



Universidade Federal Fluminense

Instituto de Computação

Bacharelado em Ciência da Computação

Uma visão sobre gerenciamento de testes através das
ferramentas Mantis e RQM

Autores: Diego Souza Paulino da Silva e Thiago de Oliveira Henriques

Professor Orientador: José Raphael Bokehi

Niterói-RJ

Abril / 2013

DIEGO SOUZA PAULINO DA SILVA

E

THIAGO DE OLIVEIRA HENRIQUES

Uma visão sobre gerenciamento de testes através das ferramentas
Mantis e RQM

Monografia apresentada ao Curso de Graduação em Ciência da Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: Prof. José Raphael Bokehi

Niterói-RJ

Abril / 2013

Ficha Catalográfica elaborada pela Biblioteca da Escola de Engenharia e Instituto de Computação da UFF

S586 Silva, Diego Souza Paulino da
Uma visão sobre gerenciamento de testes através das ferramentas
Mantis e RQM / Diego Souza Paulino da Silva, Thiago de Oliveira
Henriques. – Niterói, RJ : [s.n.], 2013.
120 f.

Trabalho (Conclusão de Curso) – Departamento de Computação,
Universidade Federal Fluminense, 2013.

Orientador: José Raphael Bokehi.

1. Software. 2. Teste. 3. Gerenciamento de software. 4.
Ferramenta de programação. I. Henriques, Thiago de Oliveira. II.
Titulo.

CDD 005.3

DIEGO SOUZA PAULINO DA SILVA

E

THIAGO DE OLIVEIRA HENRIQUES

Uma visão sobre gerenciamento de testes através das ferramentas
Mantis e RQM

Monografia apresentada ao Curso de Graduação em Ciência da Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Aprovado em Abril de 2013

Banca examinadora

José Raphael Bokehi
Universidade Federal Fluminense - UFF
Professor orientador

Cristina Nader Vasconcelos
Universidade Federal Fluminense - UFF

Marco Antonio Monteiro Silva Ramos
Universidade Federal Fluminense - UFF

If there were things in this world that you had to offer, things that you did well, better than anyone else... Things that you could do that helped people feel better about themselves... It wasn't just a good idea to do those things. It is your responsibility to do those things. Don't try to be something else. Don't try to be less. Great things are going to happen to you and your life. And with that all come great responsibility.

Benjamin Parker

Agradecimentos

Existem professores que passam pela vida das pessoas sem dizer a que eles vieram. Isso nós não podemos dizer sobre o professor José Raphael Bokehi. A sua dedicação dentro da sala de aula só é superada pela sua dedicação em melhorar as condições do nosso curso cada vez mais. Gostaríamos de agradecê-lo não só por toda sua paciência, dedicação e atenção nos auxiliando bastante na preparação deste trabalho, mas também por ter aceitado o desafio de orientar um projeto criado do zero em apenas um semestre. Temos certeza que, sem seu auxílio, não teríamos conseguido entregar nosso projeto final dessa forma que ele está. E o melhor de tudo, podemos dizer que poderemos contar com ele em qualquer momento, pois podemos ter começado nossa relação como alunos e professor. Mas temos certeza que a terminamos como amigos.

Diego Souza Paulino da Silva

Venho aqui agradecer imensamente a todas as pessoas que estiveram no meu caminho e me auxiliaram para que eu pudesse chegar até aqui: Meus pais Gilberto e Vera, meu irmão Diedro, a todos os meus tios e tias, primos e primas, por compreenderem os momentos de ausência por causa de um trabalho ou prova e por sempre me darem palavras de sabedoria nos momentos em que eu fraquejava.

Aos meus amigos piadistas, “mau caráter”, *Pokémon trainer*, beijo, Percival, casal Xbox, “biscate”, a inha e a ona e todos os outros que tiveram papel importante na minha vida dentro da faculdade, por tornar os momentos mais difíceis de provas ou trabalhos em

momentos descontraídos em que todos se ajudavam enquanto jogavam Super Trunfo nos intervalos.

A todos os professores que passaram pela minha vida acadêmica que, mesmo que alguns eu não tenha concordado com seus métodos, acabaram sempre trazendo experiência para que eu pudesse crescer não só dentro da faculdade, mas sim como pessoa.

Aos meus amigos fora do eixo da UFF, sendo atores ou não, que me ajudaram a aliviar a pressão da faculdade com boas doses de piadas, cervejas e confraternizações.

A aquela força especial e espiritual que eu chamo de Deus, mas que você pode chamar de Alá, Buda ou Power Ranger Branco, pela iluminação necessária para nos momentos em que eu fraquejei.

E um comentário especial para aquela senhora que sempre sonhou em me ver na faculdade e que nunca pôde ver isso, eu consegui vovó.

Thiago de Oliveira Henriques

Primeiramente agradeço a minha família. Agradeço aos meus pais Alcir e Neide e ao meu irmão Vinícius por sempre estarem ao meu lado por toda minha vida em todos os momentos e por sempre apoiarem minhas escolhas. Agradeço a minha tia Sandra e ao meu tio Elenilson por compreenderem a diminuição das visitas ao longo da faculdade por conta dos estudos e por todo apoio que me deram.

Agradeço aos amigos da UFF pelos inesquecíveis anos juntos nos melhores e piores momentos, fazendo cada dia especial, não só nas horas de lazer em nossas saídas e viagens como também nas longas e inacabáveis aulas, nos estudos em grupo e por darem boa parte da motivação que precisava para continuar nesse curso até o fim.

Agradeço ao Diego, que me aceitou como dupla para fazer o projeto final sobre um tema que a princípio seria tranquilo. Porém pela dificuldade teve que ter algumas mudanças e que no fim, após alguns problemas e um pouco de sorte, confiou em mim e aceitou mudar completamente o tema para um que não conhecia.

Agradeço por todos os professores da UFF que passei por contribuírem na minha formação e agradecer aos que contribuíram não só na minha vida profissional como na pessoal, não apenas ensinando o que está nos livros e sim indo além, me fazendo lembrar todos os dias que valeu a pena escolher uma universidade federal mesmo tendo que viajar diariamente para outra cidade, que era a reclamação que eu mais ouvia quando dizia que estudava na UFF.

Agradeço aos amigos fora da faculdade que por todos esses anos foram compreensíveis por eu não estar presentes em inúmeros eventos sociais durante e aos finais de

semana por estar em aula ou estudando e que mesmo assim, sempre mantivemos a mesma amizade. Principalmente ao Victor, Pedro, Alexandre por estarem sempre mantendo contato e a todo pessoal do vôlei pela compreensão do meu “desaparecimento” durante a escrita deste trabalho.

Agradeço aos colegas da PrimeUp que me proporcionaram uma ótima experiência na área de testes e principalmente a Janaína, que me auxiliou bastante na construção deste trabalho como na escolha do tema, me passando os conhecimentos da área e por ajudar a divulgar o questionário da pesquisa de mercado.

Resumo

Um dos primeiros conceitos que uma pessoa aprende desde pequena é o de testar um produto antes de utilizá-lo na sua versão final. Avisos como “tome cuidado com a temperatura da água para você não se queimar” ou “experimente e veja se está do seu gosto” ensinam a importância de testes. Em produtos de *software* não poderia ser diferente. Os testes desempenham uma importante função durante o tempo de vida de um projeto, e seu gerenciamento se mostra extremamente importante, para que o controle dos defeitos e correções de um projeto não se perca durante o seu desenvolvimento. Nesse projeto foram abordadas duas ferramentas úteis de gerenciamento de testes: O *Rational Quality Manager* e o *Mantis*. Para cada um delas é descritos seu funcionamento, sendo enfatizadas suas diferenças. Adicionalmente, buscou-se identificar a visão de profissionais do mercado sobre elas.

Palavras-chave: Testes, Software, Gerenciamento, RQM, Mantis.

Abstract

One of the first concepts that a person learns since a child is the concept of testing a product before using its completed version. Things like “Take care with the temperature of the water, for you to not burn yourself” or “Try it out and say if it is to your taste or not” taught us the importance of testing a product. Testing a product plays an important function during the lifetime of a project, and its management proves to be extremely important, to not lose the control of the bugs or the adjustments made during the development of a project. In this project were addressed two useful tools: *Rational Quality Manager* and *Mantis*. For each of them is described how they function, with emphasis on their differences. In addition, we sought to identify the vision of market professionals about them.

Key words: Testing, Software, Management, RQM, Mantis.

Lista de Figuras

FIGURA 2.1 – PAPÉIS E RESPONSABILIDADES DOS PROFISSIONAIS LIGADOS ÀS ATIVIDADES DE PLANEJAMENTO E CONTROLE.	26
FIGURA 3.1 - TECNOLOGIA JAZZ.	31
FIGURA 3.2 - ALM COLABORATIVO CONECTA ANALISTAS, DESENVOLVEDORES E TESTADORES.	32
FIGURA 3.3 - RELACIONAMENTO DOS ARTEFATOS CLM.	33
FIGURA 3.4 - SELEÇÃO DE SEÇÕES PARA O PLANO DE TESTE	34
FIGURA 3.5 - SEÇÕES DOS CASOS DE TESTES	36
FIGURA 3.6 - EXECUÇÃO DE UM TESTE PASSO A PASSO	38
FIGURA 3.7 - RESULTADO DE EXECUÇÃO DE UM SCRIPT DE TESTE	39
FIGURA 3.8 - RESULTADO DE EXECUÇÃO DE CASO DE TESTE.	40
FIGURA 3.9 - GRÁFICO COM OS TESTES EXECUTADOS E SEUS STATUS.	40
FIGURA 3.10 - CADASTRO DE UM DEFEITO APÓS UMA EXECUÇÃO DE TESTE	41
FIGURA 3.11 - FLUXO DE DEFEITOS.	42
FIGURA 3.12 - LISTA DE RELATÓRIOS DISPONÍVEIS NO RQM.	43
FIGURA 3.13 - RELATÓRIO DE SCORECARD QUE EXIBE ALGUNS DADOS DE TESTES EXIBINDO PESOS DOS TESTES E A QUANTIDADE DE TESTES EXECUTADOS NO TOTAL.	43
FIGURA 4.1 – LOGOTIPO DO MANTIS BUG TRACKER	44
FIGURA 4.2 - PÁGINA COM A VISÃO DO ESTADO ATUAL DOS ERROS REPORTADOS.	46
FIGURA 4.3 - PÁGINA DE FILTRO DOS ERROS REPORTADOS.	47
FIGURA 4.4 - PÁGINA ONDE SÃO REPORTADOS OS ERROS DO PROJETO.	48
FIGURA 4.5 - PÁGINA DE RESUMO DOS ERROS REPORTADOS NO PROJETO.	50

FIGURA 4.6 - PÁGINA PARA IMPRIMIR RELATÓRIOS REFERENTES AOS ERROS DO PROJETO.....	50
FIGURA 4.7 - PÁGINA COM DOS DADOS CADASTRADOS DO USUÁRIO.	51
FIGURA 4.8 - PÁGINA REFERENTE AO CADASTRO DOS TIPOS DE NOTIFICAÇÕES DE ERROS QUE O USUÁRIO IRÁ RECEBER.....	52
FIGURA 4.9 - PÁGINA REFERENTE À EDIÇÃO DOS DADOS QUE O USUÁRIO TERÁ ACESSO.....	53
FIGURA 4.10 - PÁGINA COM INFORMAÇÕES ADICIONAIS SOBRE O AMBIENTE ONDE O PROJETO IRÁ FUNCIONAR.	54
FIGURA 4.11 - PRINCIPAIS TELAS DO MANTIS TOUCH RODANDO EM UM DISPOSITIVO APPLE	55
FIGURA 5.1 – CRIANDO NOVO DEFEITO	58
FIGURA 5.2 – EXECUTANDO UM CASO DE TESTE	58
FIGURA 5.3 – EXECUÇÃO DE UM CASO DE TESTE	59
FIGURA 5.4 – CRIAÇÃO DE UM NOVO DEFEITO DURANTE A EXECUÇÃO DE UM TESTE..	59
FIGURA 5.5 – DETALHES DE UM NOVO DEFEITO.....	60
FIGURA 5.6 – DESCRIÇÃO PARCIALMENTE PRONTA AUTOMATICAMENTE PELO RQM..	60
FIGURA 5.7 – TESTE EXECUTADO E DEFEITO ASSOCIADO CRIADO	61
FIGURA 5.8 – ESTADOS DE UM DEFEITO.....	62
FIGURA 5.9 – HISTÓRICO DE UM DEFEITO	62
FIGURA 5.10 – TELA COM DESCRIÇÃO DO ERRO GERADO PELO RESPONSÁVEL PELO TESTE.....	64
FIGURA 5.11 – TELA DO MANTIS EXIBINDO O STATUS DO ERRO RECÉM-CRIADO	65
FIGURA 5.12 – TELA EXIBINDO AS NOVAS ATRIBUIÇÕES DO ERRO BEM COMO O SEU HISTÓRICO.....	66
FIGURA 5.13 – TELA EXIBINDO O ERRO EM QUE O DESENVOLVEDOR ESTÁ TRABALHANDO	67
FIGURA 5.14 – TELA EXIBINDO AS INFORMAÇÕES SOBRE O ERRO RESOLVIDO PELO DESENVOLVEDOR.....	67
FIGURA 5.15 – TELA EXIBINDO OS ERROS QUE FORAM CORRIGIDOS PELOS DESENVOLVEDORES.....	68
FIGURA 5.16 – TELA EXIBINDO OS ERROS QUE FORAM RESOLVIDOS E RETORNADOS PARA OS SEUS RESPONSÁVEIS PELO TESTE PARA TESTÁ-LOS DE NOVO	69
FIGURA 5.17 – POSSÍVEIS TIPOS DE MENSAGENS QUE PODERÃO SER ENCONTRADAS CASO O ERRO NÃO TENHA SIDO CONSERTADO OU NÃO TENHA SIDO SIMULADO.....	70
FIGURA 5.18 – TELA EXIBINDO OS ERROS QUE NÃO FORAM RESOLVIDOS OU NÃO FORAM SIMULADOS DA FORMA COM QUE FORAM TESTADOS	70

Apêndice

APÊNDICE A	79
APÊNDICE B	85
APÊNDICE C	93
APÊNDICE D	96

Sumário

1. INTRODUÇÃO.....	17
1.1. OBJETIVO	19
2. VISÃO GERAL SOBRE TESTES	20
2.1. INTRODUÇÃO AO TESTE DE SOFTWARE	20
2.2. NECESSIDADE DOS TESTES.....	21
2.3. TESTES E QUALIDADE	21
2.4. OS SETE PRINCÍPIOS DO TESTE.....	22
2.5. FUNDAMENTOS DO PROCESSO DE TESTE.....	24
2.5.1. PLANEJAMENTO E CONTROLE DO TESTE	25
2.5.2. ANÁLISE E MODELAGEM	26
2.5.3. IMPLEMENTAÇÃO E EXECUÇÃO	28
2.5.4. AVALIAÇÃO DOS CRITÉRIOS DE SAÍDA E RELATÓRIOS	28
2.5.5. ATIVIDADES DE ENCERRAMENTO DE TESTES.....	29
3. FERRAMENTA RATIONAL QUALITY MANAGER.....	30
3.1. O QUE É O RQM?	30
3.2. ESTRUTURA.....	31
3.3. PLANO DE TESTE NO RQM	33
3.3.1. DEFINIÇÃO DE ALGUMAS SEÇÕES	34
3.4. CASOS DE TESTES	36
3.5. SCRIPTS DE TESTES.....	37
3.6. CRIANDO OS SCRIPTS DE TESTES.....	37

3.7.	EXECUÇÃO DE TESTES	38
3.8.	REGISTRO DE EXECUÇÃO DE CASO DE TESTE	39
3.9.	CADASTRO DE DEFEITOS	41
3.10.	ACOMPANHAMENTO DE DEFEITOS	41
3.11.	ANÁLISE DE RESULTADOS	42
4.	FERRAMENTA MANTIS BUG TRACKER	44
4.1.	O QUE É O MANTIS?	44
4.2.	COMO SURTIU?	45
4.3.	PRIMEIROS PASSOS COM A FERRAMENTA.....	45
4.4.	SUAS FUNCIONALIDADES	54
5.	COMPARAÇÃO ENTRE AS FERRAMENTAS.....	57
5.1.	CADASTRO DO DEFEITO PELO RQM.....	57
5.2.	CADASTRO DO DEFEITO PELO MANTIS.....	63
6.	VISÃO DE PROFISSIONAIS DA ÁREA SOBRE O RQM E O MANTIS	72
7.	CONSIDERAÇÕES FINAIS	75
8.	REFERÊNCIAS BIBLIOGRÁFICAS	77

1. INTRODUÇÃO

Os produtos de *software* estão cada vez mais presentes na vida das pessoas. Não apenas em aparelhos como celulares e *tablets* como também em eletrodomésticos como micro-ondas e geladeiras que avisam quais alimentos estão acabando e precisam ser comprados. À medida que a necessidade do uso desses sistemas vem aumentando, sua complexidade também aumenta, com o surgimento de novas tecnologias, facilitando o aparecimento de defeitos de *software*. Assim, é imprescindível garantir a sua qualidade para evitar que ocorra algum tipo de problema no usuário final, fazendo-o perder a confiança na empresa fabricante. O *software* antigamente era desenvolvido de acordo com a experiência do desenvolvedor. Hoje em dia o cliente que diz como seu produto deve ser. E eles estão cada vez mais exigentes devido à facilidade de adquirir conhecimento técnico sobre o assunto (GETEC SISTEMAS, 2008).

