, local certo, hora certa).

Esta satisfação é buscada dentro dos programas de qualidade, tanto de forma defensiva (eliminando os fatores que desagradam o consumidor através da retroalimentação das informações que retornam do mercado), como de forma ofensiva (buscando antecipar as necessidades do consumidor e incorporando estes fatores no produto ou serviço) (Campos, 1989a).

No que se refere a produtos de software, a qualidade pode ser definida como um conjunto de atributos de software que devem ser satisfeitos de modo que o software atenda às necessidades dos usuários. A determinação dos atributos relevantes para cada software varia em função do domínio da aplicação, das tecnologias utilizadas, das características específicas do projeto e das necessidades da organização (Rosa, 1997).

A qualidade é uma combinação complexa de fatores que variam de acordo com diferentes aplicações e clientes. Os fatores são categorizados em dois grupos: fatores medidos diretamente (erros, por exemplo) e fatores medidos indiretamente (tais como usabilidade ou manutenibilidade) (Pressman, 2001).

De acordo com a norma NBR-ISO-IEC 9126 (ISO-9126, 1994), a qualidade de software é definida como a totalidade das características de um produto de software que lhe confere a capacidade de satisfazer as necessidades explícitas e implícitas de seus usuários. Existem diferentes visões de qualidade de software. Primeiro, na visão do usuário que avalia o software sem conhecer seus aspectos internos ou como o software foi desenvolvido. Um dos maiores interesses do usuário está na facilidade de uso, na confiabilidade, na eficiência, na portabilidade e no preço. Segundo, na visão da equipe de desenvolvimento, que se preocupa com o processo de desenvolvimento do software, avaliando aspectos internos como confiabilidade, facilidade de manutenção e se o software está em conformidade com seus requisitos. Terceiro, na visão do gerente que pode estar interessando mais na qualidade de forma geral do que em características específicas de qualidade, e pode também balancear os melhoramentos na qualidade com critérios gerenciais, tais como atraso no cronograma, estouro no orçamento, dentre outros (ISO-9126, 1994).

Independentemente da visão escolhida, pode-se dizer que a qualidade de um produto de software é totalmente dependente da qualidade de seu processo de desenvolvimento. Entende-se que processo é uma seqüência de operações para produzir algo. Uma empresa é um processo e dentro dela existem vários processos, não só processos de produção de bens como também de serviços.

O conceito de divisibilidade de um processo permite controlar sistematicamente cada um de seus subprocessos separadamente, podendo, desta maneira, conduzir a um controle mais eficaz sobre o processo todo. Controlando-se os processos menores (subproces-

sos) é possível localizar mais facilmente o problema e agir mais prontamente sobre sua causa (Campos, 1989b).

O controle da qualidade de um processo visa a estabelecer e melhorar continuamente um sistema de padrões atuando na causa fundamental de problemas detectados pela observação de itens de controle previamente selecionados. Procura-se, inicialmente, que o processo de produção seja estável e previsível. Após a estabilização das características da qualidade, deve-se buscar pelas melhorias no processo. Manter estável o processo de produção é uma atividade de rotina normalmente chamada de abordagem por sistemas. Já a realização de melhorias é uma abordagem por projetos, que visa a obter um processo cada vez mais definido e otimizado.

A melhoria de um processo se baseia na suposição de que o principal fator que influencia a qualidade é o processo de desenvolvimento do produto. O processo é aperfeiçoado com o objetivo de reduzir o número de defeitos do produto. A qualidade de processos de desenvolvimento de produto de software não é dependente de um processo de manufatura, mas de um processo de projeto, em que as considerações individuais humanas são importantes (Sommerville, 2003).

Dentro desse cenário, modelos de maturidade de processo de software tais como CMM (*Capability Maturity Model*), CMM-I (*Capability Maturity Model Integration*), ISO-IEC 15504 (*SPICE - Software Process Improvement and Capability dEtertmination*), MPSPE (Modelo de Processo de Software para Pequena Empresa) e MPS-Br (Melhoria de Processo do Software Brasileiro) foram definidos visando a fornecer subsídios para que as empresas possam tanto avaliar qual a qualidade de seus processos de desenvolvimento, quanto fornecer subsídios para que as mesmas possam melhorar tais processos de modo a produzirem produtos de software de melhor qualidade. A seguir, tais modelos de melhoria de processo de software são descritos brevemente.

2.2 Maturidade de Processo de Software

Processo é um sistema de operações que produz um produto ou serviço. É uma série de ações, mudanças ou funções que atinge um fim ou resultado. É um conjunto de atividades, métodos, práticas e transformações que as pessoas utilizam para desenvolver e manter um produto (ex: planos e documentos de projeto, código, casos de teste e manuais de usuário). À medida que uma organização vai se tornando madura, o processo de desenvolvimento de produto vai ficando melhor definido, possibilitando que o mesmo seja implementado de modo mais consistente em toda a organização.

Em uma organização dita imatura não há bases objetivas para a avaliação da qualidade do produto e nem para a resolução de problemas associados ao produto ou ao processo. Sendo assim, é difícil antever a qualidade do produto. As atividades que objetivam aumentar a qualidade, tais como revisões e testes, são freqüentemente reduzidas ou eliminadas em função dos atrasos ocorridos no andamento do projeto.

Por outro lado,uma organização madura possui habilidade para gerenciar o desenvolvimento do produto e os processos de manutenção do produto em toda a organização. O processo é claro e de conhecimento de todos. As atividades são realizadas de acordo com um planejamento cuidadoso. Os procedimentos são bem definidos e padronizados, sendo atualizados sempre que necessário. As melhorias são implementadas através de testes pilotos e resultados de análise de custo-benefício. As regras e as responsabilidades são claras em toda parte do projeto e da organização.

2.2.1 CMM - Modelo de Maturidade de Capacidade

O CMM é um modelo desenvolvido pelo Instituto de Engenharia de Software (SEI), para avaliar o processo, manter o controle do desenvolvimento do software e estabelecer programas de melhoria desse processo (Paulk et al., 1993).

O CMM estabelece um conjunto de critérios disponíveis ao público que descrevem as características de organizações de software maduras. Esses critérios podem ser utilizados por organizações para melhorar seus processos de desenvolvimento e manutenção de software, ou para avaliar os riscos de contratação para um projeto de software. O CMM estabelece uma estrutura comum de referência para as avaliações do processo de software e para as auditorias da capacidade do software. A estrutura do CMM consiste de cinco níveis de maturidade que determina em qual destes níveis se encontram os processos de software das empresas, identifica os pontos fortes e fracos e aponta caminhos para as melhorias necessárias, Figura 2.1.

Níveis da Maturidade

Um nível de maturidade representa um estágio evolutivo, desde o processo imaturo até o maduro. Cada nível indica um índice de capacidade do processo. Capacidade é a habilidade de um processo desempenhar sua missão de modo eficiente e confiável. Capacidade é a amplitude total da variação inerente a um processo estável, determinado utilizando-se de dados provenientes de controle estatístico. Os níveis definidos no CMM são:

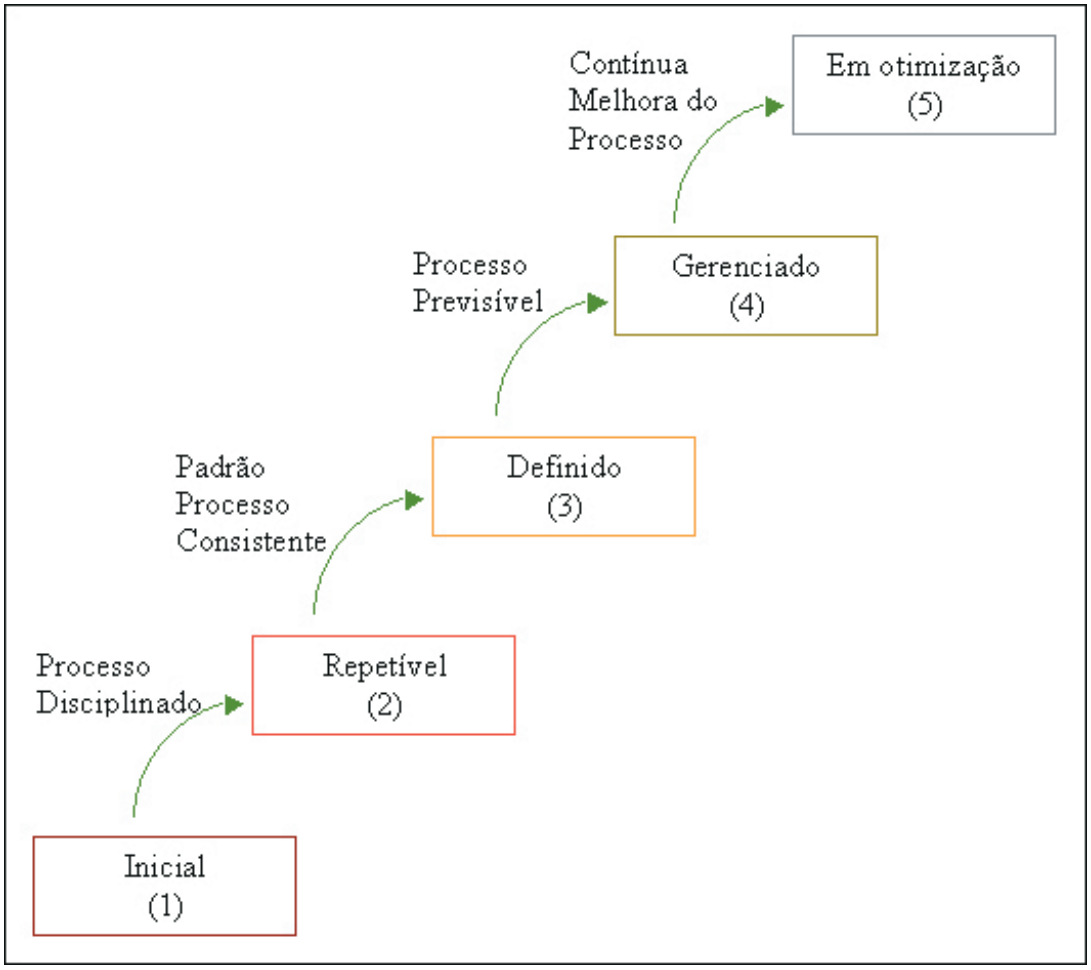

Figura 2.1: Os Cinco Níveis da Maturidade do CMM. (Paulk et al., 1993)

- ***Nível 1* - Inicial** - Processo *"ad hoc"*: O processo de desenvolvimento é informal, é uma entidade amorfa - ***uma caixa preta*** - e a visibilidade interna dos processos do projeto é limitada. Os requisitos fluem no interior do processo de uma forma descontrolada e surge o produto. Podem produzir produtos de sucesso, apesar de estarem com o orçamento e com o cronograma estourados. O sucesso do produto depende exclusivamente da competência e habilidade das pessoas.

- ***Nível 2* - Repetível** - Processo disciplinado. O processo de desenvolvimento do produto pode ser visualizado como uma ***sucessão de caixas pretas***, permitindo visibilidade da gestão nos pontos de transição como fluxo de atividades entre as caixas (marcas do projeto). O processo é dividido em etapas, em que os requisitos do cliente e o resultado da execução das tarefas são controlados no seu desenvolvimento. A gerência reage aos problemas quando os mesmos ocorrem.

- ***Nível 3*** **- Definido** - Padrão, processo consistente. Existe um processo de desenvolvimento padrão da organização. O processo de desenvolvimento de produto é definido, documentado, padronizado e integrado segundo a estrutura padrão. A ***estrutura interna das caixas pretas é visível***, isto é, as tarefas dentro do processo definido são visíveis. Como o processo é bem definido, a gerência tem uma boa percepção do progresso técnico em todos os projetos. ***Tanto as atividades de gestão como as de engenharia do produto são estáveis e repetíveis***.

- ***Nível 4*** **- Gerenciado** - Processo previsível. O foco é o ***controle de processo***. Os processos definidos são mensurados e controlados quantitativamente. Tem características do que na produção seriada é denominado de "controle estatístico de processos". O processo é gerenciado, operando de forma estável dentro de limites de qualidade pré-estabelecidos. Os gerentes são capazes de medir os progressos e os problemas. Eles possuem bases objetivas e quantitativas para tomadas de decisões. Suas habilidades de prever resultados crescem constantemente, tornando-se mais precisas à medida que a variação no processo diminui.

- ***Nível 5*** **- Em otimização** - Melhoria contínua do processo. Toda a organização está voltada para a melhoria contínua de processo. A organização tem meios de identificar as oportunidades de melhoria e fortalecer o processo de maneira pró-ativa, com o objetivo de prevenir a ocorrência de falhas. As inovações decorrentes das melhores práticas de engenharia são identificadas e transferidas para a organização toda. A melhoria contínua do processo é propiciada pelo "feedback" quantitativo do processo e pelas idéias e tecnologias inovadoras. Os gerentes são capazes de estimar e acompanhar quantitativamente o impacto e a eficiência da mudança.

Estrutura do Modelo

Com exceção do Nível 1, cada nível é decomposto em várias áreas-chave de processo, que indicam aquelas nas quais uma organização deve focar seus esforços para a melhoria de seus processos de desenvolvimento de produto.

Áreas Chave - Algumas áreas são imprescindíveis para a maturidade do processo. Elas são identificadas em função de sua eficácia e eficiência na melhoria da capacidade do processo. São consideradas como os requisitos para o nível de maturidade, isto é, para se obter um certo nível de maturidade, as áreas-chave de processo daquele nível devem ser satisfeitas. Elas representam um meio de se descrever o grau de maturidade de uma organização. Cada área-chave de processo identifica um grupo de atividades relacionadas,

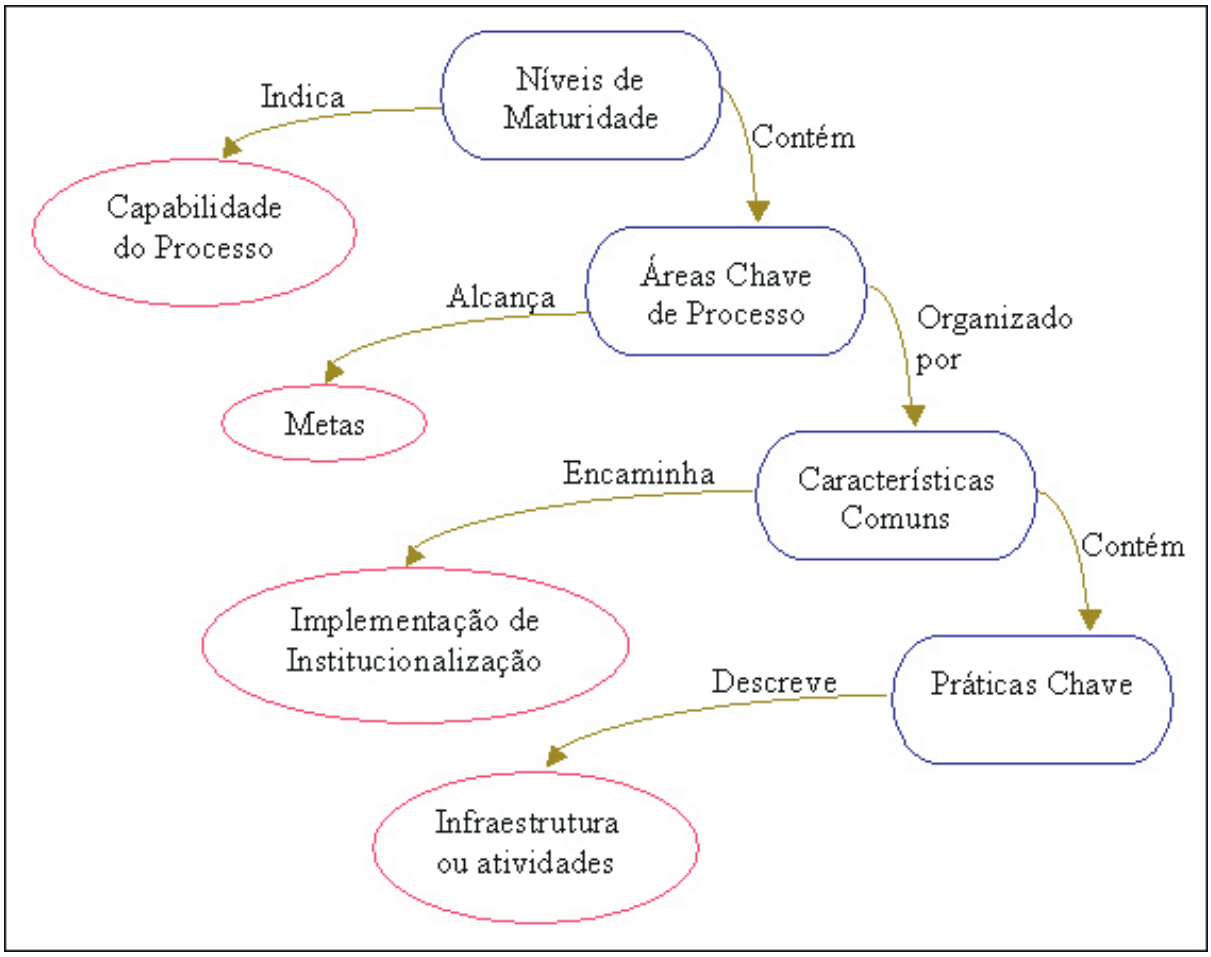

Figura 2.2: A Estrutura do CMM. (Paulk et al., 1993)

as quais, quando executadas coletivamente, atingem um conjunto de metas consideradas importantes para aumentar a capacidade do processo. Exemplo Figura 2.2.

2.2.2 CMM-I Modelo Integrado de Maturidade da Capacidade

Após o CMM surgiram vários outros modelos para cobrir áreas específicas de interesse tais como Engenharia de Sistemas (*SECM / EIA 731 - System Engineering Capability Model*), Desenvolvimento de Produto (*IPD-CMM - Integrated Product Development CMM*), Aquisição de Produto (*SA-CMM - Software Acquisition*), Administração de Equipes (*P-CMM - People CMM*) (Átila Belloquim, 2004).

Com o objetivo de integrar os diversos CMMs numa estrutura única com mesma terminologia e processos de avaliação, aproveitar a experiência de vários anos no uso do

CMM e também os compatibilizar com a ISO 15540 / SPICE, o SEI criou o modelo CMM-I (*Capability Maturity Model Integration*) (Átila Belloquim, 2004).

O CMM-I é disponível em duas representações: o modelo em estágios (herdado do CMM) e o modelo contínuo (herdado da ISO-IEC-15504). Há a possibilidade de escolher entre as duas diferentes abordagens para a melhoria de processos (Herndon et al., 2003).

No modelo em estágios o CMM-I apresenta os cinco níveis de maturidade organizacional: inicial, administrado, definido, gerenciado e otimização. A cada nível de maturidade é associado um certo número de áreas de processo. Cada área de processo contém metas e práticas relacionadas. É por um checklist de quais metas necessitam ser atingidas, quais atividades devem ser executadas, e quais ferramentas devem ser criadas e mantidas para satisfazer os requisitos para uma parte específica do processo de desenvolvimento (Academy, 2003).

A principal diferença com o CMM é no Nível 2, onde foi incluída uma nova área denominada Medição e Análise. Ela relaciona-se, principalmente, com o planejamento e controle de projetos. Refere-se a estimativas de esforço e cronograma, monitoramento do andamento, controle de requisitos e configuração (Hazan, 2004). Cada área-chave de processo situa-se em apenas um nível. Uma organização para estar no Nível 3 precisa cumprir todas as áreas do Nível 2 e todas as do Nível 3. Assim, uma organização no Nível 2 pode possuir práticas de níveis mais altos, mas ser apenas Nível 2 por não possuir o conjunto completo das áreas do nível mais alto (Átila Belloquim, 2004).

No modelo em estágios a organização é avaliada como estando em apenas num dos níveis de maturidade. No modelo contínuo há quatro categorias de áreas de processo - Administração de Projeto, Suporte, Engenharia e Administração de Processo. Cada uma destas categorias consiste de múltiplas áreas de processo. Cada área de processo contém metas e práticas genéricas relacionadas. São as mesmas áreas de processo do modelo em estágio, mas agrupadas por categoria e independente do nível (Herndon et al., 2003).

O grau que uma organização se encontra para padronizar e melhorar seus processos é determinado pela implementação das práticas genéricas. O modelo contínuo é baseado na implementação em degraus das práticas genéricas em cada área de processo, classificada por nível de capacidade - incompleto, executado, administrado, definido, gerenciado quantitativamente e em otimização (Herndon et al., 2003).

Assim, no modelo contínuo, cada área de processo possui características relativas a mais de um nível. Uma área-chave que no modelo em estágio pertence exclusivamente ao Nível 2, no modelo contínuo pode ter características que a coloquem em outros níveis (Átila Belloquim, 2004).

2.2.3 ISO/IEC 15504 – SPICE

Em outubro de 2003 a norma ISO/IEC 15504 (SPICE) para a avaliação de processos de software foi oficialmente publicada pela ISO. Esta norma define um modelo bi-dimensional que tem por objetivo a realização de avaliações de processos de software com o foco da melhoria dos processos e a determinação da capacidade dos processos. Gera um perfil dos processos, identificando os pontos fracos e fortes, que serão utilizados para a elaboração de um plano de melhorias. Viabiliza a avaliação de um fornecedor em potencial.

Esta norma estava em desenvolvimento desde 1993 pela ISO em conjunto com a comunidade internacional através do projeto SPICE (*Software Process Improvement and Capability dEtertmination*) com base nos modelos já existentes como ISO 9000 e CMM (ISO/IEC-15504, 2003). Ela é mais abrangente que o CMM e mais específica que a ISO 9000 (*International Organization for Standardization*) - Organização Internacional de Normalização.

Para executar uma avaliação, a organização deve escolher um avaliador capacitado, um modelo de processo compatível, e um método de avaliação compatível (Salviano, 2001). A Figura 2.3 ilustra o relacionamento destes principais elementos.

Figura 2.3: Modelo de Processo do Spice. (Salviano, 2001)

O modelo de referência descreve um conjunto de processos da engenharia de software considerados universais e fundamentais para a boa prática da engenharia de software. Quarenta processos e componentes de processos são descritos e organizados em cinco categorias de processo:

- Cliente - Fornecedor;
- Engenharia;
- Suporte;

- Gerência e
- Organização.

A norma 15504 possibilita a seleção de um subconjunto de processos chaves da organização direcionando a avaliação às características e necessidades específicas de uma empresa. No modelo de referência, cada processo é descrito pelo propósito e características que indiquem uma implementação do processo com sucesso, incluindo práticas bases.

Em uma determinada organização, cada um destes processos pode estar sendo excepcionalmente bem executado ou, apenas, parcialmente. Entre estes extremos, o grau de execução é dividido em seis níveis de capacidade. Estes níveis são seqüenciais e cumulativos. Eles podem ser utilizados:

- Como uma métrica para avaliar como uma organização está executando um determinado processo;
- Como determinação da capacidade dos processos, viabilizando a avaliação de um fornecedor potencial, obtendo o seu perfil de capacidade;
- Como um guia para a melhoria da execução: gerando um perfil dos processos, identificando os pontos fracos e fortes, que serão utilizados para a elaboração de um plano de melhorias.

Nível 0 **- Incompleto** - Existe uma falha geral na satisfação do propósito do processo. Os produtos resultantes dos processos existentes são poucos ou difíceis de serem identificados, porque o processo não está implantado ou geralmente não atinge seus objetivos.

Nesse nível não existe evidência de que os produtos de trabalho sejam adequadamente produzidos ou que os resultados sejam realmente alcançados (Rocha et al., 2001).

Nível 1 **- Executado** - Neste nível, o propósito do processo é geralmente alcançado embora possa não ser rigorosamente planejado e acompanhado. Existe uma concordância geral e informal entre as pessoas da organização de que uma ação deve ser executada e quando isto deve ser feito. Existem produtos do processo que evidenciam a satisfação do propósito do processo.

O processo geralmente atinge seus objetivos.

***Nível 2* - Gerenciado** - No Nível Gerenciado o processo produz produtos de acordo com procedimentos específicos. O processo é gerenciado com planejamento, acompanhamento e ajuste de suas atividades. Seus resultados são apropriadamente identificados, verificados e controlados. Os produtos estão em conformidade com os padrões e requisitos especificados.

A principal distinção entre este nível e o Nível Executado é que o processo gera produtos que satisfazem os parâmetros de qualidade, dentro do cronograma e recursos alocados.

O processo atinge seus objetivos e é gerenciado com planejamento, acompanhamento e ajuste de suas atividades e seus resultados são apropriadamente identificados, verificados, e controlados.

***Nível 3* - Estabelecido** - Neste nível o processo é definido, gerenciado e executado com base em princípios de uma boa engenharia de software. O processo é padronizado e documentado, descreve as principais atividades e técnicas, incluindo orientações para sua adaptação às necessidades específicas de cada execução. Os recursos para definição dos processos são disponibilizados.

A principal distinção deste nível e o Gerenciado, é a utilização de um processo padronizado.

***Nível 4* - Previsível** - O processo definido é executado consistentemente dentro de limites de controle definidos, para atingir as metas do processo. Medições detalhadas de desempenho são coletadas e analisadas, levando a um entendimento quantitativo da capacidade do processo e uma melhora na habilidade para prever e gerenciar a execução. A execução é gerenciada quantitativamente. A qualidade dos produtos é conhecida de forma quantitativa.

A principal distinção entre este nível e o Nível Estabelecido é que o processo é controlado dentro de limites pré-estabelecidos.

***Nível 5* - Otimizado** - O desempenho do processo é continuamente melhorado para satisfazer objetivos correntes e futuros de negócio, e o processo atinge repetitibilidade em atingir suas metas de negócio definidas. Objetivos quantitativos de eficiência e eficácia para o desempenho do processo são estabelecidos, baseados nos objetivos de negócio da organização. Um acompanhamento contínuo do processo em relação a estes objetivos é estabelecido pela obtenção de realimentações quantitativas e a melhoria é obtida a partir da análise dos resultados. A otimização contínua do processo envolve experiências de idéias e tecnologias inovadoras e a mudança de processos não efetivos para satisfazer as metas e objetivos definidos.

A principal distinção deste nível em relação ao Nível Previsível é que o processo padrão passa a ser alterado e adaptado para atingir de forma efetiva os objetivos correntes e futuros de negócio (Salviano, 2001).

Observa-se que os modelos CMM, CMM-I, ISO-IEC-15504 foram desenvolvidos visando à melhoria de processo em grandes empresas. Visando a fornecer modelos mais adequados para o contexto de micro e pequenas empresas, alguns modelos de melhoria de processos derivados dos modelos acima começaram a ser desenvolvidos. Dois desses modelos são descritos a seguir.

2.2.4 MPSPE - Modelo de Processo de Software para Pequena Empresa

O Modelo de Processo de Software para Pequena Empresa (MPSPE) foi desenvolvido em 1997 (Rosa, 1997) e é baseado nos modelos de processo de software CMM, SPICE e ISO 9000-3. Nele constam as atividades essenciais para a construção do software que são relevantes nos três modelos, mas são desconsideradas as de difícil implementação em empresas de pequeno porte.

O MPSPE é dividido em três áreas de atividades: Gerencial, Construção e Organização. A ordem de implantação de cada atividade é a indicada na Figura 2.4, e tem como objetivo facilitar a mudança de cultura da empresa.

Gerencial

É a primeira área de atividades a serem implantadas e tem por objetivo melhorar a parte administrativa da empresa. A maioria das pequenas empresas possui bons conhecimentos técnicos, mas é omissa em planejamento e administração.

Nesta etapa pretende-se melhorar a tarefa de planejamento dos projetos, quanto a recursos e cronogramas, iniciar um gerenciamento de configuração com um mínimo de formalismo, implantar documentação, revisão e métodos de garantia da qualidade.

- **Planejamento de Projeto** - compreende a determinação das atividades e tarefas a serem executadas, das estimativas de tempo e recursos necessários para cada uma delas, determinação de custos e prazos.
- **Gerenciamento de Configuração** - identifica, controla e relata as modificações que ocorrem enquanto o software está sendo desenvolvido e depois que ele é entregue

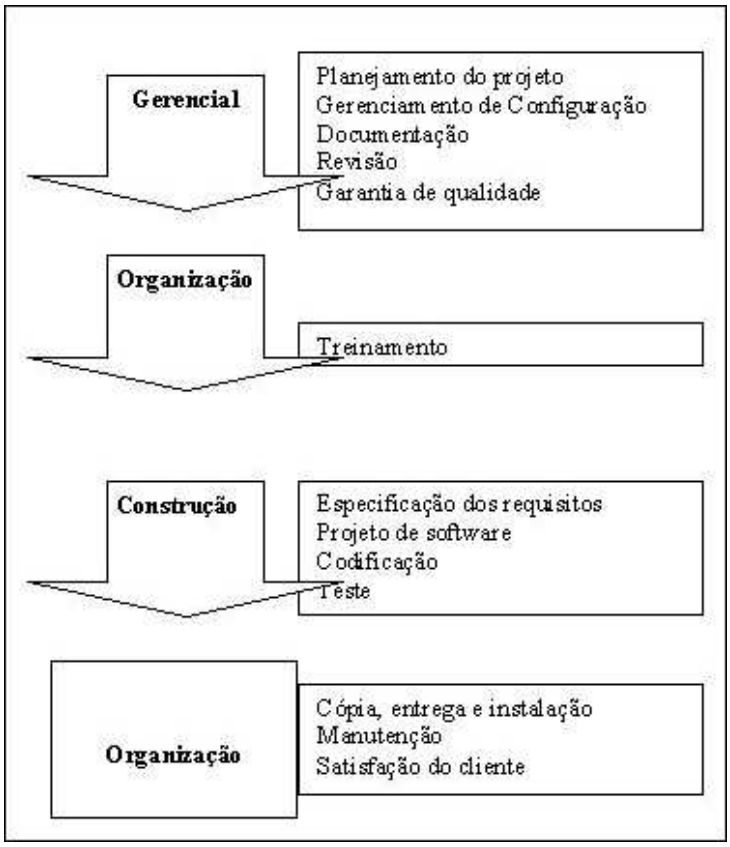

Figura 2.4: Visão Geral do MPSPE. (Rosa, 1997)

ao cliente. Este gerenciamento é realizado de forma a possibilitar o controle das mudanças.

- **Documentação** - institui os procedimentos para garantir que informações completas sobre o software estejam disponíveis para uso posterior. É o histórico do projeto.

- **Revisão** - deve ser realizada em cada etapa do processo de desenvolvimento do software para identificar erros, verificar se o software atende a seus requisitos, garantir que tenha sido representado de acordo com padrões pré-definidos. Objetiva descobrir erros precocemente e corrigi-los a um custo inferior.

- **Garantia de Qualidade** - estabelece procedimentos para garantir que as características da qualidade do produto sejam alcançadas. Por exemplo: funcionalidade, usabilidade e confiabilidade do software.

Construção

Refere-se a aspectos técnicos do desenvolvimento de software. É a institucionalização de um processo de software que inclui especificação dos requisitos, projeto de software, codificação, teste e manutenção.

- **Especificação de Requisitos** - detalhamento de todas as funções de software, características de interface, restrições do projeto e critérios de validação. Os requisitos de software devem ser estabelecidos de maneira precisa para permitir a validação durante a aceitação do produto.
- **Projeto de Software** - descrição das especificações da estrutura de dados, da arquitetura de software, do procedimento algorítmico e do projeto de interface.
- **Codificação** - representação do projeto numa linguagem artificial que resulta em instruções que são executadas pelo computador.
- **Teste** - aplicação de técnicas e estratégias para descobrir defeitos de função, lógica e implementação no software. Essa atividade permite que se remova defeitos antes do software ser entregue ao cliente.

Organização

Trata de aspectos de política da organização, treinamento dos funcionários, cópia, entrega e instalação, manutenção do sistema e da satisfação do cliente.

- **Treinamento** - desenvolvimento das habilidades individuais e conhecimentos necessários ao desempenho eficiente.
- **Cópia, Entrega e Instalação** - consiste em liberar o produto e manuais para o cliente e manter uma cópia contendo a versão e os manuais apropriados na empresa. Durante a atividade de instalação deve-se estabelecer claramente as funções, responsabilidades e obrigações do fornecedor e do comprador.
- **Manutenção** - permite que as modificações solicitadas sejam realizadas no sistema, no hardware e no software.
- **Satisfação do Cliente** - permite oferecer um bom atendimento ao consumidor durante e pós-vendas, além de um bom produto.

2.2.4.1 MAPPE - Modelo de Avaliação de Processo de Software para Pequena Empresa

Para apoiar a aplicação do MPSPE, bem como avaliar e indicar como implantar as melhorias, foi desenvolvido também por (Rosa, 1997) um modelo de avaliação de processo de software para pequena empresa, o MAPPE (Modelo de Avaliação de Processo de Software para Pequena Empresa), Figura 2.5, cuja estrutura é baseada na estrutura do SPICE.

Basicamente, o MAPPE é composto de 3 partes:

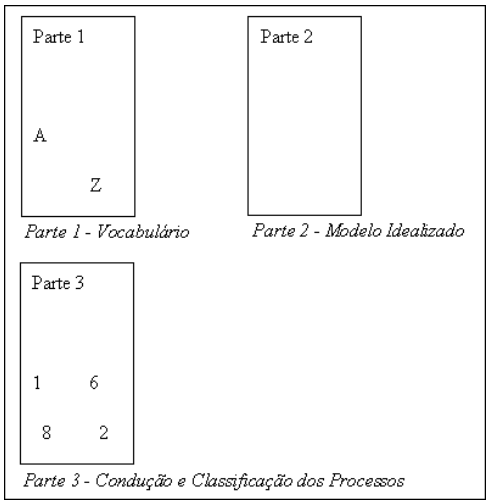

Figura 2.5: Partes Componentes do MAPPE. (Rosa, 1997)

Parte 1 - Vocabulário

Constituído de um dicionário com os termos utilizados no MAPPE.

Parte 2 - Modelo Idealizado

Apresentação do Modelo de Processo de Software para Pequena Empresa - MPSPE.

Parte 3 - Condução da Avaliação e Classificação dos Processos

Fornece orientações para avaliar e classificar os processos de software da empresa. O processo de software do projeto é avaliado individualmente pelas pessoas que participam do projeto. Para isto é utilizado um questionário que contém todas as atividades do Modelo de Processo de Software para Pequena Empresa (MPSPE) (Rosa, 1997).

2.2.5 MPS.Br - Melhoria de Processo do Software Brasileiro

Em dezembro de 2003 a SOFTEX - Associação para Promoção da Excelência do Software Brasileiro - iniciou o Projeto MPS.Br - Melhoria de Processo do Software Brasileiro.

O projeto não se propunha a definir algo novo no que se refere a normas e modelos, mas sim criar uma estratégia de implementação compatível com a realidade brasileira. É destinado preferencialmente a pequenas e médias empresas. Ele desenvolve um Modelo de Referência de Processo (MR-MPS) e um Modelo de Avaliação de Processo (MA-MPS), ambos compatíveis com o CMM-I e ISO/IEC 15504. Além disso, o MPS.Br possui também um modelo de negócio (MN-MPS) que descreve os mecanismos que podem ser utilizados pelas empresas para solicitarem a consultoria/implantação/avaliação do MPS.Br.

O modelo de referência MR-MPS é constituído de sete níveis de maturidade, seqüenciais e cumulativos. Cada nível de maturidade é composto por um conjunto de processos. Cada um dos processos, por sua vez, tem cinco níveis de capacidade, Figura 2.6.