Para o profissional que irá gerenciar os testes, como o profissional líder de teste¹ de um projeto ou até mesmo o gerente do projeto em questão, ele precisará planejá-los a fim de cobrir a maior parte do sistema de *software*. Para auxiliá-lo, existem inúmeras ferramentas de gerenciamento de teste de *software* que ajudam a organizar todo esse trabalho, como o

¹ Líder de teste - Gerente de teste; Pessoa responsável pelo gerenciamento do projeto, pelas atividades e recursos de teste e por avaliar o objeto de teste. É o indivíduo que dirige, controla, administra, planeja e regula a avaliação de um objeto de teste (GLOSSÁRIO PADRÃO DE TERMOS UTILIZADOS EM TESTE DE SOFTWARE, 2012).

Rational Quality Manager da IBM, o *Mantis Bug Tracker* da Mantisbt, o *Testlink* da Teamtest, o *HP Quality Center* da HP/Mercury Interactive e o *Bugzilla* da Mozilla Foundation. Cada uma destas ferramentas tem características e funcionamento diferentes. Para que possa escolher uma ferramenta que cubra todas as suas necessidades, cabe ao líder de teste analisar o projeto e definir qual a melhor opção dentre as disponíveis no mercado.

São inúmeras as ferramentas existentes no mercado para o auxílio dos testes. Cada uma com características diferentes para suprir as necessidades que o líder de testes precisará para determinado projeto. E assim, persistem dúvidas sobre quais as melhores ferramentas que podem ser usadas nesse momento. Para isso é aconselhável ter conhecimentos de pelo menos uma ferramenta gratuita e uma paga para obter suas características e auxiliar na escolha do usuário.

A ferramenta *Rational Quality Manager* da IBM baseada em *Web* possui como característica conter todas as tarefas para assegurar garantia de qualidade de um sistema de *software* devido ao seu completo e customizável plano de testes. Além disso, pode registrar um defeito para a correção de um desenvolvedor a partir de casos de testes, facilitando sua descrição. Outra característica da ferramenta é a integração com outras ferramentas poderosas de outras áreas, mas do mesmo fabricante, a fim de melhorar a rastreabilidade de artefatos.

Já a ferramenta *Mantis* também baseada em *Web* é um pouco mais simples e direta. Ela tem como principal objetivo a criação e o controle de defeitos e enviar uma notificação aos desenvolvedores do projeto. Outra vantagem é ser uma ferramenta de distribuição gratuita.

Para a realização deste estudo, as ferramentas foram utilizadas diariamente no mercado de trabalho no período de pelo menos um ano com auxílio de profissionais e especialistas, sendo analisadas por um período de três meses.

As diferenças entre o *Mantis* e o *Rational Quality Manager* são muitas e encontrar um lugar que faça uma comparação entre elas é uma tarefa árdua, fazendo o líder de teste gastar horas para a pesquisa.

1.1. OBJETIVO

Este trabalho tem como objetivo fazer uma análise comparativa sobre as características, funcionamento e utilidades das ferramentas *Rational Quality Manager* e *Mantis*.

Fazem parte dos objetivos específicos do projeto:

- Realizar uma revisão da literatura sobre os fundamentos básicos de testes de *software*;
- Mostrar como cada uma das ferramentas cadastra um mesmo defeito;
- Seguir o fluxo de cada ferramenta até a completa correção ou rejeição de um erro;
- Realizar pesquisa de mercado avaliando cada uma das ferramentas.

2. VISÃO GERAL SOBRE TESTES

Este capítulo traz alguns dos fundamentos básicos de testes de *software*, abordando uma visão abrangente sobre testes em um projeto, bem como sua importância no dia a dia do desenvolvimento de produtos de *software* desde sua fase inicial até a final.

2.1. INTRODUÇÃO AO TESTE DE SOFTWARE

Houve na história inúmeros casos em que a falta de testes adequados levaram a perdas irreparáveis como o do programa *Mars Surveyor '98*, que causou a destruição de uma sonda na hora do pouso em Marte. Os computadores na Terra enviavam os dados em unidades inglesas e o da nave usava o sistema métrico (MARS CLIMATE ORBITER, 2013). Este problema seria simplesmente resolvido se houvessem realizado testes de integração entre os dois sistemas. A fim de evitar essas e outras inconveniências, é necessário o investimento em testes durante todo o processo de desenvolvimento do *software* a fim de minimizar qualquer tipo de defeito futuro.

2.2. NECESSIDADE DOS TESTES

O ser humano está sujeito a cometer um erro (engano) que pode gerar um defeito (*bug*, falha) tanto em um documento quanto no código ou sistema. Caso um defeito no código seja executado, o sistema falhará ao tentar reproduzir uma determinada ação ou então reproduzirá outra não esperada, causando uma falha. Ou seja, defeitos resultam em falhas, porém nem todos eles causam falhas.

Há inúmeros fatores que podem causar falhas pelos seres humanos como pressão no prazo de entrega do produto, códigos complexos, problemas na infraestrutura, mudanças de tecnologia. Assim como também podem ser causados por fatores externos, tais como, radiação, magnetismo, poluição e campos eletrônicos que interferem em *software* embarcado ou influenciar a execução do sistema pela alteração das condições de *hardware* (FOUNDATION LEVEL SYLLABUS, 2011).

Assim, apesar desses percalços, o cliente necessita do *software* funcionando da melhor maneira possível.

2.3. TESTES E QUALIDADE

De acordo com o dicionário *Michaelis*, testar significa “*Submeter a teste; experimentar, pôr à prova*” (WEISZFLOG, 2012). Isto é o que é feito em testes de *software*, através de simulações de como o sistema deveria se comportar num ambiente real no qual ele deverá ser executado. Os testes devem se iniciar o mais cedo possível, desde a primeira fase de desenvolvimento de um projeto e não apenas na final como muitos pensam. Quanto mais cedo um problema é identificado, menor é o custo para a sua correção porque um defeito não identificado no início pode acarretar vários outros defeitos de difícil correção, obrigando o desenvolvedor a refazer boa parte do código, tendo assim um retrabalho. Ou seja, aumentando em muito o custo e a complexidade do projeto.

Contudo há atividades de teste antes e depois da fase de execução, como planejamento e controle, modelagem dos casos de testes, avaliação do critério de conclusão, checagem de resultados, geração de relatórios sobre a execução dos testes, revisão de documentos e de códigos e testes dinâmicos (testes com o sistema sendo executado) e estáticos (testes no código a fim de cobrir todos os caminhos possíveis), por exemplo, (FOUNDATION LEVEL SYLLABUS, 2011). Verificações não precisam ser feitas apenas quando existe código. Podem ser usadas técnicas de inspeções, que visam identificar inconsistências em documentos, modelagens ou em outros requisitos e com isso, evitando que algo seja desenvolvido erroneamente (MURTA, 2012).

Testes dinâmicos e estáticos podem ser usados para obter objetivos similares e informações de como melhorar o sistema e o processo de teste. Porém os testes podem ter objetivos diferentes, como encontrar defeitos, descobrir o nível de qualidade de um sistema, prover informações para tomar decisões no desenvolvimento do *software* ou para prevenir defeitos.

Com os resultados de teste, é possível medir a qualidade de um sistema pela quantidade de defeitos encontrados ou por características e requisitos funcionais e não funcionais. Encontrados pouco ou nenhum defeito, pode demonstrar confiança na sua qualidade caso o teste tenha sido projetado e executado adequadamente. Entretanto caso apareça um número de defeito alto de acordo com o tamanho do sistema, aumenta o nível de risco e conseqüentemente a qualidade diminui. Assim, quando os *bugs* são corrigidos, a qualidade aumenta (FOUNDATION LEVEL SYLLABUS, 2011).

2.4. OS SETE PRINCÍPIOS DO TESTE

Alguns dos sete princípios do teste foram sendo apresentados ao longo dos últimos 40 anos. Os princípios são como um guia geral bastante usado atualmente para o processo de teste como um todo, definido pelo livro “*Syllabus Foundation Level*” (FOUNDATION LEVEL SYLLABUS, 2011):

a) – Teste demonstra a presença de defeitos

O teste pode mostrar que o defeito existe. Porém não prova que em um sistema não exista nenhum defeito. Os testes reduzem a probabilidade de que os defeitos permaneçam, mas mesmo não encontrando nenhum, não prova que eles não existam como mostra o princípio de que o teste exaustivo é impossível.

b) – Teste exaustivo é impossível

Testar todas as combinações de entradas, condições e desvios do código é impossível. Por exemplo, se em um código de um sistema tivesse um *loop* variando de 1 a 100, deveriam ser feitos 100 testes para avaliar todas as saídas. Porém, havendo outro *loop* também variando de 1 a 100, o número de testes necessários aumentaria de 100 para 100 elevado a 100 testes. Assim seria possível apenas se estivesse disponível uma enorme quantidade de tempo para realizar todos eles, coisa que não é comum em projetos no mercado de trabalho. Então, em vez de realizar um teste exaustivo, são levados em consideração riscos e prioridades para focar nos testes.

c) – Teste antecipado

Os testes devem começar o mais cedo possível durante o desenvolvimento de um sistema e devem ser focados em objetivos definidos.

d) – Agrupamento de defeitos

Um número pequeno de módulos contém a maior parte de defeitos descobertos durante os testes ou exibe a maior parte das falhas.

e) – Paradoxo do Pesticida

Após a repetição de um mesmo tipo de teste inúmeras vezes, pode ocorrer de não serem encontrados erros novos após um determinado momento. Para contrariar este princípio, os casos de testes devem ser frequentemente revisados e atualizados a fim de obter diferentes comportamentos. E assim, aumentar a cobertura de testes e a possibilidade de identificar mais defeitos.

f) – Teste depende do contexto

Os testes são realizados de diferentes formas de acordo com o contexto. Por exemplo, jogos são testados diferentemente de um *software* para um banco, já que num jogo pode-se focar melhor na parte de jogabilidade dele enquanto em um sistema de um banco deve-se priorizar a parte de segurança das informações.

g) – A ilusão da ausência de erros

O sistema pode estar livre de defeitos, porém ele não estará correto se o que foi construído foi completamente oposto aos anseios do cliente.

2.5. FUNDAMENTOS DO PROCESSO DE TESTE

O processo de teste é composto de etapas, atividades, artefatos, papéis e responsabilidades que buscam a padronização dos trabalhos e maior organização e controle dos projetos de testes. Como qualquer outro processo, o processo de teste deve ser revisto constantemente de forma de aumentar sua área de cobertura e possibilitar aos profissionais uma maior visão e organização dos seus trabalhos, gerando um maior controle e agilidade operacional dos projetos de testes (BARTIE, 2007).

O processo de teste básico é composto pelas seguintes atividades:

- Planejamento e controle
- Análise e modelagem
- Implementação e execução
- Avaliação dos critérios de saída e relatórios
- Atividades de encerramento de teste

Apesar de normalmente serem apresentadas e executadas sequencialmente, as etapas podem sobrepor-se ou ocorrer paralelamente. Depende da forma como o teste é executado no contexto do sistema e do projeto (FOUNDATION LEVEL SYLLABUS, 2011).

2.5.1. PLANEJAMENTO E CONTROLE DO TESTE

Esta etapa caracteriza-se pela especificação das atividades de teste e objetivos a serem realizados durante todo o processo de desenvolvimento do *software*. Se baseia nos anseios do cliente em relação a custos, prazos, qualidade esperada e escopo do teste (BARTIE, 2007). Com estas informações é possível definir o tamanho da equipe e estabelecer um esforço durante todo o projeto.

O controle de teste é a atividade constante que visa comparar a evolução atual com a que foi planejada, reportando o status e as diferenças entre eles durante todo o projeto para obter o controle efetivo. Além disso, faz parte do controle a definição de novos planos e estratégias para atingir a missão e o objetivo do projeto.

Estas atividades são desempenhadas pelo coordenador de testes, tendo a colaboração de outros membros da equipe, como o líder de testes e o arquiteto de testes (BARTIE, 2007). Algumas responsabilidades desta etapa e o papel de cada um dos atores são representados na Figura 2.1.

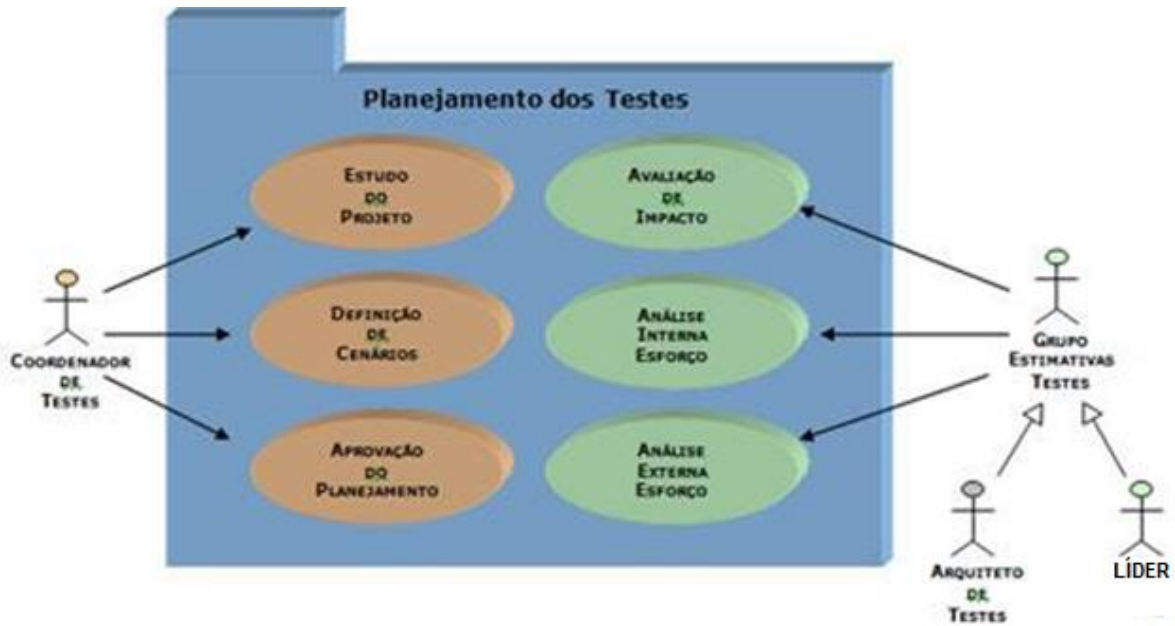


FIGURA 2.1 – PAPÉIS E RESPONSABILIDADES DOS PROFISSIONAIS LIGADOS ÀS ATIVIDADES DE PLANEJAMENTO E CONTROLE (BARTIE, 2007).

2.5.2. ANÁLISE E MODELAGEM

A análise e a modelagem de teste são atividades onde objetivos gerais do teste são transformados em condições e modelos tangíveis (FOUNDATION LEVEL SYLLABUS, 2011). São atividades que estão sujeitas a sofrer alterações ao longo do projeto e assim, deve-se sempre ficar atento as possíveis mudanças nos requisitos para que a modelagem possa estar de acordo com os desejos do cliente.

Esta etapa compõe-se de diversas atividades. Uma delas é revisar a base de teste² como requisitos, nível de integridade do sistema, a arquitetura, interface, dentre outros itens. Assim, uma inconsistência logo encontrada na base, evitaria inúmeros problemas futuros não só na implementação como também na fase de teste, já que evitaria um reporte de defeito e o controle do seu *status*.

Outra etapa é avaliar a testabilidade dos requisitos e do sistema, pois dependendo do projeto, o teste não pode ser tão simples o quanto parece. Um teste de um sistema de controle

² Base de teste - Documentação na qual os casos de testes estão baseados (GLOSSÁRIO PADRÃO DE TERMOS UTILIZADOS EM TESTE DE SOFTWARE, 2012).

de profundidade de submarino pode ser realizado em um simples navegador *web* ou pode ser feito dentro de um submarino real, no fundo do oceano. Esta etapa ajuda justamente a analisar todas as possibilidades e a adequação de cada uma delas ao projeto, de acordo com as metas definidas na etapa de planejamento.

Identificar e priorizar as condições ou requisitos de testes e dos dados que originam, na especificação, no comportamento e na estrutura, faz parte de outra atividade: análise e modelagem. Com isso, o coordenador de teste pode executar os testes prioritários caso haja necessidade de diminuir o custo ou esteja enfrentando problemas com o prazo.

Uma atividade importante nesta fase é projetar e priorizar casos de testes de alto nível. Caso de teste é um conjunto de etapas, condições e passos a serem seguidos a fim de realizar um teste específico. São elaborados para identificar defeitos no *software* através de determinadas situações (CASO DE TESTE, 2013).

Um bom caso de teste precisa ter alta probabilidade de encontrar erros. Para isso, conhecer o produto e explorar aspectos diferenciados é fundamental. É necessário estabelecer claramente o propósito de cada teste evitando redundâncias e não ser demasiadamente complexo, pois pode gerar confusão na hora da execução. Para isso, decompor os testes de forma que cada um foque em somente um objetivo é o suficiente (MURTA, 2012).

Identificar as necessidades de dados para teste suportando as condições e casos de teste e planejar a preparação do ambiente de teste e identificar a infraestrutura e ferramentas necessárias auxiliam na execução dos testes e deixando-os mais próximos do mundo real em que o sistema será utilizado.

Por fim, criar rastreabilidade bidirecional entre os requisitos e os casos de teste auxilia bastante caso seja necessária alguma modificação. Isto porque um caso de teste desatualizado pode fazer com que o testador cadastre defeitos que serão rejeitados pelos desenvolvedores, que é uma prática ruim para o profissional de teste.

2.5.3. IMPLEMENTAÇÃO E EXECUÇÃO

A implementação e a execução do teste é a atividade onde os *scripts* de teste e seus procedimentos são feitos para compor os casos de teste a fim de cobrir certa parte de um sistema. Esta fase de teste visa finalizar, implementar e priorizar os casos de teste, criar os dados a serem testados, preparar o ambiente para teste, criar suítes de teste a partir dos casos de teste, verificar e atualizar a rastreabilidade bidirecional entre a base de teste e os casos de teste. Após essas atividades de implementação, ocorre à execução que visa executar os casos de teste manualmente ou com alguma ferramentas de apoio, registrar os resultados de execução, comparar os resultados obtidos com os esperados, reportar os problemas encontrados e por fim repetir os testes que falharam após a correção dos defeitos a fim de se certificar que os *bugs* encontrados não foram replicados e que as correções não geraram outros defeitos.

2.5.4. AVALIAÇÃO DOS CRITÉRIOS DE SAÍDA E RELATÓRIOS

Avaliação do critério de saída é a atividade onde as execuções dos testes são avaliadas, buscando identificar se atingiram os objetivos definidos, sendo feita para cada etapa de teste. Suas atividades envolvem checar os registros de teste (*logs*) de acordo com o critério de encerramento proposto na fase de planejamento dos testes, avaliar se são necessários testes adicionais ou se o critério de saída especificado anteriormente deve ser alterado e por fim elaborar um relatório dos testes realizados na etapa para os interessados (*stakeholders*).

2.5.5. ATIVIDADES DE ENCERRAMENTO DE TESTES

Na atividade de encerramento de teste, são coletados todos os dados de todas as atividades para consolidar a experiência, fatos e números. Essas atividades são: conferir quais produtos planejados foram entregues ao cliente; concluir o relatório de defeitos ou exibir os registros de mudanças ainda em aberto; documentar o aceite do sistema; finalizar e arquivar o ambiente de teste e seus componentes; analisar o que foi aprendido para se determinar mudanças necessárias para futuros projetos e *releases* e, por fim, utilizar as informações coletadas para melhorar a maturidade de teste.

3. FERRAMENTA RATIONAL QUALITY MANAGER

Este capítulo traz os fundamentos e funcionalidades do *Rational Quality Manager*, mostrando o processo desde a criação do defeito até o seu encerramento com o *status* de corrigido ou não.

3.1. O QUE É O RQM?

O *Rational Quality Manager (RQM)* é uma ferramenta da *IBM* de gerenciamento de qualidade de *software* baseada em *Web* que auxilia todo o processo de testes durante as fases de desenvolvimento de um sistema. Nela podem ser definidos os planos, casos, suítes e configuração de execução de teste. A ferramenta oferece também auxílio para criação e execução de *scripts* de testes manuais e integração com outras ferramentas voltadas à criação e execução de *scripts* de testes automatizados. Com ela pode-se ter um suporte de cadastramento e acompanhamentos de *bugs* e associá-los a requisitos e artefatos de desenvolvimento e assim obter uma boa rastreabilidade (Rational Quality Manager, 2012).

A ferramenta contém um controle de permissões para cada tipo de usuário definida pelo gerente de teste atribuindo a cada um seus papéis. O gerente, por exemplo, pode ter o

acesso total às funcionalidades, o analista de teste pode ter permissão para criação de planos e casos de testes enquanto o testador encarregado pode ter acesso apenas a executar os testes (ESPINHA, 2012).

3.2. ESTRUTURA

Criado com base na plataforma do *IBM Rational® Jazz™* (Figura 3.1), foi projetado para ser utilizado tanto por pequenas, médias e grandes equipes. Pode ser acessado via interface *web* ou diretamente do *IDE Eclipse (client do Rational Team Concert)*.

Esta plataforma é baseada em colaboração para permitir uma evolução do desenvolvimento de forma mais produtiva e transparente. Ela favorece não apenas a colaboração de profissionais de *software*, e sim de todas as pessoas relevantes envolvidas na entrega do *software*. Por fim, permite a integração das ferramentas *Rational Team Concert*, *Rational Requirements Composer* e a *Rational Quality Manager* (ESPINHA, 2012).

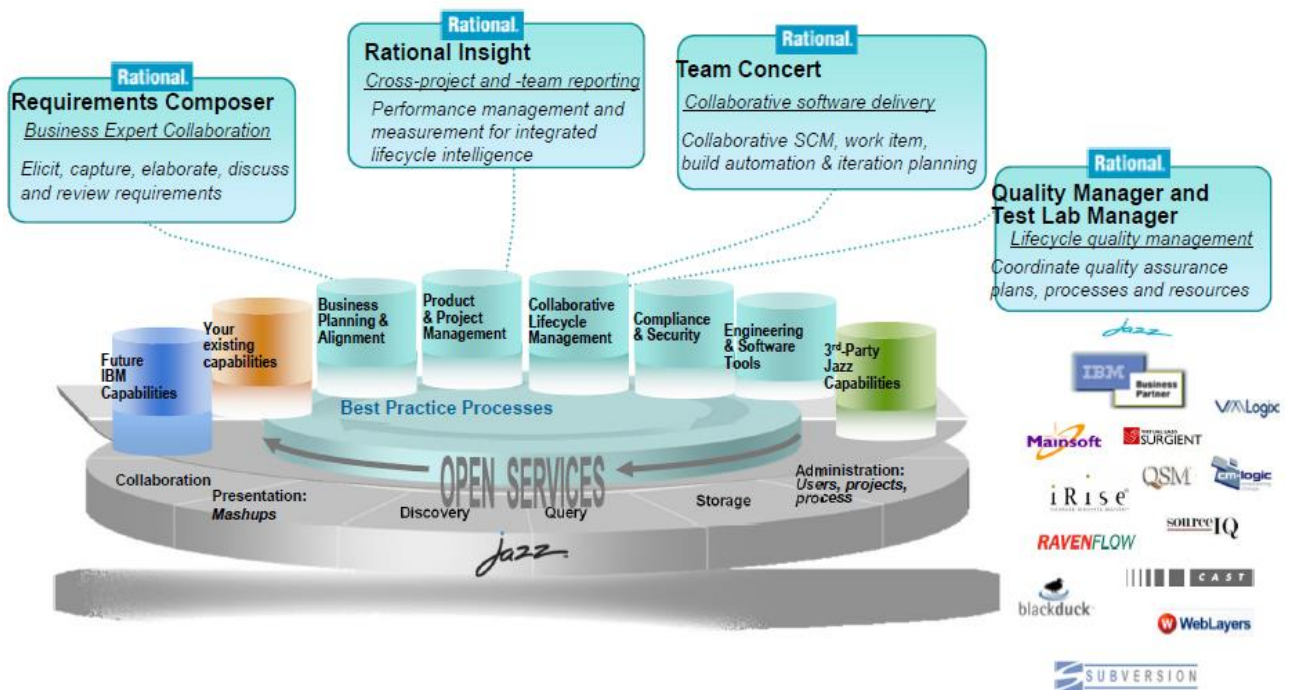


FIGURA 3.1 - TECNOLOGIA JAZZ (ESPINHA, 2012).

A integração ALM Colaborativo (*Collaborative Application Lifecycle Management*) permite a fácil conexão entre o trabalho de análise com as equipes de desenvolvimento e teste, melhor visualizado na figura 3.2. Os links dos artefatos favorecem a rastreabilidade, sua revisão, comentários, acompanhamento de *status* em repositórios de projeto. Com eles é possível visualizar artefatos das outras ferramentas integradas apenas passando o mouse por cima do *link*, facilitando e tornando ágil todo o processo.

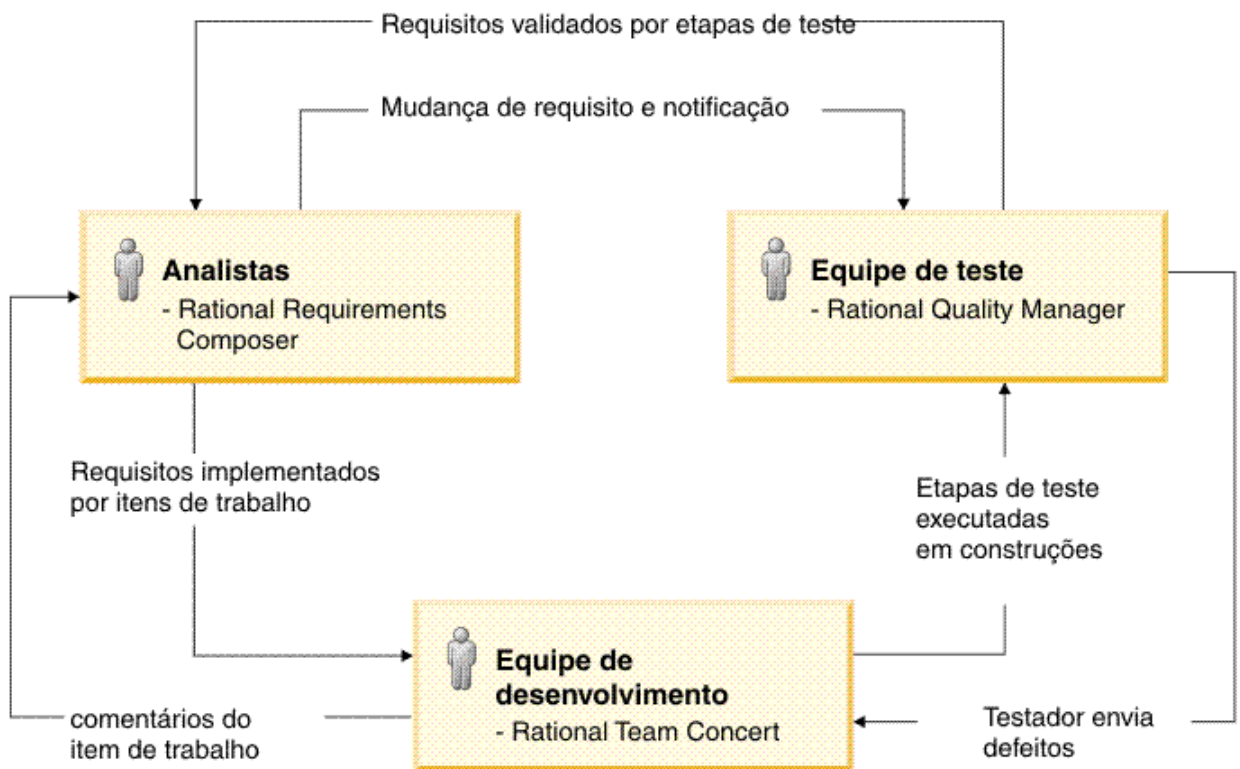


FIGURA 3.2 - ALM COLABORATIVO CONECTA ANALISTAS, DESENVOLVEDORES E TESTADORES.

A ferramenta possui inúmeras vantagens como: criação de planos e casos de testes completos e editável; controle de acesso de usuários; integração com outras ferramentas *IBM* favorecendo a rastreabilidade; compatibilidade com qualquer sistema operacional por sua arquitetura em *web*; geração de relatórios; controle de defeitos e controle e edição do fluxo de defeitos. Já as desvantagens são: Necessidade de um servidor potente, dificuldade para configurar os relatórios dinâmicos do espaço de trabalho (*dashboard*), ferramenta paga e para integrar as outras ferramentas, é necessária a compra de todas elas.