Nível	**Processo**	**Capacidade**
A (mais alto)	Inovação e Implantação na Organização	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Análise e Resolução de Causas	
B	Desempenho do Processo Organizacional	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência Quantitativa do Projeto	
C	Análise de Decisão e Resolução	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência de Riscos	
D	Desenvolvimento de Requisitos	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Solução Técnica	
	Integração do Produto	
	Instalação do Produto	
	Liberação do Produto	
	Verificação	
	Validação	
E	Treinamento	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Avaliação e Melhoria do Processo Organizacional	
	Definição do Processo Organizacional	
	Adaptação do Processo para Gerência de Projeto	
F	Medição	AP 1.1, AP 2.1 e AP 2.2
	Gerência de Configuração	
	Aquisição	
	Garantia da Qualidade	
G	Gerência de Requisitos	AP 1.1 e AP 2.1
	Gerência de Projeto	

Figura 2.6: Modelo de Referência (MR-MPS). (Weber et al., 2005)

Os níveis de maturidade estabelecem estágios de evolução de processos: A (em otimização), B (gerenciado quantitativamente), C (Definido), D (largamente definido), E (parcialmente definido), F (gerenciado) e G (parcialmente gerenciado). O Nível G é o mais imaturo e A é o mais maduro. Os níveis F, C, B e A correspondem respectivamente aos níveis 2, 3, 4, e 5 do CMM-I. O Nível G é intermediário entre os níveis 1 e 2 do CMM-I, e os níveis E, D são intermediários entre os níveis 2 e 3. O maior número de níveis possibilita uma implantação mais gradual e com visibilidade dos resultados em prazos menores, tornando-o mais adequado às pequenas e médias empresas.

A capacidade do processo possui cinco atributos de processos: AP 1.1 (o processo é executado), AP 2.1 (o processo é gerenciado), AP 2.2 (os produtos são gerenciados), AP 3.1 (o processo é definido), AP 3.2 (o processo padrão está implementado). Estes atributos correspondem às práticas genéricas do CMM-I e atributos de processo da ISO/IEC 15504.

O modelo de avaliação MA-MPS é composto pelos requisitos do método, atividades do método, indicadores de avaliação e características da qualificação dos avaliadores. O processo de avaliação é composto por quatro fases: Preparação e planejamento, Condução da avaliação, Resultados e Certificação (Weber et al., 2005).

2.3 Conclusão

Neste capítulo, foi dada uma visão geral sobre Qualidade, Qualidade de Software e Qualidade de Processo. Foram descritos alguns modelos de processo tais como CMM, CMM-I e ISO/IEC 15504 (SPICE), desenvolvidos principalmente para serem utilizados no contexto de grandes empresas.

Além disso, considerando micro e pequenas empresas, foram descritos sucintamente dois modelos mais adequados para esse contexto o MPSPE e o MPS-Br, desenvolvidos com base nos modelos de maturidade citados acima.

Conforme identificado por Staab e Burnestein, os modelos de melhoria de processo, tais como o CMM, CMM-I e ISO-IEC-15505 não contemplam adequadamente a definição de processos de teste. Utilizando o mesmo raciocínio e considerando que os modelos MPSPE e MPS-Br, por serem derivados do CMM, CMM-I e ISO-IEC-15505 sofrem do mesmo problema, sendo pertinente a adaptação do TMM para o contexto de micro e pequenas empresas.

No capítulo seguinte o Modelo de Maturidade de Teste TMM é descrito mais detalhadamente por ser ele a base para o desenvolvimento do modelo de Melhoria de Processo de Teste para Pequenas Empresas definido neste trabalho.

Capítulo

3

Maturidade do Processo de Teste: O TMM

Devido à importância da atividade de teste, os modelos de maturidade de teste começaram a ser desenvolvidos. Neste capítulo, inicialmente, é apresentado um histórico do surgimento desses modelos de maturidade. Em seguida, o modelo de maturidade de teste mais tradicional, denominado TMM (*Testing Maturity Model*), é descrito detalhadamente e confrontado com o CMM.

3.1 Histórico

O histórico apresentado a seguir foi extraído do artigo "Growth of maturity in the testing process" (Drabick, 1999).

Durante as décadas de 60, 70 e 80 a atividade de teste era executada no fim do ciclo de desenvolvimento, normalmente pelos próprios programadores. O foco era avaliar se o sistema estava funcionando e o objetivo era procurar erros. Não havia ainda técnicos especializados na atividade de testar aplicações nem metodologias estruturadas de teste. Havia inúmeras ferramentas CASE para o pessoal de desenvolvimento, mas os testadores eram forçados a continuar com métodos manuais.

Foi em 1979 que Glenford Myers, publicou o livro "The Art of Software Testing", onde entre outras coisas, lembrava que testar era procurar defeitos e não provar que o software estava funcionando.

David Gelperin e Bill Hetzel, em seu artigo "The Growth of Software Testing" lançaram a idéia de um Plano de Testes, que deveria ser escrito a partir dos requisitos do sistema. Isto por si só já ajudava a reduzir a quantidade de defeitos dos sistemas, dando aos testadores os objetivos a serem alcançados durante a atividade de teste.

No fim da década de 80, a responsabilidade pelos testes passou para os grupos de garantia de qualidade. Recentemente, essa responsabilidade coube aos grupos de teste, quando a atividade de testar passou a ser parte de um processo independente do processo de desenvolver, embora continuassem integradas.

Outros trabalhos visaram ao desenvolvimento dos chamados modelos de maturidade de teste de software e sistemas. Em 1996 Rodger Drabick publicou na QA Magazine artigos iniciais sobre o modelo "Formal Software Process" e o modelo completo foi apresentado na "Testing Computer Software Conference", em Washington - DC em 1996.

Em maio de 1996, Cavid Gelperin e Aldin Hayashi publicam um artigo na "Application Development Trends" intitulado "How to Support Better Software Testing". Este artigo descrevia um Modelo de Maturidade de Teste para avaliação de processo de teste.

Neste mesmo ano os doutores Ilene Burnstein, C. Robert Carlson e Taratip Suwannasart apresentaram o trabalho "Development of a Testing Maturity Model" na "Quality Week Conference". Simultaneamente, Susan Burgess e Rodger Drabick desenvolveram o "Testing Capability Maturity Model".

Estes três modelos de maturidade de teste têm em comum a base estabelecida no modelo "Capability Maturity Model (CMM)" para engenharia de software, desenvolvido pelo "Software Engineering Institute" da "Carnegie Mellon University". Todos os modelos de maturidade de teste foram projetados para complementar o CMM (Drabick, 1999). Os modelos foram desenvolvidos para serem utilizados em conjunto com os modelos de maturidade de processo de software, mas também podem ser empregados em separado, facilitando a introdução da atividade de teste dentro de uma empresa.

Desde o início dos estudos, pelo menos seis modelos de testes de maturidade foram definidos. A maioria desenvolvida por volta de 1996, mas nunca houve muita aceitação dos mesmos. Uma das razões da baixa aceitação é que há pouca documentação de aplicações dos modelos.

1. Modelo de Maturidade de Testabilidade - TMM (Testability Maturity Model) (Gelperin, 1996).

2. Modelo de Maturidade de Teste de Software - SW-TMM (Software Testing Maturity Model) (Staab, 2002).

3. Melhoria do Processo de Teste - TPI (Test Process Improvement) (Kooman e Pol, 2002).

4. Maturidade da Organização de Teste - TOM (Test Organization Maturity) (Evolutif, 2002).

5. Programa de Avaliação de Teste - TAP (Testing Assessment Program) (Software-Futures, 2002).

6. Modelo de Maturidade de Capacidade de Software - Áreas Chaves de Processo (Capability Maturity Model for Software - Key Process Areas) (SoftwareTest, 2002).

Mais recentemente essa situação começa a se modificar. Dos seis modelos mencionados acima, o que mais tem se destacado é o TMM (Staab, 2002). Staab tem escrito diversos artigos sobre o tema. Além disso, Burnstein (2003) lançou o livro "Practical Software Testing" que aborda em profundidade o TMM. A seguir, o TMM é descrito mais detalhadamente.

3.2 O Modelo de Maturidade de Teste de Software - TMM

Os modelos de avaliação da maturidade do processo de desenvolvimento de software tais como, CMM, CMM-I e ISO-IEC-15504 não tratam adequadamente o processo de teste, razão pela qual o Illinois Institute of Technology criou o TMM - *Test Maturity Model*. O objetivo do TMM é dar suporte às organizações na melhoria do processo de teste.

O TMM é um modelo de avaliação do processo de teste de software, constituído de cinco níveis evolutivos de maturidade, conforme ilustrado na Figura 3.1. É baseado no modelo CMM. A estrutura operacional do TMM provê uma seqüência de **níveis hierárquicos** que contêm metas, submetas e atividades, tarefas e responsabilidades (ATRs) que definem a capacidade de teste da organização. Elas identificam áreas que a organização deve focar para melhorar seu processo de teste.

O TMM é usado para:

- uma equipe de avaliação interna identificar o estado atual de capacidade de teste.
- a administração superior iniciar um programa de melhoria de teste.

- as equipes de desenvolvimento melhorarem a capacidade do teste.
- os usuários e clientes definirem seus papéis no processo de teste.

Níveis	*Metas*	*Descrição da meta*	*Explicações*
1		Teste normalmente feito pela equipe de desenvolvimento de forma rudimentar.	A atividade de testar é um processo caótico e não existe uma diferença entre testar e buscar erros. O teste é executado sem ferramentas, equipes e outros recursos.
2	1	Desenvolver os objetivos de teste.	O propósito do teste é mostrar que o software funciona, e que atende às especificações.
	2	Iniciar um processo de planejamento de teste.	
	3	Institucionalizar técnicas e métodos básicos de teste.	
3	1	Estabelecer uma organização de teste de software.	O propósito do teste é mostrar que o software não funciona.
	2	Estabelecer um programa de treinamento.	
	3	Integrar o teste no ciclo de vida do software.	
	4	Controlar e monitorar o teste.	
4	1	Estabelecer um programa amplo de revisão.	O propósito do teste não é provar nada, mas reduzir os riscos.
	2	Estabelecer um programa de medições do processo de teste.	
	3	Avaliação da qualidade de software.	
5	1	Aplicação de dados de processos para prevenção de defeitos.	A atividade de testar não é um ato, mas uma disciplina mental que resulta em baixos riscos para o software com pouco esforço de teste.
	2	Controle de qualidade.	
	3	Otimização do processo de teste.	

Figura 3.1: Níveis e Metas do TMM. (Rios, 2000)

3.2.1 Os Níveis do TMM

Nível 1 **: Inicial** - Testar é um processo caótico, mal definido e não distingüido da depuração. Os testes são desenvolvidos por prática (*"ad hoc"*) com a finalidade de mostrar

que o sistema e o software funcionam. Usualmente há falta de pessoal treinado em teste e faltam recursos de ferramentas de testes.

***Nível 2* : Fase de Definição** - O teste é função separada da depuração. É uma atividade planejada, que ocorre após o termino da codificação. O objetivo do teste é mostrar que o sistema e o software satisfazem as especificações. As técnicas e critérios básicos de teste estão alocados. Muitos problemas de qualidade ocorrem porque como o planejamento de teste ocorre tardiamente no ciclo de vida do software, os defeitos se propagam entre as fases de requisitos, projeto e codificação, aumentando os custos para a correção dos mesmos. Ou seja, a execução de teste somente após a codificação é ainda considerada uma atividade de teste primária.

***Nível 3* : Integração** - O teste não é mais uma etapa que segue a codificação; ele é integrado no ciclo de vida de software. Os objetivos de testes são agora estabelecidos em relação aos requisitos baseados nas necessidades do usuário. Existe uma organização formal de teste e o teste é reconhecido como uma atividade profissional. Há treinamento técnico com foco em testes. São utilizadas ferramentas básicas de teste como suporte às atividades chave de teste. Há revisões, mas estas ainda não ocorrem durante todo o ciclo de vida.

***Nível 4* : Gestão e Medições** - O processo de teste é medido e quantificado. Há revisões em todas as etapas do processo de desenvolvimento reconhecidas como teste e controle de qualidade. Os produtos software são testados quanto aos atributos de qualidade, confiabilidade, usabilidade e manutenibilidade. Casos de teste de todos os projetos são coletados e registrados em um banco de dados de teste para reuso e teste de regressão. Aos defeitos encontrados durante os testes são atribuídos níveis de severidade e prioridade para correção.

***Nível 5* : Otimização, Prevenção de Defeito e Controle de Qualidade** - O processo de teste é bem definido e os custos administrativos e efetividade podem ser monitorados. Há mecanismos para ajustes finos e melhoria contínua dos testes. São praticadas as técnicas de prevenção de defeitos e controle de qualidade. O processo de teste é direcionado por amostras estatísticas, medidas de níveis de confiança e confiabilidade. Há procedimentos estabelecidos para a seleção de ferramentas de teste e avaliação. Ferramentas automatizadas suportam todos os processos de teste (Burnstein et al., 1996b).

Cada nível do TMM tem associado metas de maturidade que identificam objetivos de melhorias de teste que devem ser cumpridos para atingir a maturidade naquele nível (vide Figura 3.1). O Nível 1 como Nível Inicial não possui nenhuma meta associada.

O TMM, assim como o CMM, também possui um modelo de validação, baseado num questionário, que verifica o nível de maturidade da área de Tecnologia da Informação em relação ao teste. Definido o nível, a equipe de testes deverá se submeter a um processo de migração para um nível superior de maturidade, o que acarretará uma melhoria no processo de testes. Para isto, algumas práticas devem ser adotadas baseadas nas metas estabelecidas para o nível a ser alcançado (Rios, 2000).

Cada meta é dividida nos seguintes elementos:

- Submetas de maturidade;
- Visão dos gerentes;
- Visão dos desenvolvedores e testadores;
- Visão dos usuários.

3.2.2 Descrição das Metas e Submetas

Nível 2 - Fase de Definição

Meta 1 - Desenvolver os Objetivos de Teste

A organização deve distinguir claramente entre processo de teste e depuração. A separação destes dois processos é essencial para o crescimento da maturidade de teste, desde que eles são diferentes em metas e métodos.

As submetas de maturidade incluem:

- A organização que deve formar um ou mais comitês sobre teste e depuração com suporte e recursos.
- Os comitês que devem desenvolver e registrar as metas de teste.
- Os comitês que devem desenvolver as metas de depuração.
- As metas documentadas de teste e de depuração que devem ser distribuídas por todos os gerentes de projeto e desenvolvedores.
- As metas de teste que devem ser refletidas nos planos de teste.

Meta 2 - Iniciar um Processo de Planejamento de Teste

O planejamento de teste envolve estabelecer objetivos, analisar os riscos, traçar estratégias e desenvolver especificações do projeto de teste e casos de teste. Ainda o plano de teste, designar recursos e atribuir responsabilidades de teste da unidade, integração, sistema e aceitação.

As submetas de maturidade incluem:

- Estabelecer um comitê de planejamento de teste para toda a organização, com recursos adequados.
- Estabelecer uma política de plano de teste para toda a organização, apoiada pela administração.
- Desenvolver um modelo de planejamento de teste, registrar e distribuir para os gerentes de projeto.
- Treinar os gerentes de projeto e desenvolvedores para usar o modelo e desenvolver o plano de teste.
- Implantar um procedimento que inclua os requisitos de usuários como entrada para o plano de teste.
- O planejamento básico deve ser avaliado, recomendado e aprovado. A administração deve apoiar o uso.

Meta 3 - Institucionalizar Técnicas e Métodos Básicos de Teste

Para melhorar a capacidade do processo de teste, técnicas e métodos básicos devem ser aplicados por toda a organização. Políticas para estas técnicas e ferramentas básicas de suporte devem ser claramente especificadas. Exemplos de métodos e técnicas básicas são as estratégias de caixa-preta e caixa-branca, uso de matriz de validação de requisitos, e teste de unidade, integração, sistemas e aceitação.

As submetas de maturidade incluem:

- A administração que deve instituir um conjunto de políticas que assegure que métodos e técnicas recomendados sejam consistentemente aplicados por toda a organização.

- A administração que deve formar um grupo de tecnologia para a empresa, a fim de estudar, avaliar e recomendar um conjunto de técnicas e métodos básicos de teste, e recomendar um conjunto de ferramentas simples para suportá-los.

Nível 3 - Integração

O modelo-V é citado na Meta 3 abaixo, descreve graficamente as fases individualmente em forma de "V". Neste caso "V" vem de verificação e validação; as atividades são ordenadas de forma seqüencial em níveis de abstrações de modo que as conexões com as fases de desenvolvimento ficam extremamente claras (Molinari, 2003).

Meta 1 - Estabelecer uma Organização de Teste de Software

Uma organização de teste de software é criada para identificar um grupo de pessoas que são responsáveis por teste. O grupo de teste é responsável pelo planejamento, execução e registro de teste, pelos padrões relativos a teste, métricas de teste, banco de dados de teste, reuso de teste, acompanhamento e avaliação.

As metas de maturidade incluem:

- Estabelecer um grupo de teste amplo empresarial com liderança, apoio e recursos da administração superior. Deve definir suas funcionalidades e subordinação.
- Definir regras e responsabilidades para o grupo de teste.
- Designar para o grupo membros bem treinados e motivados.
- O grupo de teste deve estabelecer conexões de comunicação, que consultam os clientes e os usuários participantes das atividades de teste, para ajudar a coletar dados e documentos, e incorporar as necessidades, desejos e requisitos do usuário no processo de teste.

Meta 2 - Estabelecer um Programa de Treinamento

Os testadores devem ser adequadamente treinados de modo a que eles possam desempenhar suas funções eficiente e eficazmente. Neste nível, o pessoal é treinado em planejamento de teste, métodos, padrões, técnicas e ferramentas de teste. O programa de treinamento também prepara o pessoal para o processo de revisão, instrui lideres de revisão e institui canais para a participação de usuários no processo de testes e revisões.

O treinamento inclui cursos internos, aprendizado autodidata, programa de mentores e suporte de instituições acadêmicas.

As submetas de maturidade incluem:

- A administração que deve estabelecer um programa da organização para treinamento, prover recursos e suporte.
- O comitê de treinamento técnico que deve desenvolver e distribuir uma política da organização para treinamento.
- Desenvolver metas e planos de treinamento com entradas dos gerentes de projeto.
- Estabelecer um grupo de treinamento interno, com ferramentas, facilidade e materiais no local.

Meta 3 - Integrar o Teste no Ciclo de Vida do Software

O planejamento de teste deve ser iniciado cedo no ciclo de vida. Os testadores e desenvolvedores usam uma variação do modelo-V (V-Model). As entradas de usuário no processo de teste são solicitadas através de canais estabelecidos para as diversas etapas de teste.

As submetas de maturidade incluem:

- Dividir a etapa de teste em sub-etapas que devem ser integradas no ciclo de vida de software, seguindo uma política organizacional escrita, e revisada com a administração.
- Desenvolver e adotar uma versão institucionalizada do modelo-V.
- Alocar recursos para apoiar a integração da atividade de teste no ciclo de vida de software.
- Definir padrões para teste de produtos e medida de sua qualidade.
- Estabelecer um procedimento que permita aos testadores trabalharem com os desenvolvedores para facilitar as atividades de teste.

Meta 4 - Controlar e Monitorar o Teste

As atividades de controle e monitoramento provêem visibilidade e asseguram que os processos de teste procedam de acordo com o plano. O suporte para o controle e monitoramento vem de padrões de teste de produtos, marcos e métricas de teste que podem ser usados para avaliar o progresso e a efetividade do teste.

As metas de maturidade incluem:

- Desenvolver mecanismos e políticas para controlar e monitorar o teste.
- Definir, registrar e distribuir um conjunto de métricas relativas ao teste.
- Desenvolver um conjunto de ações corretivas e de planos de contingência, registrados e documentados para uso quando o teste desvia significativamente do planejamento.

Nível 4 - Gestão e Medições

Meta 1 - Estabelecer um Programa Amplo de Revisão

No Nível 3, uma organização integra as atividades de teste no ciclo de vida. No Nível 4, esta integração é ampliada pelo estabelecimento de um programa de revisão. Revisões são conduzidas em todas as fases do ciclo de vida para identificar, catalogar e remover defeitos de software o mais cedo possível no desenvolvimento do produto.

As metas de maturidade incluem:

- A administração superior que deve desenvolver políticas de revisão, apoiar o processo de revisão, e assumir a responsabilidade para integrá-las na cultura organizacional.
- O grupo de teste e o grupo de garantia de qualidade de software que devem desenvolver e documentar metas, procedimento de acompanhamento, e mecanismos de registro das revisões ao longo do ciclo de vida.
- Os grupos acima que devem especificar os itens para revisão.
- O pessoal que deve ser treinado de modo a compreender e seguir adequadamente as políticas, práticas e procedimentos de revisão.

Meta 2 - Estabelecer um Programa de Medições de Teste

Um programa de medida de teste é essencial para avaliar exatamente a qualidade do processo de teste, a produtividade de pessoal e o progresso da melhoria do processo

de teste. Medidas para cada fase do ciclo de vida devem ser especificadas. As medidas incluem progresso do teste, custo de teste, dados sobre erros e defeitos, medidas de produto tais como confiabilidade de software.

As submetas de maturidade incluem:

- Definir políticas e metas organizacionais para métricas do processo de teste.
- Desenvolver um plano de medida do processo de teste, com mecanismos de coleta de dados, análises e aplicação.
- Desenvolver e documentar um plano de ação que aplica resultados medidos para melhoria do processo de teste.

Meta 3 - Avaliação da Qualidade de Software

Avaliação de qualidade de software envolve definir atributos mensuráveis da qualidade do software produzido. A qualidade do produto é dependente da adequação da qualidade do processo de teste, porque um processo de teste maduro deve conduzir para um software que ao menos seja correto, confiável, usável, portável, seguro e de fácil manutenção.

As submetas de maturidade incluem:

- A administração e os grupos de garantia de qualidade de teste de software que devem definir política, metas e atributos relativos à qualidade do produto software.
- A organização que deve desenvolver, documentar e apoiar procedimentos e políticas de avaliação da qualidade de software.
- O processo de teste que deve ser estruturado, medido e avaliado para assegurar que as metas da qualidade possam ser atingidas.

Nível 5 - Otimização, Prevenção de Defeitos e Controle da Qualidade

Meta 1 - Prevenção de Defeito

Organizações neste nível registram defeitos, analisam padrões de defeitos e identificam causas de erros. Planos de ação são desenvolvidos, ações são realizadas para prevenir

recorrência de defeitos e há um mecanismo de acompanhamento do progresso das ações. Prevenção de defeitos é aplicada em todos os projetos e em toda a organização.

As submetas de maturidade incluem:

- A organização que deve desenvolver e documentar, e apoiar procedimentos e políticas para prevenção de defeitos.
- Um time de prevenção de defeitos que deve ser formado com apoio gerencial.
- Defeitos introduzidos ou removidos que devem ser identificados e registrados durante cada fase do ciclo de vida.
- Mecanismos de análise que devem ser estabelecidos para identificar as causas de defeito.
- Planos de ação que devem ser desenvolvidos através de interação de gerentes, desenvolvedores e testadores para prevenir a recorrência de defeitos identificados.

Meta 2 - Controle de Qualidade

Neste nível organizações usam amostragem estatística, medidas de níveis de confiança e confiabilidade para conduzir o processo de teste. O custo para conseguir qualidade é medido pelo custo da não qualidade (custo das falhas e correções).

As submetas de maturidade incluem:

- A organização que deve desenvolver, documentar, e apoiar procedimentos para o controle da qualidade.
- O grupo de teste de software e o grupo de garantia de qualidade que devem estabelecer metas para a qualidade do produto tais como unidades de produto defeituoso, níveis de confiança e confiabilidade.
- Gerentes de teste que devem incorporar estas metas de qualidade em seus planos.
- Os grupos de teste que devem ser treinados em métodos estatísticos.
- Entradas de usuários que devem ser coletadas para modelagem de uso.

Meta 3 - Otimização do Processo de Teste

No mais alto nível do TMM, o processo de teste está sujeito a melhorias contínuas através dos projetos e da organização. O processo de teste é quantificado e pode ser ajustado de modo que o crescimento da capacidade seja um processo contínuo. Otimização do processo de teste envolve: identificação das práticas de teste que podem ser melhoradas, implementação de melhorias, acompanhamento do progresso de melhorias e avaliação contínua de novas ferramentas de teste e tecnologias para adaptação.

As submetas de maturidade incluem:

- A organização que deve desenvolver, documentar e apoiar procedimentos e políticas para otimização de processo de teste.
- Um grupo de melhoria de processo de teste que deve ser estabelecido para monitorar o processo de teste e identificar áreas de melhorias.
- Um mecanismo que deve ser introduzido para avaliar novas ferramentas e tecnologias que podem melhorar a capacidade e a maturidade do processo de teste.
- A efetividade do processo de teste que deve ser continuamente avaliada. As decisões de quando parar os testes devem ser feitas de maneira mensurável e ótima, e serem relacionadas com as metas de qualidade (Swinkels, 2000).

3.2.3 Atividades, Tarefas e Responsabilidades - ATRs

Associada com cada meta de maturidade há um conjunto de Atividades, Tarefas e Responsabilidades (ATRs) para suportá-la. As ATRs descrevem as práticas que necessitam ser institucionalizada para alcançar maturidade no processo de teste. As ATRs são designadas para três grupos essenciais críticos para o esforço de teste - administradores, desenvolvedores/testadores, usuários/clientes. Elas especificam regras para todos os participantes no processo de teste e fornecem suporte interno e externo para avaliação e melhoria do processo de teste.

Para cada nível existem:

i. Uma declaração de cada meta de maturidade;

ii. ATRs para os administradores;

iii. ATRs para os desenvolvedores / testadores;

iv. ATRS para usuários.

Para ilustrar como o conjunto de ATRs é definido e apresentado, segue abaixo um exemplo simplificado das ATRs definidas para a primeira meta do Nível 2 do TMM.

Nível 2 - Fase de Definição

Meta de Maturidade 2.1: Desenvolver Metas e Política de Teste e Depuração

Recordando que no Nível 2 não há requisito para um grupo dedicado a teste, assim as ATRs são formalmente designadas somente para os desenvolvedores. Se uma organização tem um grupo de especialistas de teste, então estas ATRs podem ser transferidas para este grupo.

ATRs para Administradores (superior e gerente de projeto)

- Prover liderança, recursos adequados e fundos para formar o comitê (equipe ou força tarefa) sobre teste e depuração. A constituição do comitê é gerencial, com a participação da equipe técnica.
- Tornar disponível quaisquer políticas e metas de teste/depuração pré-existentes ou amostras.
- Assumir um papel de liderança no desenvolvimento da política de teste/depuração.

ATRs para Desenvolvedores

- Trabalhar com o administrador para desenvolver metas e políticas de teste/depuração.
- Participar da equipe que supervisiona aplicação de política de teste/depuração e administra alterações.
- Familiarizar-se com o conjunto de políticas e metas de teste/depuração, manter-se atualizado com as revisões e fazer sugestões de mudança quando apropriada.

ATRs para Usuários/Clientes

- Dar entrada/feedback sobre políticas e metas de teste e depuração quando solicitado pelo administrador. (Estes grupos desempenham um importante papel na formação da políticas e metas de teste/depuração da organização, uma vez que estas metas e políticas refletem os esforços da organização para garantir a satisfação do cliente/usuário. O feedback desses grupos deve ser encorajado pelo administrador e pela Garantia de Qualidade de Software. Em geral, as necessidades de seus clientes e do mercado terão um impacto na natureza das políticas e metas de teste/depuração da organização.)

Uma descrição detalhada das ATRs é apresentada no Apêndice A (Burnstein, 2003).

3.2.4 Ferramentas de Teste

O trabalho dos testadores e administradores de processo de teste é suportado por ferramentas de teste, administração e planejamento. O grau de complexidade dessas ferramentas aumenta conforme o aumento do nível de maturidade a ser suportado. Entretanto, a existência de tais ferramentas é de fundamental importância para a melhoria da produtividade e aumento da confiabilidade das tarefas a serem realizadas.

Por exemplo, as ferramentas que suportam a Meta 1 - Desenvolver os Objetivos de Teste, do Nível 2, consistem principalmente de processadores de texto para registrar as políticas e um conjunto intraorganizacional de páginas web para promover, disponibilizar e distribuí-las por toda a organização.

A Tabela 3.1 apresenta uma síntese do conjunto de ferramentas requeridas em cada nível do TMM. No Apêndice C tais ferramentas são descritas mais detalhadamente.

3.2.5 Avaliação da Maturidade de Teste

Com base na estrutura do TMM, (Burnstein, 2003) desenvolveu um questionário que serve de guia para avaliar o estado atual da maturidade dos processos de teste das organizações. As questões procuram verificar o grau em que cada meta de maturidade é atendida. Com o diagnóstico é possível identificar o que falta e planejar ações de melhoria.

O questionário é composto de sete seções. Os conteúdos de cada seção estão descritos abaixo.

Tabela 3.1: Níveis do TMM e Ferramentas

Níveis	*Ferramentas*
Nível 1	Não há ferramentas
Nível 2	Planejadores de projetos e teste Verificadores de erros em tempo de execução Ferramentas suporte de preparação de teste Analisadores de cobertura Ferramentas de referência cruzada
Nível 3	Ferramentas de gerenciamento de configuração Registradores de requisitos Verificadores de requisitos Rastreadores de requisitos pelo teste Ferramentas de Capture-Replay Comparadores Rastreadores de defeito Medidas de complexidade Geradores de carga
Nível 4	Verificadores de código Auditores Ferramentas de compreensão de código Geradores de rotinas de teste Ferramentas de teste de performance Analisadores de rede Simuladores e emuladores Ferramentas de teste de Web Ferramentas de administração de teste
Nível 5	Ferramentas de suporte de Biblioteca de Processo Ativo(PAL) Ferramentas avançadas para roteiros de teste Verificador de asserção Geradores avançados de administração de rede Ferramentas de medida de usabilidade

Seção 1. Instruções para o respondedor.

Seção 2. Identificação e qualificação do respondedor.

Seção 3. Qualificação da organização.

Seção 4. Questões do TMM. Há para cada nível do TMM:

i. O conjunto de metas de maturidade;

ii. O conjunto de submetas de maturidade associados com cada meta de maturidade;

iii. O conjunto de questões associados com cada meta de maturidade.

Seção 5. Questões de ferramentas de teste.

Seção 6. Questões de tendências.

Seção 7. Comentários do respondedor.

Cada questão tem quatro possibilidades de resposta: SIM, NÃO, NÃO se APLICA ou NÃO CONHECE. O respondedor pode também fornecer comentários relacionados a cada questão no espaço fornecido.

Por exemplo, as questões apresentadas a seguir ilustram como as ATRs do Nível 2 do TMM são avaliadas.

Para o Nível de Maturidade 2.2, Iniciar um Processo de Planejamento de Teste, segue:

ATR para Administradores - Prover liderança, fundos e recursos para um comitê de planejamento de teste para toda a organização.

Questão 1 - Foi estabelecido um comitê ou grupo de planejamento de teste para toda a organização?

Questão 4 - Há suporte adequado e recursos para planejamento de teste para todos projetos?

ATR para Desenvolvedores - Desenvolver especificações de casos de teste, especificações de procedimento de teste e outros documentos relacionados a teste.

Questão 9 - Os desenvolvedores (testadores) têm sido treinados no uso de modelos e ferramentas de planejamento?

Questão 10 - Os desenvolvedores (testadores) são adequadamente treinados para desenvolver especificações, projeto de teste e casos de teste para plano de teste?

ATR para Usuários/Clientes - Suprir entrada e consenso para plano de teste de aceitação. O requisito funcional e atributos relativos a desempenho que são esperados pelo cliente/usuários devem ser especificados claramente e quantitativamente se possível.

Questão 17 - Há um procedimento estabelecido para solicitar entradas de usuários/clientes para planejamento de teste quando apropriado (por ex: no planejamento de teste de aceitação)?

O questionário completo utilizado para avaliar o nível de maturidade do TMM pode ser encontrado no Apêndice B (Burnstein, 2003).

3.3 Considerações Finais

Este capítulo apresentou um histórico sobre o surgimento dos modelos de maturidade de teste. Dos modelos propostos, o TMM é o que mais se destacou. Tal modelo foi descrito detalhadamente, juntamente com suas metas, submetas e ATRs. Foram também apresentadas as opções de ferramentas que suportam a realização e a organização de teste, para cada nível do TMM. O questionário de (Burnstein, 2003) para avaliação dos processos de teste das organizações também foi apresentado.

Capítulo

4

O Modelo MPT-PE

Este capítulo trata do desenvolvimento de um modelo de maturidade de processo de teste específico para as pequenas e micro empresas. É apresentado o modelo de maturidade resultante MPT-PE, bem como os métodos de avaliação, questionário de avaliação e tabela de Check-up. São, também, sugeridas as ferramentas para a automatização do modelo.

4.1 Construção

Com base nos modelos de melhoria de processo e de teste apresentados, a seguir são feitas as considerações que deram origem ao modelo de Melhoria do Processo de Teste para Pequenas Empresas - MPT-PE.

A evolução proposta pelo modelo deve ser técnica e economicamente viável para as micro e pequenas empresas. Os conceitos devem ser expressos em linguagem simples, as tarefas devem fazer sentido no contexto da pequena organização e devem estar de acordo com a cultura da empresa.

As pequenas empresas têm características peculiares e distintas das grandes. Estas características, descritas abaixo, serviram como premissas que orientaram a formulação e redação das atividades e tarefas do modelo desenvolvido.

- A maioria das empresas pequenas tem, ainda, como líder o fundador. O propósito e os valores da organização são os do fundador. Não utilizam mecanismos formais de políticas para orientar a organização. A orientação pelo empreendedor é constante.
- As pequenas empresas não apresentam a complexidade das grandes empresas, não têm que lidar com muitos grupos, departamentos e dispersão geográfica.
- Seus clientes são, em geral, organizações pequenas que requerem software menor e menos complexo.
- Não dispõem de muitos recursos e evitam métodos e ferramentas caros e sofisticados.
- Têm poucas pessoas, convivendo próximas, com poucas barreiras departamentais. A comunicação e troca de idéias são fáceis.
- A integração dos objetivos dos projetos com os do negócio ocorre naturalmente pela proximidade, comunicação e ação gerencial, sem o uso de comitês e reuniões.
- Não ocorreu, ainda, a divergência metodológica provocada pelo crescimento com o surgimento de diversos grupos de desenvolvimento, cada um com interesses e motivações próprios.
- Os processos e métodos são únicos e, em geral, implementados pelos líderes que originaram a empresa. Como não têm muitos grupos e os que têm estão muito próximos, a comunicação é mais fácil e os métodos, tanto os de desenvolvimento como os de teste, tendem a ser os mesmos para todos. Não demanda grande esforço de padronização para toda a organização.

Os processos de teste e os processos de software trabalham em conjunto e sob a mesma organização, devem, então, ter o mesmo nível de complexidade. Se o processo de software de uma empresa pequena é simples, também deve ser simples o processo de teste.

As atividades do ciclo de vida de desenvolvimento de software consideradas no CMM e SPICE são ligeiramente diferentes. (Rosa, 1997), na elaboração do MPSPE para pequenas empresas, comparou as atividades do processo de software dos dois modelos e as simplificou para um número menor e mais adequado à pequena empresa, conforme apresentado na Tabela 4.1.

Como ponto de partida, consideramos que o modelo de melhoria do processo de teste para pequenas empresas (MPT-PE) deveria seguir o MPSPE, isto é, teria as atividades do processo de teste que correspondem às do processo de software do MPSPE. Assim, algumas

Tabela 4.1: Modelo de Processo de Software para Pequena Empresa (MPSPE) - Atividades. (Rosa, 1997)

Atividades	***CMM***	***SPICE***	***ISO 9000***	***Questionário- MAPPE***	***MPSPE***
Codificação		X	X		X
Cópia, Entrega e Instalação		X	X		X
Definição do Processo	X	X			X
Documentação		X	X	X	X
Especificação dos Requisitos	X	X	X	X	X
Garantia de Qualidade	X	X	X		X
Gerenciamento de Configuração	X	X	X	X	X
Gerenciamento de Projeto	X	X			
Gerenciamento de Qualidade	X	X	X		
Gerenciamento de Subcontratados	X	X	X		
Manutenção		X	X		X
Melhoria de Processo	X	X			
Mudança de Tecnologia	X		X		
Planejamento do Projeto	X	X	X	X	X
Prevenção de Defeitos	X	X	X		
Projeto de Software		X	X	X	X
Revisão	X	X		X	X
Teste		X	X	X	X
Treinamento	X	X	X		X

das atividades do processo de software não foram consideradas devido à dificuldade de implementá-las em empresas de pequeno porte. Entre as atividades não abordadas estão:

Prevenção de defeitos - que indica as possíveis causas de defeitos e as previne da recorrência. Esta só é possível quando a empresa possui bom gerenciamento e mantém históricos de projetos passados. É difícil até mesmo para as grandes empresas.