3.3. PLANO DE TESTE NO RQM

O plano de teste agrupa os casos de teste relacionados para serem executados em conjunto. Na figura 3.3 é exibido como é feito o relacionamento dos artefatos do *Rational Quality Manager*. O plano de teste pode ser associado a um suíte de teste³, que por sua vez tem um registro de execução que gera um resultado. O caso de teste possui um conjunto de *scripts* de testes que possuem um passo a passo a seguir para que o caso seja executado. Por fim, o caso de teste gera um registro de execução de testes que fornece um resultado e por sua vez, pode ser associado a um defeito caso haja alguma falha.

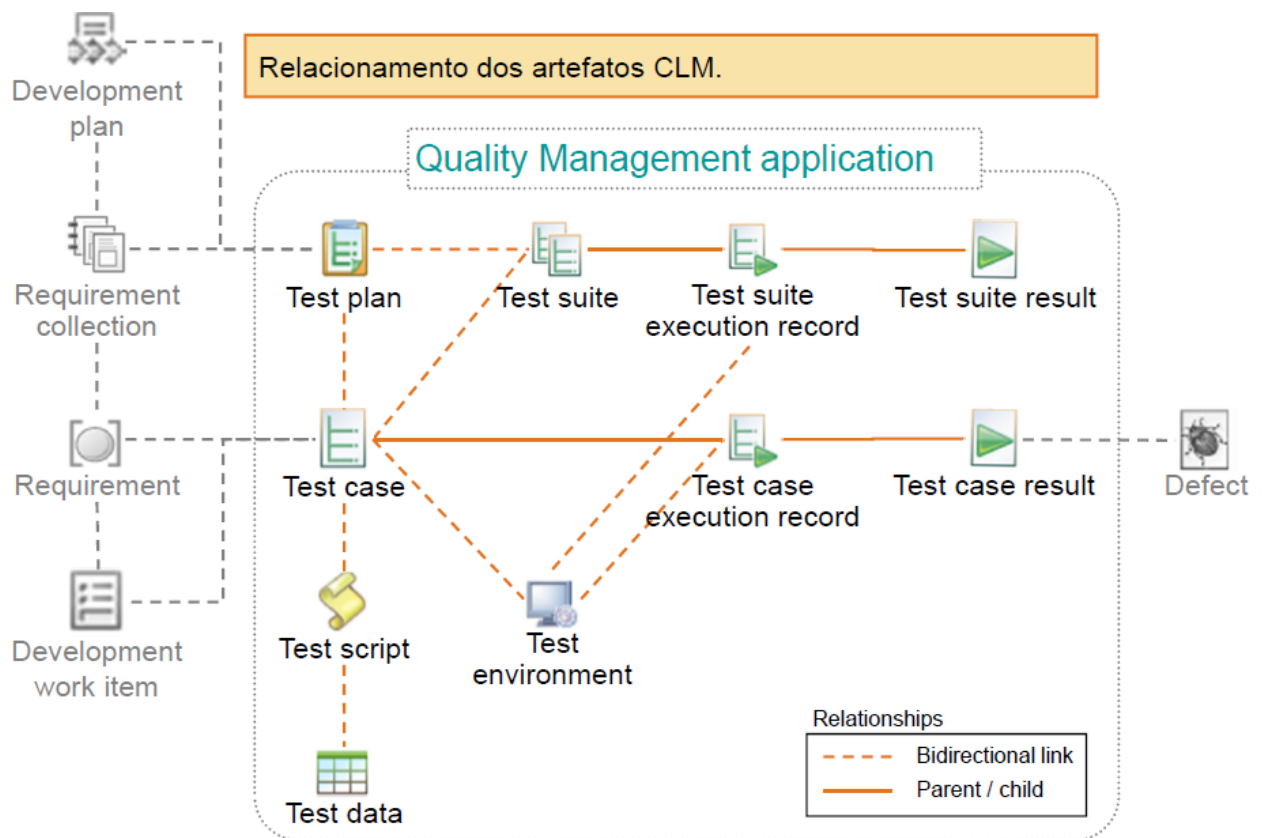


FIGURA 3.3 - RELACIONAMENTO DOS ARTEFATOS CLM (ESPINHA, 2012).

³ Suíte de teste - Conjunto de vários casos de teste para um componente ou sistema sendo testado, no qual a pós-condição de um teste é frequentemente utilizada como pré-condição para o próximo (GLOSSÁRIO PADRÃO DE TERMOS UTILIZADOS EM TESTE DE SOFTWARE, 2012).

Para criação do plano de teste, é recomendado criar anteriormente um modelo de plano contendo as seções dos planos que variam de acordo com o projeto ou com a empresa que o utilizará, já que o *RQM* fornece várias seções para auxiliar nos testes, como na figura 3.4. Porém vale ressaltar que utilizar todas estas seções seria bastante custoso para um projeto que contivesse um número muito grande de planos de testes diferentes.

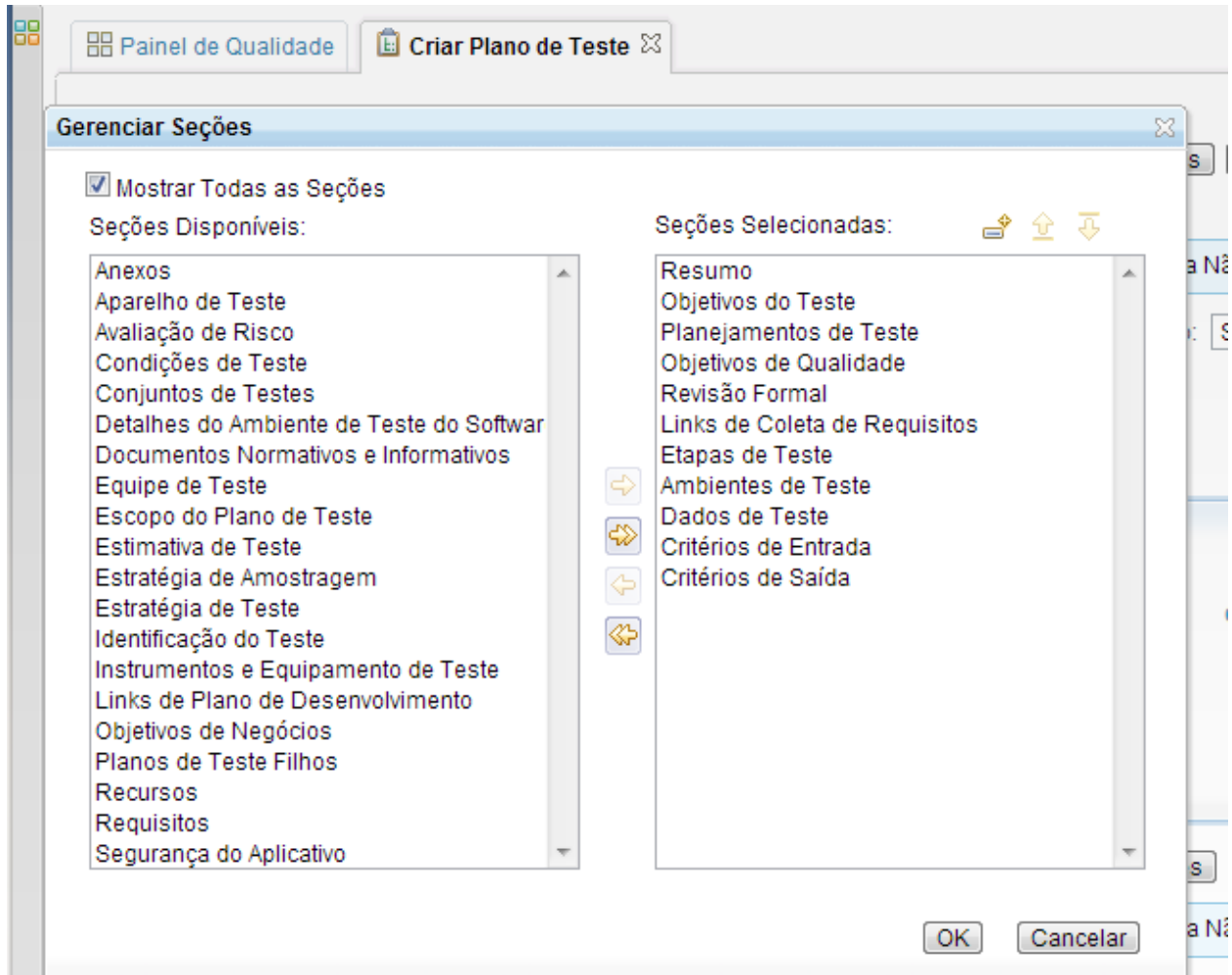


FIGURA 3.4 - SELEÇÃO DE SEÇÕES PARA O PLANO DE TESTE

3.3.1. DEFINIÇÃO DE ALGUMAS SEÇÕES

Todas as seções possuem uma opção de criação de tarefas para a realização do preenchimento de cada seção, permitindo associá-las a uma pessoa específica.

- a) Resumo – Contém a visão geral do plano de teste. Nele podem ser criadas categorias a serem selecionadas para ser seu tipo de teste.

- b) Escopo – Determina o que será testado no plano atual

- c) Objetivos – São estipuladas as metas de execução geral e a estratégia para a fase do teste. Fornece quais características serão testadas.

- d) Links de Coleta de Requisitos – Podem ser criadas ou selecionadas uma coleção de requisitos do *Rational Requirements Composer* associados ao plano de teste e assim auxiliando na rastreabilidade

- e) Ambientes de Teste – Lista os ambientes suportados e testados pelo plano de teste. Podem ser inclusas plataformas como navegadores, sistemas operacionais, bancos de dados, dentre outras.

- f) Equipe de Teste – Especifica que equipe ficará responsável pela execução do plano de teste

- g) Planejamento de Teste – Define os marcos de teste para o plano.

- h) Estimativa de Teste – Planeja o esforço a ser realizado pela equipe para planejar e executar os testes

- i) Etapas de teste – A seção mais importante, pois é aonde são associados os casos de testes para o plano.

3.4. CASOS DE TESTES

A estrutura do caso de teste no *RQM* é um pouco parecida com a do plano de teste. Também é dividida por seções (figura 3.5), pode ser criada uma tarefa para cada e pode ser feita a partir de um modelo pré-definido.

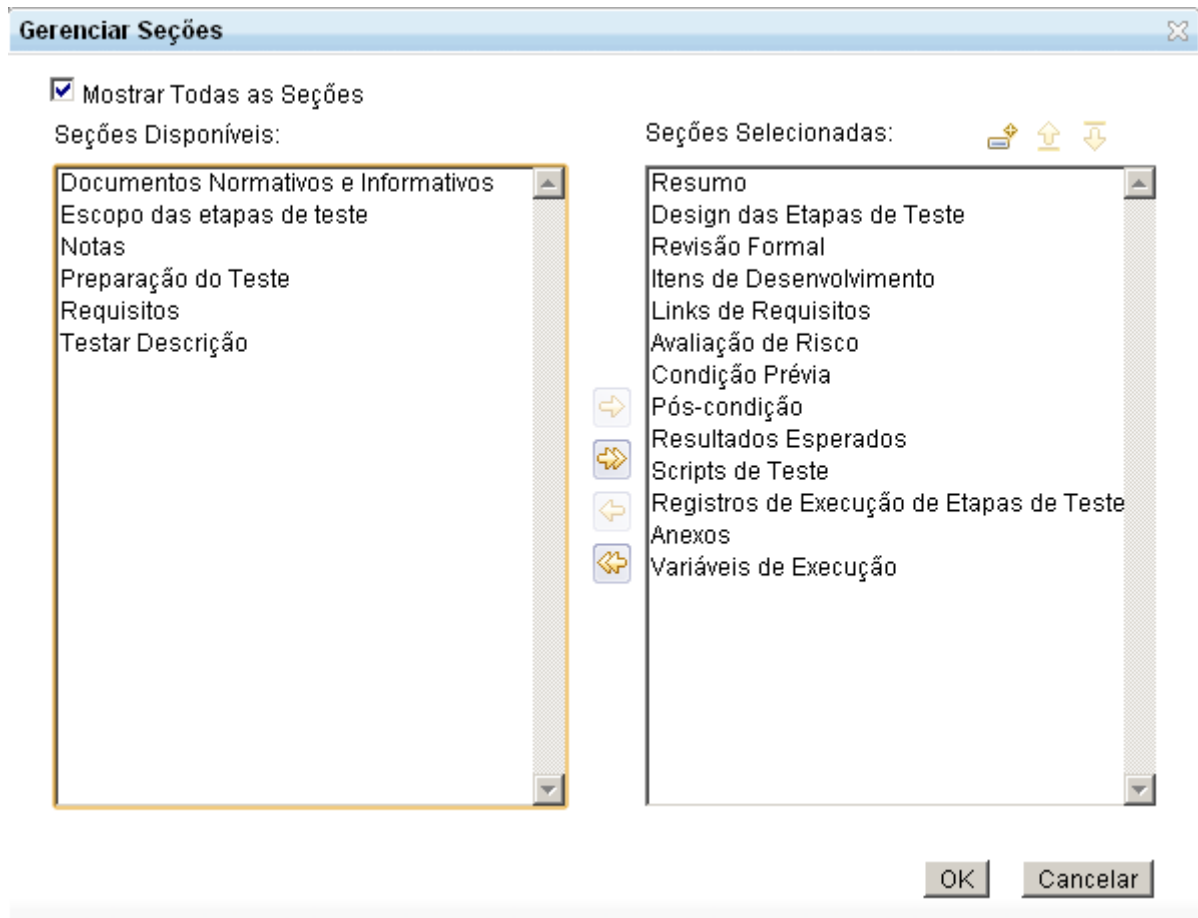


FIGURA 3.5 - SEÇÕES DOS CASOS DE TESTES

Para criar um novo caso de teste, há duas maneiras. Uma delas é pelo *menu* “*Construção>Criar caso de teste*”, porém este não ficaria associado a um plano de teste, tendo esta ligação que ser feita posteriormente. Outro método normalmente usado é criar o teste dentro do plano de teste a que irá pertencer e assim, seria possível criar todos os casos de

teste apenas com o título e sua descrição para posteriormente serem definidas tarefas para o analista de teste desenvolvê-los.

Há inúmeras seções disponíveis para serem utilizadas no caso de teste, porém a principal é a de *Scripts* de Testes, pois é onde ficam os testes e seus respectivos passo a passo.

3.5. SCRIPTS DE TESTES

Como já citado anteriormente, a seção de *Script* de Testes é onde ficam apontados os scripts manuais dos testes e, dentro deles, seu passo a passo. O mesmo *script* pode ser apontado por vários casos de testes diferentes e, por isso, ficam armazenados em outra região específica e não dentro de cada caso de teste, facilitando sua manutenção. Assim, os *scripts* poderiam ficar dentro dos casos de testes completos, dos testes de fumaça (ou *smoke*), de regressão e também de integração, por exemplo. E quando fosse necessário alterá-lo, seria preciso apenas alterar em um lugar.

3.6. CRIANDO OS SCRIPTS DE TESTES

Há três maneiras para criar os *scripts* de testes. Uma delas é pelo menu "*Construção > Criar Script de Teste*". Assim, o usuário pode preencher todos os campos e deixar todo o *script* já pronto. Entretanto seria preciso associá-lo a um caso de teste posteriormente. A segunda maneira é dentro do Caso de Teste na seção *Script* de Testes clicando no botão de "*Incluir novo script de teste*". Assim, ele já fica armazenado dentro do caso de teste e disponível para futuras associações em outros casos de testes. Deste modo, o analista apenas escolhe inicialmente o "Nome", a "Descrição" e o "Tipo" e pode criar vários scripts de uma vez só, deixando o detalhamento para uma etapa posterior ou delegar à outra pessoa a tarefa. Uma terceira forma de criar é também dentro do caso de teste, porém clicando o botão "*Criar script de teste manual a partir do design de etapas de teste*". Pelo próprio nome já se pode ter uma noção de como ele funciona. O *RQM* cria um *script* de teste com os passos já criados

definidos na seção "Design de Etapas de Teste", facilitando na criação de scripts que tenham muitos passos parecidos.

3.7. EXECUÇÃO DE TESTES

Após a criação dos *scripts* de testes, eles já podem ser executados. Basta, dentro do caso de teste, clicar no botão com a seta verde "Executar caso de teste". Após selecionar o registro, plano, marco, ambiente e o *script*, pode-se optar por criar o registro executando o passo a passo ou então, sem ele, selecionando apenas o veredicto final da execução. Escolhendo executar passo a passo, o testador marca cada um deles se passou ou não, como é exibida na figura 3.6. Além disso, ainda pode ser descrito pelo testador o resultado real obtido na execução do teste, caso seja diferente do resultado esperado definido no caso de teste.

The screenshot shows a test execution window titled "Cópia de OP20 - Manter Posição de Inspeção Execução". It includes a progress bar at 75% and a table of test steps. The table has four columns: "Etapa", "Resultado/Descrição", "Resultados Esperados", and "Resultados Reais".

Etapa	Resultado/Descrição	Resultados Esperados	Resultados Reais
1	Logar no sistema >> Escolher a instalação >> Ir em Operação >> Posições de Inspeção	Sistema habilita uma tela de filtro composto por campos que poderão ser informados pelo ator para serem utilizados como critérios de consulta, de acordo com (ED1).	
2	Preencher os campos do filtro	Sistema apresenta a relação de Posições de Inspeção cadastrados no sistema que atendam aos campos informados e visualiza a relação de Posições de Inspeção existentes e opta entre: incluir, alterar ou excluir	
3	Clicar em Excluir Posição de Inspeção	Sistema solicita a confirmação da exclusão.	Sistema não solicitou a confirmação de exclusão
4	Confirmar a exclusão.	Sistema exclui a Posição de Inspeção.	

FIGURA 3.6 - EXECUÇÃO DE UM TESTE PASSO A PASSO

Logo após a execução, o *RQM* gera um relatório de execução do teste como na figura 3.7 mostrando quantos e quais passos passaram ou não. Esses relatórios podem ser reexibidos posteriormente, a qualquer fase do projeto.

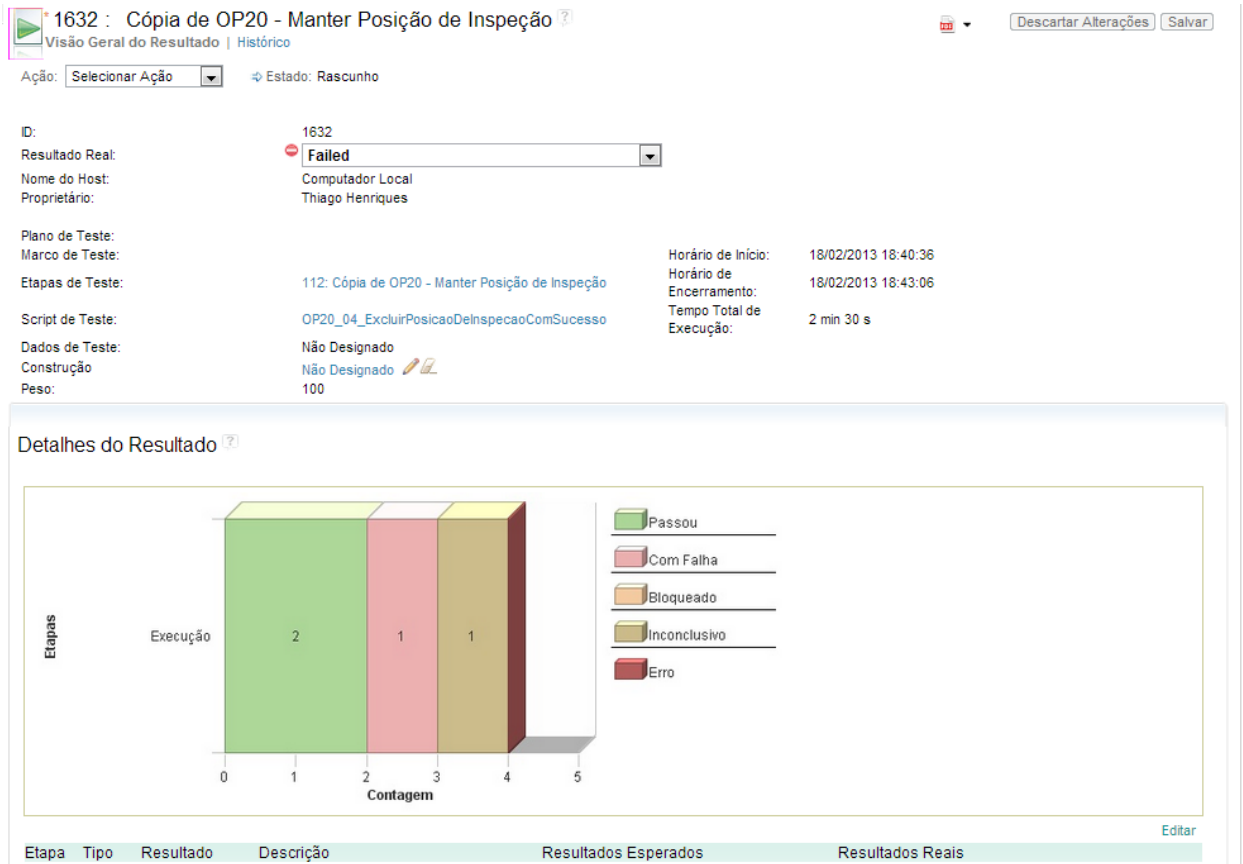


FIGURA 3.7 - RESULTADO DE EXECUÇÃO DE UM SCRIPT DE TESTE

3.8. REGISTRO DE EXECUÇÃO DE CASO DE TESTE

A seção "*Registros de Execução de Etapas de Teste*" contém os registros das execuções dos testes manuais. Nela fica armazenada a informação sobre a qual plano um teste específico pertence, em qual marco, o ambiente e a máquina em que foi executado (figura 3.8). Assim, o gerente de teste fica sabendo quais testes passaram ou não em cada etapa de teste.

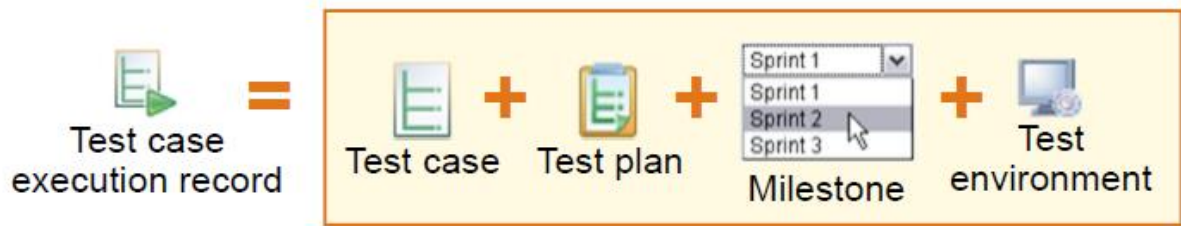


FIGURA 3.8 - RESULTADO DE EXECUÇÃO DE CASO DE TESTE (ESPINHA, 2012).

Com esses dados podem ser gerados relatórios que ficam a mostra no *Dashboard* (figura 3.9), facilitando o acompanhamento dos testes na própria página inicial do *RQM*, sem que o gerente de teste tenha que solicitar a geração de um novo relatório.

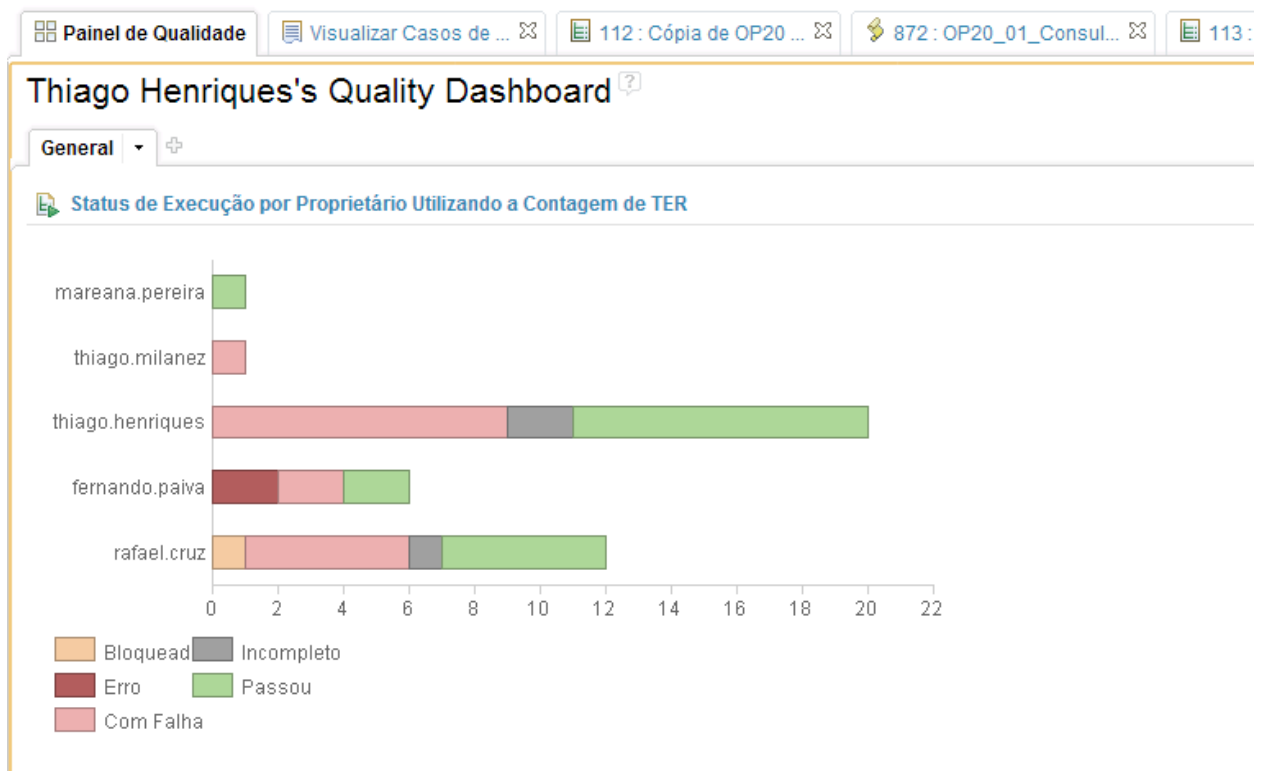


FIGURA 3.9 - GRÁFICO COM OS TESTES EXECUTADOS E SEUS STATUS.

3.9. CADASTRO DE DEFEITOS

Os defeitos cadastrados ficam todos armazenados no RTC. Ele se comporta como se fosse uma tarefa para o desenvolvedor, que pode inclusive registrar as horas que utilizou durante a correção dos defeitos.

Para registrar um defeito no *RQM*, o analista pode ir pelo *menu "Defeitos>Criar Defeito"*. Contudo seria necessária a associação do defeito com alguns *links* como a uma tarefa pai, a um plano de teste e a um requisito para ajudar na rastreabilidade. Outra maneira para criá-lo seria durante a execução manual de um caso de teste e caso seja encontrado um erro, bastaria clicar no ícone de "*Criar novo defeito*" ou durante o passo a passo ou após a execução dele dentro do item "*Defeitos*" mostrado na Figura 3.10. Após preencher os campos detalhando o defeito e clicando em "*OK*" o *RQM* já associa o defeito aos *links* associados ao caso de teste executado, como o caso de teste e o requisito e assim, economizando tempo.

The screenshot shows the RQM interface with the following details:

- Defeitos** (Defects) section is active.
- ID:** 1632
- Resultado Real:** Passed
- Nome do Host:** Computador Local
- Proprietário:** Thiago Henriques
- Plano de Teste:** 112: Cópia de OP20 - Manter Posição de Inspeção
- Horário de Início:** 18/02/2013 18:40:36
- Horário de Encerramento:** 18/02/2013 18:43:06
- Tempo Total de Execução:** 2 min 30 s
- Script de Teste:** OP20_04_ExcluirPosicaoDeInspecaoComSucesso
- Dados de Teste:** Não Designado
- Construção:** Não Designado
- Peso:** 100

The table below shows no defects found:

Status de Bloqueio	ID	Resumo	Estado
Nenhum item localizado.			

FIGURA 3.10 - CADASTRO DE UM DEFEITO APÓS UMA EXECUÇÃO DE TESTE

3.10. ACOMPANHAMENTO DE DEFEITOS

Ao registrar um defeito, ele é carregado com o estado "Novo". Quando o desenvolvedor encarregado na correção do *bug* o pega para corrigi-lo, o estado é alterado para

"Em Andamento". Caso não seja possível a correção por algum motivo, ele pode voltar para o estado "Novo". Porém se foi corrigido ou se é inválido, o defeito vai para "Resolvido". Neste estado, o testador confere se ele realmente foi corrigido ou inválido. Caso verdade, ele vai para "Verificado". Senão volta para "Novo" voltando novamente para ser analisado pelo desenvolvedor (figura 3.11).

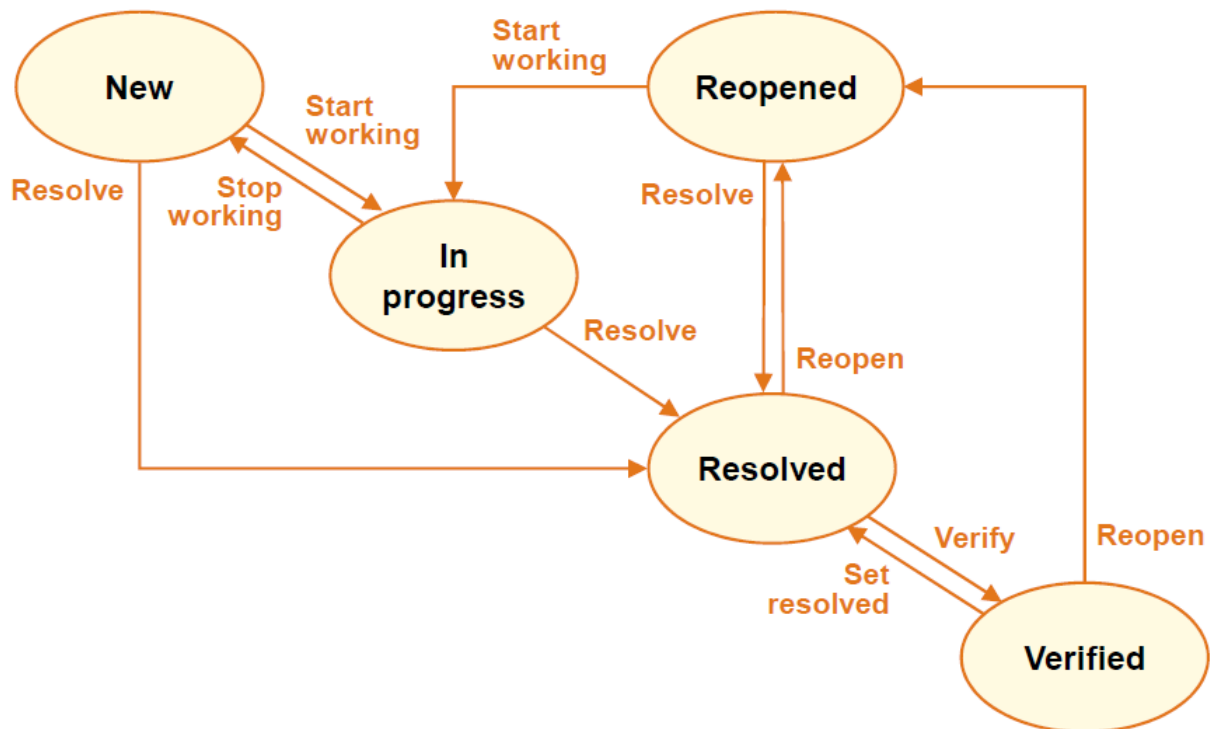


FIGURA 3.11 - FLUXO DE DEFEITOS (ESPINHA, 2012).