Definição do processo - que consiste em padronizar um processo para toda a empresa.

Gerenciamento de qualidade - que visa gerenciar a qualidade do processo de desenvolvimento através de métricas. Corresponde ao Nível 4 do CMM, difícil de ser alcançado mesmo por empresas maiores.

Gerenciamento de sub-contratos - normalmente a empresa pequena não terceiriza seus serviços.

Melhoria de processo - que trata da otimização do processo, aumentando a produtividade e a confiabilidade, e diminuindo o tempo para desenvolvimento do produto. Não é abordada devido às limitações de uma empresa de pequeno porte. Requer Nível 5 no CMM, raramente alcançado mesmo pelas grandes empresas.

Mudança de tecnologia - que trata do gerenciamento de mudança de tecnologia da empresa. Identificação e teste de novas ferramentas, métodos e processos úteis para a organização. É abordado pelo Nível 5 do CMM, que é um nível difícil e demorado de

ser atingido por empresa de porte maior. Só é viável a empresas que possuem um bom controle de seus processos de software. Não condiz com a realidade da pequena empresa.

Gerenciamento de projeto - que compreende tratar cada desenvolvimento como um projeto. Compreende o objetivo, os recursos exigidos, os riscos, as tarefas a serem executadas, os marcos de acompanhamento, o esforço despendido e a programação a ser seguida. Empregado com equipe multidisciplinar e em projetos complexos. Não são comuns em empresas pequenas que ainda enfrentam dificuldades básicas na parte gerencial.

Para a construção do MPT-PE foi elaborada a tabela apresentada nas Figuras 4.1 e 4.2, como segue:

1. As atividades do processo de software que constam em pelo menos um dos modelos CMM, SPICE E ISO 9000, conforme mostra a Tabela 4.1, constam na coluna A. Elas constituem o que seria o processo mais simples e considerado pelo MPSPE estão indicadas na coluna B.

2. Num paralelo com as atividades do processo de software, foram indicadas na coluna C as atividades do processo de teste consideradas necessárias e viáveis para a pequena empresa, como segue:

   a) Inicialmente foram consideradas as atividades de teste correspondentes somente às atividades do processo de software constantes do MPSPE, pelas mesmas razões justificadas por Rosa (1997) e esplanadas anteriormente.

   b) As atividades Documentação de Teste, Requisitos de Teste, Garantia de Qualidade, Gerenciamento de Configuração, Plano de Teste, Revisão, Teste e Treinamento são correspondentes com as atividades do MPSPE, e têm os mesmos significados.

   c) Foram acrescentadas as atividades Definição da Organização de Teste, Planejamento e Controle, e Integração de Testes no Ciclo de Vida, que não têm atividades correspondentes no MPSPE:

   - A Definição da Organização de Teste e a atividade de Planejamento e Controle estão presentes em qualquer sistema de administração. São necessárias ao suporte e controle administrativo do processo.
   - A atividade Integração de Testes no Ciclo de Vida foi acrescentada como meio de prevenção de defeitos. Visa identificar cedo os defeitos nas etapas de desenvolvimento de software.

3. Cada ATR do TMM foi avaliada quanto ao grau de importância, dificuldade e recurso requerido. As ATRs que atendiam às características da empresa de pequeno porte para cada atividade do processo de teste foram selecionadas e constam das colunas D, E e F. As ATRs óbvias ao ato de administrar, as redundantes e as menos relevantes para a qualidade foram eliminadas. Parte das ATRs resultantes foram reescritas de maneira mais simples. Isto resultou num menor número de ATRs que no TMM e com linguajar compatível ao ambiente da pequena empresa.

4. Com base no TMM e nas dificuldades enumeradas, cada ATR foi classificada em um dos quatro níveis - Inicial, Fase de Definição, Definido e Integrado - indicados nas colunas G, H e I. Como o Nível Inicial não tem metas e ATRs ele não foi indicado.

5. Para visualizar o nível das tarefas foram utilizadas cores, atividades de um mesmo nível são de cor igual. Para selecionar as atividades e tarefas por nível com o uso do autofiltro, disponível em planilhas eletrônicas, o nível de cada tarefa foi indicado com x, y e z. Assim, as atividades do Nível 2 estão indicadas em preto e com um "x" na coluna G; as do Nível 3a estão em verde e com "y" na coluna H; as do Nível 3b estão em vermelho e com "z" na coluna I. Isto facilitou o trabalho, pois em qualquer momento foi possível agrupar as ATRs segundo critério e analisar o grupo quanto a compatibilidades, redundâncias e conveniências.

Isto resultou num modelo de TMM mais leve, com degraus menores entre os níveis, com menor número de ATRs e com redação mais simples. Mais fácil de entender, de ser utilizado para avaliação e para projetar situações futuras para o processo de teste. Corresponde aos três níveis do TMM, no qual o Nível 3 aparece subdividido em dois sub-níveis - definido e integrado.

No nível *Definido* existe uma organização formal de teste, o teste é uma atividade profissional, existem os métodos, ferramentas e meios para a detecção e correção de defeitos, mas ainda é uma etapa que segue a codificação. No nível *Integrado* o teste é distribuído ao longo do ciclo de vida e utilizado desde o início do processo.

Uma das razões de separar o Nível 3 em dois sub-níveis é que a conveniência de atingir o Nível *Integrado* depende do balanço entre os benefícios e os custos. Testar cedo no ciclo de desenvolvimento revela problemas quando as correções são ainda fáceis para implementar. Entretanto, testar cedo pode ser caro e freqüentemente requer sistemas de suporte custosos. Testar tarde no ciclo de desenvolvimento pode ser eficiente, mas identifica defeitos quando o re-trabalho é caro, tanto em dinheiro como em tempo. Muitas empresas

erram em colocar muita ênfase nos testes tardios. Entretanto, a pequena empresa, principalmente a com poucos recursos, deve analisar cuidadosamente se o custo do teste cedo pode ser compensado pelo custo mais baixo do re-trabalho. Quando a redução de custo do re-trabalho excede o aumento de custo dos testes, deve-se intensificar os testes cedo. Isto pode ocorrer diferentemente em cada situação, para cada tipo ou linha de produto software.

4.2 Descrição

O MPT-PE é um modelo de referência para a avaliação e Melhoria do Processo de Teste para Pequenas Empresas. É constituído de uma escala de níveis hierárquicos e cada nível contém um conjunto de metas, submetas e atividades, tarefas e responsabilidades (ATRs) que definem a capacidade de teste de uma organização. Cada nível de maturidade representa um estágio evolutivo, desde o processo imaturo até o maduro. Representa o grau de capacidade e a habilidade de um processo de teste desempenhar a sua missão de modo eficiente e confiável, conforme ilustrado na Figura 4.3.

Como modelo é utilizado para avaliar o estado atual do processo de teste e identificar o que falta para o nível acima. As metas e ATRs servem de guia para administradores, testadores e clientes definirem os seus papeis e melhorarem seu desempenho nas áreas críticas.

4.3 Os Níveis do MPT-PE

Nível 1: Inicial - O teste é um processo caótico, desordenado e mal definido, não distinto da depuração. É normalmente feito pela equipe de desenvolvimento de forma rudimentar. Os desenvolvedores realizam os testes por prática, baseado na sua própria experiência, habilidade (*"ad hoc"*) e executados sem ferramentas. Há falta de pessoal treinado e ferramentas de teste. Tem a finalidade de mostrar que o sistema e o software funcionam.

Nível 2: Fase de Definição - O teste é uma função separada da depuração. É uma fase bem definida e planejada, que ocorre após o término da codificação. São executados por desenvolvedores e ou testadores. Os métodos e técnicas básicos de teste estão alocados, mas a atividade de teste ainda é considerada primária. O objetivo do teste é mostrar que

A	B	C	D	E	F	G	H	I
CMM/SPICE/ISO	MPSPE	MPT-PE	MPT-PE			em definição	definido (mpspe)	integrado
Atividades do processo de software		Atividades do processo de teste	Atividades, tarefas e resonsabilidades - ATRs			Niv.2	Nivel 3a	Nivel 3b
			resp.	meta				
Codificação	X		Des					
Cópia, Entrega e Instalação	X		Des					
Definição de processo para toda a organização		Definição da organização de teste	Adm	2.1	Liderar e prover recursos para teste e depuração.	x		
			Adm	2.1	Designar responsabilidades para teste e depuração.	x		
			Adm	2.2	Designar comitê de planej. de teste para a organização.			
			Adm	2.3	Prover liderança e recursos para implementar técnicas, métodos e ferramentas básicas de teste.	x		
			Adm	3.1	Definir um setor especializado em teste.		y	
			Adm	3.1	Designar responsabilidades do setor de teste.		y	
			Adm	3.1	Estabelecer padrões, requisitos e plano de carreira do profissional de teste.		y	
			Adm	3.1	Prover recursos para a organização de teste.		y	
			Adm	3.1	Avaliar e supervisionar pessoal de teste.		y	
			Adm	3.1	Encorajar e prover caminho de comunicação para interação de testadores e clientes.		y	
Documentação	X	Documentação de teste	Adm	3.1	Assegurar que todos os documentos de teste são preparados, tal como diário de teste e relatório de incidências.		y	
			Adm	3.1	Preparar e apresentar relatório resumo de teste.		y	
			Test	3.1	Preparar documentos de pré e pós teste.		y	
			Test	3.4	Preparar diário de teste e relatório de incidências.		y	
Especificação dos Requisitos	X	Requerimentos de Teste	Adm	2.1	Encorajar a participação do cliente.	x		
			Des/Test	2.1	Registrar defeitos no repositório.	x		
			Clien	2.2	Enunciar claramente os requisitos.	x		
			Clien	2.3	Trabalhar com analistas para que os requisitos sejam claros e testáveis.	x		
Garantia de Qualidade	X	Garantia da qualidade	Adm	2.1	Implementar classificação e repositório de defeitos.	x		
			Adm	2.1	Assegurar que desenvolvedores registrem defeitos.	x		
			Des/Test	2.1	Desenvolver esquema de classificação de defeitos.	x		
			Des/Test	2.2	Assegurar que questões de testabilidade são tratadas durante as fases de requisitos e projeto.	x		
			Des/Test	2.3	Referir-se ao repositório de defeitos passados.	x		
			Test	3.1	Trabalhar com analistas, projetistas, desenvolvedores e gerentes para planejar o teste e desenvolver produtos de qualidade.		y	
			Test	3.1	Trabalhar para estabelecer padrões de produto e processo.		y	
Gerenciamento de Configuração	X	Gerenciamento de configuração	Adm	3.4	Suportar a instalação de um sistema de administração de configuração e controle de mudança.		y	
			Test	3.4	Assegurar que itens de teste estão sob o controle do sistema de administração de configuração.		y	
Gerenciamento de projeto								
		Planejamento e controle	Des/Test	2.2	Auxiliar o gerente no planejamento de teste.			
			Des/Test	2.3	Incluir no plano de teste técnicas de caixa preta/branca.	x		
			Des/Test	2.3	Trabalhar com clientes para desenvolver critérios para teste de aceitação.	x		
			Adm	3.4	Assegurar, planejar, monitorar e controlar as atividades do plano de teste.		y	
			Adm	3.4	Participar de reuniões de andamento e auditoria para soluções de problemas e follow-up de ações corretivas.		y	
			Adm	3.4	Designar responsabilidades aos testadores para a correção de problemas relativos a teste.		y	
			Adm	3.4	Atribuir responsabilidades para monitoramento e controle de teste.			2
			Test	3.4	Trabalhar com o gerente de teste para selecionar medidas apropriadas de controle e monitoramento.			2
			Test	3.4	Trabalhar com o gerente de projeto para desenvolver planos de contingência.			2
			Clien	3.4	Dar feedback pós-release com dados úteis aos testes.			2
			Clien	3.4	Participar de reuniões de verificação de progresso e melhoria dos testes.		y	
Gerenciamento da Qualidade (por métricas)								
Gerenciamento de Subcontratos								
Manutenção	X		Des					
Melhoria do Processo								

Figura 4.1: Construção MPT-PE - Parte 1 de 2.

Mudança de Tecnologia (gerenciamento)								
Planejamento do Projeto	X	Plano de Teste	Des/Test	2.1	Desenvolver planos de teste.	x		
			Adm	2.2	Selecionar ferramentas para planejamento de teste.	x		
			Adm	2.2	Assegurar que testes de múltiplos níveis são executados.	x		
			Adm	2.3	Assegurar que o plano de teste inclua o uso de métodos de caixa preta e caixa branca.	x		
			Des/Test	2.3	Auxiliar o gerente na seleção de ferramentas e métodos.	x		
			Clien	2.3	Dar feedback de problemas de soft para inclusão em teste de aceitação.	x		
			Test	3.1	Trabalhar com gerentes de projeto/teste sobre planejamento e monitoramento do esforço de teste.		y	
			Test	3.4	Executar follow-up das atividades de ações corretivas.		y	
			Test	3.4	Coletar e analisar medidas do processo de teste.			z
Prevenção de Defeitos								
		Integração de testes no ciclo de vida	Adm	3.3	Assegurar que a integração das atividades de teste é aplicada por toda a organização e todos os projetos.			z
			Adm	3.3	Prover liderança e recursos para as atividades de integração do teste no ciclo de vida de software.			z
			Adm	3.3	Trabalhar com os testadores para desenvolver padrões para teste do resultado do trabalho de cada fase do ciclo de vida.			z
			Adm	3.3	Revisar, aprovar e adotar modelo de teste de integração (ex: modelo V).			z
			Adm	3.3	Prover treinamento para suporte das atividades de integração.			z
			Test	3.3	Planejar as atividades integradas de teste.			z
			Test	3.3	Executar atividades de testes integradas no ciclo de vida.			z
			Test	3.3	Trabalhar com analistas, projetistas, desenvolvedores e cliente sobre questões de teste designados a cada fase do ciclo de vida.			z
			Clien	3.3	Participar como convidado para prover consenso sobre atividades integradas de teste.			z
Projeto de Software (desenvolvimento)	X	Projeto de Teste	Des/Test	2.1	Definir metas de teste e de depuração.	x		
			Clien	2.1	Participar da definição de metas de teste e depuração.	x		
			Des/Test	2.2	Determinar casos e procedimentos de teste.	x		
			Des/Test	2.2	Trabalhar com clientes para desenvolver casos de uso e plano de aceitação.	x		
			Clien	2.2	Suprir informações para plano de teste de aceitação.	x		
			Clien	2.2	Participar da determinação de casos de uso.	x		
			Des/Test	2.3	Projetar uso de casos, especificações e procedimentos de teste baseados na estratégia de teste, técnicas e métodos.	x		
			Test	3.1	Projetar casos, seus procedimentos e execução de teste.		y	
			Test	3.1	Trabalhar com clientes para desenvolver casos de uso e plano de aceitação e perfil de uso.		y	
			Clien	3.1	Trabalhar com os testadores no planejamento de teste de instalação e perfil de uso.		y	
Revisão	X	Revisão	Des/Test	2.3	Revisar as representações de software.	x		
Teste	X	Teste	Des/Test	2.2	Coletar medidas quanto a realização das metas de teste.	x		
			Des/Test	2.3	Conduzir atividades de teste.	x		
			Des/Test	2.3	Executar casos de teste.	x		
			Des/Test	2.3	Registrar dados relativos à teste.	x		
			Des/Test	2.3	Registrar dados relativos a defeitos.	x		
			Des/Test	2.3	Assegurar abordagem equilibrada em todos os níveis, unidade, integração e sistema.	x		
			Clien	2.3	Participar nos testes de aceitação, teste alfa e beta.	x		
			Clien	2.3	Participar no desenvolvimento de caso de uso.	x		
			Test	3.1	Avaliar e aplicar novas técnicas de teste e ferramentas.		y	
Treinamento	X	Treinamento	Adm	2.1	Assegurar treinamento e ferramentas para depuração.	x		
			Adm	2.1	Assegurar treinamento e ferramentas para teste.	x		
			Test	3.1	Manter-se atualizado com novas técnicas, métodos e ferramentas de teste.		y	
			Adm	3.2	Assegurar educação e treinamento dos membros do grupo de teste.		y	
			Test	3.2	Aplicar novos conhecimentos e habilidades adquiridas nos projetos da organização.		y	
			Clien	3.2	Participar de sessões de treinamento relativas a revisão, teste de aceitação e casos de uso.		y	

Figura 4.2: Construção MPT-PE - Parte 2 de 2.

Nível		Meta		Explicações
1.	Inicial		Não há metas.	Teste executado pelos desenvolvedores de forma prática, com o objetivo de mostrar que o software funciona.
2.	Fase de Definição	2.1	Desenvolver os objetivos de teste.	O propósito do teste é mostrar que o sistema e o software funcionam e que atendem às especificações.
		2.2	Iniciar um processo de planejamento de teste.	
		2.3	Institucionalizar técnicas e métodos básicos de teste.	
3a.	Definido	3.1	Estabelecer uma organização de teste de software.	O propósito do teste é mostrar que o software não funciona, é detectar falhas.
		3.2	Estabelecer um programa de treinamento.	
3b.	Integrado	3.3	Integrar o teste no ciclo de vida do software.	O objetivo é detectar defeitos e prevenir falhas, o mais cedo possível no processo.
		3.4	Controlar e monitorar o teste.	

Figura 4.3: Níveis e Metas do MPT-PE.

o sistema e o software satisfazem às especificações. Muitos problemas ocorrem, com retrabalho caro, porque o planejamento de teste ocorrendo tardiamente no ciclo de vida do software, não impede a propagação dos defeitos desde requisitos e fase de projeto, até o código.

Nível 3a: Definido - existe uma organização formal encarregada dos testes. O teste é reconhecido como uma atividade profissional. Existem os métodos, ferramentas e meios para a detecção e correção de defeitos. Há programas e treinamento técnico com foco em testes.

É ainda uma etapa após o término da codificação, mas o propósito agora é mostrar que o software não funciona, é detectar falhas. Os objetivos de testes são estabelecidos com respeito a requisitos baseados nas necessidades do usuário e cliente. São usados para teste caso de projeto e critérios de sucesso.

Nível 3b: Integrado - o teste não é mais concentrado em uma única etapa após a codificação, mas sim é dividido em partes distribuídas ao longo das diferentes etapas do processo de desenvolvimento do software. É distribuído ao longo do ciclo de vida e planejado desde de cedo no processo - *é integrado no ciclo de vida do software.* O objetivo é detectar defeitos e prevenir falhas, o mais cedo possível no processo. Há métricas e revisões, mas não ocorrem ainda através de todo o ciclo de vida.

4.4 Descrição das Metas e Submetas

Cada nível do MPT-PE tem associado metas e submetas de maturidade que identificam objetivos de melhorias de teste que devem ser cumpridas para atingir a maturidade naquele nível (vide Figura 4.4).

Nível 1 - Inicial

O Nível 1 como nível inicial não possui nenhuma meta para atingi-lo.

Nível 2 - Fase de Definição

Meta 2.1: Desenvolver os Objetivos de Teste

A organização deve distinguir claramente entre processo de teste e depuração. A separação destes dois processos é essencial para o crescimento da maturidade de teste, desde que eles são diferentes em metas e métodos.

As submetas de maturidade incluem:

- A organização que deve formar um comitê sobre teste e depuração com suporte e recursos. Em todo o Nível 2, um comitê pode, em casos especiais, ser formado por uma única pessoa quando a empresa não tiver pessoal suficiente.
- O comitê que deve desenvolver e registrar as metas de teste.
- O comitê que deve desenvolver as metas de depuração.
- As metas documentadas de teste e de depuração que devem ser distribuídas para a gerência e desenvolvedores.
- As metas de teste que devem ser refletidas nos planos de teste.

Meta 2.2: Iniciar um Processo de Planejamento de Teste

O planejamento de teste envolve estabelecer objetivos, analisar os riscos, traçar estratégias e desenvolver especificações do projeto de teste e casos de teste. Ainda o plano

de teste, designar recursos e atribuir responsabilidades de teste de unidade, integração, sistema e aceitação.

As submetas de maturidade incluem:

- Estabelecer um comitê de planejamento de teste para toda a organização, com recursos adequados.
- Estabelecer uma política de plano de teste para toda a organização, apoiada pela administração.
- Desenvolver um modelo de planejamento de teste, registrar e distribuir para os gerentes de projeto.
- Treinar os gerentes de projeto e desenvolvedores para usar o modelo e desenvolver o plano de teste.
- Implantar um procedimento que inclua os requisitos de usuários como entrada para o plano de teste.
- O planejamento básico deve ser avaliado, recomendado e aprovado. A administração deve apoiar o uso.

Meta 2.3: Institucionalizar Técnicas e Métodos Básicos de Teste

Para melhorar a capacidade do processo de teste, técnicas e métodos básicos devem ser aplicados por toda a organização. Políticas para estas técnicas e ferramentas básicas de suporte devem ser claramente especificadas. Exemplos de métodos e técnicas básicas são as estratégias de caixa-preta e caixa-branca, uso de matriz de validação de requisitos, e teste de unidade, integração, sistemas e aceitação.

As submetas de maturidade incluem:

- A administração que deve instituir um conjunto de políticas que assegure que métodos e técnicas recomendados sejam consistentemente aplicados por toda a organização.

Nível		*Meta*		*Submeta*	
1.	Inicial		Não há meta		
2.	Fase de Definição	2.1	Desenvolver os objetivos de teste	2.1.1	A organização deve formar comitês sobre teste e depuração.
				2.1.2	Os comitês devem desenvolver e registrar as metas de teste.
				2.1.3	Os comitês devem desenvolver e registrar as metas de depuração.
				2.1.4	As metas de teste e de depuração devem ser distribuídas por todos os gerentes e desenvolvedores.
				2.1.5	As metas de teste devem ser refletidas nos planos de teste.
		2.2	Iniciar um processo de planejamento de teste	2.2.1	Estabelecer comitê de planejamento de teste para toda a organização.
				2.2.2	Estabelecer uma política de plano de teste para toda a organização.
				2.2.3	Desenvolver um modelo de planejamento de teste e distribuir aos gerentes de projeto.
				2.2.4	Treinar os gerentes e desenvolvedores para usar o modelo e desenvolver plano de teste.
				2.2.5	Implantar procedimento com os requisitos de usuários como entrada para o plano de teste.
				2.2.6	O planejamento básico deve ser avaliado, aprovado e recomendado.
		2.3	Institucionalizar técnicas e métodos básicos de teste	2.3.1	Estabelecer políticas para que os métodos e técnicas recomendadas sejam aplicados.
				2.3.2	Formar grupo de tecnologia para recomendar técnicas, métodos e ferramentas de teste.
3a.	Definido	3.1	Estabelecer uma organização de teste de software	3.1.1	Estabelecer um grupo de teste para toda a organização.
				3.1.2	Definir regras e responsabilidades para o grupo de teste.
				3.1.3	Designar para o grupo membros bem treinados e motivados.
				3.1.4	O grupo de teste deve estabelecer conexões com o cliente e usuários, para coletar dados, desejos e requisitos.
		3.2	Estabelecer um programa de treinamento	3.2.1	Estabelecer um programa da empresa para treinamento, prover recursos e apoio.
				3.2.2	Formar um comitê técnico para desenvolver e divulgar uma política para treinamento.
				3.2.3	Desenvolver metas e planos de treinamento.
				3.2.4	Estabelecer um grupo de treinamento interno, com ferramentas e facilidades.
3b.	Integrado	3.3	Integrar o teste no ciclo de vida do software	3.3.1	Dividir a etapa de teste em sub-etapas e distribuir ao longo do ciclo de vida.
				3.3.2	Desenvolver e adotar uma versão do modelo V.
				3.3.3	Alocar recursos e apoiar a integração do teste no ciclo de vida.
				3.3.4	Definir padrões para teste de produto e medida de sua qualidade.
				3.3.5	Estabelecer procedimento que facilite o trabalho dos testadores junto aos desenvolvedores.
		3.4	Controlar e monitorar o teste	3.4.1	Desenvolver mecanismos e políticas para controlar e monitorar o teste.
				3.4.2	Definir, registrar e distribui um conjunto de métricas relativas ao teste.
				3.4.3	Desenvolver planos de ações corretivas para uso quando o teste desvia do planejado.

Figura 4.4: Metas e Submetas do MPT-PE.

- A administração que deve formar um grupo de tecnologia para a empresa, a fim de estudar, avaliar e recomendar um conjunto de técnicas e métodos básicos de teste, e recomendar um conjunto de ferramentas simples para suportá-los.

Nível 3a. - Definido

Meta 3.1: Estabelecer uma Organização de Teste de Software

Uma organização de teste de software é criada para identificar um grupo de pessoas que são responsáveis por teste. O grupo de teste é responsável pelo planejamento, execução e registro de teste, pelos padrões relativos a teste, métricas de teste, banco de dados de teste, reuso de teste, acompanhamento e avaliação.

As submetas de maturidade incluem:

- Estabelecer um grupo de teste com liderança, apoio e recursos da administração superior. Deve definir suas funcionalidades e subordinação. Em casos especiais, devido a restrições de pessoal o grupo poderá ser reduzido a uma única pessoa.
- Definir regras e responsabilidades para o grupo de teste.
- Designar para o grupo membros bem treinados e motivados.
- O grupo de teste deve estabelecer conexões de comunicação, que consultam os clientes e os usuários participantes das atividades de teste, para ajudar a coletar dados e documentos, e incorporar as necessidades, desejos e requisitos do usuário no processo de teste.

Meta 3.2: Estabelecer um Programa de Treinamento

Os testadores devem ser adequadamente treinados de modo a que eles possam desempenhar suas funções eficiente e eficazmente. Neste nível, o pessoal é treinado em planejamento de teste, métodos, padrões, técnicas e ferramentas de teste. O programa de treinamento também prepara o pessoal para o processo de revisão, instrui lideres de revisão e institui canais para a participação de usuários no processo de testes e revisões. O treinamento inclui cursos internos, aprendizado autodidata, programa de mentores e suporte de instituições acadêmicas.

As submetas de maturidade incluem:

- A administração que deve estabelecer um programa da organização para treinamento, prover recursos e suporte.
- O comitê de treinamento técnico que deve desenvolver e distribuir uma política da organização para treinamento.
- Desenvolver metas e planos de treinamento com entradas dos gerentes de projeto.
- Estabelecer um grupo de treinamento interno, com ferramentas, facilidade e materiais no local.

Nível 3b. - Integrado

Meta 3.3: Integrar o Teste no Ciclo de Vida do Software

O planejamento de teste deve ser iniciado cedo no ciclo de vida. Os testadores e desenvolvedores usam uma variação do modelo-V. As entradas de usuário no processo de teste são solicitadas através de canais estabelecidos para as diversas etapas de teste.

As submetas de maturidade incluem:

- Dividir a etapa de teste em partes que devem ser integradas no ciclo de vida de software, seguindo uma política organizacional escrita, e revisada com a administração.
- Desenvolver e adotar uma versão institucionalizada do modelo-V.
- Alocar recursos para apoiar a integração da atividade de teste no ciclo de vida de software.
- Definir padrões para teste de produtos e medida de sua qualidade.
- Estabelecer um procedimento que permita aos testadores trabalharem com os desenvolvedores para facilitar as atividades de teste.

Meta 3.4: Controlar e Monitorar o Teste

As atividades de controle e monitoramento provêem visibilidade e asseguram que os processos de teste procedam de acordo com o plano. O suporte para o controle e monitoramento vem de padrões de teste de produtos, marcos e métricas de teste que podem ser usados para avaliar o progresso e a efetividade do teste.

As submetas de maturidade incluem:

- Desenvolver mecanismos e políticas para controlar e monitorar o de teste.
- Definir, registrar e distribuir um conjunto de métricas relativas ao teste.
- Desenvolver um conjunto de ações corretivas e de planos de contingência, registrados e documentados para uso quando o teste desvia significativamente do planejamento.

4.5 Atividades, Tarefas e Responsabilidades (ATRs)

Conforme item 4.1, Construção, as ATRs foram agrupadas visando cada atividade do processo para pequena empresa. As ATRs que atendiam às características da empresa de pequeno porte para cada atividade do processo de teste foram mantidas. As ATRs óbvias ao ato de administrar, as redundantes e as menos relevantes para a qualidade foram eliminadas. Parte das ATRs resultantes foram reescritas de maneira mais simples. Isto resultou num menor número de ATRs que no TMM, 82 em vez de 385, e com linguajar compatível ao ambiente da pequena empresa.

Para cada meta estão associadas às atividades e tarefas necessárias para alcançar aquele nível de maturidade no processo de teste. Para cada uma são definidos os responsáveis. As ATRs descrevem as práticas que necessitam ser institucionalizada.

As ATRs são designadas para as três visões críticas no esforço de teste - administradores, desenvolvedores / testadores, usuários / clientes. Elas definem regras para os participantes no processo de teste e fornecem suporte interno e externo para avaliação e melhoria do processo. Vide Figura 4.4.

Nível 2 - Fase de Definição

Meta de Maturidade 2.1: Desenvolver os Objetivos de Teste

Meta 1 desenvolver as metas de teste e depuração e políticas. Recordando que no Nível 2 não há requisito para um grupo dedicado a teste, assim as ATRs são formalmente designadas somente para os desenvolvedores. Se uma organização tem um grupo de especialistas de teste, então estas ATRs podem ser transferidas para este grupo. Vide Figura 4.5.

ATRs para Administradores (superior e gerente de projeto)

- Liderar e prover recursos para teste e depuração.
- Designar responsabilidades para teste e depuração.
- Encorajar a participação do cliente.
- Implementar classificação e repositório de defeitos.
- Assegurar que desenvolvedores registrem defeitos.
- Assegurar treinamento e ferramentas para depuração.
- Assegurar treinamento e ferramentas para teste.

ATRs para Desenvolvedores

- Registrar defeitos no repositório.
- Desenvolver esquema de classificação de defeitos.
- Desenvolver planos de teste.
- Definir metas de teste e de depuração.

ATRs para Usuários/Clientes

- Participar da definição de metas de teste e depuração.

Meta de Maturidade 2.2: Iniciar um Processo de Planejamento de Teste

ATRs para Administradores

- Designar comitê de planejamento de teste para a organização.
- Selecionar ferramentas para planejamento de teste.
- Assegurar que testes de múltiplos níveis são executados.

ATRs para Desenvolvedores

- Assegurar que questões de testabilidade são tratadas durante as fases de requisitos e projeto.
- Auxiliar o gerente no planejamento de teste.
- Determinar casos e procedimentos de teste.
- Trabalhar com clientes para desenvolver casos de uso e plano de aceitação.
- Coletar medidas quanto à realização das metas de teste.

ATRs para Usuários/Clientes

- Enunciar claramente os requisitos.
- Suprir informações para plano de teste de aceitação.
- Participar da determinação de casos de uso.

Meta de Maturidade 2.3: Institucionalizar Técnicas e Métodos Básicos de Teste

ATRs para Administradores

- Prover liderança e recursos para implementar técnicas, métodos e ferramentas básicas de teste.
- Assegurar que o plano de teste inclua o uso de métodos de caixa preta e caixa branca.

ATRs para Desenvolvedores

Nível		Meta		ATRs	
1.	Inicial		Não há		Não há.
2.	Fase de Definição	2.1	Desenvolver os objetivos de teste	Adm.	Liderar e prover recursos para teste e depuração.
				Adm.	Designar responsabilidades para teste e depuração.
				Adm.	Encorajar a participação do cliente.
				Adm.	Implementar classificação e repositório de defeitos.
				Adm.	Assegurar que desenvolvedores registrem defeitos.
				Adm.	Assegurar treinamento e ferramentas para depuração.
				Adm.	Assegurar treinamento e ferramentas para teste.
				Des/Test	Registrar defeitos no repositório.
				Des/Test	Desenvolver esquema de classificação de defeitos.
				Des/Test	Desenvolver planos de teste.
				Des/Test	Definir metas de teste e de depuração.
				Clien.	Participar da definição de metas de teste e depuração.
		2.2	Iniciar um processo de planejamento de teste	Adm.	Designar comitê de planejamento de teste para a organização.
				Adm.	Selecionar ferramentas para planejamento de teste.
				Adm.	Assegurar que testes de múltiplos níveis são executados.
				Des/Test	Assegurar que questões de testabilidade são tratadas durante as fases de requisitos e projeto.
				Des/Test	Auxiliar o gerente no planejamento de teste.
				Des/Test	Determinar casos e procedimentos de teste.
				Des/Test	Trabalhar com clientes para desenvolver casos de uso e plano de aceitação.
				Des/Test	Coletar medidas quanto à realização das metas de teste.
				Clien.	Enunciar claramente os requisitos.
				Clien.	Suprir informações para plano de teste de aceitação.
				Clien.	Participar da determinação de casos de uso.
		2.3	Institucionalizar técnicas e métodos básicos de teste	Adm.	Prover liderança e recursos para implementar técnicas, métodos e ferramentas básicas de teste.
				Adm.	Assegurar que o plano de teste inclua o uso de métodos de caixa preta e caixa branca.
				Des/Test	Referir-se ao repositório de defeitos passados.
				Des/Test	Incluir no plano de teste técnicas de caixa preta/branca.
				Des/Test	Trabalhar com clientes para desenvolver critérios para teste de aceitação.
				Des/Test	Auxiliar o gerente na seleção de ferramentas e métodos.
				Des/Test	Projetar uso de casos, especificações e procedimentos de teste baseados na estratégia de teste, técnicas e métodos.
				Des/Test	Revisar as representações de software.
				Des/Test	Conduzir atividades de teste.
				Des/Test	Executar casos de teste.
				Des/Test	Registrar dados relativos à teste.
				Des/Test	Registrar dados relativos a defeitos.
				Des/Test	Assegurar abordagem equilibrada em todos os níveis, unidade, integração e sistema.
				Clien.	Trabalhar com analistas para que os requisitos sejam claros e testáveis.
				Clien.	Dar feedback de problemas de software para inclusão em teste de aceitação.
				Clien.	Participar nos testes de aceitação, teste alfa e beta.
				Clien.	Participar no desenvolvimento de caso de uso.

Figura 4.5: Nível 2 - Fase de Definição - Metas e ATRs

- Referir-se ao repositório de defeitos passados.
- Incluir no plano de teste técnicas de caixa preta/branca.
- Trabalhar com clientes para desenvolver critérios para teste de aceitação.
- Auxiliar o gerente na seleção de ferramentas e métodos.
- Projetar uso de casos, especificações e procedimentos de teste baseados na estratégia de teste, técnicas e métodos.
- Revisar as representações de software.
- Conduzir atividades de teste.
- Executar casos de teste.
- Registrar dados relativos a teste.
- Registrar dados relativos a defeitos.
- Assegurar abordagem equilibrada em todos os níveis, unidade, integração e sistema.

ATRs para Usuários/Clientes

- Trabalhar com analistas para que os requisitos sejam claros e testáveis.
- Dar feedback de problemas de software para inclusão em teste de aceitação.
- Participar nos testes de aceitação, teste alfa e beta.
- Participar no desenvolvimento de caso de uso.

Nível 3a. - Definido

Meta de Maturidade 3.1: Estabelecer uma Organização de Teste de Software

ATRs para Administradores.

Vide Figura 4.6.

- Definir um setor especializado em teste.
- Designar responsabilidades do setor de teste.
- Estabelecer padrões, requisitos e plano de carreira do profissional de teste.
- Prover recursos para a organização de teste.
- Avaliar e supervisionar pessoal de teste.
- Encorajar e prover caminho de comunicação para interação de testadores e clientes.
- Assegurar que todos os documentos de teste são preparados, tal como diário de teste e relatório de incidências.
- Preparar e apresentar relatório resumo de teste.