3.11. ANÁLISE DE RESULTADOS

Com as execuções dos testes manuais, criação dos casos de teste, criação de defeitos dentre outras atividades, é possível a geração de inúmeros relatórios disponíveis no *RQM*, que são acessados através do menu "*Relatórios >> Visualizar Relatórios*". Há relatórios de defeitos, execuções de casos de testes manuais, Requisitos, casos de teste, dentre outros, como na figura 3.12.

<input type="checkbox"/>	▼ Defects (1 - 2 de 2)		
<input type="checkbox"/>	Defect list	Defects status listing	Sim
<input type="checkbox"/>	Defects arrival and resolution	Defects created and resolved trend	Sim
<input type="checkbox"/>	▼ Execution (1 - 10 de 10)		
<input type="checkbox"/>	Execution and defects by owner	Execution and defects by owner	Sim
<input type="checkbox"/>	Execution Status by Machine using TER Count	Execution Status by Machine using TER Count	Sim
<input type="checkbox"/>	Execution Status by Machine using Weight	Execution Status by Machine using Weight	Sim
<input type="checkbox"/>	Execution Status by Owner using TER Count	Execution Status by Owner using TER Count	Sim
<input type="checkbox"/>	Execution Status by Owner using Weight	Execution Status by Owner using Weight	Sim
<input type="checkbox"/>	Execution Status using TER Count	Execution Status by TER Count	Sim
<input type="checkbox"/>	Execution Status using Weight	Execution Status by Weight	Sim
<input type="checkbox"/>	Execution Trend	Execution Trend	Sim
<input type="checkbox"/>	TER Listing	TER Listing	Sim
<input type="checkbox"/>	TER Status Counts	TER Status Counts	Sim
<input type="checkbox"/>	▼ Lab Manager (1 - 3 de 3)		
<input type="checkbox"/>	Lab Resource Utilization	Lab resource utilization in machine groups	Sim
<input type="checkbox"/>	Request Response Time	Average request response time by user	Sim
<input type="checkbox"/>	Requests by State	Requests by State	Sim
<input type="checkbox"/>	▶ Relatórios Compartilhados (18 itens)		
<input type="checkbox"/>	▶ Requirements (8 itens)		
<input type="checkbox"/>	▶ Scorecard (1 itens)		
<input type="checkbox"/>	▶ Shared Reports (13 itens)		
<input type="checkbox"/>	▶ Summary (3 itens)		
<input type="checkbox"/>	▶ Test case (6 itens)		

Anterior | 1 - 64 de 64 | Próximo

FIGURA 3.12 - LISTA DE RELATÓRIOS DISPONÍVEIS NO RQM

Ao selecionar um relatório a ser exibido, o *RQM* pede para serem informados alguns itens para que o relatório possa ser gerado. Depois de selecionado um relatório, basta clicar em "Executar" para que o relatório seja exibido, como o exemplo da Figura 3.13.

Scorecard ?

Status dos casos de teste

Total de etapas de teste	Peso total	Rascunho	Em revisão	Aprovado	Aposentado
51	5100	35	8	8	0

Contagens de execução de teste

TER	Execuções	Passed	Blocked	Partially Blocked	Failed	Não Iniciado
45	703	20	2	0	18	0
Inconclusive	Incomplete	PermFailed	Deferred	Error	In Progress	Paused
0	3	0	0	2	0	0

Pontos do resultado do teste

Peso total do TER	Passou	Bloqueado	Com Falha	Tentado	Inconclusivo	Não Iniciado
4500	3149	83	905	4207	153	293

FIGURA 3.13 - RELATÓRIO DE SCORECARD QUE EXIBE ALGUNS DADOS DE TESTES EXIBINDO PESOS DOS TESTES E A QUANTIDADE DE TESTES EXECUTADOS NO TOTAL.

4. FERRAMENTA MANTIS BUG TRACKER

Este capítulo traz os fundamentos e funcionalidades do *Mantis Bug Tracker* (Figura 4.1), mostrando o processo desde a criação do erro até o seu encerramento com o erro corrigido ou não.



FIGURA 4.1 – LOGOTIPO DO MANTIS BUG TRACKER

Curiosidade: A escolha de um *Mantis* (Louva-a-deus) como mascote oficial do projeto devido a sua natureza em rastrear *bugs*, que significam erros na linguagem popular dentro da computação.

4.1. O QUE É O MANTIS?

Mantis é uma ferramenta criada com o propósito de auxiliar no gerenciamento das falhas e suas correções dentro de um projeto, ajudando o gerente do projeto a distribuir e controlar os erros que estão sendo corrigidos pelos desenvolvedores do projeto ou os erros

que estão sendo testados, para verificar se eles foram corrigidos, pela equipe de testes do projeto.

4.2. COMO SURTIU?

O projeto do *Mantis*, também conhecido como *Mantisbt*, data de 2000 quando Kenzaburo Ito começou o seu desenvolvimento. Não podendo dar prosseguimento ao projeto em 2002, Victor Boctor assumiu o projeto e vem dando suporte a ele até as suas versões mais atuais. A sua primeira versão para o público saiu em janeiro de 2006, baseada na ferramenta de controle de versão *Apache Subversion* (conhecido como SVN), mas com o passar das novas edições, os seus desenvolvedores passaram a adotar ferramentas atuais como a Git, dando mais velocidade ao controle das revisões de cada projeto. (Mantis Bug Tracker, 2012).

4.3. PRIMEIROS PASSOS COM A FERRAMENTA

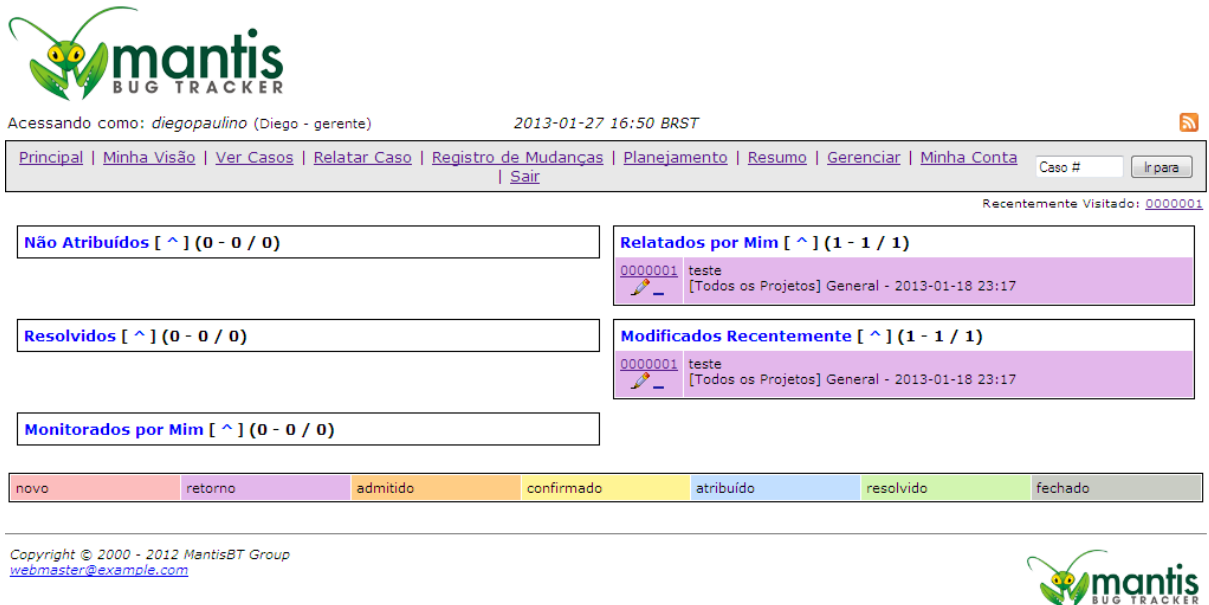
Para começar a utilizá-la, basta acessar o site oficial do *Mantis*, <http://www.mantisbt.org/>, na aba *Download*, e baixar a versão que seja compatível com o sistema operacional utilizado. Mas baixar a sua última versão não é o suficiente. Para que o ambiente de gerenciamento de testes possa funcionar corretamente é necessário instalar as últimas versões do servidor *Apache*, do banco de dados *Mysql* e da linguagem de programação *PHP*. Ao ter o primeiro contato com o *Mantis*, o usuário será redirecionado para a tela de *login* do sistema, onde o usuário poderá utilizar o seu *login* e senha liberados pelo administrador do servidor local do *Mantis*.

O *Mantis* é dividido em abas onde todas as opções estão habilitadas para todos os usuários, não importando o seu nível perante o projeto, sendo elas:

- a) Principal – Acessa a última visita do usuário e o total de erros atribuídos ao

usuário e relatados por ele;

- b) Minha visão – Página inicial do *Mantis*, onde o usuário poderá ter acesso ao andamento das últimas atualizações dos erros reportados, bem como quais erros estão alocados para ele (Figura 4.2);



The screenshot displays the Mantis Bug Tracker interface. At the top left is the Mantis logo with the text "mantis BUG TRACKER". Below the logo, it shows the user is logged in as "diegopaulino (Diego - gerente)" on "2013-01-27 16:50 BRST". A navigation menu includes links for "Principal", "Minha Visão", "Ver Casos", "Relatar Caso", "Registro de Mudanças", "Planejamento", "Resumo", "Gerenciar", "Minha Conta", and "Sair". There is a search box labeled "Caso #" and a "Ir para" button. A "Recentemente Visitado" section shows "0000001".


The main content area is divided into several sections:


- Não Atribuídos [^] (0 - 0 / 0)**: A box indicating no unassigned bugs.
- Relatados por Mim [^] (1 - 1 / 1)**: A box showing one bug reported by the user. The bug details are: ID "0000001", reporter "teste", project "[Todos os Projetos] General", and date "2013-01-18 23:17".
- Resolvidos [^] (0 - 0 / 0)**: A box indicating no resolved bugs.
- Modificados Recentemente [^] (1 - 1 / 1)**: A box showing one recently modified bug, identical to the one in the "Relatados por Mim" section.
- Monitorados por Mim [^] (0 - 0 / 0)**: A box indicating no bugs monitored by the user.

At the bottom, there is a status bar with colored buttons for "novo", "retorno", "admitido", "confirmado", "atribuído", "resolvido", and "fechado". The footer contains copyright information: "Copyright © 2000 - 2012 MantisBT Group" and "webmaster@example.com", along with the Mantis logo.

FIGURA 4.2 - PÁGINA COM A VISÃO DO ESTADO ATUAL DOS ERROS REPORTADOS.

- c) Ver casos - Página onde o usuário pode filtrar todos os erros reportados no sistema seja por quem o relatou, para quem, quem está monitorando, sua categoria e entre outras opções (Figura 4.3);



Acessando como: *diegopaulino* (Diego - gerente) 2013-01-27 19:10 BRST 

[Principal](#) | [Minha Visão](#) | [Ver Casos](#) | [Relatar Caso](#) | [Registro de Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) |

Recentemente Visitado: 000001

Relator:	Monitorado Por:	Atribuído a:	Categoria:	Gravidade:	Resolução:	Perfil:
qualquer	qualquer	qualquer	qualquer	qualquer	qualquer	qualquer
Estado:	Ocultar Status:					Prioridade:
qualquer	fechado (e acima)					qualquer
Exibir:	Visibilidade:	Mostrar Casos "Pequenos":	Alterado(hrs):	Usar Filtros de Data:	Relações:	
50	qualquer	Sim	6	Não	qualquer	
Plataforma:	SO:	Versão SO:	Marcadores:			
qualquer	qualquer	qualquer				
Nota Por:	qualquer	Ordenar por:	Atualizado Descendente			
Match Type:	All Conditions					

[[Filtros Avançados](#)] [[Criar Link Permanente](#)]

Visualizando Casos (1 - 1 / 1) [[Imprimir Relatórios](#)] [[Exportar CSV](#)] [[Exportação para Excel](#)]

	P	Núm	#	U	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>		000001	1		General	pequeno	retorno (thiagoholiveira)	2013-01-18	teste

Selecionar Tudo

novo	retorno	admitido	confirmado	atribuído	resolvido	fechado
------	---------	----------	------------	-----------	-----------	---------

FIGURA 4.3 - PÁGINA DE FILTRO DOS ERROS REPORTADOS.

- d) Relatar caso – Página onde o erro encontrado é relatado, onde o usuário poderá informar como o erro ocorreu, com direito a imagens do momento exato do erro, relatando ele para o desenvolvedor que possa resolvê-lo (Figura 4.4);

digite os Detalhes do Relatório

* **Categoria** (seleccione) ▾

Frequência não se tentou ▾

Gravidade pequeno ▾

Prioridade normal ▾

Selecionar Perfil

OU Preencha

Plataforma

SO

Versão SO

Atribuir a ▾

* **Resumo**

* **Descrição**

Passos para Reproduzir

Informações Adicionais

Carregar Arquivo (Tamanho máximo: 2,097k)

Visibilidade público privado

Continuar Relatando seleccione para relatar mais casos

* requerido

FIGURA 4.4 - PÁGINA ONDE SÃO REPORTADOS OS ERROS DO PROJETO.

- e) Registro de mudanças – Lista possíveis mudanças e correções realizadas no projeto;
- f) Planejamento – Lista os testes que estão abertos para correção e estão sendo tratados;
- g) Resumo – Página onde são exibidas as estatísticas do projeto, informando o andamento da correção dos erros por seus tipos, por quem relatou ou por quem corrigiu (Figura 4.5). Caso queira, o usuário também pode imprimi-los em forma de relatório (Figura 4.6);



Acessando como: diegopaulino (Diego - gerente)

2013-01-23 00:26 BRST



[Principal](#) | [Minha Visão](#) | [Ver Casos](#) | [Relatar Caso](#) | [Registro de Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) | [Minha Conta](#) | [Sair](#)

Caso # [[Imprimir Relatórios](#)] [[Resumo](#)]

Resumo										
Por Status	aberto	resolvido	fechado	total						
retorno	1	-	-	1						
Por Gravidade	aberto	resolvido	fechado	total						
pequeno	1	0	0	1						
Por Categoria	aberto	resolvido	fechado	total						
General	1	0	0	1						
Estatísticas de Tempo para Casos Resolvidos (dias)										
Caso aberto há mais tempo										
Aberto há quanto tempo	0.00									
Tempo médio	0.00									
Tempo total	0.00									
Estatísticas de Desenvolvedores	aberto	resolvido	fechado	total						
thiagoholiveira	1	0	0	1						
Por Dia	Aberto	Resolvido	Saldo							
1	0	0	0							
2	0	0	0							
3	0	0	0							
7	1	0	+1							
30	1	0	+1							
60	1	0	+1							
90	1	0	+1							
180	1	0	+1							
365	1	0	+1							
Mais Ativos				Mudanças						
0000001 - teste				12						
Aberto há quanto tempo				Dias						
0000001 - teste				4						
Por Resolução	aberto	resolvido	fechado	total						
reaberto	1	0	0	1						
Por Prioridade	aberto	resolvido	fechado	total						
normal	1	0	0	1						
Estatísticas de Relatores	aberto	resolvido	fechado	total						
diegopaulino	1	0	0	1						
Eficiência do Relator	Gravidade	Falso	Total							
diegopaulino	5	0	5							
Relator por Resolução	aberto	corrigido	reaberto	incapaz de reproduzir	não corrigível	duplicado	não é um caso	suspenso	não será corrigido	% Falsos
diegopaulino	0	0	1	0	0	0	0	0	0	0%
Desenvolvedor por Resolução	aberto	corrigido	reaberto	incapaz de reproduzir	não corrigível	duplicado	não é um caso	suspenso	não será corrigido	% Corrigidos
thiagoholiveira	0	0	1	0	0	0	0	0	0	100%

Copyright © 2000 - 2012 MantisBT Group
webmaster@example.com



FIGURA 4.5 - PÁGINA DE RESUMO DOS ERROS REPORTADOS NO PROJETO.

MantisBT - Projeto Final									
Visualizando Casos(1 - 1)									
P	Núm	#	0	Categoria	Gravidade	Estado	Atualizado	Resumo	
<input type="checkbox"/>	-	0000001	1	General	pequeno	retorno (thiagoholiveira)	2013-01-18	teste	
<input type="button" value="Mostrar apenas os selecionados"/>									

FIGURA 4.6 - PÁGINA PARA IMPRIMIR RELATÓRIOS REFERENTES AOS ERROS DO PROJETO.

- h) Gerenciar – Página que, dependendo do nível cadastrado, poderá ter mais opções. Caso não tenha o perfil de Administrador ou Gerente, o usuário terá apenas informação sobre as últimas versões do *Mantis*. Caso seja Gerente, além das opções que os outros usuários possuem, terá acesso ao escopo do seu projeto, podendo adicionar marcador e com a função de metas ou poderá informar os perfis globais do projeto, de qual plataforma ou sistema organizacional ele irá funcionar. Caso seja Administrador, além das funções de gerente, ele terá acesso para modificar os dados cadastrais de cada usuário, podendo alterar a prioridade de acesso de cada um dele, além de poder criar campos personalizados para a identificação de cada projeto, visualizar a última atualização de plug-ins do *Mantis* e visualizar os relatórios de configuração de cada um dos usuários;
- i) Minha conta – Exibe as informações cadastradas do usuário, podendo ser alteradas caso ele deseje (Figura 4.7, 4.8, 4.9, 4.10);

Logo: mantis BUG TRACKER

Acessando como: diegopaulino (Diego - gerente) 2013-01-23 00:53 BRST

Principal | Minha_Visão | Ver_Casos | Relatar_Caso | Registro_de_Mudanças | Planejamento | Resumo | Gerenciar | Minha_Conta | Caso # Ir para

Alterar Conta [Minha Conta] [Preferências] [Gerenciar_Colunas] [Perfis]	
Nome de usuário	diegopaulino
Senha	<input type="password"/>
Confirmar Senha	<input type="password"/>
E-Mail	<input type="text"/>
Nome Verdadeiro	<input type="text"/>
Nível de Acesso	gerente
Nível de Acesso ao Projeto	gerente
Projetos Atribuídos	Projeto Final [gerente] (privado)
<input type="button" value="Atualizar Usuário"/>	

FIGURA 4.7 - PÁGINA COM DOS DADOS CADASTRADOS DO USUÁRIO.



Acessando como: diegopaulino (Diego - gerente)

2013-01-23 00:45 BRST



[Principal](#) | [Minha_Visão](#) | [Ver_Casos](#) | [Relatar_Caso](#) | [Registro_de_Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) | [Minha_Conta](#) | [Sair](#)

Caso #

Ir para

Preferências da Conta	
[Minha Conta]	[Preferências]
[Gerenciar Colunas]	[Perfis]
Projeto Padrão	Projeto Final ▾
Tempo de Renovação	30 minutos
Tempo de Redirecionamento	2 segundos
Ordem de Classificação de Anotações	<input checked="" type="radio"/> Ascendente <input type="radio"/> Descendente
Enviar e-mail para Novos Casos	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Mudança de Atribuição	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Retornos	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Casos Resolvidos	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Casos Fechados	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Casos Reabertos	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Anotações Adicionadas	<input checked="" type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Mudança de Status	<input type="checkbox"/> Com a gravidade mínima qualquer ▾
Enviar e-mail para Mudança de Prioridade	<input type="checkbox"/> Com a gravidade mínima qualquer ▾
Limite de Anotações por e-mail	0
Fuso Horário	Sao Paulo ▾
Idioma	portuguese_brazil ▾
<input type="button" value="Atualizar Preferências"/>	

Copyright © 2000 - 2012 MantisBT Group
webmaster@example.com



FIGURA 4.8 - PÁGINA REFERENTE AO CADASTRO DOS TIPOS DE NOTIFICAÇÕES DE ERROS QUE O USUÁRIO IRÁ RECEBER.



Acessando como: diegopaulino (Diego - gerente)

2013-01-23 00:47 BRST



[Principal](#) | [Minha Visão](#) | [Ver Casos](#) | [Relatar Caso](#) | [Registro de Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) | [Minha Conta](#) | [Sair](#)

Caso # Ir para

Gerenciar Colunas	
[Minha Conta] [Preferências] [Gerenciar Colunas] [Perfil]	
Todas as Colunas Disponíveis	id, project_id, reporter_id, handler_id, duplicate_id, priority, severity, reproducibility, status, resolution, category_id, date_submitted, last_updated, os, os_build, platform, version, fixed_in_version, target_version, view_state, summary, sponsorship_total, due_date, description, steps_to_reproduce, additional_information, attachment_count, bugnotes_count, selection, edit, overdue
Ver Colunas dos Casos*	selection, edit, priority, id, sponsorship_total, bugnotes_count, attachment_count, category_id, severity, status, last_updated, summary
Imprimir Colunas dos Casos*	selection, priority, id, sponsorship_total, bugnotes_count, attachment_count, category_id, severity, status, last_updated, summary
Colunas CSV*	id, project_id, reporter_id, handler_id, priority, severity, reproducibility, version, category_id, date_submitted, os, os_build, platform, view_state, last_updated, summary, status, resolution, fixed_in_version
Colunas do Excel*	id, project_id, reporter_id, handler_id, priority, severity, reproducibility, version, category_id, date_submitted, os, os_build, platform, view_state, last_updated, summary, status, resolution, fixed_in_version

* requerido

Copyright © 2000 - 2012 MantisBT Group
webmaster@example.com



FIGURA 4.9 - PÁGINA REFERENTE À EDIÇÃO DOS DADOS QUE O USUÁRIO TERÁ ACESSO.

Acessando como: diegopaulino (Diego - gerente) 2013-01-23 00:49 BRST

Principal | Minha Visão | Ver Casos | Relatar Caso | Registro de Mudanças | Planejamento | Resumo | Gerenciar | Minha Conta | Sair

Adicionar Perfil [Minha Conta] [Preferências] [Gerenciar Colunas] [Perfis]

*Plataforma

*Sistema Operacional

*Versão SO

Descrição Adicional

* requerido

Copyright © 2000 - 2012 MantisBT Group
webmaster@example.com

mantis
BUG TRACKER

FIGURA 4.10 - PÁGINA COM INFORMAÇÕES ADICIONAIS SOBRE O AMBIENTE ONDE O PROJETO IRÁ FUNCIONAR.

j) Sair – Executa o *logout* do usuário da página do *Mantis*.

4.4. SUAS FUNCIONALIDADES

O *Mantis* é uma ferramenta que está sendo usada por muitas empresas, seja por projetos pequenos ou grandes, por projetos universitários ou mesmo para projetos pessoais devido a sua facilidade em conseguir uma cópia dele, pois, como o *Mantis* é uma ferramenta extremamente leve comparado a ferramentas em que o usuário deve conseguir uma cópia pagando por ela e que ela é aberta para que qualquer um possa baixá-lo, utilizá-lo e adaptá-lo para o seu melhor proveito. Outra grande vantagem do *Mantis* comparado aos seus concorrentes é a sua versão adaptada para dispositivos móvel como *iPad*, *iPod* e *smartphones* com o *Mantis Touch* (Figura 4.11). Com ele o usuário pode ter acesso no seu dispositivo móvel a todas as funcionalidades que o *Mantis* oferece no seu navegador convencional,

tornando mais fácil o controle do gerenciamento e controle dos erros que foram gerados e atualizados.

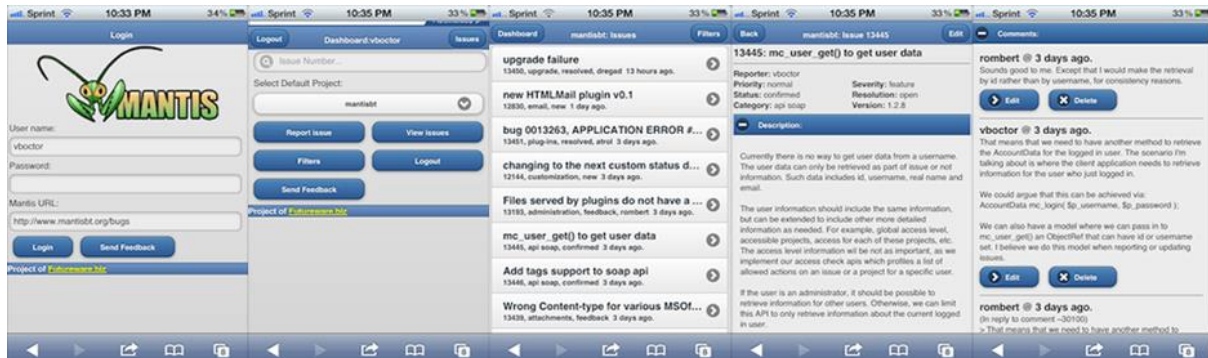


FIGURA 4.11 - PRINCIPAIS TELAS DO MANTIS TOUCH RODANDO EM UM DISPOSITIVO APPLE

Por outro lado o *Mantis Touch* não pode ser conseguido de graça que nem a sua versão de navegador. Pagando 100 dólares o usuário pode conseguir a licença para o uso do *Mantis Touch* para ele e quem mais ele passar o *login* de acesso.

Já com relação a sua versão *web*, ela tem um problema que acaba por afastar os entusiastas que conhecem a ferramenta por pouco tempo: ela é extremamente difícil para se configurar caso você não conheça a ferramenta. O usuário precisará quebrar a cabeça no começo até conseguir ter os primeiros acessos como usuários do sistema e não como administrador, além de precisar configurar por conta própria funcionalidades básicas em outras ferramentas de gerenciamento de testes como o envio de e-mails para os usuários relacionados aos erros abertos, por exemplo.

Outro ponto negativo relativo ao *Mantis* é a sua instabilidade, pois a sua funcionalidade depende exclusivamente da máquina em que o *Mantis* e o pacote *Apache / Mysql / PHP* tiver sido instalada. Caso ela saia do ar por algum problema os usuários não poderão ter acesso ao *Mantis* e seus erros abertos, tornando o trabalho de gerencia dos erros mais complexa, fazendo com que o gerente tenha que atuar com maior presença ao lado dos desenvolvedores para controlar o estado atual de cada *bug* em que eles estejam trabalhando.

Outro ponto em que o *Mantis* pode ser considerado um problema é a falta de hierarquia na mudança de estados de um erro. Em outras ferramentas de gerenciamento de

testes a mudança de estado de um erro de novo para resolvido, por exemplo, deve ser feito seguindo a hierarquia para sair de um para chegar ao outro. Já com o *Mantis* pode-se mudar do estado novo para o resolvido de forma direta, podendo atrapalhar o trabalho do gerente para controlar a quantidade de erros que estão ou passaram por cada estado.

Pode ser visto que o *Mantis* possui muitas características que o fazem ser escolhido para gerenciar os testes de um projeto, mas alguns problemas que ainda não o tornam a melhor escolha de gerenciamento de testes do mercado, fazendo com que o líder de testes do projeto tenha que escolhê-lo pelas características do projeto em que ele está atuando, como o nível de prioridade do projeto, seja no seu tempo de execução como no investimento que a empresa fará para executá-lo.

5. COMPARAÇÃO ENTRE AS FERRAMENTAS

Este capítulo traz a comparação entre o *RQM* e o *Mantis*, onde será utilizado um mesmo defeito como exemplo para que seja visto o tempo de vida dele dentro das duas ferramentas.

O exemplo utilizado foi tirado de um trabalho passado pelo professor Leonardo Murta da disciplina “Engenharia de *Software* II” na Universidade Federal Fluminense no segundo semestre de 2012. Nele é pedido que se desenvolvesse um jogo no estilo do jogo de tabuleiro *WAR* e incluía etapas de teste no desenvolvimento do trabalho. Um dos problemas ocorrido foi em uma tela em que havia botões de “Atacar” e “Movimentar” e que na primeira rodada, os botões teriam que estar desabilitados. Assim, o teste do exemplo verificava se ao clicar no botão “Atacar”, ele realmente permanecia desabilitado. Porém ao clicá-lo, uma janela de ataque era aberta e assim, constituía um defeito. Este defeito que foi cadastrado por ambas as ferramentas a fim de compará-las.

5.1. CADASTRO DO DEFEITO PELO RQM

Para registrar um novo defeito, há duas maneiras. Uma é acessando o menu acessando “*Change Requests*>>*Create Defect*” (figura 5.1). Este modo é mais utilizado para registrar um *bug* encontrado sem executar nenhum caso de teste, como em um teste exploratório por

exemplo. Porém criando desta forma, daria mais trabalho para detalhá-lo e para associá-lo a casos de uso e a tarefas para auxiliar na rastreabilidade.