ATRs para Testadores

- Preparar documentos de pré e pós teste.
- Trabalhar com analistas, projetistas, desenvolvedores e gerentes para planejar o teste e desenvolver produtos de qualidade.
- Trabalhar para estabelecer padrões de produto e processo.
- Trabalhar com gerentes de projeto/teste sobre planejamento e monitoramento do esforço de teste.
- Projetar casos, seus procedimentos e execução de teste.
- Trabalhar com clientes para desenvolver casos de uso e plano de aceitação e perfil de uso.
- Avaliar e aplicar novas técnicas de teste e ferramentas.
- Manter-se atualizado com novas técnicas, métodos e ferramentas de teste.

ATRs para Usuários/Clientes

3a.	*Definido*	3.1	Estabelecer uma organização de teste de software	Adm.	Definir um setor especializado em teste.
				Adm.	Designar responsabilidades do setor de teste.
				Adm.	Estabelecer padrões, requisitos e plano de carreira do profissional de teste.
				Adm.	Prover recursos para a organização de teste.
				Adm.	Avaliar e supervisionar pessoal de teste.
				Adm.	Encorajar e prover caminho de comunicação para interação de testadores e clientes.
				Adm.	Assegurar que todos os documentos de teste são preparados, tal como diário de teste e relatório de incidências.
				Adm.	Preparar e apresentar relatório resumo de teste.
				Test.	Preparar documentos de pré e pós teste.
				Test.	Trabalhar com analistas, projetistas, desenvolvedores e gerentes para planejar o teste e desenvolver produtos de qualidade.
				Test.	Trabalhar para estabelecer padrões de produto e processo.
				Test.	Trabalhar com gerentes de projeto/teste sobre planejamento e monitoramento do esforço de teste.
				Test.	Projetar casos, seus procedimentos e execução de teste.
				Test.	Trabalhar com clientes para desenvolver casos de uso e plano de aceitação e perfil de uso.
				Test.	Avaliar e aplicar novas técnicas de teste e ferramentas.
				Test.	Manter-se atualizado com novas técnicas, métodos e ferramentas de teste.
				Clien.	Trabalhar com os testadores no planejamento de teste de instalação e perfil de uso.
		3.2	Estabelecer um programa de treinamento	Adm.	Assegurar educação e treinamento dos membros do grupo de teste.
				Test.	Aplicar novos conhecimentos e habilidades adquiridas nos projetos da organização.
				Clien.	Participar de sessões de treinamento relativas a revisão, teste de aceitação e casos de uso.

Figura 4.6: Nível 3a. - Definido - Metas e ATRs

- Trabalhar com os testadores no planejamento de teste de instalação e perfil de uso.

Meta de Maturidade 3.2: Estabelecer um Programa de Treinamento

ATRs para Administradores

- Assegurar educação e treinamento dos membros do grupo de teste.

ATRs para Testadores

- Aplicar novos conhecimentos e habilidades adquiridas nos projetos da organização.

ATRs para Usuários/Clientes

- Participar de sessões de treinamento relativas a revisão, teste de aceitação e casos de uso.

Nível 3b. - Integrado

Meta de Maturidade 3.3: Integrar o Teste no Ciclo de Vida do Software

ATRs para Administradores.
Vide Figura 4.7.

- Assegurar que a integração das atividades de teste é aplicada por toda a organização e todos os projetos.
- Prover liderança e recursos para as atividades de integração do teste no ciclo de vida de software.
- Trabalhar com os testadores para desenvolver padrões para teste do resultado do trabalho de cada fase do ciclo de vida.
- Revisar, aprovar e adotar modelo de teste de integração (ex: modelo V).
- Prover treinamento para suporte das atividades de integração.

ATRs para Testadores

- Planejar as atividades integradas de teste.
- Executar atividades de testes integradas no ciclo de vida.
- Trabalhar com analistas, projetistas, desenvolvedores e cliente sobre questões de teste designados a cada fase do ciclo de vida.

ATRs para Usuários/Clientes

- Participar como convidado para prover consenso sobre atividades integradas de teste.

Meta de Maturidade 3.4: Controlar e Monitorar o Teste

ATRs para Administradores

3b.	***Integrado***	3.3	Integrar o teste no ciclo de vida do software	Adm.	Assegurar que a integração das atividades de teste é aplicada por toda a organização e todos os projetos.
				Adm.	Prover liderança e recursos para as atividades de integração do teste no ciclo de vida de software.
				Adm.	Trabalhar com os testadores para desenvolver padrões para teste do resultado do trabalho de cada fase do ciclo de vida.
				Adm.	Revisar, aprovar e adotar modelo de teste de integração (ex: modelo V).
				Adm.	Prover treinamento para suporte das atividades de integração.
				Test.	Planejar as atividades integradas de teste.
				Test.	Executar atividades de testes integradas no ciclo de vida.
				Test.	Trabalhar com analistas, projetistas, desenvolvedores e cliente sobre questões de teste designados a cada fase do ciclo de vida.
				Clien.	Participar como convidado para prover consenso sobre atividades integradas de teste.
		3.4	Controlar e monitorar o teste	Adm.	Suportar a instalação de um sistema de administração de configuração e controle de mudança.
				Adm.	Atribuir responsabilidades para monitoramento e controle do processo de teste.
				Adm.	Assegurar, planejar, monitorar e controlar as atividades do plano de teste.
				Adm.	Participar de reuniões de andamento e auditoria para soluções de problemas e follow-up de ações corretivas.
				Adm.	Designar responsabilidades aos testadores para a correção de problemas relativos a teste.
				Test.	Preparar diário de teste e relatório de incidências.
				Test.	Assegurar que itens de teste estão sob o controle do sistema de administração de configuração.
				Test.	Trabalhar com o gerente de teste para selecionar medidas apropriadas de controle e monitoramento.
				Test.	Trabalhar com o gerente de projeto para desenvolver planos de contingência.
				Test.	Executar follow-up das atividades de ações corretivas.
				Test.	Coletar e analisar medidas do processo de teste.
				Clien.	Dar feedback pós-release com dados úteis aos testes.
				Clien.	Participar de reuniões de verificação de progresso e melhoria dos testes.

Figura 4.7: Nível 3b. - Integrado - Metas e ATRs

- Suportar a instalação de um sistema de administração de configuração e controle de mudança.
- Atribuir responsabilidades para monitoramento e controle de teste.
- Assegurar, planejar, monitorar e controlar as atividades do plano de teste.
- Participar de reuniões de andamento e auditoria para soluções de problemas e follow-up de ações corretivas.
- Designar responsabilidades aos testadores para a correção de problemas relativos a teste.

ATRs para Testadores

- Preparar diário de teste e relatório de incidências.
- Assegurar que itens de teste estão sob o controle do sistema de administração de configuração.
- Trabalhar com o gerente de teste para selecionar medidas apropriadas de controle e monitoramento.
- Trabalhar com o gerente de projeto para desenvolver planos de contingência.
- Executar follow-up das atividades de ações corretivas.
- Coletar e analisar medidas do processo de teste.

ATRs para Usuários/Clientes

- Dar feedback pós-release com dados úteis aos testes.
- Participar de reuniões de acompanhamento de teste e verificação de progresso.

4.6 Questionário de Avaliação

Para melhorar uma organização precisa avaliar a situação atual de seus processos, comparar o encontrado com os modelos de referências e assim identificar os pontos de melhoria. Para isso é preciso ter meios de identificar o nível e o grau de atendimento das diferentes metas e submetas. Para tal, com base no questionário (Burnstein, 2003), foi desenvolvido um questionário baseado no modelo de referência, metas, submetas e ATRs do MPT-PE.

O questionário é aplicado por um avaliador ou equipe de avaliação externa ao grupo avaliado. É distribuído as pessoas que participam do processo avaliado. Os resultados são ponderados pelo avaliador.

Cada questão tem quatro possibilidades de resposta: SIM, NÃO, NÃO se APLICA ou NÃO CONHECE. Deve ser respondida com base no conhecimento e experiência adquirida na realização dos projetos corrente na organização. A resposta será considerada como confidencial.

1. **Sim** quando a prática é bem estabelecida e consistentemente executada.
2. **Não** quando a prática não é bem estabelecida ou não é consistentemente executada.
3. **Não se aplica** quando você tem o conhecimento requerido do projeto ou da organização, mas você acredita que a questão não se aplicada para o projeto ou organização.
4. **Não conhece** quando você está indeciso como responder esta questão, ou você não têm experiência ou informações apropriadas.

Identificação e qualificação do respondedor. Informações sobre conhecimento, experiência, habilidades técnicas e responsabilidades atuais do respondedor sobre engenharia de software.

1. Identificação do Respondedor

   Nome

   Posição

   Empresa

   Telefone

   E-mail

   Data

2. Experiência do Respondedor

   Qual a melhor descrição de sua posição atual?

   Gerente

   Administrador Sênior ou superior

   Gerente de projeto

   Gerente de teste

   Líder de grupo Garantia de Qualidade de Software

   Líder de grupo de Processo de Engenharia de Software

   Líder de subgrupo relacionado a teste (*por favor, especifique o nome do subgrupo*).

   Outros (*por favor, especificar*)

Assistente Técnico

Engenheiro de Software

Engenheiro de Teste

Programador (desenvolvedor)

Analista

Membro de grupo Garantia de Qualidade de Software

Membro de grupo de Processo de Engenharia de Software

Membro de subgrupo relacionado a teste *(por favor, especificar)*

Outros (*por favor, especificar*)

3. Responsabilidades e Obrigações Atuais

   Quais atividades relacionadas a teste você esta efetivamente envolvido (você pode marcar mais de uma)?

   Política de teste e desenvolvimento de metas

   Planejamento de teste

   Projeto de caso e procedimento de teste

   Execução de teste

   Coleta e análise de medidas relacionadas a teste

   Coleta de dados de defeitos

   Manutenção de banco de dados de defeitos

   Desenvolvimento de padrões

   Revisões e auditorias

   Acompanhamento do status

   Treinamento

   Definições de métricas

   Recrutamento e contratação

   Comunicação com Usuário/Cliente

   Controle de processo

   Prevenção de defeito

Transferência de tecnologia

Avaliação de processo

Melhoria de processo

Engenharia e modelagem de confiabilidade

Teste de usabilidade

Avaliação de ferramentas

Reuso de processo

4. Qual é a extensão de sua experiência na industria de software?

Na organização atual	Número de anos
Na industria de software	Número de anos
Experiência em teste	Número de anos

4.6.1 Qualificação da Organização

1. Descreva o melhor que puder o tipo de sua organização.

   Desenvolve software militar

   Desenvolve software de aplicação

   Desenvolve software de telecomunicação

   Garantia de Qualidade de Software/teste de software/certificação

   Outros (*por favor, descreva*)

2. A maioria (acima de 50%) dos software desenvolvidos é para uso interno ou externo?

   Interno Externo Não aplicado

3. Quantas pessoas são empregadas na organização em avaliação?

   Número total de empregados

   Número de empregados engajados em desenvolvimento e/ou manutenção de software

Número de empregados engajados em teste de software

4. Por favor, descreva o percentual da equipe empenhada em testes como segue:

   Em tempo integral

   Em tempo parcial

   Consultores

5. A organização sob avaliação tem um grupo de processo de engenharia de software ou uma unidade similar?

   Sim Não

6. Como o grupo de teste é organizado? (Por favor, selecione um)

   Os desenvolvedores fazem o teste

   Grupo de teste dentro do desenvolvimento, reportando ao gerente de projeto

   Grupo de teste separado, reportando ao gerente de teste

   Parte do grupo de Garantia de Qualidade de Software

7. Como você poderia caracterizar a natureza de seu processo de teste?

   "Ad hoc"

   Informal

   Levemente estruturado

   Altamente estruturado

8. Com que freqüência os gerentes de projeto têm a necessidade de modificar os requisitos de cliente?

   Nunca Raramente Freqüentemente Muito freqüentemente

4.6.2 Questões sobre Metas de Maturidade

Nível 2 - Fase de Definição

Meta de Maturidade 2.1: Desenvolver os Objetivos de Teste

Questões

1. Metas, atividades e ferramentas para o processo de teste têm sido identificadas, documentadas e aprovadas?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. O processo de teste é definido?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. O processo de depuração é definido?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4. Há medidas básicas usadas para determinar a realização das metas de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

5. Há medidas básicas utilizadas para determinar a realização das metas de depuração?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

6. Metas de teste têm sido desenvolvidas com entradas de grupos usuário/cliente com respeito a suas necessidades?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

7. Foi desenvolvido um esquema de classificação dos defeitos básicos?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

8. Foi estabelecido um repositório de defeitos?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

9. Desenvolvedores/testadores depositam os defeitos no repositório consistentemente?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

Meta de Maturidade 2.2: Iniciar um Processo de Planejamento de Teste

Questões

1. Foi estabelecido um comitê ou grupo de planejamento de teste para toda a organização?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. Existem procedimentos para planejamento de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. Há suporte adequado e recursos para planejamento de teste para todos projetos?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4. Metas/objetivos de teste são utilizados como base para o planejamento de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

5. Têm sido desenvolvidos e distribuídos modelos de plano de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

6. Há ferramentas de planejamento apropriadas disponíveis para planejamento de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

7. Os desenvolvedores (testadores) têm sido treinados no uso de modelos e ferramentas de planejamento?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

8. Os desenvolvedores (testadores) são adequadamente treinados para desenvolver especificações, projeto de teste e casos de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

9. Estão disponíveis estimativas (tempo, orçamento, ferramentas) de projetos passados para uso no planejamento de teste.

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

10. É feito o planejamento de teste no nível de unidade?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

11. É feito o planejamento de teste no nível de integração?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

12. É feito o planejamento de teste no nível de sistema?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

13. É feito o planejamento de teste de aceitação?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

14. Outros itens relativos à teste tais como relatórios de andamento de teste, diários de teste, relatórios de incidências e sumários de teste são definidos em documentos da organização?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

15. Outros itens relativos à teste tais como relatório de andamento, diários de teste, relatórios de incidências e sumários de teste são completados a cada projeto?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

16. As medidas de teste são especificadas no plano de teste para todos os níveis?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

17. Desenvolvedores (testadores) coletam e armazenam medidas básicas relativas a teste?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

18. As medidas básicas de teste são utilizadas para assegurar que as metas de teste foram atingidas?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

Meta de Maturidade 2.3: Institucionalizar Técnicas e Métodos Básicos de Teste

Questões

1. Formulários e modelos adequados foram projetados para suportar técnicas básicas de teste?

    ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. São providos pela alta gerência recursos adequados para suportar o uso de técnicas e métodos básicos de teste, assim como ferramentas básicas de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. Desenvolvedores (testadores) foram treinados para aplicar as ferramentas básicas, formulários e métodos.

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4. Técnicas e métodos básicos de teste são aplicados por toda a organização?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

5. Ferramentas básicas de teste são aplicadas por toda a organização?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

6. As técnicas e ferramentas de teste a serem aplicadas são descritas nos planos de teste de múltiplos níveis?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

Nível 3a. - Definido

Meta de Maturidade 3.1: Estabelecer uma Organização de Teste de Software

Questões

1. O teste é reconhecido pela organização como uma atividade profissional?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. Há uma organização de teste de software responsável pelo teste de cada projeto?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. Há passos de carreira a fim de que membros do grupo de teste podem seguir?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4. Há recursos adequado designados para a organização de teste de software?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

5. Os membros da organização de teste estão treinados nos métodos de teste, planejamento de teste, teoria, ferramentas e técnicas?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

6. As responsabilidades e regras do grupo estão definidas?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

7. Há mecanismos formais de interação entre testadores e usuário/cliente?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

Meta de Maturidade 3.2: Estabelecer um Programa de Treinamento

Questões

1. Foram desenvolvidos planos de treinamento com a participação de gerentes de projeto/teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. São providos recursos suficientes para a implementação do programa de treinamento técnico?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. São utilizadas medidas para determinar a qualidade e a efetividade do programa de treinamento?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

Nível 3b. - Integrado

Meta de Maturidade 3.3: Integrar o Teste no Ciclo de Vida do Software

Questões

1. Um grupo ou comitê foi estabelecido para suportar integração de atividades de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. Foi adotado um modelo de ciclo de vida de software que suporta integração de atividades de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. Foram identificadas atividades de teste e requisitos de teste associados com cada fase do ciclo de vida?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4. É provido treinamento adequado para o esforço de integração de testes?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

5. São providos recursos suficientes para o esforço de integração de testes?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

6. O grupo de teste e o grupo de Garantia de Qualidade de Software têm desenvolvido um conjunto de padrões documentados para todos os produtos produzidos em cada fase do ciclo de vida?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

Meta de Maturidade 3.4: Controlar e Monitorar o Teste

Questões

1. Foi estabelecido um comitê ou grupo para suportar o monitoramento e controle de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

2. São disponíveis ferramentas e treinamento para suportar controle e monitoramento de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

3. Foram definidas e distribuídas medidas para acompanhamento de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4. Gerentes de projeto e gerentes de teste trabalham em conjunto nos planos de monitoramento e controle?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

5. São desenvolvidos planos de contingência para cada projeto para suportar controle de um teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

6. A organização coleta e armazena métricas de acompanhamento e controle para cada projeto?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

7. A organização de teste, apoiada pelo gerente de projeto, desenvolve planos de contingência para riscos de teste?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

8. Itens de teste estão sob controle de um sistema de administração de configuração?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

9. As atividades de controle e monitoramento de processo de teste são revisadas periodicamente?

   ( ) Sim ( ) Não ( ) Não se aplica ( ) Não conhece

4.6.3 Avaliação do Questionário

Cada meta de maturidade tem um conjunto de questões associadas. A meta é satisfeita quando a maioria das questões tem a resposta "sim". O grau de satisfação é calculado por (Burnstein, 2003):

Grau de satisfação é muito alto	se % de respostas "sim" for > 90
Grau de satisfação é alto	se % de respostas "sim" for 70-90
Grau de satisfação médio	se % de respostas "sim" for 50-69
Grau de satisfação baixo	se % de respostas "sim" for 30-49
Grau de satisfação muito baixo	se % de respostas "sim" for < 30

A meta é considerada aprovada, se receber grau de satisfação alto ou muito alto.

O nível é considerado atingido se todas as suas metas forem aprovadas.

4.7 Sugestões de Ferramentas de Auxílio ao Teste

O trabalho de planejamento, controle e execução dos testes, e ainda, a monitoração e controle do próprio processo de teste, pode ser auxiliado pelo uso de ferramentas. A seguir são sugeridas as ferramentas adequadas para cada nível de maturidade.

***Nível 1* - Ferramentas para o nível 1**

O micro computador (PC) de cada desenvolvedor deve ser equipado com: i) um processador de texto; ii) uma planilha; iii) um comparador de arquivo que indica se dois arquivos são iguais ou diferentes; iv) um e-mail para assegurar a comunicação adequada e; v) um programa de captura de tela que permita enviar o conteúdo da tela para um arquivo ou para impressão.

1. *Depurador interativo.* Auxilia o desenvolvedor na compreensão do código e localização de defeitos. Deve ter capacidade de TRACE BACK e BREAKPOINT para habilitar os desenvolvedores a entender a dinâmica de execução do programa e para identificar áreas suspeitas de código.

2. *Ferramenta de Construção da Configuração.* Esta ferramenta (por exemplo: o UNIX "MAKE") permite a construção de configurações de sistemas de software em um ambiente controlado. Elas suportam um processo de construção de sistema bem ordenado, administrado e repetível para desenvolvedores e testadores.

3. *Contador de Linha de Código (LOC).* Mede automaticamente o tamanho do software. Úteis para estimar custo, cálculo de volume de defeitos e da produtividade.

A planilha requerida neste Nível 1 do TMM é útil para registrar medidas simples como tempo real gasto em atividades de teste, medidas LOC para cada projeto e o número de cada tipo de defeito encontrado em cada projeto. Serve de repositório de defeitos.

***Nível 2* - Ferramentas para Fase de Definição**

1. *Planejador de Projetos e Teste.* Automatiza e padroniza o processo de planejamento de teste na organização. Suporta as especificações e registros de itens necessários para plano de teste. Incluem metas de teste, recursos de teste, custos, agendas, técnica a ser utilizada, projeto de teste e designação de responsáveis. Incluem teste de unidade, integração e de sistema.

2. *Verificadores de Erros em Tempo de Execução.* Estas ferramentas são conhecidas também como marcadores de limite, teste de memória e detectores de vazamento. Eles detectam problemas de memória, limites de campos, locação de memória livre e ocupada e o uso de locadores de memória. Esse grupo de ferramentas ajudará na descoberta de defeitos e fornecem normalmente mensagens de erros detalhadas que ajuda os usuários seguir a pista de defeitos.

3. *Ferramentas Suporte de Preparação de Teste.* Ferramentas que produzem dados de teste caixa preta utilizando método algorítmico. Por exemplo, ferramentas que requerem um gráfico de causa e efeito, ou uma entrada de classes de equivalência e análise de valor limite. Analisadores de controle de fluxo que geram gráficos de controle de fluxo, e analisadores de fluxo de dados. Ajudam a identificar ramos, caminhos básicos e uso de variáveis. Auxilia no projeto de casos de teste que satisfazem a cobertura das metas.

4. *Analisadores de Cobertura.* Auxilia no teste de caixa branca, no desenvolvimento de metas mensuráveis de conclusão de testes e assegura que as metas são satisfeitas.

   Executando o código alvo sob o controle de uma ferramenta de cobertura dá ao desenvolvedor a medida do grau de declaração e / ou cobertura de ramos e indicam quais as estruturas (caminhos) dos programas têm ou não sido exercitadas pelo conjunto de casos de teste. Se metas de cobertura não são satisfeitas com o conjunto de caso de teste atual, o desenvolvedor pode projetar casos de teste adicionais e re-executar o código sob o controle da ferramenta para determinar se as metas de cobertura foram agora satisfeitas.

5. *Ferramentas de Referência Cruzada.* Permite localizar a ocorrências de itens como eles aparecem em diferentes artefatos de software. É útil para construir modelos mentais de software com propósito de fazer mudanças, desenvolver testes, e re-execução de teste. Por exemplo, um desenvolvedor ou testador pode querer determinar onde uma variável específica aparece em toda fonte de código listada de modo a centrar os testes sobre as variáveis a completar. Outros itens que podem ser traçados ou referenciados são labels, literais, parâmetros e chamadas de sub-rotinas.

***Nível 3* - Ferramentas para as Fases Definido 3a. e Integrado 3b.**

1. *Ferramentas de Gerenciamento de Configuração.* Controla e monitora as mudanças de configuração para todos os artefatos relacionados ao projeto. Artefatos, chama-

dos itens de configuração, sob controle destas ferramentas inclui versões de código, mudança de requisitos, assim como itens relativos à teste tais como planos de teste, procedimentos de teste e casos de teste. É suporte para administrar, coordenar, e manter as dependências e relacionamentos entre todos os artefatos de software.

Por exemplo, o relacionamento entre requisitos e casos de teste, e elementos de projeto e casos de teste, pode ser estabelecido e disponibilizado utilizando uma ferramenta de gerenciamento de configuração.

2. *Registradores de Requisitos (Registradores de Caso de Uso).* Muitas organizações registram requisitos em uma linguagem de formato nativo que utiliza um processador de texto. Outras utilizam ferramentas de modelagem de requisitos, e registram informações em um formato gráfico tal como diagramas de fluxo de dados. Estas representações de requisitos podem não prover suporte adequado para testadores.

   De especial interesse são os registradores de casos de uso que auxiliam com a geração de casos de teste com base em casos de uso.

3. *Verificadores de Requisitos.* Checam requisitos por ambigüidade, consistência e demonstração de integridade. Porém, eles não são um substituto para uma revisão de requisitos, a qual, entre outras coisas, checará a integridade e a exatidão da especificação de requisitos.

4. *Rastreadores de Requisitos pelo Teste.* Consiste de uma matriz de rastreamento de requisitos, mais completa que a planilha do Nível 2. Provêem conexão automática entre requisitos, projeto, código fonte e casos de teste. As ferramentas também desempenham um papel no monitoramento do processo de teste indicando quais requisitos têm/não têm sido coberto por um caso de teste.

5. *Ferramentas de Replay-Capture.* Durante a execução do programa alvo a ferramenta registra todas as informações de entradas e saídas e em modo replay ela repete até tudo estar registrado. Tais ferramentas capturam os movimentos do mouse, batidas de teclado e imagens da tela. Após uma mudança no software a ferramenta pode repetir os testes registrados e validar os resultados da modificação, comparando estes resultados com os que serviram de base na versão anterior. Diferentes tipos de relatórios podem ser gerados pelas ferramentas, por exemplo: Relatório de tempo que lista o tempo de execução para cada teste; Relatório de falha que lista os testes que falharam; Relatório de regressão que lista somente aqueles testes cujos resultados tenham sido alterado desde a ativação do teste anterior; Relatório cumulativo que lista resultados de testes correntes e passados para cada teste executado.

Ferramentas de replay-capture podem ser nativas, a ferramenta e o software sendo testado residem no mesmo sistema, e ferramentas não nativas que requerem um sistema de hardware adicional para a ferramenta. As últimas são muito úteis quando testam sistemas incorporados.

6. *Comparadores.* Estas ferramentas comparam os resultados reais dos testes com os resultados esperados e marca as diferenças. Eles têm capacidades mais complexas do que os comparadores de arquivo. O software sob teste passará se a saída real e a esperada forem às mesmas dentro de uma tolerância permitida.

   Ferramentas mais sofisticadas têm habilidade para comparar outros tipos de saída por igualdade; por exemplo, telas e dados gráficos.

7. *Medidas de Complexidade.* São algumas vezes chamadas de relatórios de métricas. Medem a complexidade ciclomática e muitas vezes estarão integradas com outras ferramentas, tais como um medidor de tamanho (contador de linha de código), um analisador de fluxo de dados, e/ou um analisador de controle de fluxo. Algumas destas ferramentas poderão medir a complexidade na etapa de detalhamento do projeto se um módulo é escrito em um pseudo-código, linguagem estruturada e padronizada. Outras também poderão gerar métricas de ciência de software (Halstead's metrics) que são relativas à complexidade. Exemplos de tais métricas são número de um único operador, número de um único operando e número total de operandos. O nível de complexidade de um módulo dá uma indicação do risco do módulo em termos de probabilidade de defeitos, e o número de casos de teste necessários para o testar adequadamente. Organizações devem fixar limites sobre o nível de complexidade admitidos e módulos exibindo valores acima do limite devem ser considerados para re-projeto.

8. *Geradores de carga.* Estas ferramentas geram grande volume de dados necessários para teste de sistema, tais como testes de esforço e de performance. Geradores de carga podem ser usados para produzir um fluxo de transações. Por exemplo, se você estiver testando um sistema de telecomunicação você poderia necessitar simular uma série de transações na forma de chamadas telefônicas de diferentes tipos e duração, chegando de diferentes locais. Uma vez que um grande volume de dados é produzido quando o gerador de carga é utilizado, ferramentas para coleta e análise de dados devem ser disponíveis para os testadores.

9. *Rastreador de Defeitos.* Permite ao testador e ao desenvolvedor registrar e gerenciar defeitos por todo o ciclo de vida. Depois de um defeito ter sido detectado este é

registrado em um repositório de defeito suportado pela ferramenta. O status da solução de cada defeito é sempre atualizado. Código em que o defeito foi corrigido é submetido ao re-teste pela equipe de teste. Uma vez aprovado, um relatório de correção de defeito é feito e o status do defeito é atualizado no registro do rastreador de defeito. Rastreadores de defeito mais sofisticados suportam a comunicação entre desenvolvedores e testadores para promover a solução de defeitos, e a integração das atividades de teste através do ciclo de vida. Rastreador de defeitos pode também emitir (i) relatórios de defeitos que rastreiam os esforços de solução, e (ii) relatório sumário que inclui tipos de defeitos, sua época e freqüência de ocorrência.

4.8 Check-up do Processo de Teste

A resposta a questões isoladas no contexto de um questionário não é confortável. Não dá uma visão de conjunto e não capta o grau de desempenho, obrigando o respondente a opção de sim ou não. Para contornar estas dificuldades foi desenvolvido um novo procedimento mais amigável, denominado Check-up (Carvalho e Ferreira, 2004).

O Check-up é uma proposta de avaliação da maturidade do processo de teste de software de uma organização. Ele se diferencia do Questionário em alguns aspectos.

O Questionário tem uma estrutura bipolar, onde há só duas opções de desempenho na questão: Sim ou Não. É tudo ou nada. Cabe ao respondente, que pertence à organização avaliada, julgar se a organização merece o Sim ou o Não em cada quesito. O conjunto de respostas indica a classificação do nível de maturidade atual. Para evoluir a empresa deve se guiar pelas metas, submetas e ATRs do nível acima.

O Check-up tem uma estrutura de avaliação de desempenho. Consiste de uma tabela que apresenta as ATRs agrupadas pelas metas dos níveis de maturidade. Vide Figura 4.8, 4.9 e 4.10. O avaliador faz uma avaliação disciplinada dos processos da organização, e atribui um grau de desempenho em cada atividade. Utiliza-se uma escala Likert de cinco pontos para avaliar o nível de atendimento do aspecto observado, sendo que 0 significa "não atende" e 4 significa "atende plenamente".

O objetivo do Check-up, além de determinar o nível atual da empresa, é identificar os pontos de deficiência. ATRs com nota **menor ou igual a dois** devem ser priorizadas.

A evolução do processo de teste pode ser acompanhada pela aplicação periódica do check-up, por meio de um avaliador interno.

4.9 Considerações Finais

Neste capítulo apresentou-se um Modelo de Avaliação e Melhoria do Processo de Teste para Pequenas Empresas (MPT-PE) que foi desenvolvido com base nos modelos de maturidade de processo de software e de teste estudados, e respeitando as características particularidades e limitações das pequenas empresas. O processo de construção do modelo foi apresentado em detalhe. Foram também desenvolvidos e apresentados um Questionário de Avaliação, uma tabela de Check-up e Ferramentas para automatização do modelo.

Comparando-se o MPT-PE com o TMM, observa-se uma simplificação do modelo nos seguintes aspectos (Tabela 4.2):

Tabela 4.2: Comparação entre o TMM e o MPT-PE.

Características	***TMM***	***MPT-PE***
Níveis	5	4
Metas	13	7
Submetas	53	29
ATRs	385	82
Questões no questionário	201	58
Questões no check-up	não tem	60 (mais fácil de aplicar)
Redação dos textos	abrangente e complexa	simples e específica

Espera-se que com a redução apresentada acima, o MPT-PE seja um processo viável de ser implantado por micro e pequenas empresas, o que deve ser avaliado por meio de estudos empíricos.

Check-Up MPT-PE		*Nível de Atendimento*				
Nível 2 - *Fase de Definição*		De (0) - Não atende a (4) - Atende totalmente				
ATRs		0	1	2	3	4
2.1 Desenvolver os objetivos de teste						
Adm.	Liderar e prover recursos para teste e depuração.					
Adm.	Designar responsabilidades para teste e depuração.					
Adm.	Assegurar treinamento e ferramentas para depuração.					
Adm.	Assegurar treinamento e ferramentas para teste.					
Adm.	Encorajar a participação do cliente.					
Adm.	Assegurar que desenvolvedores registrem defeitos.					
Adm.	Implementar classificação e repositório de defeitos.					
Des/Test	Registrar defeitos no repositório.					
Des/Test	Desenvolver esquema de classificação de defeitos.					
Des/Test	Definir metas de teste e de depuração.					
Des/Test	Desenvolver planos de teste.					
Clien.	Participar da definição de metas de teste e depuração.					
2.2 Iniciar um processo de planejamento de teste						
Adm.	Designar comitê de planejamento de teste para a organização.					
Adm.	Selecionar ferramentas para planejamento de teste.					
Adm.	Assegurar que testes de múltiplos níveis são executados.					
Des/Test	Assegurar que questões de testabilidade são tratadas durante as fases de requisitos e projeto.					
Des/Test	Auxiliar o gerente no planejamento de teste.					
Des/Test	Determinar casos e procedimentos de teste.					
Des/Test	Trabalhar com clientes para desenvolver casos de uso e plano de aceitação.					
Des/Test	Coletar medidas quanto à realização das metas de teste.					
Clien.	Suprir informações para plano de teste de aceitação.					
Clien.	Participar da determinação de casos de uso.					
Clien.	Enunciar claramente os requisitos.					
2.3 Institucionalizar técnicas e métodos básicos de teste						
Adm.	Assegurar que o plano de teste inclua o uso de métodos de caixa preta e caixa branca.					
Adm.	Prover liderança e recursos para implementar técnicas, métodos e ferramentas básicas de teste.					
Des/Test	Auxiliar o gerente na seleção de ferramentas e métodos.					
Des/Test	Incluir no plano de teste técnicas de caixa preta/branca.					
Des/Test	Trabalhar com clientes para desenvolver critérios para teste de aceitação.					
Des/Test	Referir-se ao repositório de defeitos passados.					
Des/Test	Projetar uso de casos, especificações e procedimentos de teste baseados na estratégia de teste, técnicas e métodos.					
Des/Test	Revisar as representações de software.					
Des/Test	Conduzir atividades de teste.					
Des/Test	Executar casos de teste.					
Des/Test	Registrar dados relativos à teste.					
Des/Test	Registrar dados relativos a defeitos.					
Des/Test	Assegurar abordagem equilibrada em todos os níveis, unidade, integração e sistema.					
Clien.	Dar feedback de problemas de software para inclusão em teste de aceitação.					
Clien.	Participar nos testes de aceitação, teste alfa e beta.					
Clien.	Participar no desenvolvimento de caso de uso.					
Clien.	Trabalhar com analistas para que os requisitos sejam claros e testáveis.					

Figura 4.8: Check-up - MPT-PE - Nível 2 - Fase de Definição

Check-Up MPT-PE		*Nível de Atendimento*				
Nível 3a - *Definido*		De (0) - Não atende a (4) - Atende totalmente				
ATRs		0	1	2	3	4
3.1 Estabelecer uma organização de teste de software						
Adm.	Definir um setor especializado em teste.					
Adm.	Prover recursos para a organização de teste.					
Adm.	Estabelecer padrões, requisitos e plano de carreira do profissional de teste.					
Adm.	Avaliar e supervisionar pessoal de teste.					
Adm.	Designar responsabilidades do setor de teste.					
Adm.	Assegurar que todos os documentos de teste são preparados, tal como diário de teste e relatório de incidências.					
Adm.	Preparar e apresentar relatório resumo de teste.					
Adm.	Encorajar e prover caminho de comunicação para interação de testadores e clientes.					
Test.	Trabalhar com clientes para desenvolver casos de uso e plano de aceitação e perfil de uso.					
Test.	Trabalhar com analistas, projetistas, desenvolvedores e gerentes para planejar o teste e desenvolver produtos de qualidade.					
Test.	Trabalhar para estabelecer padrões de produto e processo.					
Test.	Trabalhar com gerentes de projeto/teste sobre planejamento e monitoramento do esforço de teste.					
Test.	Preparar documentos de pré e pós-teste.					
Test.	Projetar casos, seus procedimentos e execução de teste.					
Test.	Avaliar e aplicar novas técnicas de teste e ferramentas.					
Test.	Manter-se atualizado com novas técnicas, métodos e ferramentas de teste.					
Clien.	Trabalhar com os testadores no planejamento de teste de instalação e perfil de uso.					
3.2 Estabelecer um programa de treinamento.						
Adm.	Assegurar educação e treinamento dos membros do grupo de teste.					
Test.	Aplicar novos conhecimentos e habilidades adquiridas nos projetos da organização.					
Clien.	Participar de sessões de treinamento relativas a revisão, teste de aceitação e casos de uso.					

Figura 4.9: Check-up - MPT-PE - Nível 3a - Definido

Check-Up MPT-PE		Nível de Atendimento				
Nível 3b - *Integrado*		De (0) - Não atende a (4) - Atende totalmente				
ATRs		0	1	2	3	4
3.3 Integrar o teste no ciclo de vida do software						
Adm.	Trabalhar com os testadores para desenvolver padrões para teste do resultado do trabalho de cada fase do ciclo de vida.					
Adm.	Revisar, aprovar e adotar modelo de teste de integração (ex: modelo V).					
Adm.	Prover liderança e recursos para as atividades de integração do teste no ciclo de vida de software.					
Adm.	Prover treinamento para suporte das atividades de integração.					
Adm.	Assegurar que a integração das atividades de teste é aplicada por toda a organização e todos os projetos.					
Test.	Planejar as atividades integradas de teste.					
Test.	Executar atividades de testes integradas no ciclo de vida.					
Test.	Trabalhar com analistas, projetistas, desenvolvedores e cliente sobre questões de teste designados a cada fase do ciclo de vida.					
Clien.	Participar como convidado para prover consenso sobre atividades integradas de teste.					
3.4 Controlar e monitorar o teste.						
Adm.	Atribuir responsabilidades para monitoramento e controle do processo de teste.					
Adm.	Assegurar, planejar, monitorar e controlar as atividades do plano de teste.					
Adm.	Participar de reuniões de andamento e auditoria para soluções de problemas e follow-up de ações corretivas.					
Adm.	Designar responsabilidades aos testadores para a correção de problemas relativos a teste.					
Adm.	Suportar a instalação de um sistema de administração de configuração e controle de mudança.					
Test.	Preparar diário de teste e relatório de incidências.					
Test.	Coletar e analisar medidas do processo de teste.					
Test.	Trabalhar com o gerente de teste para selecionar medidas apropriadas de controle e monitoramento.					
Test.	Trabalhar com o gerente de projeto para desenvolver planos de contingência.					
Test.	Executar follow-up das atividades de ações corretivas.					
Test.	Assegurar que itens de teste estão sob o controle do sistema de administração de configuração.					
Clien.	Dar feedback pós-release com dados úteis aos testes.					
Clien.	Participar de reuniões de verificação de progresso e melhoria dos testes.					

Figura 4.10: Check-up - MPT-PE - Nível 3b - Integrado

Capítulo

5

Conclusão

Entender o teste como um processo e que evolui em paralelo com a evolução do processo de software possibilita entender o equilíbrio que deve haver entre eles, um não pode ser mais complexo que o outro, e tirar proveito dos estudos sobre maturidade da capacidade de processos para a busca de melhorias.

A avaliação de um processo de teste é o primeiro passo para melhorá-lo. A identificação das suas deficiências, com base em modelos de referência, permite elaborar um plano de evolução gradativa.

A primeira parte desse trabalho foi realizada através de levantamento bibliográfico em que se procurou entender cada um dos modelos de maturidade de capacidade de processo de software CMM, CMM-I, SPICE, MPSPE e MPS.Br.

O modelo de maturidade de teste TMM e seu método de avaliação foram estudados detalhadamente.

Como resultado desses estudos, foi proposto um modelo de Melhoria do Processo de Teste para Pequenas Empresas, suficientemente simplificado, mas conceitualmente de acordo com os modelos de referência CMM, SPICE e TMM. As características de aplicação às pequenas empresas foram baseadas nos modelos MPSPE e MPS.Br. O modelo desenvolvido MPT-PE resultou num escalonamento de níveis mais suave, menor número de ATRs e redação mais simples e adequada ao ambiente da pequena empresa. Considera-se o MPT-PE mais leve e fácil de ser aplicado que o modelo TMM.

Simples e econômico por apresentar um menor número de níveis, corresponde a três níveis do TMM. Não tem os dois níveis mais elevados que são mais complexos, mais caros e difí-ceis de implementar, e que não são o foco imediato da pequena empresa. Por ter um escalo-namento de níveis mais suave, com dois níveis graduais correspondendo ao Nível 3 do TMM.

Tem menor número de ATRs, e com redação apropriada ao ambiente das empresas menores. Não exige comitês, reuniões e procedimentos burocráticos complexos que não são viáveis e não fazem parte da realidade das empresas com pequeno número de funcionário.

Deixa a opção de integração dos testes no ciclo de vida condicionada ao julgamento do seu custo beneficio, representando melhor uso de recursos escassos.

Foram analisadas as ferramentas para cada um dos níveis e o questionário desenvolvido para o TMM foi simplificado e adaptado ao novo modelo MPT-PE.

Também para a avaliação do processo de teste e com base no novo modelo MPT-PE foi desenvolvida uma tabela de Check-up, que diferentemente do questionário, permite avaliar o grau de desempenho de cada ATR. Identifica, assim, os pontos fracos e fortes do processo avaliado, e apresenta visualmente em uma única tabela o perfil de desempenho da organização. Contribuí para a estratégia de melhoria priorizando as ATRs que devem ser trabalhadas primeiro.

Comparando-se o MPT-PE com o TMM, observa-se uma simplificação do modelo.

O TMM apresenta 5 níveis, 13 metas, 53 sub-metas, 385 ATRs, 201 questões do questionário, redação abrangente, mas complexas.

Já o MPT-PE apresenta 4 níveis, 7 metas, 29 sub-metas, 58 questões do questionário, redação simples e adequada a pequena empresa. Apresenta avaliação mais fácil por meio do Check-up.

5.1 Trabalhos Futuros

Com base no modelo proposto no presente trabalho alguns temas e estudos importantes ainda precisam ser realizados e dão margem para uma série de trabalhos futuros:

1. Desenvolver estudos práticos de aplicação do modelo proposto MPT-PE em micro e pequenas empresas da região, verificando compreensão, adequação e usabilidade do modelo.

2. Testar a eficiência do Questionário de Avaliação do processo de teste em um grupo de empresas selecionadas.

3. Testar a efetividade da tabela de Check-up na avaliação do processo de teste das empresas selecionadas.

4. Comparar os resultados do Questionário de Avaliação e da tabela de Check-up, aplicados nas mesmas empresas.

5. Mapeamento dos processos atuais de software e de teste, de micros e pequenas empresas. Verificar a integração entre eles e no ciclo de vida. Avaliar as capacidades, pontos fortes e pontos fortes. Eleger as melhores práticas como referência.

6. Analisar, testar e desenvolver ferramentas de teste para as pequenas e micros empresas, principalmente considerando ferramentas disponíveis em repositórios de software livre.

Referências Bibliográficas

Academy, P. Process academy's white paper what is the cmmi? version:1.00. Process Academy's Inc, disponível em: http://www.dovico.com/documents/What-is-the-CMMI-process-improvement-v1%-00-20030730.pdf [Acesso em: 17 de agosto de 2005, 16:23:48], 2003.

Belloquim Cmmi: O futuro do cmm. Choose Technologies, disponível em: http://www.choose.com.br/infochoose/artigos/42art01.htm [Acesso em: 17 de agosto de 2005, 16:23:46], 2004.

Burnstein, I. *Practical software testing*. Springer-Verlag, 2003.

Burnstein, I.; Suwannasart, T.; Carlson, C. *Developing a testing maturity model: Part i*. Relatório Técnico, Illinois Institute of Technology, disponível em: http://www.improveqs.nl/ [Acesso em: 23 de março de 2004, 15:55:48], 1996a.

Burnstein, I.; Suwannasart, T.; Carlson, C. *Developing a testing maturity model: Part ii*. Relatório Técnico, Illinois Institute of Technology, disponível em: http://www.improveqs.nl/ [Acesso em: 23 de março de 2004, 15:55:48], 1996b.

Campos, V. F. *Gerência da qualidade total*, cáp. Capítulo 1 - Introdução à Produtividade Bloch Editores S.A., p. 20, 1989a.

Campos, V. F. *Gerência da qualidade total*, cáp. Capítulo 3.1.2. Conceito de Processo Bloch Editores S.A., p. 39 – 42, 1989b.

Carvalho, R. B.; Ferreira, M. A. T. Check-up do portal: Uma proposta de metodologia para avaliação técnica-gerencial de portais corporativos, p. 166. 2004.

Drabick, R. Growth of maturity in the testing process. International Software Testing Institute, disponível em: http://www.softtest.org/articles/rdrabick3.htm [Acesso em: 29 de março de 2004, 09:38:27], 1999.

Evolutif Test organization maturity. jqai, disponível em: http://www.evolutif.co.uk/tom/overview.html, 2002.

FEIGENBAUM, A. V. *Total quality control third edition*, cáp. Tema1 O Sistema da Qualidade McGraw-Hill Book Company, p. 7, 1983.

GELPERIN, D. How to support better software testing. *Application Trends*, 1996.

GRIFO, E. *Iniciando os conceitos da qualidade total 2 edição*, cáp. Tema1 O Sistema da Qualidade Editora Pioneira, p. 5, 1994.

HAZAN, C. Uma ferramenta na busca da excelência medições de software. Portal do SERPRO, disponível em: `http://www.serpro.gov.br/publicacao/tematec/tematec/2004/ttec75` [Acesso em: 17 de agosto de 2005, 16:23:46], 2004.

HERNDON, M. A.; MOORE, R.; PHILLIPS, M.; WALKER, J.; WEST, L. Interpreting capability maturity model integration (cmmi) for service orgabizations - a systems engineering and integration services example. CMU/SEI-2003-TN-005, disponível em: `http://ftp.sei.cmu.edu/pub/documents/03.reports/pdf/03tn005.pdf` [Acesso em: 19 de agosto de 2005, 09:32:58], 2003.

ISO-9126 *Nbr iso/iec 9126 - tecnologia de informação.* Relatório Técnico, ABNT Associação Brasileira de Normas Técnicas, 1994.

ISO/IEC-15504 Publicada a norma iso 15504 (spice). Disponível em: `http://www.inovacaotecnologica.com.br/noticias/010150031114.html` [Acesso em: 15 de abril de 2004 03:38:37], 2003.

KOOMAN, T.; POL, M. Test process improvement. jqai, 2002.

MCT Porte das organizações. Disponível em: `http://www.mct.gov.br/Temas/info/Dsi/Quali2001/PorteOrg2001.htm` [Acesso em: 09 de março de 2004, 16:54:54], 2000.

MOLINARI, L. *Testes de software produzindo sistemas melhores e mais confiáveis*, cáp. Capítulo I - Evolução da Qualidade de Software Editora Érica Ltda, p. 41 – 45, 2003.

PAULK, M. C.; CURTIS, B.; CHRISSIS, M. B.; WEBER, C. V. *Capability maturity model sm for software, version 1.1.* Relatório Técnico CMU/SEI-93-TR-024 ESC-TR-93-177, SEI Joint Program Office Hanscom AFB, MA 01731-2116, 1993.

PRESSMAN, R. S. *Engenharia de software.* Makron Books, 2001.

RIOS, E. Considerações sobre o tmm - test maturity model em comparação com o cmm. Disponível em: `http://www.alats.org.br/artigos/` [Acesso em: 15 de abril de 2004 11:02:01], 2000.

ROCHA, A. R. Modelo de referência para melhoria de processo do software (mr-mps). softex, disponível em: `http://www.softex.br/media/MPSBR-Modelo-de-Referencia.pdf`[17 de agosto de 2005, 11:29:59], 2005.

ROCHA, A. R. C.; MALDONADO, J. C.; WEBER, K. C. *Qualidade de software teoria e prática.* São Paulo - SP: Prentice Hall, 2001.

ROSA, P. G. *Modelo de avaliação de processo de software para a pequena empresa.* Dissertação de Mestrado, ICMSC-USP, São Carlos - SP, 1997.

SALVIANO, C. F. *Qualidade e produtividade em software.* Campinas, São Paulo: Makron Books, 2001.

SOFTEX; COPPE/UFRJ; CESAR; CENPRA mpsbr - melhoria de processo do software brasileiro. Disponível em:`http://www.softex.com.br/mpsbr` [Acesso em: 17 de Agosto de 2005, 18:49:02], 2003.

SOFTWAREFUTURES Testing assessment program. jqai, disponível em: `http://www.softwarefutures.ltd.uk`, 2002.

SOFTWARETEST Technology builders inc. jqai, disponível em: `http://www.softtest.com/pages/kpbai.htm`, 2002.

SOMMERVILLE, I. *Engenharia de software 6ł edição*, cáp. Capítulo 25 Melhoria de processo Addison Wesley, p. 479 – 480, 2003.

STAAB, T. C. Can a testing maturity model help improve my testing process? *jqai*, p. 10–16, 2002.

SWINKELS, R. *A comparison of tmm and other test process improvement models.* Frits Philips Institute, disponível em: `http://tmitwww.tm.tue.nl/research/v2m2/wp1/12-4-1-FPdef.pdf/>`. [Acesso em: 12 de março de 2004, 15:35:43], 2000.

WEBER, K. C.; ARAUJO, E.; MACHADO, C. A. F.; SCALET, D.; SALVIANO, C. F.; ROCHA, A. R. C. Modelo de referência e método de avaliação para melhoria de processo de software - versão 1.0 (mr-mps e ma-mps). IV - Simpósio Brasileiro de Qualidade de Software, disponível em:`http://www.softex.br/media/MR_MA-MPS.pdf`[Acesso em: 17 de agosto de 2005, 16:20:36], 2005.

Apêndice

A

Conjunto de Atividades, Tarefas e Resposabilidades do TMM

Conforme mencionado no Capítulo 3, o TMM é um modelo de maturidade de teste que foi desenvolvido tendo como base o CMM, com o qual é compatível e pode ser utilizado em conjunto. Da mesma forma que o CMM, o TMM também apresenta uma estrutura hierárquica dividida em cinco níveis de maturidade. Cada nível é dividido em metas, submetas e um conjunto de atividades, tarefas e responsabilidades, referenciadas por ATRs, que definem a capacidade de teste da organização. O objetivo deste apêndice é descrever o conjunto completo de ATRs do TMM visando tornar mais fácil a comparação do TMM com o MPT-PE. Destaca-se que o texto apresentado a seguir consiste de uma tradução, com modificações, do Capítulo do livro Practical Software Testing de Ilene Burnstein Burnstein (2003).

A.1 TMM - ATRs

Parte 1 - TMM Atividades, Testes e Responsabilidades (ATRs)

Esta parte contem um conjunto de atividades, tarefas e responsabilidades (ATRs) de três visões críticas como descrito no TMM. Esta seção é organizada pelo nível do TMM,

pelas metas de maturidade dentro de cada nível e pelas três visões críticas. Existe em cada nível:

i. Uma declaração de cada meta de maturidade;

ii. ATRs para os administradores;

iii. ATRs para os desenvolvedores / testadores;

iv. ATRS para usuários / clientes.

Atividades, Tarefas e Responsabilidades (ATRs)

Associada com cada meta de maturidade é também necessário um conjunto de ATRs para suporta-lo. As ATRs descrevem as práticas que necessitam ser institucionalizada para alcançar maturidade do processo de teste. As ATRs são designadas para três grupos essenciais críticos para o esforço de teste - administradores, desenvolvedores/testadores, usuários/clientes. A inclusão da visão dos três grupos críticos é específica do TMM. As visões especificam regras para todos os participantes no processo de teste e fornecem suporte interno e externo para avaliação e melhoria do processo de teste.

Nível 2 - Fase de Definição

Meta de Maturidade 2.1: Desenvolver Metas e Política de Teste e Depuração

Recordando que no nível 2 não há requisito para um grupo dedicado a teste, assim as ATRs são formalmente designadas somente para os desenvolvedores. Se uma organização tem um grupo de especialistas de teste, então estas ATRs podem ser transferidas para este grupo.

ATRs para Administradores (superior e gerente de projeto)

- Prover liderança, recursos adequados e fundos para formar o comitê (equipe ou força tarefa) sobre teste e depuração. A constituição do comitê é gerencial, com a participação da equipe técnica.
- Tornar disponível qualquer políticas e metas de teste/depuração pré-existentes ou amostras.

- Assumir um papel de liderança no desenvolvimento da política de teste/depuração.
- Apoiar as recomendações e políticas do comitê:
  - distribuindo documentos de política e meta de teste/depuração para gerentes de projeto, desenvolvedores/testadores e outros interessados e solicitando feedback desses grupos;
  - designando uma equipe permanente para supervisionar conformidade e modificação da política;
- Assegurar treinamento, educação e ferramentas necessárias para elaborar metas de teste/depuração bem definidas e disponibilizar as políticas.
- Promover a mudança cultural necessária para implementar as políticas de teste/depuração.
- Designar responsabilidades para teste e depuração.
- Encorajar entradas/feedback dos grupos chave de usuários/clientes para políticas de teste/depuração.
- Assegurar que há suporte para desenvolvimento de um esquema de classificação e repositório de defeito. O esquema de classificação e o repositório devem estar disponíveis para todo o pessoal do projeto para estudo e avaliação.
- Assegurar que desenvolvedores (testadores) estejam familiarizados com o esquema de classificação de defeitos e registrem no repositório os defeitos ocorridos para cada projeto.
- Revisar periodicamente as metas e políticas de teste/depuração.

ATRs para Desenvolvedores

- Trabalhar com o administrador para desenvolver metas e políticas de teste/depuração.
- Participar da equipe que supervisiona aplicação de política de teste/depuração e administra alterações.
- Familiarizar-se com o conjunto de políticas e metas de teste/depuração, manter-se atualizado com as revisões e fazer sugestões de mudança quando apropriada.

- Fixar metas de teste para o projeto e para cada nível de teste que reflitam as metas e as políticas de teste da organização.
- Desenvolver planos de teste que estão em conformidade com a política de teste.
- Conduzir atividades de teste que estão em conformidade com a política organizacional.
- Trabalhar com os administradores de projeto para desenvolver um esquema de classificação e um repositório de defeito.
- Registrar no repositório os defeitos que ocorrem no projeto.
- Participar nas revisões periódicas das políticas e metas de teste/depuração.

ATRs para Usuários/Clientes

- Dar entrada/feedback sobre políticas e metas de teste e depuração quando solicitado pelo administrador. (Estes grupos desempenham um importante papel na formação da políticas e metas de teste/depuração da organização, uma vez que estas metas e políticas refletem os esforços da organização para garantir a satisfação do cliente/usuário. O feedback desses grupos deve ser encorajado pelo administrador e pela Garantia de Qualidade de Software. Em geral, as necessidades de seus clientes e do mercado terão um impacto na natureza das políticas e metas de teste/depuração da organização.)

Meta de Maturidade 2.2: Iniciar um Processo de Planejamento de Teste

ATRs para Administradores

- Prover liderança, fundos e recursos para um comitê de planejamento de teste para toda a organização.
- Assegurar que as declarações da política de planejamento de teste são discutidas e aprovadas.
- Promover a mudança cultural para suporte de planejamento de teste.

- Assegurar que as declarações da política de teste, padrões de qualidade e modelos de plano de teste suportam o planejamento de teste com comprometimento de recursos, ferramentas e treinamento.

- Assegurar que as declarações da política de teste contenham um mecanismo formal para entrada de usuário/cliente no planejamento de processo de teste, especialmente para teste de aceitação. (No nível mais alto do TMM este inclui também o planejamento de teste de usabilidade.)

- Assegurar que todos os projetos estão em conformidade com a política de planejamento de teste.

- Assegurar que os modelos de plano de teste são aplicados uniformemente em todos os projetos.

- Assegurar que todos os desenvolvedores (testadores) completem todos os documentos necessários, tais como diários de teste e relatórios de incidência de teste.

- Assegurar que os planos de teste são preparados para todos os níveis de teste: unidade, integração, sistema e aceitação.

- Participar de aulas de treinamento para planejamento de teste, uso de modelos de plano de teste, identificação/estimação de riscos de teste, ferramentas de planejamento (estas se aplicam para administradores de projeto e administradores de teste quando há um grupo de teste.)

- Selecionar ferramentas apropriadas para o planejamento de teste.

- Preparar planos de teste de múltiplos níveis para cada projeto com entradas e suporte de desenvolvedores. (Este se aplica para os administradores de projeto e administradores de teste quando há um grupo de teste.) Administradores de projeto/teste usam os modelos de plano de teste da organização como guia na preparação de planos de teste. Riscos de teste são identificados, e métricas simples são selecionadas e incluídas no plano de teste para assegurar que as metas de teste estão sendo alcançadas. Dados de defeitos de projetos passados são usados como apropriado para ajudar no planejamento de teste.

- Assegurar que desenvolvedores (testadores) preparem plano de teste tais como casos de teste e procedimentos de teste.

- Assegurar que documentos auxiliares de teste são preparados tais como relatórios informativos de teste e diários de teste.
- Revisar planos de teste com os desenvolvedores (testadores).
- Assegurar que todos os desenvolvedores (testadores) completem todos os documentos necessários de teste tais como diários de teste e relatórios de incidência de teste.
- Preparar um relatório sumário de teste (administradores de projeto/teste).
- Promover interações com desenvolvedores (testadores) e clientes para desenvolver planos de teste de aceitação, casos de uso e/ou qualquer outra interação típica de usuário/computador.
- Revisão periódica das políticas de planejamento de teste com a equipe técnica.

ATRs para Desenvolvedores

- Participar como membros do comitê de planejamento de teste.
- Assistir aulas de treinamento para planejamento de teste, uso de ferramentas e modelos para planejamento e para identificação de riscos de teste.
- Auxiliar o gerente de projeto (teste) na determinação de metas de teste, riscos de teste, e custos de teste para planejamento em todos os níveis de teste.
- Auxiliar o gerente de projeto (teste) na seleção de métodos de teste, procedimentos e ferramentas.
- Desenvolver especificações de casos de teste, especificações de procedimento de teste e outros documentos relacionados a teste.
- Trabalhar com analistas e projetistas para assegurar que as questões de testabilidade são tratadas durante as fases de requisitos e projeto para suportar o planejamento e projeto de teste.
- Coletar medidas simples relativas a teste para assegurar que as metas de teste estão sendo realizadas.
- Completar todos os documentos necessários antes e após os teste tais como relatórios de comunicação de teste, relatório de incidentes de teste e diário de teste.

- Trabalhar com clientes para desenvolver casos de uso e/ou qualquer outra descrição típica de interação usuário/computador e planos de teste de aceitação.
- Participar em revisões de políticas de planejamento de teste.

ATRs para Usuários/Clientes

- Enunciar claramente os requisitos.
- Suprir entrada e consenso para plano de teste de aceitação. O requisito funcional e atributos relativos a desempenho que são esperados pelo cliente/usuários devem ser especificados claramente e quantitativamente se possível.
- Prover entrada para o desenvolvimento de casos de uso e/ou qualquer outra descrição de interação típica de usuário/computador.

Meta de Maturidade 2.3: Institucionalizar Técnicas e Métodos Básicos de Teste

ATRs para Administradores

- Prover liderança, suporte e fundos para um comitê ou um grupo responsável por identificar, avaliar e recomendar técnicas básicas de teste, métodos e ferramentas.
- Assegurar que as recomendações dos grupos são documentadas, distribuídas e aprovadas.
- Assegurar que políticas e padrões das organizações são projetados para promover a institucionalização de métodos de projeto de teste da caixa preta/caixa branca.
- Assegurar que políticas e padrões de teste requeiram teste de múltiplos níveis.
- Assegurar que desenvolvedores (testadores) recebam educação e treinamento necessários para entender e aplicar métodos de caixa preta e branca, e desenvolver casos de teste, procedimentos de teste, diários de teste e relatórios de incidência de teste.
- Assegurar que desenvolvedores (testadores) tenham a educação e treinamento necessário para aplicar teste de múltiplos níveis.

- Prover recursos para suportar o uso de métodos de teste de caixa preta/branca tal como ferramentas e modelos.
- Encorajar a cooperação o entre desenvolvedor (testador), analistas de requisitos e projetistas sobre questões de teste.
- Assegurar que planos de teste incluam o uso de métodos de projeto de teste de caixa preta/branca.
- Assegurar que testes de múltiplos níveis são cobertos nos planos de teste.
- Promover mudanças culturais necessárias para suportar a aplicação de técnicas e ferramentas básicas de teste por toda a organização.
- Promover mudanças culturais necessárias para suportar teste de múltiplo nível.
- Alocar tempo e recursos adequados para projetar e executar os testes de caixa preta/branca, testes de múltiplos níveis e analisar os resultados de teste.
- Ajustar o cronograma de projeto para que os teste de múltiplo níveis possam ser executados adequadamente.
- Prover visibilidade para sucesso da aplicação métodos e técnicas de teste.
- Revisar periodicamente técnicas, métodos e ferramentas básicas de teste.

ATRs para Desenvolvedores

- Participar como membro de um comitê responsável pela avaliação e recomendação de técnicas de teste, métodos e ferramentas.
- Assistir aulas e sessões de treinamento, ler materiais, adquirir ferramentas, trabalhar com colegas instruídos, e ganhar experiência na aplicação de métodos de projeto de teste caixa branca e caixa preta.
- Assistir aulas e sessões de treinamento, ler materiais, adquirir ferramentas, trabalhar com colegas instruídos, e ganhar experiência em teste de unidade, integração, sistema e aceitação.
- Assegurar que um equilíbrio de abordagens de teste é utilizado no projeto de caso de teste em todos os níveis do teste.

- Projetar casos de teste, especificações de teste e procedimentos de teste baseados no conhecimento de estratégias de teste, técnicas e métodos.
- Montar ambiente de software/hardware necessário para executar testes. Desligar as facilidades quando os testes são completados.
- Executar casos de teste em todos os níveis do teste.
- Registrar dados relativos a teste.
- Registrar dados relativos a defeito.
- Interagir com especialistas e projetistas para revisar suas representações do software. Representações incluem especificações de entrada/saída, pseudocódigo, diagramas de estado e gráfico de controle de fluxo que são ricas fontes para desenvolvimento de caso de teste. Estas representações são vitais para projetar casos de teste de caixa branco/preta.
- Referir-se ao repositório de defeito para aprender sobre defeitos passados em projetos similares, para ajudar no próprio projeto de teste.
- Assistir o administrador (projeto/teste) para assegurar que o teste de múltiplos níveis e uso de técnicas de teste de caixa preta/branca são parte da política organizacional, estão incorporados nos planos de teste e são aplicados em toda a organização.
- Trabalhar com gerentes de projeto (teste) para assegurar que há tempo e recursos para teste em todos os níveis.
- Orientar os colegas que desejam adquirir o conhecimento e experiência necessária para realizar testes de múltiplos níveis.
- Trabalhar com usuários/clientes para desenvolver casos de uso e/ou qualquer outra descrição de interação do tipo usuário/computador e critérios de aceitação necessários para testes de múltiplos níveis.
- Trabalhar com usuários/clientes por ocasião dos testes de aceitação para assegurar que problemas sejam resolvidos para a satisfação deles.
- Participar em revisões periódicas das técnicas e métodos básicos de teste.

ATRs para Usuários/Clientes

- Servir de elo de ligação para interagir com a equipe de desenvolvimento (testes) nas questões relativas a teste.
- Trabalhar com analistas para que os requisitos do sistema sejam completos, claros e testáveis.
- Participar nos testes de aceitação e/ou no teste alfa e beta.
- Prover feedback e registrar os problemas imediatamente a fim de que problemas possam ser tratados durante o teste de aceitação (e teste de instalação).
- Participar no desenvolvimento de caso de uso (e de perfis de uso no nível mais alto do TMM).

Nível 3 - Integração

Meta de Maturidade 3.1: Estabelecer uma Organização de Teste

ATRs para Administradores

- Prover liderança, recursos e fundos para um comitê definir a estrutura da organização de teste.
- Assegurar que o papel e responsabilidades da organização de teste estão estabelecidos na declaração da política de teste.
- Assegurar que os negócios do cliente são refletidos na política de teste da organização.
- Assegurar que quaisquer mudanças necessárias para suportar uma organização de teste são incorporadas na estrutura de representação organizacional.
- Estabelecer padrões, requisitos, níveis de equilíbrio, responsabilidades e carreira do profissional de teste.
- Suportar as mudanças culturais necessárias para colocar em prática uma organização de teste.
- Prover recursos, equipe e fundos para a organização de teste.

- Assegurar a cooperação entre desenvolvedores, especialistas de teste e a organização de garantia de qualidade de software.
- Recrutar e empregar especialistas de teste e gerentes de teste.
- Avaliar e supervisionar o pessoal de teste.
- Iniciar periodicamente ações para avaliar a maturidade, eficiência e performance da organização do teste.
- Suportar educação e treinamento dos membros do grupo de teste.
- Suportar as mudanças culturais necessárias para manter uma organização de teste.
- Monitorar a performance do grupo de teste.
- Suportar o crescimento da organização de teste e esforços de melhoria processo de teste.
- Encorajar e prover caminhos de comunicação para interação de testadores/clientes.
- Promover os sucessos que resultem do trabalho da organização de teste.

ATRs para Testadores

- Servir como membros do comitê de teste da organização.
- Estar atento a tarefas, responsabilidades e passos de carreira do especialista de teste.
- Manter-se atualizado com respeito as novas técnicas, métodos e ferramentas de teste.
- Manter-se atualizado sobre técnicas de planejamento de teste e administração de risco.
- Trabalhar com analistas, projetistas, desenvolvedores, gerentes de projeto e equipe de Garantia de Qualidade de Software para planejar o teste e desenvolver produtos de software de qualidade.
- Trabalhar com clientes sobre questões tais como planejamento de teste de aceitação, casos de uso (ou equivalente) e perfil de uso (onde apropriado).

- Trabalhar com gerentes de projeto/teste sobre planejamento de teste e monitoramento dos esforços de teste.
- Trabalhar com gerentes de projeto/teste para identificar e priorizar riscos de teste.
- Projetar casos de teste, procedimentos de teste e executar testes.
- Preparar documentos de pré- e pós-teste.
- Coletar, analisar e aplicar medidas relacionadas a teste.
- Contribuir para elaborar política de teste.
- Manter o repositório de teste.
- Manter o repositório de defeito.
- Recrutar novos membros.
- Orientar novos membros da equipe de teste.
- Trabalhar para estabelecer padrões de produto e processo.
- Avaliar e aplicar novas técnicas de teste e ferramentas.
- Participar em revisões técnicas.
- Contribuir para a avaliação e melhoria do processo de teste.
- Participar de revisões periódicas da organização de teste.

ATRs para Usuários/Clientes

- Expressar suas preocupações e necessidades para administração das questões de teste e da organização de teste quando solicitado.
- Trabalhar com testadores nos itens tais como planejamento de teste de aceitação, planejamento de teste de instalação, casos de uso (ou equivalente) e perfis de uso (onde apropriado).

Meta de Maturidade 3.2: Estabelecer um Programa de Treinamento Técnico

ATRs para Administradores

- Prover liderança, fundos e suporte para um comitê de programa de treinamento técnico.
- Desenvolver uma política organizacional para treinamento com a participação da equipe técnica e dos administradores de projeto/teste. Obter aprovação e distribuir para todas as partes relevantes.
- Promover o desenvolvimento de um programa de treinamento técnico provendo fundos para o programa, equipe, recursos, materiais de treinamento, ferramentas e laboratórios.
- Recrutar e empregar equipe qualificada para a organização de treinamento.
- Assegurar que planos de treinamento são desenvolvidos para suportar as necessidades e metas de treinamento para todos os projetos.
- Monitorar e revisar o programa de treinamento com a equipe técnica para avaliar a eficácia e para identificar áreas para melhoria.
- Promover a mudança cultural necessária para suportar um programa de treinamento.
- Prover a visibilidade para o programa de treinamento.
- Promover os sucessos resultantes do programa de treinamento.
- Recomendar os membros da equipe para as seções de treinamento.

ATRs para Testadores

- Participar como membro do comitê de programa de treinamento técnico.
- Atender sessões de treinamento para melhorar suas habilidades e capacidades de teste.

- Aplicar novos conhecimentos e habilidades adquiridas nos projetos da organização.
- Requisitar o desenvolvimento de novas sessões de treinamento para adquirir as habilidades necessárias.
- Identificar aqueles nos grupos de teste que poderiam se beneficiar do treinamento.
- Participar em revisões periódicas do programa de treinamento.

ATRs para Usuários/Clientes

- Se a política da organizacional permitir, atender a sessões de treinamento para promover a participação em áreas específicas, tais como revisões técnicas, planejamento de teste de aceitação e desenvolvimento de casos de uso (ou representações equivalentes).

Meta de Maturidade 3.3: Integrar o Teste no Ciclo de Vida do Software

ATRs para Administradores

- Prover liderança, recursos e suporte para um comitê focar nas atividades de integração de teste e modelos.
- Revisar, aprovar e adotar (com a participação de testadores e outros técnicos da equipe) um modelo de integração de teste (o Modelo-V é um exemplo deste tipo).
- Assegurar que a atividade de integração de teste é uma parte da política e documentos padrão de teste.
- Prover treinamento para suporte das atividades de integração.
- Assegurar que a integração de atividades de teste é aplicada por toda a organização para todos os projetos.
- Assegurar que todos os testadores são treinados para executar atividades integradas de teste.
- Promover mudanças culturais necessárias para as atividades de integração.

- Monitorar e revisar as atividades integradas de teste, distribuição, avaliação e se necessário propor melhorias.
- Trabalhar com testadores para desenvolver padrões para teste do produto do trabalho resultante em cada fase do ciclo de vida.
- Assegurar que os padrões do produto do trabalho de cada fase são preservados.
- Promover sucessos dos esforços de integração.

ATRs para Testadores

- Servir no comitê dedicado para revisar, aprovar e instituir um modelo de integração de atividades de teste e procedimentos associados.
- Atender a sessões de treinamento para preparar para a integração das atividades de teste.
- Planejar as atividades integradas de teste (políticas de planejamento de teste podem requerer modificações).
- Aplicar as atividades normalizadas de integração de teste para cada projeto.
- Executar as atividades de teste requeridas através do ciclo de vida do software como especificado no modelo aprovado, na política organizacional e documentos padrão.
- Preparar todos os testes entregues que são requeridos em cada atividade integrada de teste.
- Trabalhar com a administração e a Garantia de Qualidade de Software para desenvolver padrões de teste para os produtos do trabalho entregues.
- Trabalhar com analistas, projetistas, desenvolvedores e clientes sobre questões de teste designados a cada fase do ciclo de vida.
- Assegurar que cada trabalho de teste do produto/entrega atende os padrões organizacionais.
- Participar em revisões periódicas das atividades de integração.

ATRs para Usuários/Clientes

- Prover consenso e suporte para atividades integradas de teste e planejamento dos testes cedo no ciclo de vida. Por exemplo: prover suporte para planejamento de teste de aceitação, caso de uso (ou representações equivalentes) e desenvolvimento de perfil de uso durante as fases de requisitos e especificações.

Meta de Maturidade 3.4: Controlar e Monitorar o Processo de Teste

TRs para Administradores

- Prove liderança, recursos e fundos para um comitê ou grupo de monitoramento e controle de processo de teste.
- Assegurar que declarações de política de teste sejam modificadas de modo que mecanismos para controlar e monitorar os testes sejam descritos em detalhes.
- Promover a cooperação entre administradores de projeto e teste para planejamento, monitoramento e atividades de controle.
- Promover mudanças culturais necessárias para implementação das atividades de controle e monitoramento.
- Promover o compartilhamento de ferramentas e técnicas entre administradores de projeto e teste para monitoramento e controle.
- Assegurar que atividades de controle e monitoramento são partes de cada plano de teste.
- Assegurar que fundos adequados, treinamento, ferramentas e recursos são fornecidos para suportar as atividades de controle e monitoramento.
- Atribuir responsabilidades para controle e monitoramento.
- Suportar a identificação e seleção de medidas de controle/monitoramento.
- Participar de reuniões de andamento e auditoria, contribuir para sessões de solução de problemas e suportar follow-up de ações corretivas.

- Revisar periodicamente o sistema de controle e monitoramento.
- Promover sucessos resultantes de atividades de controle e monitoramento.

As ATRs seguintes são designadas primeiramente para os administradores de teste.

- Servir no comitê para desenvolver políticas para controle e monitoramento de teste.
- Desenvolver planos de teste com cronograma apropriado e recursos disponíveis para controle e monitoramento.
- Cooperar com os administradores de projeto para preparar planos de contingência para cobrir áreas de teste relativas a risco.
- Selecionar medidas apropriadas para guiar o controle e monitoramento de testes.
- Assegurar que todos os documentos apropriados de teste são preparados tais como diários de teste e relatórios de incidência de teste.
- Coletar e analisar as medidas relativas à teste e apresentar relatórios para a administração e membros da equipe.
- Marcar reuniões de status, conduzir discussões e relatar o progresso presente.
- Iniciar e follow-up de ações corretivas quando os testes desencaminharem.
- Designar responsabilidades aos testadores para ações corretivas de problemas relacionados a teste.
- Acompanhar e relatar as ações corretivas tomadas.
- Suportar a instalação de um sistema de administração de configuração e desempenhar um papel no conselho de controle de mudança.
- Preparar e apresentar o relatório resumo do teste.
- Participar em revisões periódicas do sistema de controle e monitoramento.

ATRs para Testadores

- Atender as sessões de treinamento sobre controle e monitoramento (incluir treinamento sobre administração de configuração).
- Trabalhar com os administradores de teste para planejar o controle e monitoramento de testes.
- Trabalhar com os administradores de teste para selecionar medidas apropriadas de controle e monitoramento.
- Trabalhar com os administradores de projeto/teste para desenvolver planos de contingência.
- Coletar e analisar medidas de teste.
- Participar em reuniões de status de teste.
- Executar follow-up das atividades de ações corretivas.
- Preparar os documentos relativos a testes tal como diário de teste e relatórios de incidência de teste.
- Contribuir para o relatório resumo de teste.
- Assegurar que itens de teste estão sob o controle de um sistema de administração de configuração.
- Servir como membro do conselho de controle de mudança.
- Participar em revisões periódicas do sistema de controle e monitoramento.

ATRs para Usuários/Clientes

- Participar de reuniões de acompanhamento de teste quando apropriado. Se o software está sendo desenvolvido para um cliente específico, a organização de desenvolvimento pode convidar o usuário ou grupo de cliente para participar da reunião de acompanhamento de teste para ver o progresso.
- Contribuir com dados necessários para avaliação pós-teste quando apropriado. Se uma organização está medindo a efetividade do teste com o uso de relatórios de problemas pós-release, os usuários/clientes necessitarão completar relatórios de problemas e assegurar que eles retornem a pessoa certa na organização de desenvolvimento.

Nível 4 - Gestão e Medições

Meta de Maturidade 4.1: Estabelecer um Programa Amplo de Revisão

ATRs para Administradores

- Prover liderança, recursos e fundos para um programa de revisão do comitê.
- Assegurar que uma política de revisão é desenvolvida, documentada, aprovada, e avaliada pelo administrador e equipe técnica.
- Prover treinamento e recursos para o programa de revisão.
- Promover a mudança cultural necessária para implementar o programa de revisão.
- Assegurar que a revisão dos requisitos de itens selecionados do software foram seguidos.
- Alta administração e gerentes de projeto/teste trabalham juntos para assegurar que os planos de projeto/teste provêem tempo e recursos de revisão de projeto entregues.
- Assegurar que medidas de revisão são identificadas, coletadas e aplicadas na revisão do processo de melhorias.
- Assegurar que durante as revisões os defeitos sejam identificados.
- Assegurar que durante as revisões os defeitos são processados identificados e as correções são conduzidas até o fim.
- Assegurar que revisão do dado é encaminhada para o planejamento de projeto/teste.
- Avaliar o programa de revisão periodicamente e apoiar melhorias aonde necessitada.
- Promover o sucesso do programa de revisão.

ATRs para Testadores

- Participar como membro de um comitê para desenvolver políticas e planos de revisão.
- Atender as seções de treinamento de revisão.

- Servir como líder de revisão, instrutores de revisão e participantes de revisão como descrito nas declarações da política de revisão.
- Assegurar que a revisão do trabalho é completada como descrito na política de revisão e nos planos de revisão.
- Trabalhar com Garantia de Qualidade de Software para identificar classes e níveis de severidade de defeitos relatados na revisão.
- Acumular e analisar dados de defeito na revisão.
- Coletar e medir as revisões relatadas.
- Trabalhar com a equipe de Garantia de Qualidade de Software e outras formas de suporte de revisão.
- Analisar os dados de defeitos passados e atuais para o planejamento de teste e projeto de teste.
- Participar periodicamente das avaliações de revisão de programa.

ATRs para Usuários/Clientes

- Atender as seções de treinamento aonde apropriado.
- Atender as seções de revisão como descrito na política de revisão. (Por exemplo, tratamento de requisitos, aceitação do plano de teste e uso de manuais de revisão são vitais para assegurar a qualidade do software e satisfação das necessidades e requisitos do usuário).

Meta de Maturidade 4.2: Estabelecer um Programa de Medições de Teste

ATRs para Administradores

- Prover liderança, recursos e fundos para um programa de medida de teste.
- Assegurar que uma medida de programa/política é desenvolvida, documentada, aprovada e avaliada pela administração e assistente técnico.

- Prover treinamento e recursos de medida de programa.
- Assegurar que são identificadas as eficácias do processo de teste e medida de produto.
- Rever as medidas selecionadas periodicamente, e adicionar, modificar, ou excluir medidas quando julgar apropriado.
- Determinar responsabilidades de definição, coleta, armazenagem (repositório de dados de teste), análise e aplicação de dados de medida. Decidir quando e onde os dados podem ser aplicados, e por quem.
- Assegurar que o plano de teste inclui medidas apropriadas para que o processo de teste possa suportar metas de qualidade de produto da empresa de software.
- Trabalhar junto com outros administradores (administradores do alto nível, administradores de projeto/teste) para assegurar que os planos projeto/teste provê tempo e recursos e concluir o produto e as medidas de processo.
- Assegurar que dados de projeto/processo/teste são encaminhados para o planejamento de projeto/teste e suportam o desenvolvimento de teste e metas de projeto.
- Assegura que o dado medido é utilizado para monitorar, controlar e melhorar o processo de teste (administradores de teste).
- Aplicar o dado de medida para desenvolver um plano de ação apropriado quando medidos (e suporta avaliações) indica que o processo de teste pode ser alterado e melhorado (administradores de teste).
- Trabalhar com clientes para coletar entradas sobre os atributos de qualidade de software que são importantes para eles.
- Usar os dados coletados do produto durante o teste para avaliar e melhorar a qualidade do produto.
- Revisar/avaliar e medir periodicamente o programa e suportar melhorias aonde for necessário.
- Promover resultados de sucesso da medida do programa.
- Promover a mudança cultural necessária para o sucesso do programa medido.

ATRs para Testadores

- Participar no comitê de melhoria em medida de programa planejamento e execução do plano de ação.
- Participar em medidas de seleção de ferramentas e métodos.
- Participar dentro da identificação dos atributos de qualidade e métricas de produtos relatados.
- Participar dentro da identificação de atributos e métricas de teste do processo relatado.
- Atender as medidas de treinamento de classes.
- Desenvolver e manter a base de dados de teste medidos (e repositório de defeito).
- Coletar produto e dados de processo durante todo os níveis de teste.
- Suportar análise de dados e resultados de aplicação.
- Suportar o uso de medidas de avaliação de processo de teste e melhoria apropriadas.
- Prover entradas para administradores e equipe de Garantia de Qualidade de Software para medir dados de cada produto com respeito aos atributos de qualidade.
- Ajudar na realização de metas de qualidade e requisitos de cada projeto.
- Participar de revisões periódicas de medidas de programa.

ATRs par Usuários/Clientes

- Dar consenso nos atributos de qualidade dos produtos.

Meta de Maturidade 4.3: Avaliação da Qualidade de Software

ATRs para Administradores

- Prover liderança, recursos e fundos para um comitê de avaliação da qualidade de software.
- Assegurar que políticas de avaliação da qualidade do software, procedimentos, formulários e padrões são desenvolvidos, distribuídos e aprovados.
- Assegurar que um conjunto completo de atributos e de métricas de qualidade de software é desenvolvido, distribuído e aprovado.
- Promover mudanças culturais necessárias para implementar as atividades de avaliação da qualidade do software.
- Assegurar que cada projeto segue os procedimentos normalizados de avaliação da qualidade de software.
- Assegurar que as metas quantitativas de qualidade são estabelecidas em cada projeto.
- Assegurar que as metas de qualidade são alcançadas em cada projeto.
- Prover fundos, treinamento, ferramentas e outros recursos para ajudar na avaliação da qualidade do software.
- Revisar periodicamente com a equipe técnica a política de avaliação, padrões, métricas e procedimentos da qualidade do software.
- Revisar/avaliar periodicamente o processo de teste para assegurar que ele suporta a avaliação dos atributos de qualidade e a realização de metas de qualidade.
- Assegurar que há mecanismos alocados para entrada do cliente na fixação das metas de qualidade dos projetos.
- Assegurar que a entrada do cliente é solicitada para identificar os atributos chave da qualidade.
- Promover os sucessos resultantes de atividades de avaliação da qualidade de software.

ATRs para Testadores

- Atender a classes de treinamento de avaliação da qualidade de software.

- Participar de desenvolvimento de políticas, procedimentos, formulários e padrões de avaliação da qualidade de software.
- Participar na identificação e aplicação de atributos e métricas de qualidade de software.
- Trabalhar com administradores de projeto, administradores de teste e clientes para identificar atributos da qualidade, métricas e metas de qualidade relevantes para cada projeto.
- Coletar medidas de atributos de qualidade de software durante o teste e outros procedimentos de avaliação da qualidade.
- Participar nos procedimentos de validação de métricas da qualidade do software.
- Usar medidas de qualidade de software e técnicas de análises para avaliar a qualidade dos artefatos de software emergentes.
- Participar na avaliação do processo de teste e procedimentos de avaliação para assegurar que os testes dão forte suporte para avaliação e realização das metas de qualidade.
- Participar na revisão de qualidade de software, e avaliação dos padrões e procedimentos.
- Apoiar na realização das metas de qualidade e requisitos para cada projeto.

ATRs para Usuários/Clientes

- Dar consenso sobre atributos de qualidade dos produtos.
- Prover entrada e aprovação dos requisitos e metas da qualidade que são importantes para eles com respeito ao software em desenvolvimento. Isto aparece nos documentos de requisitos.
- Prover entradas durante testes de aceitação para assegurar que requisitos de qualidade são atendidos.

Nível 5 - Otimização / Prevenção de Defeito e Controle de Qualidade

Meta de Maturidade 5.1: Prevenção de Defeito

ATRs para Administradores

- Prover liderança, recursos e fundos para um comitê focado no desenvolvimento de políticas, procedimentos e programas de prevenção de defeito documentados.
- Assegurar que os documentos de prevenção de defeito são distribuídos e aprovados.
- Designar responsabilidades para atividades de prevenção de defeito.
- Assegurar que treinamento, ferramentas e outros recursos estão disponíveis para as atividades de prevenção de defeito.
- Promover a mudança cultural necessária para o sucesso das atividades de prevenção de defeito.
- Promover visibilidade para as ações de prevenção de defeito.
- Assegurar que as mudanças de processo que resultam das atividades de prevenção de defeito são incorporadas nas políticas e padrões documentados.
- Participar, e servir como líder, nas atividades de prevenção de defeito tais como ação de planejamento e monitoração. Liderar as reuniões no início do projeto. (“Lead project kick-off meetings”)
- Selecionar projetos piloto para implementação de planos de ação para prevenção de defeito.
- Promover discussão e distribuição de listas de defeitos comuns e informação de mudança de processo a todos os membros da equipe do projeto.
- Promover a inclusão das atividades de prevenção de defeito como parte dos planos de projeto e testes.
- Revisar periodicamente o programa de prevenção de defeito.

ATRs para Testadores

- Servir como membro do comitê de política de prevenção de defeito.
- Assistir aulas de treinamento em atividades e métodos de prevenção de defeito.
- Coletar e armazenar dados atualizados dos defeitos ocorridos nas diferentes fases do ciclo de vida.
- Servir como membro de equipes de análise de causa e prevenção de defeito.
- Suprir informações para o desenvolvimento de políticas e procedimentos de prevenção de defeito.
- Participar no planejamento de ações de prevenção de defeito especialmente as aplicadas no processo de teste.
- Servir como membro de equipes piloto que implementam planos de ação para prevenção de defeito.
- Acompanhar e monitorar as mudanças no processo de teste resultante de atividades de prevenção de defeito.
- Assegurar que as mudanças de processo resultante de ações de prevenção bem sucedidas são documentadas e seguidas.
- Suportar mudança cultural necessária para o sucesso dos programas de prevenção de defeito.
- Participar dos programas de revisões de prevenção de defeito.

ATRs para Usuários/Clientes

- Relatar os defeitos e problemas na operação do software de modo a ser incorporado no banco de dado de defeito para acompanhamento e análises de causa.

Meta de Maturidade 5.2: Controle da Qualidade

ATRs para Administradores

- Prover liderança, recursos e fundos para o comitê focado no desenvolvimento de políticas, procedimentos e programas de controle de qualidade documentados.
- Assegurar que os documentos de controle de qualidade são distribuídos e aprovados.
- Designar responsabilidades para atividades de controle de qualidade.
- Assegurar que treinamento, ferramentas, laboratórios e outros recursos estão disponíveis para as atividades de controle de qualidade.
- Promover a mudança cultural necessária para sucesso das atividades de controle de qualidade.
- Promover visibilidade para as ações de controle de qualidade.
- Assegurar que metas qualitativas e quantitativas sejam fixadas para todos os projetos e são incluídas em documentos requeridos, planos de teste e projeto.
- Assegurar que planos de projeto e teste aloquem tempo e recursos suficientes para as atividades de controle de qualidade de software.
- Assegure que há um mecanismo para entrada regular de usuário/cliente na modelagem.
- Planejar testes de confiabilidade e usabilidade.
- Promover o desenvolvimento de padrões de usabilidade.
- Designar responsabilidades para testes estatísticos e testes de usabilidade. (Os grupos de teste e de garantia de qualidade de software devem receber estas responsabilidades após o necessário treinamento. Como alternativa, o administrador pode decidir apoiar o estabelecimento de grupos especializados em engenharia de confiabilidade e usabilidade. Contratar tais grupos de especialistas pode ser pela natureza dos produtos de software em desenvolvimento e devido às necessidades da operação do cliente.)
- Assegurar que as decisões de parar os testes são baseadas em critérios quantitativos, e inclui estes nos planos de teste.
- Monitorar testes e outras atividades relativas à qualidade para assegurar que as metas de qualidade de software para cada projeto sejam atendidas, e que as necessidades dos clientes são satisfeitas.

- Revisar periodicamente o programa de controle de qualidade.

ATRs para Testadores

Se uma organização decide contratar engenheiros especialistas em confiabilidade e usabilidade, então testadores podem ter um papel de apoio e liderança nas ATRs seguintes.

- Participar como membro do comitê de controle de qualidade.
- Assistir aulas de controle de qualidade; inclusive treinamento em teste estatístico, desenvolvimento de perfis de uso, e testes de usabilidade.
- Desenvolver e manter perfis operacionais com entradas e feedback de grupos de usuários.
- Desenvolver níveis hierárquicos de severidade para faltas e falhas.
- Apoiar planos de teste estatístico.
- Realizar teste estatístico e analisar resultados.
- Compreender e aplicar modelos de confiabilidade.
- Coletar e armazenar medidas relativas à qualidade, por exemplo, dos testes de confiabilidade e usabilidade. (Estes dados podem ser usados para fixar padrões para futuros projetos).
- Apoiar o desenvolvimento de critério quantitativo para parar testes e coletar medidas para suportar sua aplicação.
- Projetar e executar testes para medida de atributos tais como exatidão, portabilidade, confiabilidade, usabilidade, confiança, disponibilidade e validade.
- Apoiar planejamento de testes de usabilidade.
- Identificar grupos de usuários para teste de usabilidade.
- Projetar testes de usabilidade.
- Apoiar implantação de testes laboratoriais de usabilidade.
- Monitorar, registrar, analisar e relatar os resultados de testes de usabilidade.

- Solicitar feedback do usuário por meio de teste de usabilidade e assegurar que o trabalho acompanhado é completado.
- Ajudar no desenvolvimento de padrões de usabilidade de produtos de software.

ATRs para Usuários/Clientes

- Participar no desenvolvimento de requisitos de qualidade, qualitativos e quantitativos para os projetos que estão envolvidos.
- Participar no desenvolvimento de um perfil operacional.
- Participar no teste de usabilidade pelo uso do software para realizar tarefas típicas e dando feedback à equipe de teste. É importante o usuário dar sua opinião, apontando os pontos fracos e fortes do sistema de software em desenvolvimento.

Meta de Maturidade 5.3: Otimização do Processo de Teste

ATRs para Administradores

- Assegurar que um grupo ou força tarefa responsável pela melhoria do processo de teste é estabelecido, encarregado, apoiado, financiado e treinado. Assuma uma posição de liderança no grupo.
- Assegurar que políticas, procedimentos e programas de controle de processo de teste são desenvolvidos, documentados, distribuídos e aprovados.
- Assegurar que avaliações, procedimentos e programas de melhorias de processo de teste são desenvolvidos, documentados, distribuídos e aprovados.
- Assegurar que políticas, procedimentos e programas de transferência de tecnologia são desenvolvidos, documentados, distribuídos e aprovados.
- Assegurar que políticas, procedimentos e programas de reuso de processo de teste são desenvolvidos, documentados, distribuídos e aprovados.
- Designar responsabilidades pelo controle do processo de teste.

- Designar responsabilidades pela transferência de tecnologia.
- Designar responsabilidades pelo reuso de processo de teste.
- Designar responsabilidades pela avaliação do processo de teste.
- Assegurar que treinamento, ferramentas, laboratórios e outros recursos estão disponíveis para controle, reuso, avaliação e melhoria do processo de teste.
- Promover a mudança cultural necessária para o sucesso do controle, do reuso, da avaliação e das atividades de melhoria do processo de teste.
- Promover a mudança cultural necessária para o sucesso das atividades de transferência de tecnologia.
- Promover a visibilidade para ações bem sucedidas de reuso, controle e melhoria de processo de teste.
- Selecionar e prover suporte para projetos piloto que estão envolvidos em implementação de controle de processo, avaliação e melhoria de processo, e teste de tecnologias.
- Estar a par das ferramentas e tecnologias atuais para suporte e otimização do processo de teste.
- Suportar avaliações periódicas do processo de teste com o TMM.
- Supervisionar as atividades de melhoria de processo de teste resultante das avaliações TMM.
- Suportar o desenvolvimento e a manutenção de ambas uma Biblioteca de Ativos de Processos e uma Biblioteca de Ativos de Processo para Reuso.
- Identificar processos reusáveis para a inclusão na biblioteca de ativos de processos.
- Ajustar processos recuperados da Biblioteca de Processos e aplicar para novos projetos (administradores de teste).
- Desenvolver cartas de controle, identificar variações de processo e suas causas, e preparar relatórios (administradores de teste).
- Revisar periodicamente os programas de reuso, controle, avaliação e melhoria do processo de teste.

- Revisar periodicamente o programa de transferência de tecnologia.

ATRs para Testadores

- Servir como membro dos grupos e forças tarefas para a melhoria do processo de teste.
- Ajudar no desenvolvimento de políticas, planos, padrões, e outros documentos relacionados ao controle, reuso, avaliação e melhoria do processo de teste, e à transferência de tecnologia.
- Participar de aulas de treinamento em controle, reuso, avaliação e melhoria do processo de teste.
- Participar de aulas de treinamento em transferência de tecnologia.
- Estar a par das ferramentas e tecnologias atuais para suporte e otimização do processo de teste, e identificar as mais promissoras aos administradores e equipes de melhorias.
- Avaliar novas ferramentas e tecnologias, e prover feedback para administração.
- Participar como membro das equipes de avaliação do processo de teste.
- Servir como membro de equipes de planejamento de ações.
- Participar em projetos piloto para mudança de processo e transferência de tecnologia.
- Coletar dados de medidas de projetos em estudo nos grupos e forças tarefas.
- Ajudar no desenvolvimento de cartas de controle, identificação de variações e causas, e preparo de relatórios.
- Ajudar a identificar ativos de processos, apoiar e manter a Biblioteca de Ativos de Processos.
- Suportar integração de mudança de processos e novas tecnologias aprovadas.
- Participar nas revisões dos programas de controle, reuso, avaliação, melhoria de processos de teste e dos programas de transferência de tecnologia.

ATRs para Usuários/Clientes

- Usuários e clientes não têm nenhum papel significante no suporte desta meta de maturidade.

Apêndice

B

Questionário de Avaliação do TMM

Todo processo de melhoria, antes de ser iniciado, passa por um processo de conhecimento do estado atual da organização. Somente após conhecer o seu estado atual e sabendo-se o que se deseja atingir é que é possível começar o processo de melhoria. Nesse sentido, visando fornecer um mecanismo de auto-avaliação, bem como de avaliação para certificação, é de fundamental importância que o modelo ofereça um questionário. A seguir é apresentado o questionário de avaliação desenvolvido por Ilene Burnstein Burnstein (2003) para ser utilizado pelo TMM. Parte desse questionário foi adaptada para servir como questionário para o MPT-PE. O objetivo de inseri-lo neste apêndice é o de permitir ao leitor comparar o que foi eliminado para o desenvolvimento do questionário para o MPT-PE (Capítulo 4).

B.1 TMM-Questionário

Este apêndice têm duas partes. A Parte 1 contêm um conjunto completo de Atividades, Tarefas e Responsabilidades (ATRs) recomendadas para cada nível do TMM.A Parte 2 contêm o questionário completo do TMM.

Parte 2 - Questionário do TMM (Testing Maturity Model)

Estas partes do Apêndice III contêm o sumário documentado do questionário TMM versão 2.0. O questionário é composto de sete seções. Os conteúdos de cada seção estão descritos abaixo.

Seção 1. Instruções para o respondedor.

Seção 2. Identificação e qualificação do respondedor.

Seção 3. Qualificação da organização.

Seção 4. Questões do TMM. Há para cada nível do TMM:

i. O conjunto de metas de maturidade;

ii. O conjunto de submetas de maturidade associados com cada meta de maturidade;

iii. O conjunto de questões associados com cada meta de maturidade.

Seção 5. Questões de ferramentas de teste.

Seção 6. Questões de tendências.

Seção 7. Comentários do respondedor.

Cada questão têm quatro possibilidades de resposta: SIM, NÃO, NÃO se APLICA ou NÃO CONHECE. O respondedor pode também fornecer comentários relacionados a cada questão no espaço fornecido.

Seção 1. Instrução para o Respondedor

Por favor, leia e responda as seguintes questões cuidadosamente utilizando o conhecimento e experiência adquirida dos projetos sendo realizados na sua organização. Desejando comentar qualquer uma das questões ou qualificar a sua resposta, por favor, utilize o espaço para comentário fornecido. As respostas fornecidas são confidenciais.

Para as questões de maturidade, quatro possíveis escolhas são oferecidas como seguem:

1. **Sim** Quando uma prática é bem estabelecida e consistentemente executada.

2. **Não** Quando a prática não é bem estabelecida ou não é consistentemente executada.

3. **Não é aplicado** Quando você tem o conhecimento requerido do projeto ou da organização, mas você acredita que a questão não se aplica para o projeto ou organização.

4. **Não conhece** Quando você está indeciso como responder esta questão, ou você não tem experiência ou informações apropriadas.

Somente uma escolha para cada questão poderá ser selecionada, e todas as questões deverão ser consideradas. Por favor continue no questionário e obrigado pela sua ajuda.

Seção 2. Identificação e Qualificação do Respondedor

Informações sobre conhecimento, experiência, habilidades técnicas e responsabilidades atuais do respondedor sobre engenharia de software.

1. Identificação do Respondedor

   Nome

   Posição

   Nome do projeto

   Telefone

   E-mail

   Data

2. Experiência do Respondedor

   Qual a melhor descrição de sua posição atual?

   Gerente

   Administrador Sênior ou superior

   Gerente de projeto

   Gerente de teste

   Líder de grupo Garantia de Qualidade de Software

   Líder de grupo de Processo de Engenharia de Software

   Líder de subgrupo relacionado a teste (*por favor, especifique o nome do subgrupo*).

   Outros (*por favor, especificar*)

   Equipe Técnica

   Engenheiro de Software

Engenheiro de Teste

Programador (desenvolvedor)

Analista

Membro de grupo Garantia de Qualidade de Software

Membro de grupo de Processo de Engenharia de Software

Membro de subgrupo relacionado a teste *(por favor, especificar)*

Outros (*por favor, especificar*)

3. Responsabilidades e Obrigações Atuais

   Quais atividades relacionadas a teste você esta efetivamente envolvido (você pode marcar mais de uma)?

   Política de teste e desenvolvimento de metas

   Planejamento de teste

   Projeto de caso e procedimento de teste

   Execução de teste

   Coleta e análise de medidas relacionadas a teste

   Coleta de dados de defeitos

   Manutenção de banco de dados de defeitos

   Desenvolvimento de padrões

   Revisões e auditorias

   Acompanhamento do status

   Treinamento

   Definições de métricas

   Recrutamento e contratação

   Comunicação com Usuário/Cliente

   Controle de processo

   Prevenção de defeito

   Transferência de tecnologia

   Avaliação de processo

Melhoria de processo

Engenharia e modelagem de confiabilidade

Teste de usabilidade

Avaliação de ferramentas

Reuso de processo

4. Você recebeu algum treinamento de TMM?

   Sim *(por favor, descreva)*

   Não

5. Qual é a extensão de sua experiência na industria de software?

Na organização atual	Número de anos
Na industria de software	Número de anos
Experiência em teste	Número de anos

6. Você tem participado de outros tipos de avaliações de software?

   Sim (*por favor, descreva*)

   Não

Seção 3. Qualificação da Organização

1. Descreva o melhor que puder o tipo de sua organização.

   Desenvolve software de segurança

   Desenvolve software de aplicação

   Desenvolve software de telecomunicação

   Garantia de Qualidade de Software/teste de software/certificação

   Outros (*por favor, descreva*)

2. A maioria (acima de 50%) dos software desenvolvidos é para uso interno ou externo?

   Interno Externo Não aplicado

3. Quantas pessoas são empregadas na organização em avaliação?

   Número total de empregados

   Número de empregados engajados em desenvolvimento e/ou manutenção de software

   Número de empregados engajados em teste de software

4. Por favor, descreva o percentual da equipe empenhada em testes como segue:

   Em tempo integral

   Em tempo parcial

   Consultores

5. As pessoas envolvidas em melhoria de processo de sua organização são bem respeitadas em relação as suas habilidades gerenciais e técnicas? (por favor, marque uma: 1 quer dizer não respeitada e 5 altamente respeitada)

   1 2 3 4 5

6. As responsabilidades para melhoria do processo de teste são claramente definidas e apoiadas? (por favor, marque uma: 1 quer dizer não definida ou não apoiada, e 5 quer dizer bem definida e apoiada)

   1 2 3 4 5

7. A organização sob avaliação tem um grupo de processo de engenharia de software ou uma unidade similar?

   Sim Não

8. Como o grupo de teste é organizado? (Por favor, selecione um)

   Os desenvolvedores fazem o teste.

   Grupo de teste dentro do desenvolvimento, reportando ao gerente de projeto.

   Grupo de teste separado, reportando ao gerente de teste.

   Parte do grupo de Garantia de Qualidade de Software.

9. Como você poderia caracterizar a natureza de seu processo de teste?

   "Ad hoc"

   Informal

   Levemente estruturado

   Altamente estruturado

10. Com que freqüência os gerentes de projeto têm a necessidade de modificar os requisitos de cliente?

    Nunca Raramente Freqüentemente Muito freqüentemente

Seção 4. Questões sobre Metas de Maturidade

Por favor, responda cada uma das seguintes informações com um SIM, NÃO, NÃO SE APLICA ou NÃO CONHECE. Comentários podem ser feitos após cada questão.

Nível 2 - Fase de Definição

Meta de Maturidade 2.1: Desenvolver Metas e Política de Teste e Depuração

O propósito desta meta é diferenciar claramente os processos de teste e de depuração. As metas, tarefas, atividades e ferramentas de cada um podem ser identificadas. Responsabilidades de cada um devem ser designadas. Políticas devem ser estabelecidas pela administração para acomodar e estabelecer ambos os processos.

Submetas de maturidade que suportam esta meta são:

2.1.1. Um comitê para toda a organização ou grupo de teste e depuração é formado e provido com recursos e suporte. O comitê desenvolve documentos, distribui e suporta procedimentos, metas e políticas para teste e depuração. As metas, políticas e procedimentos, uma vez aprovados, são postos em prática e revisados periodicamente.

2.1.2. Políticas e metas de teste e depuração são refletidas nos planos de projeto e teste.

2.1.3. São estabelecidos um esquema de classificação e um repositório de defeitos básicos.

2.1.4. São identificadas e coletadas medidas simples de teste e depuração.

Questões

1. Foi estabelecido um comitê(s) sobre teste e depuração?
2. Políticas, metas, atividades e ferramentas para o processo de teste têm sido identificadas, documentadas e aprovadas?
3. Políticas, metas, atividades e ferramentas para o processo de depuração têm sido identificadas, documentadas e aprovadas?
4. O processo de teste é definido?
5. O processo de depuração é definido?
6. Documentos sobre políticas de teste têm sido distribuídos para os gerentes de projetos e desenvolvedores (testadores)?
7. Documentos sobre política de depuração têm sido distribuídos para os gerentes de projetos e desenvolvedores (testadores)?
8. Os desenvolvedores de software (testadores) seguem uma política escrita da organização para teste quando planejam o teste?
9. Os desenvolvedores seguem uma política escrita da organização para depuração?
10. Há medidas básicas usadas para determinar a realização das metas de teste?
11. Há medidas básicas utilizadas para determinar a realização das metas de depuração?
12. Políticas e metas de teste têm sido desenvolvidas com entradas de grupos usuário/cliente com respeito a suas necessidades?
13. Políticas e metas de depuração têm sido desenvolvidas com entrada e feedback de grupos de usuários /clientes com respeito a suas necessidades?
14. Foi desenvolvido um esquema de classificação dos defeitos básicos?
15. Foi estabelecido um repositório de defeitos?

16. Desenvolvedores/testadores depositam os defeitos no repositório consistentemente?

17. Políticas e metas de teste/depuração são revistas periodicamente?

Meta de Maturidade 2.2: Iniciar um Processo de Planejamento de Teste

O propósito desta meta é estabelecer um processo de planejamento de teste para toda a organização. Planejar teste envolve estabelecer os objetivos do teste, análise de risco, formulação de estratégias e desenvolvimento de especificações de projeto de teste e de casos de teste. Um plano de teste deve conter a alocação de recursos, custos e responsabilidades para teste de unidade, integração, sistema e aceitação.

Submetas de maturidade o que suportam esta meta são:

2.2.1 Um comitê para toda a organização, ou um grupo de planejamento de teste é formado e provido com recursos e suporte. O comitê desenvolve, documenta, distribui, suporta procedimentos, metas e políticas para o planejamento de teste. As metas, políticas e procedimentos uma vez aprovados, são colocados em prática e periodicamente revisados.

2.2.2 Modelos de plano para todos os níveis de teste são desenvolvidos, registrados e distribuídos para gerentes de projeto e desenvolvedores/testadores para uso em projetos da organização. Outros documentos relativos à teste são identificados e prescritos de acordo com a política da organização.

2.2.3. Treinamento técnico para o uso de modelos de plano de teste e desenvolvimento de planos de testes são disponíveis.

2.2.4. Um procedimento é colocado em prática para incluir os requisitos gerados pelos usuários como entrada para o plano de teste.

2.2.5. Ferramentas básicas de planejamento e medidas de teste são avaliadas e aplicadas.

Questões

1. Foi estabelecido um comitê ou grupo de planejamento de teste para toda a organização?

2. Há uma política organizacional e existem procedimentos para planejamento de teste?

3. Foram distribuídos e aprovados políticas e procedimentos?
4. Há suporte adequado e recursos para planejamento de teste para todos projetos?
5. Metas/objetivos de teste são utilizados como base para o planejamento de teste?
6. Têm sido desenvolvidos e distribuídos modelos de plano de teste para os gerentes de projeto?
7. Há ferramentas de planejamento apropriadas disponíveis para planejamento de teste?
8. Os gerentes de projeto têm sido treinados no uso de modelos e ferramentas de planejamento?
9. Os desenvolvedores (testadores) têm sido treinados no uso de modelos e ferramentas de planejamento?
10. Os desenvolvedores (testadores) são adequadamente treinados para desenvolver especificações, projeto de teste e casos de teste para plano de teste?
11. São considerados os riscos relativos a teste quando se desenvolvem os planos de teste?
12. Estão disponíveis estimativas (tempo, orçamento, ferramentas) de projetos passados para uso no planejamento de teste.
13. É feito o planejamento de teste no nível de unidade?
14. É feito o planejamento de teste no nível de integração?
15. É feito o planejamento de teste no nível de sistema?
16. É feito o planejamento de teste de aceitação?
17. Há um procedimento estabelecido para solicitar entradas de usuários/clientes para planejamento de teste quando apropriado (por ex: no planejamento de teste de aceitação)?
18. Os desenvolvedores (testadores) têm oportunidade de dar entradas para o plano de teste em todos os níveis?
19. O processo de planejamento de teste é revisado periodicamente e/ou direcionado por eventos?

20. Outros itens relativos à teste tais como relatórios de andamento de teste, diários de teste, relatórios de incidências e sumários de teste são definidos em documentos da organização?

21. Outros itens relativos à teste tais como relatório de andamento, diários de teste, relatórios de incidências e sumários de teste são completados a cada projeto?

22. Há suporte administrativo para interações entre gerente de projeto (teste), desenvolvedores (testadores), projetistas e analistas para suportar planejamento de testes?

23. As medidas de teste são especificadas no plano de teste para todos os níveis?

24. Desenvolvedores (testadores) coletam e armazenam medidas básicas relativas a teste?

25. As medidas básicas de teste são utilizadas para assegurar que as metas de teste foram atingidas?

Meta de Maturidade 2.3: Institucionalizar Técnicas e Métodos Básicos de Teste

O propósito desta meta de maturidade é melhorar a capabilidade do processo de teste pela aplicação métodos e técnicas básicas de teste. Como e quando estas técnicas e métodos são aplicados e qualquer suporte de ferramenta básica para eles, devem ser claramente especificados em planos e política de teste. Várias técnicas básicas e métodos que são freqüentemente utilizados no processo de teste são estratégias de projeto caixa preta e caixa branca, uso de uma matriz de validação de requisitos e teste como na execução em subfases tais como unidade, integração, sistema e teste de aceitação. Algumas ferramentas de teste que suporta uso destas técnicas e métodos são analisadores estático e dinâmico, analisadores de cobertura, geradores de dados de teste e ferramentas de verificação de erros.

Submetas de maturidade que suportam esta meta são:

2.3.1 Um comitê para toda a organização ou grupo sobre tecnologia de teste é formado e provido com fundos e suporte. O comitê estuda, avalia e recomenda um conjunto de técnicas básicas de teste e métodos e um conjunto de formulários simples e ferramentas para suportá-los. Desenvolve política relevante, procedimentos e documentos, e estes são distribuídos. Quando aprovados, eles são colocados em prática e revisados periodicamente.

2.3.2 Treinamento técnico e ferramentas básicas estão disponíveis para suportar uso de métodos e técnicas de teste.

2.3.3 Teste de software é planejado e implementado para os níveis de unidade, integração, sistema e aceitação de acordo com a política.

2.3.4 Estratégias básicas de teste (caixa branca/preta), técnicas e métodos são utilizados por toda a organização para projeto de casos de teste. Interação entre desenvolvedores (testadores) e outras equipes técnicas (por exemplo, projetistas e analistas de requisitos) é promovida para identificar questões de testabilidade, encorajar desenvolvimento de representações de software úteis para métodos de teste de caixa branca/preta, e suportar teste de múltiplos níveis.

Questões

1. Foi formado um comitê ou grupo para avaliar e recomendar um conjunto de técnicas básicas de teste, métodos e ferramentas?
2. As recomendações do grupo foram documentadas, distribuídas e aprovadas?
3. Foram incluídas ferramentas e técnicas básicas na política de teste?
4. Formulários e modelos adequados foram projetados para suportar técnicas básicas de teste?
5. São providos pela alta gerência recursos adequados para suportar o uso de técnicas e métodos básicos de teste, assim como ferramentas básicas de teste?
6. Desenvolvedores (testadores) foram treinados para aplicar as ferramentas básicas, formulários e métodos.
7. Técnicas e métodos básicos de teste são aplicados por toda a organização?
8. Ferramentas básicas de teste são aplicadas por toda a organização?
9. Técnicas básicas de teste e métodos são revisados periodicamente?
10. Declarações da política de teste incluem o requisito de teste de múltiplos níveis?
11. O teste é planejado e implementado para múltiplos níveis (unidade, integração, sistema, etc)?

12. As técnicas e ferramentas básicas de teste são descritas em políticas aplicadas a todos os níveis de teste?

13. As técnicas e ferramentas de teste a serem aplicadas são descritas nos planos de teste de múltiplos níveis?

14. A administração superior suporta interação entre analistas, projetistas e desenvolvedores (testadores) para assegurar que as questões de teste são resolvidas por esses grupos?

Nível 3 - Integração

Meta de Maturidade 3.1: Estabelecer uma Organização de Teste

O propósito desta meta de maturidade é identificar e organizar um grupo de pessoas de grande habilidade que é responsável pelo teste. O grupo de teste deverá ser responsável pelo plano, execução e registros de teste, formulação de padrões de teste, métricas de teste, banco de dados de teste, reuso de teste, andamento de teste e avaliação. O grupo deverá também ser responsável pela manutenção do repositório de defeitos.

Submetas de maturidade que suportam esta meta são:

3.1.1 Um comitê da organização é formado para mapear a estrutura organizacional do grupo ou organização de teste. São providos liderança, suporte e orçamento para o grupo de teste. Regras, responsabilidades e passos de carreira são definidos para o grupo de teste.

3.1.2 Um grupo de teste para toda a organização é formado através de força tarefa e a sua funcionalidade e posição hierárquica são definidas. Membros bem motivados e treinados são designados para o grupo de teste. São estabelecidos canais de comunicação bem definidos entre o grupo de teste e usuário/cliente, desenvolverdores e Garantia de Qualidade de Software. O grupo de teste é revisado periodicamente pela gerência.

3.1.3 Treinamento é disponível para assegurar que o grupo de teste tem a perícia técnica para aplicar ferramentas e técnicas de teste apropriadas, avaliar novas ferramentas e técnicas e planejar o esforço de teste.

Questões

1. O teste é reconhecido pela organização como uma atividade profissional?
2. Foi formado uma força tarefa ou um comitê para determinar a estrutura para a organização ou grupo de teste?
3. Há uma organização de teste de software responsável pelo teste de cada projeto?
4. Há passos de carreira a fim de que membros do grupo de teste podem seguir?
5. Há recursos adequado designados para a organização de teste de software?
6. Os membros da organização de teste estão treinados nos métodos de teste, planejamento de teste, teoria, ferramentas e técnicas?
7. Os testadores são respeitados adequadamente pelos outros engenheiros de software?
8. As responsabilidades e regras do grupo estão definidas?
9. As atividades do grupo são documentadas e relatadas para a administração superior?
10. A organização de teste de software esta coordenada com o grupo de Garantia de Qualidade de Software para realçar a efetividade de teste e melhorar a qualidade do software com respeito aos requisitos do cliente?
11. Quando surgem problemas há canais de comunicação entre o grupo de teste e os desenvolvedores para planejamento de teste, projeto de teste e correção do código?
12. Considerações de clientes são solicitadas como entrada para política organizacional de teste?
13. Há mecanismos formais de interação entre testadores e usuário/cliente?
14. O grupo é periodicamente revisado pelo gerente?

Meta de Maturidade 3.2: Estabelecer um Programa de Treinamento Técnico

O propósito desta meta de maturidade pelo ponto de vista do grupo de teste é assegurar que uma equipe habilitada está disponível para executar tarefas de teste. O programa formal de treinamento é baseado na política de treinamento. Também chamada de metas específicas de treinamento e desenvolve planos e materiais específicos de treinamento.

Aulas de treinamento são disponíveis para todos os membros da equipe. O impacto do programa de treinamento é capacitar os testadores em técnicas atualizadas de teste, métodos e ferramentas. Também prepara testadores para planejar testes, para tarefas envolvendo integração de teste no ciclo de vida de software, para processo de revisão, para identificação e priorização de riscos relativos à teste. Nos altos níveis do TMM prepara-se testadores para controle de processo de teste, para programa de medidas, testes estatísticos, planejamento de ações de processo de teste, confiabilidade, modelagem e outras atividades de teste de alto nível.

Submetas de maturidade que suportam esta meta são:

3.2.1. Um comitê ou grupo de treinamento para toda a organização é estabelecido com fundos e recursos. O comitê de treinamento técnico desenvolve, ganha aprovação e distribui os documentos de política organizacional de treinamento. A política e programas de treinamento são revistos periodicamente.

3.2.2. Um grupo interno de treinamento é estabelecido e provido de liderança, ferramentas e facilidades conforme a política. O grupo desenvolve um programa de treinamento. Metas e planos de treinamento são desenvolvidos pelo grupo com a participação de gerentes de projeto/teste. Material de treinamento é desenvolvido pelo grupo e membros do grupo servem de instrutores de treinamento.

Questões

1. Foi estabelecido um comitê ou grupo de treinamento com fundos e suporte?
2. As políticas e metas relativas a treinamento estão documentadas, aprovadas e distribuídas?
3. A organização segue a política organizacional para atender suas necessidades de treinamento?
4. Foi estabelecido um programa de treinamento para melhorar habilidades da equipe técnica?
5. Foram desenvolvidos planos de treinamento com a participação de gerentes de projeto/teste?

6. São providos recursos suficientes para a implementação do programa de treinamento técnico?

7. A gerência recomenda regularmente cursos de treinamento para a equipe técnica?

8. O grupo de treinamento desenvolve material de treinamento e serve como instrutor para cursos de treinamento?

9. Os participantes do grupo recebem o treinamento necessário para desenvolver as habilidades e adquirir o conhecimento requerido para executar suas tarefas de treinamento?

10. São utilizadas medidas para determinar a qualidade e a efetividade do programa de treinamento?

11. Os programas de treinamento são revistos periodicamente?

Meta de Maturidade 3.3: Integrar o Teste no Ciclo de Vida do Software

O propósito desta meta de maturidade é promover o desempenho de atividades de teste em paralelo com outras fases do ciclo de vida, iniciando cedo ciclo de vida de software. Uma organização madura não posterga atividades de teste até o código estar completo. Exemplos de boas práticas promovidas por esta meta são: planejamentos de teste de sistema e mestre são iniciados cedo no ciclo de vida na fase dos requisitos e planejamento de teste de integração e de unidades são iniciados na fase de projeto. Uma variação do Modelo-V é utilizada por gerentes, testadores e desenvolvedores para guiar atividades de integração. Recursos que suportam a integração dos esforços de teste são, por exemplo, equipe qualificada, modelos de ciclo de vida, padrões e políticas documentadas, planejamento e cronograma apropriados.

Submetas de maturidade que suportam esta meta são:

3.3.1 Um comitê ou grupo de integração das atividades de teste é estabelecido para toda a empresa, com fundos e suporte. O comitê desenvolve, documenta, distribui, procedimentos suportes, metas e políticas para integração de testes. As metas, políticas e procedimentos, uma vez aprovados, são postos em prática e revisados periodicamente.

3.3.2 Atividades de teste são integradas dentro do ciclo de vida do software usando o modelo padrão de ciclo de vida estabelecido segundo política organizacional escrita.

Políticas de planejamento de projeto e de teste são ajustadas para promover integração de testes. Padrões e diretrizes de qualidade são desenvolvidos para teste dos produtos produzido em cada fase do ciclo de vida.

3.3.3 Recursos e treinamento são providos para suportar a integração das atividades de teste dentro do ciclo de vida de software.

Questões

1. Um grupo ou comitê foi estabelecido para suportar integração de atividades de teste?
2. Foi adotado um modelo de ciclo de vida de software que suporta integração de atividades de teste?
3. Foram identificadas atividades de teste e requisitos de teste associados com cada fase do ciclo de vida?
4. Foram desenvolvidos, aprovados e distribuídos uma política de integração e um conjunto de procedimentos documentados baseados no modelo adotado?
5. É provido treinamento adequado para o esforço de integração de testes?
6. São providos recursos suficientes para o esforço de integração de testes?
7. As atividades de integração de testes são revisadas periodicamente no ciclo de vida de software?
8. Foram modificados os procedimentos de planejamento de projeto e teste para compatibilizar e acomodar atividades de integração?
9. Cada projeto segue política organizacional escrita para a integração dos esforços de testes?
10. O grupo de teste e o grupo de Garantia de Qualidade de Software têm desenvolvido um conjunto de padrões documentados para todos os produtos produzidos em cada fase do ciclo de vida?
11. Há uma política para tratar não concordâncias com padrões?

Meta de Maturidade 3.4: Controlar e Monitorar o Processo de Teste

O objetivo desta meta de maturidade é promover o desenvolvimento de um sistema de monitoramento e controle para o processo de teste de modo que desvios de plano de teste possam ser detectados o mais cedo possível, e o gerenciamento possa tomar ações efetivas para corrigir os desvios. Tendo esta capacidade de suporte, um processo de teste tem mais chance de estar em dia e dentro do orçamento. Controle e monitoração de teste também dá visibilidade para o processo de teste, suporta teste como uma atividade profissional e pode conduzir para produtos de software de alta qualidade.

Submetas de maturidade que suportam esta meta são:

3.4.1 Um comitê ou grupo de controle e monitoração de teste para toda a empresa é formado e provido com recursos e suporte. O comitê desenvolve, documenta, distribui, suporta procedimentos, metas, políticas e medidas para controle e monitoramento de teste. As metas, políticas, procedimentos e medidas, uma vez aprovados, são postos em prática e revisados periodicamente.

3.4.2 Medidas relativas à teste para controle e monitoração são coletadas para cada projeto. Relatórios de status são produzidos em base regulares para cada projeto de acordo com a política. Planos de contingência são desenvolvidos, registrados e documentados com planos de teste para cada projeto para serem usados quando o acompanhamento indicar desvios significativos em relação ao planejado.

3.4.3 Treinamento, ferramentas e outros recursos são postos a disposição para suportar controle e monitoramento de teste.

Questões

1. Foi estabelecido um comitê ou grupo para suportar o monitoramento e controle de teste?
2. Há uma política organizacional para monitoramento e controle de teste?
3. São disponíveis ferramentas e treinamento para suportar controle e monitoramento de teste?
4. Foram definidas e distribuídas medidas para acompanhamento do progresso de teste?
5. Gerentes de projeto e gerentes de teste trabalham em conjunto nos planos de monitoramento e controle?

6. Cada projeto segue política organizacional escrita para controle e monitoramento de processo de teste?

7. São desenvolvidos planos de contingência para cada projeto para suportar controle de um teste?

8. A organização coleta e armazena métricas de acompanhamento e controle para cada projeto?

9. As informações de status de teste baseadas em decisões em reuniões regulares são reportadas periodicamente aos gerentes de teste, de projeto e administração superior?

10. A organizaçãode teste, apoiada pelo gerente de projeto, desenvolve planos de contingência para riscos de teste?

11. Itens de teste estão sob controle de um sistema de administração de configuração?

12. As atividades de controle e monitoramento de processo de teste são revisadas periodicamente?

Nível 4 - Gestão e Medições

Meta de Maturidade 4.1: Estabelecer um Programa Amplo de Revisão

Revisões são um tipo de técnica de teste que podem ser utilizadas para remover defeitos de artefatos de software. Atingir este nível de maturidade resulta em um programa de revisão que ajuda a organização identificar, catalogar e remover defeitos de artefatos de software, efetivamente e cedo no ciclo de vida de software. Também suporta revisões de avaliação da qualidade de itens relativos a software. Exemplos de itens que podem ser revisados são documentos de requisitos, documentos de projeto, planos de teste e especificações de caso de teste.

Submetas de maturidade que suportam esta meta são:

4.1.1. É formado um comitê ou grupo para toda a empresa focando no desenvolvimento de um programa de revisão com fundos e suporte adequados. O comitê desenvolve, documenta, distribui e suportam procedimentos, metas, políticas e medidas para revisão

de produtos resultantes de todas as etapas do ciclo de vida de software. As metas, políticas, procedimentos e medidas, uma vez aprovados, são colocados em prática e revisados periodicamente.

4.1.2. Pessoal é treinado de modo a entender, e seguir, políticas, práticas e procedimentos próprios de revisão. Também são treinados na coleta, armazenamento e aplicação de medidas de revisão.

4.1.3. Artefatos de software são revisados para cada projeto como descrito na política de revisão, e refletidos no plano de projeto. Medidas de revisão são coletadas e aplicadas para melhoria de qualidade de produto e processo.

Questões

1. Foi estabelecido um comitê ou um grupo de revisão para toda a empresa, com fundos e recursos?
2. Foi desenvolvida, aprovada e distribuída uma política de revisão?
3. Preocupações com cliente são refletidas na política de revisão?
4. Foram definidos, documentados e aprovados, procedimentos, sistemas de medidas e relatórios de revisão?
5. São providos recursos adequados (ex., fundos, materiais e ferramentas de revisão) para implementar o programa de revisão?
6. São treinados revisores e líderes de revisão?
7. Cada projeto segue política escrita de revisão para realizar revisões?
8. Os cronogramas de projeto refletem necessidades de revisão?
9. As revisões são planejadas e os resultados relatados e documentados?
10. São revisados os produtos do trabalho de software e teste realizados nas diferentes etapas do ciclo de vida?
11. Os defeitos identificados são encontrados durante as revisões e armazenados em um repositório de defeitos?
12. As ações corretivas de defeitos são encontradas nas revisões acompanhadas até a solução?

13. São coletadas e analisadas medidas relativas a revisão?

14. São coletadas medidas relativas aos produtos do trabalho de software durante as revisões?

15. O programa de revisão é avaliado periodicamente?

Meta de Maturidade 4.2: Estabelecer um Programa de Medições de Teste

O propósito de um programa de medida de teste é identificar, coletar, analisar e aplicar métricas para auxiliar uma organização na determinação do progresso de teste, avaliação da qualidade e efetividade de seu processo de teste, determinar a produtividade de sua equipe de teste, determinar os resultados de esforço de melhoria de teste e avaliar a qualidade de seus produtos software. Exemplos de métricas relativas à teste são custos de teste, produtividade de testador, número de casos de teste executados e número de defeitos detectados.

Submetas de maturidade que suportam esta meta são:

4.2.1. Um comitê ou grupo para toda a organização focando no desenvolvimento de um programa de medida de teste é formado e com fundos e suporte adequados. O comitê desenvolve, documenta, distribui e apóia procedimentos, metas, políticas e métricas aplicadas a artefatos de software e ao processo de teste. As metas, políticas, procedimentos e métricas, uma vez aprovados, são postos em prática e revisados periodicamente.

4.2.2. Um programa de medida de teste é desenvolvido de acordo com política com um sistema de reportar medida. Medidas são coletadas, armazenadas e analisadas. Elas são aplicadas por toda a organização para fixar metas de teste e de projeto, melhorar a qualidade de produto e de processo de teste. Métricas são aplicadas pela organização para apoiar tomada de decisão, planejamento de projeto e teste, acompanhamento de projeto e teste, monitoramento e planejamento de ações.

4.2.3. Treinamento, ferramentas e outros recursos são providos para suportar o programa de medida de teste.

Questões

1. Foi estabelecido um comitê ou grupo para toda a organização, com fundos e recursos, responsável por medidas de teste?

2. Foi desenvolvida uma política de medida de teste para toda a organização, distribuída e aprovada?
3. As preocupações de clientes foram contempladas na política e planos de medida?
4. Foram definidos, aprovados e documentados, procedimentos de medida e sistemas de relato?
5. Foram especificadas e documentadas métricas adequadas para cada etapa do ciclo de vida?
6. São providos recursos adequados (ex., fundos, materiais, ferramentas) para implementar o programa de medida de teste?
7. É disponível para gerentes e equipe técnica, treinamento na identificação, coleta e análise de medidas?
8. Cada projeto segue política organizacional escrita para realização de medidas?
9. Há um repositório de dados de teste disponível para uso da equipe técnica e gerência?
10. São fixadas metas quantitativas para cada projeto?
11. As métricas são utilizadas para acompanhar e monitorar teste?
12. Há um conjunto definido de atributos e métricas de qualidade de software?
13. São utilizadas medidas de atributos de qualidade de software para acompanhar qualidade de software durante o teste?
14. São utilizados dados de itens de teste para suportar planejamento de ações de melhoria de processo de teste?
15. O programa de medida é avaliado periodicamente?

Meta de Maturidade 4.3: Avaliação da Qualidade de Software

O propósito da meta de maturidade de avaliação de qualidade de software é relacionar questões de qualidade de software à adequação do processo de teste, definir e promover uso de atributos de qualidade de software mensuráveis, e definir metas de qualidade para avaliação do produto de trabalho de software. Amostras de atributos de qualidade são

exatidão, integridade, usabilidade, manutenibilidade, flexibilidade, testabilidade, portabilidade, reusabilidade e interoperabilidade.

Submetas de maturidade que suportam esta meta são:

4.3.1 Um comitê ou grupo para toda a organização focando na avaliação de qualidade de software é formado, e provido com fundos e suporte. O comitê desenvolve, documenta, distribui e apóiam procedimentos, metas, políticas, padrões e métricas para avaliação de qualidade de software. As metas, políticas, procedimentos, padrões e métricas, uma vez aprovados, são postos em prática e revisados periodicamente.

4.3.2 Treinamento, ferramentas e outros recursos são providos para suportar avaliação de qualidade de software.

4.3.3 Metas de qualidade são desenvolvidas para cada projeto de acordo com política. O processo de teste é estruturado, medido e avaliado para garantir que metas de qualidade sejam alcançadas. Entrada de usuário/cliente é solicitada para o desenvolvimento de metas de qualidade.

Questões

1. Um comitê ou grupo para toda a organização para avaliação de qualidade de software foi estabelecido e provido de fundos e suporte?
2. Foi desenvolvida uma política relativa à medida da qualidade de software, aprovada e distribuída pelo comitê?
3. As preocupações de cliente são refletidas na política da qualidade?
4. Foi desenvolvido um conjunto de padrões e procedimentos de avaliação, documentado e aprovado?
5. Há recurso adequado (ex., fundos, materiais, ferramentas) provido para programas de qualidade e de avaliação de qualidade?
6. Gerentes e equipe técnica são treinados para fixar metas de qualidade mensuráveis, desenvolver e compreender padrões de qualidade, coletar medidas de qualidade e avaliar atributos de qualidade de artefatos de software?
7. Cada projeto segue política organizacional escrita para avaliação de qualidade de software?

8. Estão fixadas metas para avaliação de produtos do trabalho de software para cada projeto?

9. Foi especificado, verificado, distribuído e aprovado um conjunto de atributos de qualidade de software mensuráveis?

10. Há metas de qualidade mensuráveis fixadas para cada projeto?

11. Há procedimento formal alocado para entrada de cliente no processo de avaliação da qualidade de cada projeto?

12. O processo de teste é verificado periodicamente para avaliar seu impacto na qualidade de software?

13. São feitos melhoramentos no teste e outras avaliações da qualidade para aumentar o nível de qualidade de software?

14. As atividades de avaliação da qualidade de software são revisadas periodicamente?

Nível 5 - Otimização / Prevenção de Defeito e Controle de Qualidade

Meta de Maturidade 5.1: Prevenção de Defeito

O propósito desta meta de maturidade é encorajar uma organização a formalmente classificar, registrar e analisar seus defeitos. A organização é também encorajada a usar uma combinação de análise de causa de defeitos e planos de ação para guiar mudanças no processo de modo a eliminar estes defeitos nos produtos futuros. Atividades de prevenção de defeitos recomendadas incluem registro e acompanhamento de defeitos, análise de causa, plano de ação, ações de implementação e acompanhamento; e treinamento em métodos de prevenção de defeitos.

Submetas de maturidade que suportam esta meta são:

5.1.1 Um comitê ou grupo para toda a organização focando na prevenção de defeitos é formado e provido com fundos e suporte. O comitê desenvolve, documenta, distribui e suporta procedimentos, metas, políticas, padrões e métricas de prevenção de defeitos, que uma vez aprovados, são postos em prática e revisados periodicamente.

5.1.2 Treinamento, ferramentas e outros recursos são providos para suportar atividades de prevenção de defeitos.

5.1.3 Equipes de prevenção de defeitos são estabelecidas conforme política, com suporte gerencial. As equipes asseguram que defeitos injetados/removidos são identificados e registrados para cada fase do ciclo de vida, um procedimento de análise de causa é formalmente estabelecida para identificar a causa raiz de defeitos, e que planos de ação são desenvolvidos através da interação de gerentes, desenvolvedores e testadores para prevenir reincidência de defeitos. Estes planos são acompanhados e ocorrem mudanças no processo como um resultado de sucesso em projetos piloto.

Questões

1. Um comitê ou grupo para toda a organização sobre prevenção de defeitos foi estabelecido com fundos e suporte?
2. Foram desenvolvidos políticas, programas e procedimentos para suportarem prevenção de defeitos, distribuídos pelo comitê e aprovados?
3. As preocupações de cliente são refletidas na política de prevenção?
4. Há recursos adequados (ex., fundos, material, ferramentas) para atividades de prevenção de defeitos?
5. Os gerentes e equipe técnica são treinados nas atividades de prevenção de defeitos?
6. Foram estabelecidas equipes de prevenção de defeitos (membros podem ser parte de desenvolvimento, teste, Garantia de Qualidade de Software)?
7. Cada projeto segue política organizacional escrita para prevenção de defeitos?
8. São planejadas as atividades de prevenção de defeitos?
9. Para cada fase do ciclo de vida, todos os defeitos injetados/removidos são classificados e formalmente registrados no repositório de defeitos?
10. Uma vez identificadas, as causa comuns dos defeitos são analisadas e eliminadas automaticamente?
11. São desenvolvidos planos de ação específicos para evitar a recorrência de defeitos?

12. As ações de prevenção de defeitos são medidas e acompanhadas para garantir progresso e avaliar efetividade?

13. Ações de prevenção de defeitos são implementadas na organização na forma de mudança de processo documentada?

14. As atividades e programas de prevenção de defeitos são revistos periodicamente?

Meta de Maturidade 5.2: Controle da Qualidade

O propósito desta meta de maturidade é desenvolver um conjunto de procedimentos e práticas que suporte a entrega de software de alta qualidade que atende completamente os requisitos do cliente. Atendimento desta meta permite uma organização incorporar métricas, técnicas e ferramentas avançadas para melhorar a efetividade de seu processo de teste, reduzir defeitos de software, melhorar a confiabilidade de software e melhorar usabilidade. Atividades de controle de qualidade requerem ferramentas automáticas para suportar a execução e re-execução de casos de teste e coleta e análise de defeito. Técnicas estatísticas e perfis de usuário são utilizados para suportar os esforços de teste e atendimento das metas de qualidade. Níveis de confiança podem ser estabelecidos que indicam a probabilidade de um software ser livre de falhas. O software é testado para usabilidade. Critérios quantitativos são usados para determinar quando parar os testes. Estes critérios quantitativos podem ser baseados em, por exemplo, atingir um certo nível de confiabilidade, ou quando o número de defeitos por unidade de tempo a um nível de severidade selecionado atinge determinado nível.

Submetas de maturidade que suportam esta meta são:

5.2.1 Um comitê ou grupo para toda a organização focando em controle de qualidade é formado e provido com fundos e suporte. O comitê desenvolve e atualiza documentos, distribui e suportam procedimentos, metas, políticas, padrões e métricas para controle de qualidades. As metas, políticas, procedimentos, padrões e métricas, uma vez aprovados, são postos em prática e revisados periodicamente.

5.2.2 Os grupos de teste de software e Garantia de Qualidade de Software identificam atributos de software viáveis e estabelecem metas qualitativas e quantitativas para produtos software com entrada de usuário e cliente, de acordo com políticas. Estas metas são incluídas nos planos de teste. São usadas ferramentas e técnicas de teste para determinar

se a qualidade foi atingida. Critérios quantitativos são utilizados para tomar decisões de parar teste.

5.2.3 Os grupos de teste e grupos relativos à qualidade são treinados e responsabilidades designadas para o uso de métodos estatísticos e outras atividades de avaliação relativas a qualidade de software tal como teste de usabilidade. O grupo de teste ou grupo relacionado interage com usuários e clientes para obter entradas para modelagem de uso e teste de usabilidade.

Questões

1. Foi formado um comitê ou grupo para toda organização focado em controle de qualidade e provido de fundos e suporte?
2. Foram desenvolvidos políticas, programas e procedimentos para suportarem prevenção de defeitos, distribuídos pelo comitê e aprovados?
3. As preocupações de cliente são refletidas na política de controle de qualidade de software?
4. Há recursos adequados (ex., fundos, materiais, ferramentas) para atividades de controle de qualidade?
5. Os gerentes e equipe técnica são treinado nas atividades de controle de qualidade?
6. São utilizados métodos estatísticos durante os testes de avaliação de qualidade de software?
7. Há metas de qualidade de software, quantitativas e qualitativas, identificadas pelos grupos de teste e Garantia de Qualidade de Software, com entrada de usuários/clientes?
8. As metas de qualidade são incorporadas nos planos de teste?
9. Os atributos são relativos à qualidade identificados, medidos e documentados?
10. É coletada entrada de cliente para modelagem de uso?
11. Foi designada responsabilidade por teste de usabilidade?
12. O software é avaliado cuidadosamente quanto à usabilidade com respeito às necessidades do cliente?

13. Há mecanismos para a participação de usuários no teste de usabilidade?

14. O feedback de usuário é aplicado para fazer melhorias no software em desenvolvimento?

15. Há critério quantitativo para decisões de parar o teste especificado nos planos de teste?

16. As atividades de controle de qualidade são revisadas periodicamente?

Meta de Maturidade 5.3: Otimização do Processo de Teste

O propósito desta meta é promover melhorias contínuas no processo de teste e de reuso de teste. Uma organização é encorajada a identificar práticas de teste que necessita ser melhorada, para implementar melhorias e acompanhar o progresso da melhoria. Ela é também encorajada a aplicar atividades de controle de processo para teste, para continuadamente avaliar novas ferramentas e tecnologias de teste para adaptação e suportar transferência de tecnologia. Finalmente, uma organização é encorajada a identificar subprocesso de teste de alta qualidade, armazenar na Biblioteca de Processo, e cadastro para reuso em projetos futuros.

Submetas de maturidade que suportam esta meta são:

5.3.1 Um grupo para toda a organização focado em melhorias do processo de teste é formado e provido com fundos e suporte. Ele tem responsabilidades contínuas de supervisão das questões de processo de teste que incluem reuso de processo de teste, controle de processo de teste, avaliação e melhoria de processo de teste e transferência de tecnologia. Ele provê liderança para esforços de melhoria de processo de teste. O grupo desenvolve e atualiza documentos, distribui e suportam procedimentos, metas, políticas, padrões e métricas para atividades de melhoria de processo de teste. As metas, políticas, procedimentos, padrões e métricas, uma vez aprovados, são postos em prática e revisados periodicamente.

5.3.2 Treinamento é disponível para gerência e equipe técnica nas áreas de planejamento, avaliação de processo de teste, controle de processo de teste, reuso de processo de teste e transferência de tecnologia.

5.3.3 O processo de teste é submetido a avaliações periódicas de acordo com a política e planos de ação são implementados para prover melhorias. Novas ferramentas e técnicas estão sendo continuamente avaliadas e integradas.

5.3.4 Componentes de processo de teste de alta qualidade são reconhecidos como ativos e são armazenados e reusados por toda a organização.

Questões

1. Foi estabelecido um grupo ou força tarefa focada em melhorias de processo de teste, com fundos e suporte?
2. Foram desenvolvidos para toda a organização, aprovados e distribuídos, políticas, programas e procedimentos, para suportarem avaliações, melhorias e reuso de processos de teste?
3. Foram desenvolvidos para toda a organização, aprovados e distribuídos, políticas, programas e procedimentos para suportarem controle de processo de teste?
4. Foram desenvolvidos para toda a organização, aprovados e distribuídos, políticas, programas e procedimentos para suportarem transferência de tecnologia?
5. São providos recursos adequados (ex., fundos, materiais e ferramentas) para transferência de tecnologia?
6. São providos recursos adequados (ex., fundos, materiais e ferramentas) para controle de processo de teste?
7. São providos recursos adequados (ex., fundos, materiais e ferramentas) para avaliação, melhoria e reuso de processo de teste?
8. São treinados gerentes e equipe técnica em controle de processo?
9. Gerentes e equipe técnica são treinados em avaliações, melhoria e reuso de processo de teste?
10. Gerentes e equipe técnica são treinados em transferência de tecnologia?
11. Foi estabelecida uma Biblioteca de Processos?
12. Os processos na Biblioteca são catalogados e reusados em projetos subseqüentes?
13. São avaliadas continuamente ferramentas de teste para uso do grupo de teste?
14. Há procedimento para adaptação e integração de novas ferramentas e novas tecnologias no processo de teste (transferência de tecnologia)?

15. O processo de teste é avaliado periodicamente?

16. Os resultados de uma avaliação produz ações de melhorias?

17. Ações de melhorias de sucesso tem conduzido a alterações no processo de teste da organização, documentos e padrões associados?

18. São definidas medidas para o controle de processo de teste?

19. As avaliações periódicas das atividades de controle de processos estão resultando em ajustes no processo de teste?

20. O programa de transferência de tecnologia é revisado periodicamente?

21. Os programas de avaliação e melhoria de processo de teste são revisados periodicamente?

22. O controle de processo de teste é revisado periodicamente?

23. O programa de reuso de processo de teste é revisado periodicamente?

Seção 5. Questões de Ferramentas de Teste

Os assessores devem notar que questão de ferramenta de teste não tem impacto formal sobre os resultados de classificação do processo de avaliação do TMM. Esta seção do questionário é utilizada pelos assessores para ganhar experiência do estado atual de um processo de teste na organização. Responda estas questões são proveitosos também para construir ferramentas para os testadores.

Para cada tipo de ferramenta listada abaixo o responsável deve decidir de qual forma eles são aplicados ”nunca”, ”raramente”, ”freqüentemente”ou ”sempre”.

I. Ferramentas Recursos de Teste Administrativo

1. Administração de configuração (monitorar e controlar os esforços de mudanças através do desenvolvimento, manutenção e preserva a integridade do desenvolvedor e versões de released).

2. Ferramentas gerenciamento de projeto (ajuda projetar/plano administrativo, cronograma e rota de desenvolvimento, teste e manutenção de sistemas).

3. Planejar teste (ajudar desenvolvedor/testador/administrador em planejamento e aprovação de definição, sistema, integração e teste de nível de unidade).

II. Ferramentas de Requisitos e Projeto Suporte de Teste

1. Requisitos e análise de especificações (avaliação de especificações de consistência, perfeição e conformidade para estabelecer padrões específicos).
2. Sistemas/protótipos simuladores (unir análises e atividades de projeto com teste).
3. Determinar requisitos (reduzir o esforço de trabalho para determinar requisitos para informações de projetos associados, código fonte e amplos projetos de casos de teste).

III. Implementação e Ferramentas de Suporte e Manutenção de Teste

1. Compiladores.
2. Análise estática do código fonte (analisa o código fonte sem executa-lo).
    - Auditores (análise do código para assegurar conformidade para estabelecer regras e padrões).
    - Medir complexidade (computar métricas de código fonte para determinar vários atributos complexos associados com o código fonte ou escrever projetos em uma linguagem de programação de projeto).
    - Ferramentas de referência cruzada (prover referências entre várias entidades).
    - Medidores de tamanho (contar as linhas de código fonte, SLOC).
    - Conferir a estrutura (identificar anomalias estruturais e desenvolver gráfico ou código de representações textual).
    - Sintaxe e análises semânticas (identificar tipo de conflitos em argumentos de chamada de sub-rotinas compilados separadamente).
3. Ferramentas de preparação de teste (suporte preparação de dados de teste ou informações de casos de teste).

- Extratores de dados (construir dados de teste de banco de dados existentes ou conjunto de testes).
- Requisitos baseados em geradores de caso de teste (ajuda o desenvolvedor avaliar requisitos pela construção de casos de teste de requisitos escritos seguindo as regras de ferramentas formais de especificação de linguagem).
- Geradores de dado de teste (suporta o desenvolvimento de entradas de teste que são formatados ou podem ser formatados leituras nos arquivos requisitados).

4. Ferramentas de execução de teste (analisa dinamicamente o software sendo testados).

- Declaração dos analisadores (instrumento de código com expressões lógicas de condições específicas ou relação entre as variáveis do programa).
- Ferramentas de replay-captura (registra automaticamente as entradas/saídas do teste utilizando captura de scripts, repetir o teste utilizando a repetição de scripts. O re-teste é proveitoso quando são realizadas mudanças).
- Cobertura/analisadores de freqüência (avalia o grau de cobertura do caso de teste com respeito à declaração executada, ramos, caminhos ou módulos).
- Depuração (uma ferramenta de teste não cobre; este suporta a localização de defeitos relevantes durante o teste).
- Emuladores (pode ser utilizado na falta de locais ou sistema de componentes inválidos e usualmente opera em tempo real de velocidade dos componentes sendo emulados).
- Analisadores de rede (analisa o tráfego sobre a rede para identificar áreas e condições problemáticas assim como permite a simulação das atividades de múltiplos terminais).
- Performance/analisadores de tempo (monitora as características do timing dos componentes do software ou sistemas todo).
- Verificadores de erros em tempo real (monitora programas de referência de memória, vazamento de memória ou a localização de erros na memória).
- Simuladores (são utilizados em locais de perda ou sistema de componentes inválidos).
- Telas de status/seções documentadas (provê status de informações e seleção de registro de informação sobre uma execução de teste).

- Administradores de execução de teste (automatiza variáveis de um conjunto de testes executados, realiza uma variedade de testes e limpam após um teste de reset do sistema).

5. Avaliadores de teste (realiza consumo de tempo e funções propenso a erro).

   - Comparadores (compara entrada com qualquer outro após um software de teste e anota as diferenças).
   - Reduzir os dados e analises (converter dados para uma forma que possa ser interpretado mais rapidamente e algumas vezes analise estatística de performance sobre os dados).
   - Defeitos/mudança de caminhos (manter caminhos das informações de defeito e gerar registros de defeitos).

Seção 6. Questões de Tendências de Teste

Como no caso de uso de questões em ferramenta de teste, as questões de tendência de teste são projetados para prover a visão ampla para os assessores de processo de teste. O assessor pode utilizar e responder como auxiliar no processo de avaliação.

1. Conforme sua perspectiva, quais são as maiores forças no processo de teste em sua organização hoje?

2. Conforme sua perspectiva, quais são as maiores fraquezas no processo de teste em sua organização hoje?

3. Quais mudanças à organização têm feito para melhorar o processo de teste nos últimos dois - cinco anos?

4. Como você poderia avaliar a eficiência total do processo de teste hoje comparados nos últimos dois anos? Por favor selecione uma das seguintes alternativas: mesmo, melhorou, grande melhoria, pouca melhoria, pior, não conhece.

5. Comparar as frações de tempo e recursos de teste alocados hoje e de dois anos atrás. Por favor selecione uma das seguintes alternativas: mesmo, aumentou muito, diminuiu muito, continua o mesmo, não conhece.

6. Comparar o número corrente de testadores em tempo integral na organização com o número avaliado há dois anos atrás. Por favor selecione uma das seguintes alternativas: mesmo, aumentou, diminuiu, não conhece.

7. Comparar o número corrente de testadores em tempo parcial na organização com o número avaliado há dois anos atrás. Por favor selecione uma das seguintes alternativas: mesmo, aumentou, diminuiu, não conhece.

8. Conforme sua perspectiva, nos últimos dois anos a comunicação entre testadores, desenvolvedores e administradores: melhorou, continua o mesmo, ficou pior, não conhece?

9. Conforme sua perspectiva a administração cumpriu com as necessidades dos testadores/desenvolvedores? Sim, não, não muito, não conhece.

Seção 7. Comentários dos Respondedores

Esta seção é reservada para os comentários do respondedor sobre a natureza dos questionários do TMM. Os respondedores podem comentar sobre qualquer aspecto do questionário, por exemplo, sobre a clareza das questões, a organização das questões, o conteúdo e usabilidade do documento.

Apêndice

C

Ferramentas de Auxílio ao TMM

A existência de ferramentas automatizadas é de fundamental importância para viabilizar a utilização dos modelos de maturidade. Sem o apoio de tais ferramentas, as atividades a serem realizadas para a implantação dos modelos tornam-se extremamente caras e sujeitas as falhas introduzidas pela intervenção humana. Neste apêndice são descritas as principais ferramentas que podem ser utilizadas durante o processo de implantação do modelo TMM que, conseqüentemente, também se aplicam, em parte, ao MPT-PE. Destaca-se que o texto apresentado a seguir consiste de uma tradução adaptada do Capítulo 14 do livro Practical Software Testing de Ilene Burnstein Burnstein (2003).

C.1 TMM - TOOLs

Ferramentas

Nível 1 - Ferramentas para o nível 1 do TMM

Embora este nível não tenha metas de maturidade, uma organização pode começar reunir componentes para a bancada de Testadores. A meta é prover um conjunto mínimo de ferramentas básicas para cada desenvolvedor. Estas ferramentas não requerem treinamento avançado e poderiam ser avaliados por todos os desenvolvedores. Acesso às

ferramentas é através de um computador pessoal ou estação de trabalho individual. O PC ou estação de trabalho de cada desenvolvedor deve ser equipado com: (i) um processador de texto, (ii) uma planilha, (iii) um comparador de arquivo que indica se dois arquivos são iguais ou diferentes, (iv) um e-mail para assegurar a comunicação adequada e (v) um programa de captura de tela que permita enviar o conteúdo da tela para um arquivo ou para impressão.

O objetivo do teste é mostrar que o software funciona.

1. *Depuradores interativos.* Estas ferramentas auxiliam os desenvolvedores na compreensão do código e localização de defeitos. Elas devem ter capacidade de TRACE BACK e BREAKPOINT para habilitar os desenvolvedores a entender a dinâmica de execução do programa e para identificar áreas suspeitas do código. Ferramentas de depuração estabelecem a base para a separação dos processos de teste e depuração, pela ilustração das diferentes habilidades, psicologias e modelos requeridos por ambos processos.

2. *Ferramentas de Construção da Configuração.* Estas ferramentas (por exemplo: o UNIX "MAKE") permite a construção de configurações de sistema de software em um ambiente controlado. Elas suportam um processo de construção de sistema bem ordenado, administrado e repetível para desenvolvedores e testadores. Estas serão suplementadas pelas ferramentas de administração de configuração quando a organização está pronta para o nível três do TMM.

3. *Contadores de Linha de Código (LOC).* Medir o tamanho do software tem muitas aplicações úteis. Uma ferramenta que mede automaticamente o tamanho resultará em medidas de tamanho consistentes e repetíveis que são úteis para vários propósitos incluindo estimação de custo que é essencial para gerentes de projeto e de teste, cálculos de volume de defeito (número de defeitos/KLOC) e medidas de produtividade (LOC produzido/Unidade de tempo). O contador LOC pode requerer o desenvolvimento ou adaptação de padrões de contagem de linha que deverão ser seguidos.

A planilha requerida neste nível 1 do TMM é útil para registrar medidas simples como tempo real gasto em atividades de teste, medidas LOC para cada projeto e o número de cada tipo de defeito encontrado em cada projeto. Os dados coletados desta maneira ajudarão uma organização a desenvolver uma base mensurável para seus processos de teste e mais tarde fornecerá os dados para o repositório de defeito. As planilhas podem

também suportar o desenvolvimento de uma matriz de referência cruzada para teste. Nos níveis mais altos do TMM, onde a equipe de teste é bem educado e treinado, a planilha é eventualmente substituída por ferramentas mais avançadas.

Nível 2 - Ferramentas para Fase de Definição

No nível 2 do TMM uma organização estabelece uma etapa de teste distinta como parte de seu ciclo de vida de desenvolvimento. A etapa de teste é suportada pelas políticas, planos, técnicas e práticas. Fazer políticas requer recursos orientados a pessoas. As ferramentas que suportam esta meta consiste principalmente de processadores de texto para registrar as políticas. Um conjunto intraorganizacional de paginas web contendo as políticas para ajudar a promover, disponibilizar e distribuir por toda a organização.

Ferramentas de planejamento do Nível 2 do TMM são especialmente necessários para suportar a meta de maturidade do plano de teste desde que o planejamento é essencial para um processo de teste definido e administrado. As ferramentas também são necessárias para suportar as técnicas básicas de teste e indicar as diferentes tarefas e atividades associadas com teste e depuração. As ferramentas introduzidas no nível 2 do TMM devem ser simples e fáceis de serem dominadas pelos desenvolvedores desde que antes não havia um grupo dedicado à teste de software e nenhum programa de treinamento técnico formal. As ferramentas devem suportar planos de melhoria, aumento da produtividade e qualidade de software. Resultados positivos de uso de ferramenta devem ser mostrados claramente de modo a prover um forte incentivo para igualar a adaptação pelos colegas e pelo apoio gerencial.

1. *Planejadores de Projeto e Planejadores de Teste.* Estas ferramentas são necessárias para automatizar e padronizar o processo de planejamento de teste na organização. As ferramentas suportarão as especificações e registros de itens necessários para plano de teste de alta qualidade. Exemplo itens incluem metas de teste, recursos de teste requeridos, custos, agendas, técnicas a serem utilizadas, projetos de teste e a designação da equipe responsável pelas tarefas de teste. Devido à falta de ferramentas comerciais especificas para planejamento de processo de teste, uma organização pode decidir desenvolver seus próprios modelos de plano de teste e ferramentas de suporte e torna-los disponível para todos grupos internos.

   Ferramentas de planejamento de projeto poderiam aumentar os modelos de teste domésticos e serem usadas para desenvolver e registrar agendas, custos, listas de tarefas e assim por diante. No nível 2 do TMM as organizações devem testar a

unidade, integração e sistema. Ferramentas que suportam o planejamento de teste podem ajudar na diferenciação das sub-fases de teste e planos individuais podem ser desenvolvidos em cada sub-fase.

2. *Verificadores de Erro em Tempo de Execução.* Estas ferramentas são conhecidas também como marcadores de limite, teste de memória e detectores de vazamento. Eles detectam problemas de memória, limites de campos, alocação de memória livre e ocupada e o uso de alocadores de memória. Esse grupo de ferramentas ajudará na descoberta de defeitos e fornecem normalmente mensagens de erros detalhadas que ajuda os usuários seguir a pista de defeitos.

3. *Ferramentas de Suporte de Preparação de Teste.* (Nível iniciante). Desde que não há um grupo dedicado a teste no nível 2 do TMM, e os desenvolvedores estão apenas começando usar técnicas de teste e estratégias básicas, é melhor introduzir ferramentas simples de suporte à preparação de teste para ajudar no desenvolvimento de casos de teste de caixa preta e caixa branca.

   Uma das ferramentas recomendadas é uma que produz dados de teste caixa preta utilizando métodos algorítmicos. Por exemplo, ferramentas que requerem um gráfico de causa e efeito, ou uma entrada de classes de equivalência e análise do valor limite. Os usos destas ferramentas darão incentivos adicionais aos desenvolvedores para conhecer os conceitos de teste e forçar seu uso na organização. Ferramentas que suportam teste de caixa branca no nível 2 do TMM são analisadores de controle de fluxo que geram gráficos de controle de fluxo e analisadores de fluxo de dados que produz informações do fluxo de dados. Estas ferramentas podem ser introduzidas para ajudar os desenvolvedores identificar ramos, caminhos básicos e uso de variáveis. Utilizando estas informações os desenvolvedores podem projetar casos de teste que satisfazem a cobertura das metas. As ferramentas mais avançadas de geração de dados de teste poderão ser recomendadas nos níveis mais altos do TMM.

4. *Analisadores de Cobertura.* Adaptação de analisadores de cobertura para ajudar com o teste de caixa branca. As ferramentas ajudam o desenvolvimento de metas mensuráveis de conclusão de teste para planos de teste e assegura que as metas são satisfeitas.

   Executando o código alvo sob o controle de uma ferramenta de cobertura dá ao desenvolvedor uma medida do grau de declaração e / ou cobertura de ramos e indicam quais as estruturas (caminhos) do programa têm sido, ou não têm sido, exercitado pelo conjunto de casos de teste. Se metas de cobertura não são satisfeitas

com o conjunto de caso de teste atual, o desenvolvedor pode projetar casos de teste adicionais e re-executar o código sob o controle da ferramenta para determinar se as metas de cobertura foram agora satisfeitas.

5. *Ferramentas de Referência Cruzada.* Estas são ferramentas simples que permite aos usuários localizar as ocorrências de itens como eles aparecem em diferentes artefatos de software. Ferramentas de referência cruzada são úteis para construir modelos mentais de software com propósito de fazer mudanças, desenvolver testes e re-execução de testes. Por exemplo, um desenvolvedor ou testador pode querer determinar onde uma variável específica aparece em toda lista de código fonte de modo a centrar os testes sobre as variáveis a completar. Outros itens que podem ser traçados ou referenciados são labels, literais, parâmetros e chamadas de sub-rotinas.

Nível 3 - Ferramentas para Integração

No nível 3 do TMM uma organização terá um grupo dedicado para teste assim como um programa de treinamento técnico. Ferramentas de teste podem ser introduzidas, e pode ser esperado que especialistas de teste terão os recursos e habilidades para avaliar, comprar, usar, integrar e institucionalizar as ferramentas. Diferente das organizações menos maduras, aquelas com nível 3 do TMM devem exibir uma alta taxa de uso de ferramenta quando todas as metas de maturidade deste nível são atingidas.

As ferramentas recomendadas do nível 3 do TMM são distintas em essência e são selecionados para suportar as metas de maturidade deste nível e para:

- Melhorar a qualidade de software;
- Controlar e monitorar teste;
- Melhorar a produtividade do testador/desenvolvedor;
- Integrar o teste por todo o ciclo de vida;
- Dar visibilidade para a organização de teste;
- Ilustrar os benefícios de ter ambos uma organização dedicada à teste e um programa de treinamento técnico.

As ferramentas provêem suporte contínuo para planejamento de teste, projeto de teste e execução automática. Suporte para controle, monitoramento e acompanhamento de

processo de teste são também produzidos. Note há uma ênfase muito forte nas ferramentas relacionadas à coleta de requisitos e rastreamento para teste. Estes servem para suportar a integração das atividades de teste com outras atividades no ciclo de vida, uma meta de maturidade no nível 3. O uso destas ferramentas permite a integração começar cedo no ciclo de vida do software. Elas permitem também a definição do papel de usuários/clientes no processo de teste.

1. *Ferramentas de Gerenciamento de Configuração.* Estas são ferramentas complexas que são essenciais para garantir que a execução de mudanças é monitorada e controlada para todos os artefatos relacionados ao projeto. Artefatos, chamados itens de configuração, sob controle destas ferramentas inclui versões de código, mudança de requisitos, assim como itens relativos à teste tais como planos de teste, procedimentos de teste e casos de teste. Para o sucesso operacional destas ferramentas de estruturas organizacionais tal como um controle de mudanças é essencial. Especialistas de teste devem estar entre os membros desse conselho.

   Uma ferramenta de gerenciamento de configuração suportará duas metas de maturidade no nível 3 do TMM, "integração de atividades de teste no ciclo de vida" e "controle e monitoramento de teste". Suporte para estas metas é na forma de administrar, coordenar e manter as dependências e relacionamentos entre todos os artefatos de software.

   Por exemplo, o relacionamento entre requisitos e casos de teste, elementos de projeto e casos de teste, pode ser estabelecido e disponibilizado utilizando uma ferramenta de gerenciamento de configuração. As ferramentas também têm funções de controle e coordenação para supervisionar todas as mudanças feitas para itens de configuração. Elas também suportam acessos privilegiados aos itens de configuração para todos os desenvolvedores/testadores. A ferramenta também provê visibilidade para o processo de teste na organização e ajuda a estabelecer a necessidade de um grupo de especialistas de teste desenvolver os produtos de trabalho relacionado a teste.

2. *Registradores de Requisitos* (Registradores de Caso de Uso). No nível 3 do TMM uma meta primária é introdução cedo das atividades de teste no ciclo de vida do software. Boas práticas de teste requerem uma organização começar desenvolver um plano de teste de alto nível na fase de requisitos. Testadores precisam ter certeza que requisitos representam um produto testável. Muitas organizações registram requisitos em uma linguagem de formato nativo utilizando um processador de texto. Outras utilizam ferramentas de modelagem de requisitos e registram informações em

um formato gráfico tal como diagramas de fluxo de dados. Estas representações de requisitos podem não prover suporte adequado para testadores. Existem ferramentas de modelagem de requisitos disponíveis que darão forte suporte para os testadores.

De especial interesse são os registradores de casos de uso que auxiliam com a geração de casos de teste com base em casos de uso.

3. *Verificadores de Requisitos.* Verificadores de requisitos têm a habilidade para checar requisitos por ambigüidade, consistência e demonstração de integridade. Porém, eles não são um substituto para uma revisão de requisitos, a qual, entre outras coisas, checará a integridade e a exatidão da especificação de requisitos.

4. *Rastreadores de Requisitos pelo Teste.* Estes proverão assistência automática para uso de uma matriz de rastreamento de requisitos como introduzida no nível 2 do TMM. No nível 2 do TMM foi sugerido que o uso da matriz poderia ser suportada de uma maneira simples usando uma planilha. Ferramentas de rastreamento de requisitos de teste adicionam mais funcionalidade e capacidade. Elas provêem conexão entre requisitos, projeto, código fonte e casos de teste. O que formalmente era uma tarefa monótona consumidora de tempo é agora automática com estas ferramentas. As ferramentas também desempenham um papel no monitoramento do processo de teste indicando quais requisitos têm/não têm sido coberto por um caso de teste.

5. *Ferramentas de Capture-Replay.* Estas ferramentas são essenciais para automatizar a execução e re-execução de testes. Eles têm um impacto positivo na produtividade do testador. As ferramentas são usualmente combinadas com um comparador. Um testador executa o programa alvo sobre o controle da ferramenta de capture-replay. A ferramenta registra todas as informações de entrada e saída e em modo replay ela repetirá até tudo estar registrado. Tais ferramentas capturam movimentos do mouse, batidas de teclado e imagens da tela. Após uma mudança no software, os testes registrados podem ser facilmente re-executado (teste de regressão). A ferramenta pode repetir os testes registrados e validar os resultados para o software modificado, comparando estes resultados com os que serviram de base na versão anterior. Diferentes tipos de relatórios podem ser gerados pelas ferramentas, por exemplo:

   (i) um relatório de tempo que lista o tempo de execução para cada teste; (ii) um relatório de falha que lista os testes que falharam; (iii) um relatório de regressão que lista somente aqueles testes cujos resultados tenham sido alterado desde a ati-

vação do teste anterior; (iv) um relatório cumulativo que lista os resultados de teste corrente e passados para cada teste executado.

A maioria destas ferramentas assiste os testadores no desenvolvimento de roteiros de teste em uma linguagem que descreverá todos os passos necessários para executar / repetir / re-executar um teste particular. Os roteiros de teste têm um conjunto de comandos cada qual executará um requisito do usuário. Alguns exemplos de comandos são:

mouse()	enviará um evento do mouse para uma aplicação
log()	escreverá um texto em branco para o script log do arquivo
real-time	determinará um modo on/off em tempo real
screen_shot	capturará uma imagem e local da tela no arquivo de imagem de playback
delay()	atrasará o playback da próxima linha escrita por um tempo especificado

As linguagens escritas também têm linguagem de programação elaborados tais como for/next loops, que repetirá uma vez ou bloco de declarações de um certo número de vezes, e uma declaração de chamada que chamará um roteiro de teste. As ferramentas de Capture-replay podem ser categorizados como nativa onde a ferramenta e o software sendo testado residem no mesmo sistema. Ferramentas não nativas requerem um sistema de hardware adicional para a ferramenta. As últimas são muito úteis quando testam sistemas embarcados.

As ferramentas de capture-replay executarão os testes automaticamente assim os testadores não terão repetidamente que re-executa-los manualmente. Os testes podem ser executados sem atenção por longos períodos de tempo. É uma ferramenta muito útil e que deve ser incorporada na bancada do testador. Mas ela deveria ser introduzida quando a organização tem a infraestrutura de modo a suportar o seu uso. Investir em ferramentas é freqüentemente custoso. Para obter benefícios, os testadores precisam de suporte administrativo para assegurar que eles têm a educação apropriada, treinamento, software e hardware para tirar vantagem das muitas capacidades que as ferramentas têm para oferecer. Este provavelmente ocorre no nível 3 do TMM onde o atendimento das metas de maturidade associadas constrói uma infraestrutura que suporta sua adaptação.

6. *Comparadores.* Estas ferramentas comparam os resultados reais dos testes com os resultados esperados e marca as diferenças. Eles têm capacidades mais complexas do que simples comparadores de arquivo recomendados como uma ferramenta básica no nível 1 do TMM. O software sob teste passará se a saída real e a esperada forem

às mesmas dentro de uma tolerância permitida. Um simples comparador como o "diff", facilidade no sistema UNIX, compara igualdade de arquivos de texto. Ferramentas mais sofisticadas podem apresentar melhor performance e ter a habilidade para comparar outros tipos de saída por igualdade; por exemplo, telas e dados gráficos. Em muitos casos um comparador pode ser um componente em um pacote de ferramenta tal como a ferramenta de capture-replay descrita acima.

7. *Rastreadores de Defeito.* Essas ferramentas são chamadas também de administradores de problema. Para controlar e monitorar processos de teste este tipo de ferramenta é essencial. As ferramentas se usadas corretamente tem os benefícios adicionais de melhorar a satisfação do cliente, a produtividade, a qualidade do software e a moral [17]. No nível 3 do TMM uma equipe de especialistas treinados em teste está disponível para tomar vantagem das capacidades de tal ferramenta. Previamente nos níveis 2 e 3 do TMM as planilhas são sugeridas como uma maneira de registro de dados de defeito, mas elas não têm as funcionalidades necessárias para aplicações mais avançadas que as organizações maduras precisam. Rastreadores de defeito permite ao testadores e ao desenvolvedores registrar, resgatar, administrar e gerenciar defeitos por todo o ciclo de vida. Para que a ferramenta seja efetiva, um processo de rastrear defeito deve ser estabelecido, o que requer o suporte de uma organização de teste e treinamento para a equipe de teste e desenvolvimento. Depois de um defeito ter sido detectado, este deve ser registrado em um banco de dados de defeito suportado pela ferramenta. Uma ferramenta de rastrear defeito permite aos usuários construírem um repositório de defeitos mais sofisticado do que é possível com um simples programa de planilha.

   A solução do problema segue as informações registradas disponíveis sobre o defeito. A ordem das soluções deve depender do impacto do defeito e da prioridade atribuída. Na medida que o reparo dos defeitos continuam o status de cada defeito é atualizado para refletir seu estado atual de solução. O código em que o defeito foi corrigido é submetido ao re-teste pela equipe de teste. Uma vez aprovado, um relatório de correção de defeito deve ser feito e o status do defeito é atualizado no registro do rastreador de defeito.

   Um processo de solução de defeito suportado pelas capacidades de um rastreador de defeito permite monitoramento contínuo de defeitos e ajuda a assegurar que todos os defeitos são solucionados. Rastreadores de defeito mais sofisticados suportam a comunicação entre desenvolvedores e testadores para promover a solução de defeito e a integração das atividades de teste por todo do ciclo de vida. Os rastreadores

de defeito pode também emitir (i) relatórios de defeito que rastreiam os esforços de solução, e (ii) relatório sumário que inclui tipos de defeitos, sua época e freqüência de ocorrência.

Rastreadores de defeito também suportam metas de maturidade para níveis mais altos do TMM tais como o desenvolvimento de metas de teste qualitativas e atividades de prevenção de defeito.

Sistemas rastreadores de defeito introduzem muitas mudanças para uma organização. Deve suportar particularmente por um grupo de teste que tem sua própria educação, treinamento e atitude adequados para fazer uso correto das ferramentas. Sob o ponto de vista dos testadores a adaptação de ferramentas de rastrear defeitos serve para aumentar a visibilidade deles na organização e clarear seu papel na melhoria da qualidade de software. Começando no nível 1 do TMM com uma simples planilha de registro de defeitos, passando para o nível 2 com sua ênfase em políticas de teste/depuração e classificação de defeito, e então no nível 3 com um foco em grupo de teste, treinamento e mecanismos de controle e monitoramento, uma organização pode preparar por si pela introdução e integração destas ferramentas importantes. Os processos de rastreamento e manipulação de defeito devem ser ampliados no nível 4 do TMM para incluir defeitos detectados em revisões assim como em testes de execução. Finalmente, no nível 5 do TMM as capacidades das ferramentas de rastreamento de defeito devem ser aplicadas para o processo de prevenção de defeito e para o processo de melhoria contínua do processo de teste.

8. *Medidas de Complexidade.* Estas ferramentas são algumas vezes chamadas de relatórios de métricas. Poderão medir a complexidade ciclomática e muitas vezes estarão integradas com outras ferramentas, tais como um medidor de tamanho (contador de linha de código), um analisador de fluxo de dados, e/ou um analisador de controle de fluxo. Algumas destas ferramentas poderão medir a complexidade na etapa de detalhamento do projeto se um módulo é escrito em um pseudo-códico, linguagem estruturada e padronizada. Outras também poderão gerar Métricas de Ciência de software (Halstead's metrics) que são relativas à complexidade. Exemplos de tais métricas são número de um único operador, número de um único operando e número total de operandos. No nível 3 do TMM a complexidade de um módulo pode desempenhar um importante papel em teste. O nível de complexidade de um módulo dá uma indicação de como o risco do módulo é em termos de probabilidade de defeitos [18], e o número de casos de teste necessários para testar adequadamente. As

organizações devem fixar limites sobre o nível de complexidade admitido e módulos exibindo valores acima do limite devem ser considerado para re-projeto.

9. *Geradores de carga.* Agora que um grupo de teste treinado e dedicado está estabelecido, e o teste é executado em todos os níveis (unidade, integração e sistema), é apropriado introduzir ferramentas de geração de carga na bancada do testador. Estas ferramentas gerarão grande volume de dados necessários para teste de sistema, tais como testes de esforço e de performance. Geradores de carga podem ser usados para produzir um fluxo de transações. Por exemplo, se você estiver testando um sistema de telecomunicação você poderia necessitar simular uma série de transações na forma de chamadas telefônicas de diferentes tipos e durações, chegando de diferentes locais. Uma vez que um grande volume de dados é produzido quando os geradores de carga são utilizados, ferramentas para coleta e análise de dados devem ser disponíveis para os testadores.

Nível 4 - Ferramentas para Medida e Gerência

No nível 4 do TMM a definição de uma atividade de teste é expandida para incluir revisões e isto é expresso como uma meta de maturidade. Medidas e avaliação da qualidade de software são também metas importantes. Algumas das ferramentas suportam revisões como uma atividade de detecção de defeito. Revisões são atividades intensivas em pessoal e têm muito pouco suporte automático, mas há um grupo de ferramentas que podem ajudar revisores no entendimento do software e na detecção de defeitos em artefatos de software. As ferramentas de suporte de revisão usualmente realizam algum tipo de análise estática sobre o artefato em revisão. Por exemplo, introdução de um registro de requisito e uma medida de complexidade nos níveis mais baixos do TMM para suportar metas de maturidade desses níveis.

O uso contínuo dessas ferramentas suportará a implementação de requisitos e revisões de projeto. No nível 4 do TMM às ferramentas que realizam análises estáticas de código são introduzidas. Estas podem ser utilizadas tanto para pré como para pós-revisão para detectar os defeitos antes da execução real do código. Estas ferramentas detectam certos tipos de defeitos - por exemplo, anomalias de fluxo de dados - e devem ser utilizadas em conjunto com as atividades de revisão. Outros tipos de ferramentas chamadas entendedoras de programa são úteis também para revisores de código e testadores para ajudar a construir modelos mentais do código para tarefas de entendimento de programa. Exemplos destes tipos de ferramentas são:

1. *Verificadores de Código.* Algumas vezes são chamados analisadores estáticos estas ferramentas pesquisam apontadores fora do lugar, variáveis não inicializadas e outras anomalias de fluxo de dados. Algumas são capazes de identificar código não alcançável.

2. *Auditores.* Estas ferramentas examinarão o código para identificar violações dos padrões de código estabelecido e/ou formatos de código.

3. *Ferramenta de Compreensão de Código.* Algumas das características destas ferramentas podem ser duplicadas pelos verificadores de código. Porém, muitas outras propriedades de código podem ser reveladas através do uso destas ferramentas. Por exemplo, algumas executam partes de programa para frente e para trás e fornecem dados detalhados e informação de controle de fluxo. Versões mais sofisticadas dessas ferramentas tem componentes de inteligência artificial e contem bases de conhecimento especiais de planos de programas (padrões de código estereotipa). ("Stereotypical code patterns".)

   Eles atendem a realização de tarefas de engenharia reversa e unem o código aos planos para reconhecimento de conceito. Infelizmente, não temos muitos exemplos do último tipo de ferramenta disponível comercialmente no momento.

   Para suportar no nível 4 do TMM as metas de maturidade de avaliação da qualidade de software são sugeridas as ferramentas listadas abaixo para ajudar nos testes automáticos de sistemas de larga escala e permitir a coleção de dados relativos aos atributos de qualidade de software. Ferramentas introduzidas nos níveis mais baixos do TMM também suportam estas metas, por exemplo, ferramentas de capture-replay, medidas de complexidade, rastreadores de defeito, contadores de linhas de código e rastreadores de requisitos.

4. *Geradores de rotinas (harness) de Teste.* Para executar teste de unidade, integração e sistema, o código auxiliar deve ser escrito que é em adição para o código desenvolvido para o sistema sob teste. Este código extra, freqüentemente chamado de rotina de teste, pode incluir drives, stubs, interfaces para um sistema de operação e interfaces para um sistema de banco de dados. A rotina de teste pode ser muito grande em tamanho, representando esforço considerável e muitas vezes é construída especialmente para uma determinada aplicação. Avanços recentes no desenvolvimento de padrões de interface e abordagem padrão para descrever aplicações de interfaces tem capacitado a introdução de ferramentas comerciais para auxiliar na preparação

de rotina teste [4]. Estas ferramentas podem ser muito proveitosas; eles reduzem tempo e esforço em teste e produz rotinas de teste re-usável.

5. *Ferramentas de Teste de Performance.* Estas ferramentas monitoram as características de tempo de componentes de software. Elas suportam testes de carga e de esforço e são essenciais para suporte de teste em sistemas de tempo real para avaliar a qualidade de performance. A disponibilidade destas ferramentas pode ajudar determinar se as metas de desempenho foram atingidas.

6. *Analisadores de Rede.* Estas são ferramentas proveitosas para teste de sistemas de rede assim como software que funcionam em sistemas cliente/servidor, ambiente web e sistemas múltiplos. As ferramentas têm a habilidade para analisar tráfego na rede e identificar condições e áreas de problema. Muito das ferramentas de teste de rede permite um testador monitorar e diagnosticar o desempenho através de uma rede.

7. *Simuladores e emuladores.* Estas ferramentas podem ser utilizadas para substituir componentes de hardware e software que estão perdidos, atualmente indisponíveis ou de custo de reposição muito alto. Ambas as ferramentas são usadas por razões econômicas ou de segurança. Exemplos são emuladores terminais e emuladores ou simuladores para substituir componentes em plantas de energia nuclear. As ferramentas podem ser agrupadas por analisadores de desempenho e/ou geradores de carga.

   O último produzirá grande volume de dados de teste para teste, por exemplo, transações básicas de operações e sistemas de telecomunicações.

8. *Ferramentas de teste de Web.* Estas são ferramentas especializadas que suportam o teste de aplicações com base web. Muitas são por natureza similares aos simuladores emuladores uma vez que elas simulam padrões de tráfego real pela web o que permite aos testadores avaliar o desempenho de aplicações web. Isto permite ao desenvolvedor ajustar uma aplicação web de modo que ela opere efetivamente. Algumas ferramentas de teste Web têm também a capacidade que permite aos usuários validar conexões web.

9. *Ferramentas de administração de teste.* Diversas medidas foram sugeridas para coleta nos níveis inferiores do TMM. Formulários e modelos para facilitar estas coletas devem ser formalizados no nível 4 do TMMM. Devem ser designadas responsabilidades para análise de dados e disseminação de informações. Ferramentas como planilhas e banco de dados podem ser usadas para organizar e armazenar alguns

dados para análise eventual e aplicação para teste de melhoria de processo de teste. Rastreadores de defeito complementam estas ferramentas para armazenar, administrar e analisar dados de defeitos. Outra ferramenta mais sofisticada é útil para coleta e recuperação de dados de teste é uma ferramenta de administração de teste. Um banco de dados de casos de teste é parte do repositório da ferramenta. Idealmente esta ferramenta pode ser um componente do sistema de capture-replay, ou forma uma interface com um sistema capture-replay. Uma ferramenta administrativa completa prove de diversas capacidades:

- Uma interface de usuário para auxiliar os usuários em administrar testes;
- Habilidade para organizar testes e para facilitar recuperação e manutenção;
- Habilidade para administrar a execução de teste para os testes selecionados pelo usuário;
- Habilidade para gerar relatórios de teste, por exemplo, sobre status do teste e sobre o número casos de teste executado durante um período de tempo especificado.

Nível 5 - Ferramentas Otimização, Prevenção e Controle

Como agora os processos de teste são bem definidos e administrados, seu custo e sua efetividade podem ser monitorados.

No nível 5, existem mecanismos que permitem que o processo seja ajustado e continuamente melhorado. Técnicas de controle de qualidade são aplicadas no processo de teste de modo que ele se torne mais previsível e ajustável. Prevenção de defeitos e controle de qualidade são praticados. Amostras estatísticas, medidas de intervalos de confiança e confiabilidade, direcionam o processo de teste. Há procedimentos para seleção e avaliação de ferramentas de teste. Ferramentas automáticas suportam totalmente a execução e re-execução de casos de testes. Existem ferramentas para projeto de casos de teste, teste de itens em manutenção, para coleta e análises de defeitos. No nível 5 do TMMM a qualidade dos processos de teste e depuração é continuamente melhorada pelo uso de ferramentas, tais como rastreadores de defeitos, geradores de carga, analisadores de rede e de desempenho, que fornecem e coletam os dados necessários para a avaliação de confiabilidade.

Uma organização movendo-se para o nível cinco utiliza um conjunto de ferramentas automáticas que suporta todas as etapas do ciclo de vida de teste e prove suporte contínuo para metas de maturidade que estão vigorando.

No nível 5 do TMM, uma organização deve selecionar ferramentas adicionais que ajudam nas atividades de prevenção de defeito, asseguram alta qualidade dos produtos de software e provê suporte contínuo para melhoria de processo de teste. Neste nível, muitos sub-processos de teste maduros estão disponíveis e podem ser armazenados na Biblioteca de Processo Ativo e para subsequente reuso.

Ferramentas podem ser utilizadas para suportar esta atividade.

1. *Ferramentas de Suporte de Biblioteca de Processo Ativo (PAL).* Atualmente não há ferramentas comerciais projetadas especificamente para suportar reuso de componente de processo de teste. Organizações podem usar ferramentas convencionais para esta tarefa, tais como sistemas de administração de bancos de dados ou sistemas de administração de configuração, para armazenar e recuperar componentes de processo de teste. Outra alternativa é desenvolver a sua própria ferramenta para este propósito.

2. *Ferramentas avançadas para roteiros de teste.* Muitos sistemas capture-replay incluem uma linguagem de roteiro de teste. Esta linguagem capacita o testador a executar os testes de maneira completamente automática. Neste nível todas as capacidades da linguagem devem ser usadas pela organização de teste, desde que a equipe de teste é agora bem treinado, motivado e tem as habilidades necessárias para uso de todo potencial da linguagem.

3. *Verificador de asserção (assertion).* Asserções são declarações lógicas sobre as condições do programa que avalia para ”verdadeiro”ou ”falso”. Elas descrevem o comportamento correto do programa. Uma linguagem especializada é freqüentemente associada com estas ferramentas para permitir usuários entrar as asserções. Informações relativas à teste tais como classes equivalentes e pré e pós-condições provê informações úteis para projetar as asserções. O código sob teste é executado sob controle do verificador de asserção e se uma asserção é violada o usuário é notificado. A informação é útil para avaliar o código e localizar defeitos.

4. *Geradores avançados de dados de teste.* Muitas ferramentas de administração de requisitos, tais como as descritas no nível 3, têm capacidades avançadas. Elas podem ser acopladas com um gerador de teste; a informação de requisitos é utilizada para criar caso de testes por métodos de estatística, algoritmo ou heurísticos. Utilizando métodos estatísticos (ou aleatório) a ferramenta gera dados de teste baseados nas estruturas de entrada e valores para formar uma distribuição estatística aleatória.

Com a abordagem de algoritmos, a ferramenta usa um conjunto de regras ou procedimentos para gerar dados de teste. Partição em classes equivalentes, análise de valores limite e gráficos de causa e efeito podem ser usados para direcionar o algoritmo de geração de dados de teste. Métodos heurísticos, ou métodos de geração de teste baseado em falhas requerem que as ferramentas tenham base de conhecimento especializado. A base de conhecimento contém registros de falhas freqüentemente descobertas no passado com entrada por um usuário. A ferramenta usa esta informação para gerar dados de teste.

5. *Sistemas avançados de administração de teste.* No nível 4 foi descrita uma ferramenta administrativa de teste que administra casos de teste, execução de teste e gera relatórios de teste. Há mais sistemas avançados de administração de teste que prove uma localização centralizada e compartilhada para todos os itens relativos a teste incluindo teste de rotinas (harness) e teste de saídas. Estes podem ser ferramentas importantes para suporte de otimização de processo de teste, controle de qualidade e prevenção de defeito. Podem incluir:

   - Conexão com outros pacotes de aplicação tais como e-mail, ferramentas de administração de projeto, planilhas e ferramentas de relatório;
   - Habilidade para definir, rastrear e modificar requisitos e conectá-los aos testes para possibilitar a rastreabilidade;
   - Habilidade para acompanhar construções e componentes do software em desenvolvimento;
   - Habilidade para definir teste de rotina (harness) a diferentes níveis de detalhes (unidade, tese de interação, etc.);
   - Habilidade de acompanhar execução de teste com conexão com ferramentas de automação de teste
   - Plena capacidade para rastrear defeito, incluindo follow-up;
   - Habilidade para desenvolver, armazenar e acompanhar planos de teste e outros documentos de teste;
   - Medidas integradas para facilitar métricas de coleta, armazenagem e recuperação relativas às atividades de teste.

Estes tipos de ferramentas são descritos pela literatura, mas não há ainda versões comerciais com todas estas capacidades. Outro benefício destas ferramentas é prover

um site central onde todos os membros da equipe de desenvolvedores podem ter acesso às informações relativas ao teste. Isto acelera as discussões sobre a qualidade dos testes e dos produtos.

6. Ferramentas de medida de usabilidade. No nível 5 uma organização tem a capacidade de realizar testes de usabilidade como uma atividade para suportar o controle de qualidade de software. Há programas de treinamentos, uma equipe motivada, políticas, métricas e uma cultura organizacional para suportar este tipo de teste. Ferramentas simples para apoiar teste de usabilidade são gravações em áudio e vídeo de sessões de usuários para análise posterior. Outras ferramentas são disponíveis para tornar os fatores de usabilidade mais visíveis para desenvolvedores e testadores. Ferramentas deste tipo podem automaticamente registrar ações de usuário, notas de observadores, identificar dificuldades operacionais, solicitação de usuário, análise da tomada de decisão e gerar relatórios.

Foram descritas muitas ferramentas para cada nível, mas cabe a cada organização selecionar e montar o seu conjunto de ferramentas que satisfaçam as suas metas e políticas organizacionais e que sejam compatíveis com a sua cultura e ambiente.