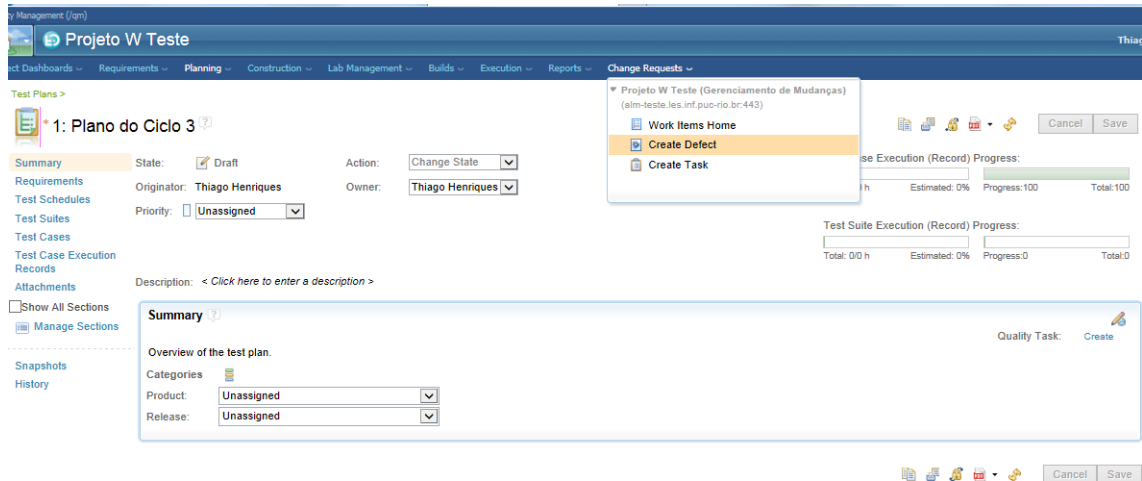


FIGURA 5.1 – CRIANDO NOVO DEFEITO

Outro método seria após a execução de um *script* de teste. Para executá-lo, basta selecionar o caso de teste que deseja executar, escolher o *script*, clicar na seta verde e selecionar “Run” como mostra a figura 5.2. Será aberta uma tela para selecionar os dados de execução mostrado na figura 5.3.

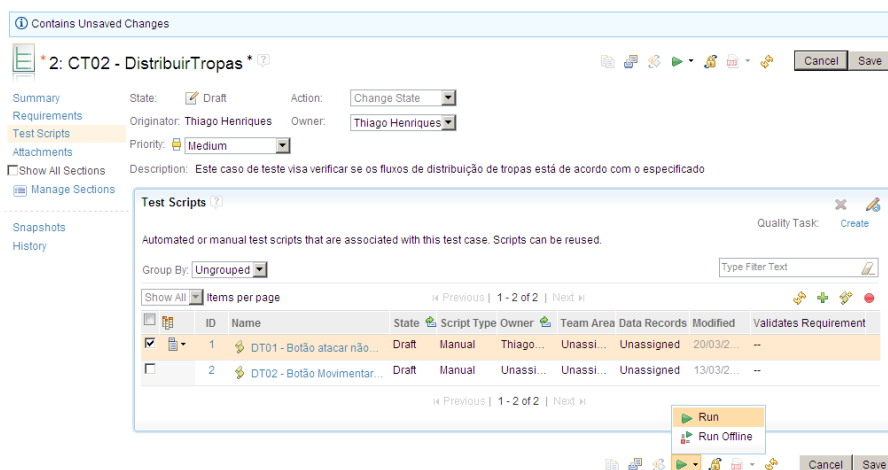


FIGURA 5.2 – EXECUTANDO UM CASO DE TESTE

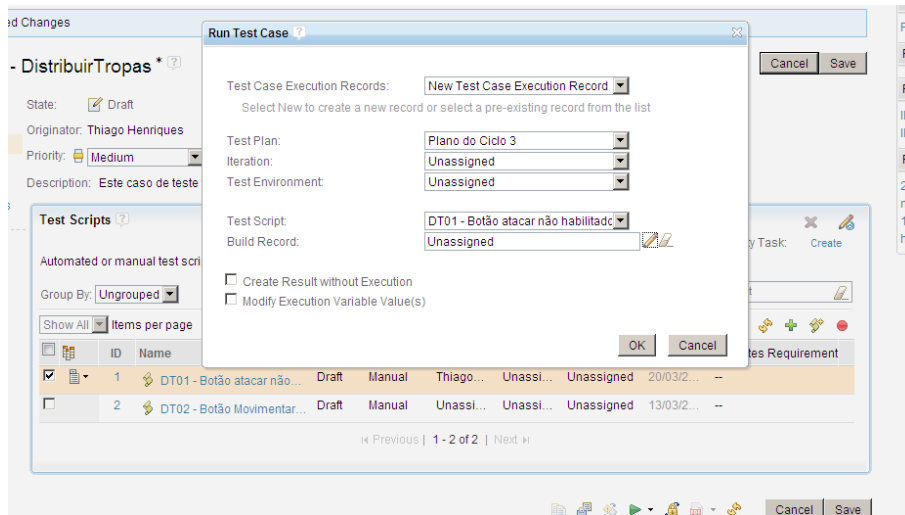


FIGURA 5.3 – EXECUÇÃO DE UM CASO DE TESTE

Após confirmar, o teste começa a ser executado. Uma página é exibida contendo o passo a passo do teste e o resultado esperado. Caso o passo ocorra conforme esperado, deve ser sinalizado com um “v” em verde. Porém caso falhe, deverá ser preenchido o que realmente ocorreu na coluna “*Actual Results*”. Após preencher e antes de sinalizar de vermelho informando que não passou, é recomendado cadastrar o defeito clicando no botão marcado em vermelho na figura 5.4.

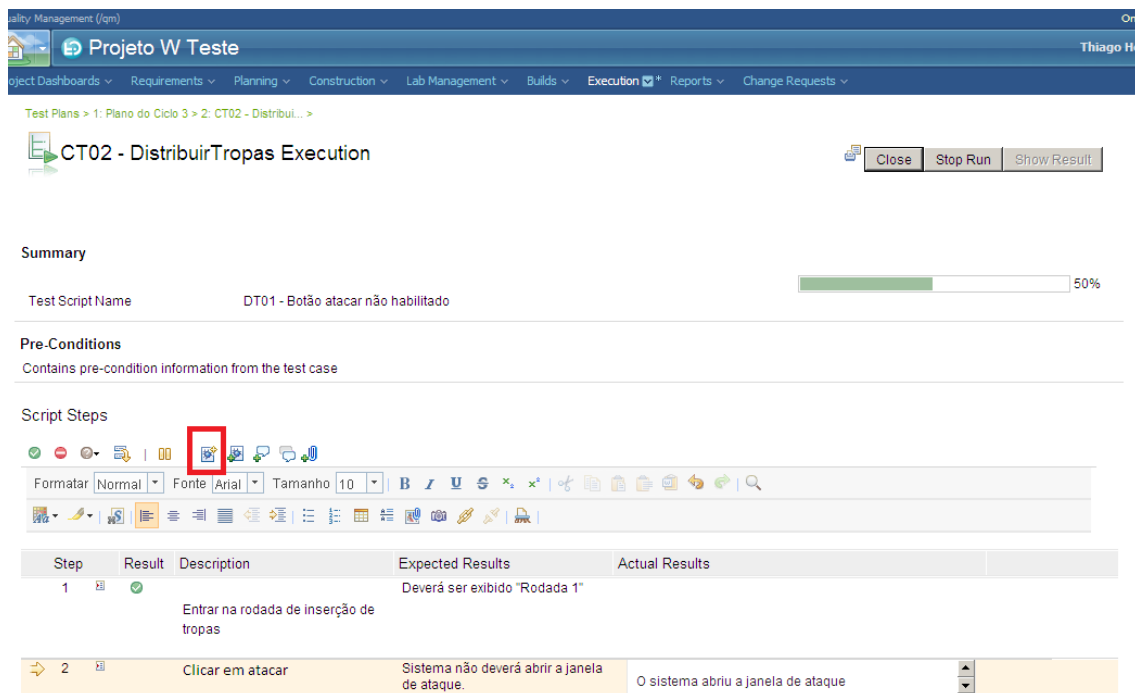


FIGURA 5.4 – CRIAÇÃO DE UM NOVO DEFEITO DURANTE A EXECUÇÃO DE UM TESTE

Será aberta uma janela pedindo para informar os dados do defeito, mostrado na figura 5.5. Fazendo desta forma executando passo a passo e criando o *bug* no passo que falhou, o passo a passo já é inserido automaticamente até o passo que falhou na descrição, como mostra a figura 5.6. Assim, o criador necessita apenas detalhar o que ocorreu de fato para ocasionar a falha, economizando tempo.

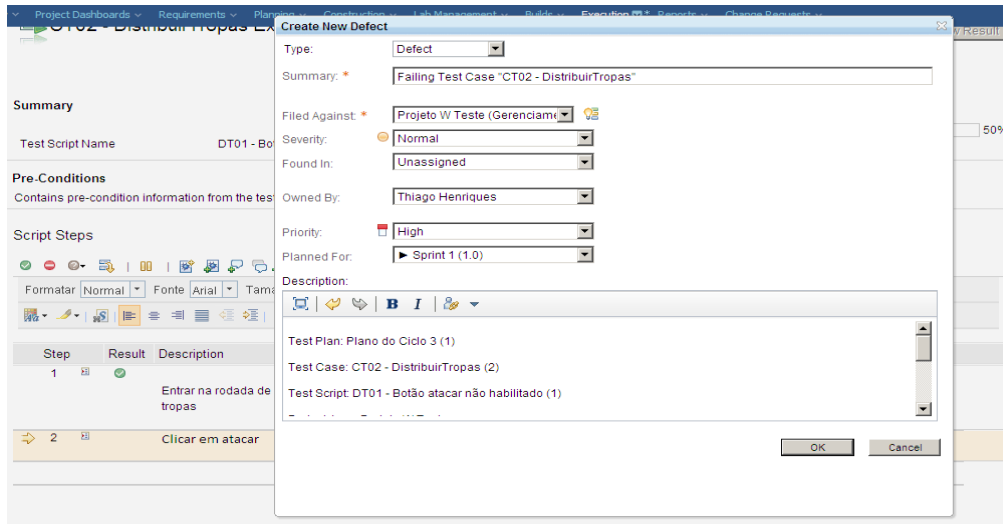


FIGURA 5.5 – DETALHES DE UM NOVO DEFEITO

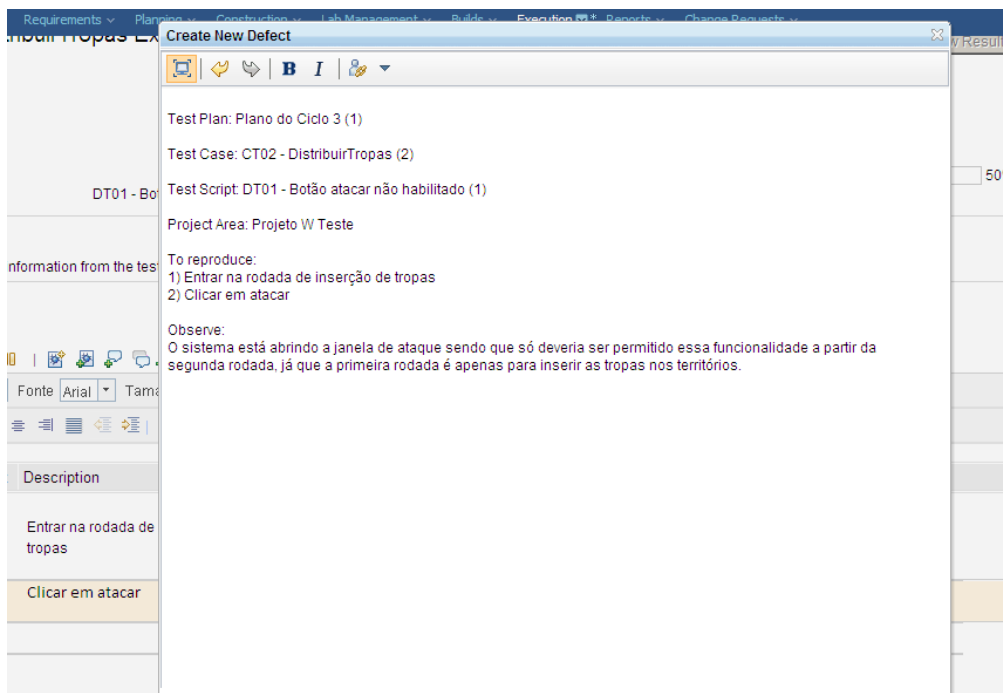


FIGURA 5.6 – DESCRIÇÃO PARCIALMENTE PRONTA AUTOMATICAMENTE PELO RQM

Após confirmar a sua criação, um ícone é adicionado ao passo falho, como é exibido na figura 5.7. Com isso todos os *links* associados ao caso de teste são automaticamente associados ao defeito.

The screenshot displays a test execution window titled "CT02 - DistribuirTropas Execution". At the top, a red banner indicates "Execution failed". Below this, a summary section shows the test script name "DT01 - Botão atacar não habilitado" with a 100% progress bar. The "Script Steps" section contains a table with two rows:

Step	Result	Description	Expected Results	Actual Results
1	✓	Entrar na rodada de inserção de tropas	Deverá ser exibido "Rodada 1"	
2	✗	Clicar em atacar	Sistema não deverá abrir a janela de ataque.	O sistema abriu a janela de ataque

At the bottom of the table, a yellow banner states "Execution completed".

FIGURA 5.7 – TESTE EXECUTADO E DEFEITO ASSOCIADO CRIADO

Depois do defeito criado, ele passará a ter alguns estados pré-definidos no momento da criação da área de trabalho do projeto, como mostra a figura 5.8. Assim, o defeito será analisado e irá para correção caso confirmado o problema. Caso contrário, ele será recusado. E assim, a cada estado que ele passa, novos vão surgindo até ele ser verificado. Este fluxo é detalhado no capítulo 3 seção 3.9. Acompanhamento de defeitos.

Defect 833 ?

Summary: *

Start Working ▼ Save

Overview Links Approvals History

Details

Type: Owned By:

Filed Against: Priority:

Severity: Planned For:

Found In: Estimate: Correction:

Project Area: Time Remaining:

Team Area: Due Date:

Creation Date: 25/03/2013 18:58:38

Created By:

Tags:

Quick Information

- Subscribers (1): TH
- Affects Test Result (1): 1
- Related Test Case (1): 1
- Related Test Plan (1): 1

Description Edit

Test Plan: Plano do Ciclo 3 (1)

Test Case: CT02 - DistribuirTropas (2)

Test Script: DT01 - Botão atacar não habilitado (1)

Project Area: Projeto W Teste

To reproduce:

- 1) Entrar na rodada de inserção de tropas
- 2) Clicar em atacar

Observe:

FIGURA 5.8 – ESTADOS DE UM DEFEITO

E toda modificação feita no defeito ao decorrer todo o seu fluxo, é exibido na aba histórico, como mostra a figura 5.9. Nela é exibida alterações de estado como associações e não associações de *links*, alterações de descrição e inserções de comentários.

Defect 833 ?

Summary: *

Verified ▼ Fixed ▼

Overview Links Approvals **History** Saved: 25/03/2013 10:46:44

History

Change by Thiago Henriques (25/03/2013 19:03:46)

Status	Resolved → Verified
--------	---------------------

Change by Thiago Henriques (25/03/2013 19:03:31)

Status	In Progress → Resolved
Resolution	Unresolved → Fixed
Resolved By	Unassigned → Thiago Henriques
Resolution Date	<Unassigned> → 25/03/2013

Change by Thiago Henriques (25/03/2013 19:03:18)

Status	Reopened → In Progress
--------	------------------------

Change by Thiago Henriques (25/03/2013 19:03:11)

Status	Resolved → Reopened
Resolution	Fixed → Unresolved
Resolved By	Thiago Henriques → Unassigned
Resolution Date	25/03/2013 → <Unassigned>

Change by Thiago Henriques (25/03/2013 19:03:03)

Status	In Progress → Resolved
Resolution	Unresolved → Fixed
Resolved By	Unassigned → Thiago Henriques
Resolution Date	<Unassigned> → 25/03/2013

Change by Thiago Henriques (25/03/2013 19:02:49)

Status	New → In Progress
--------	-------------------

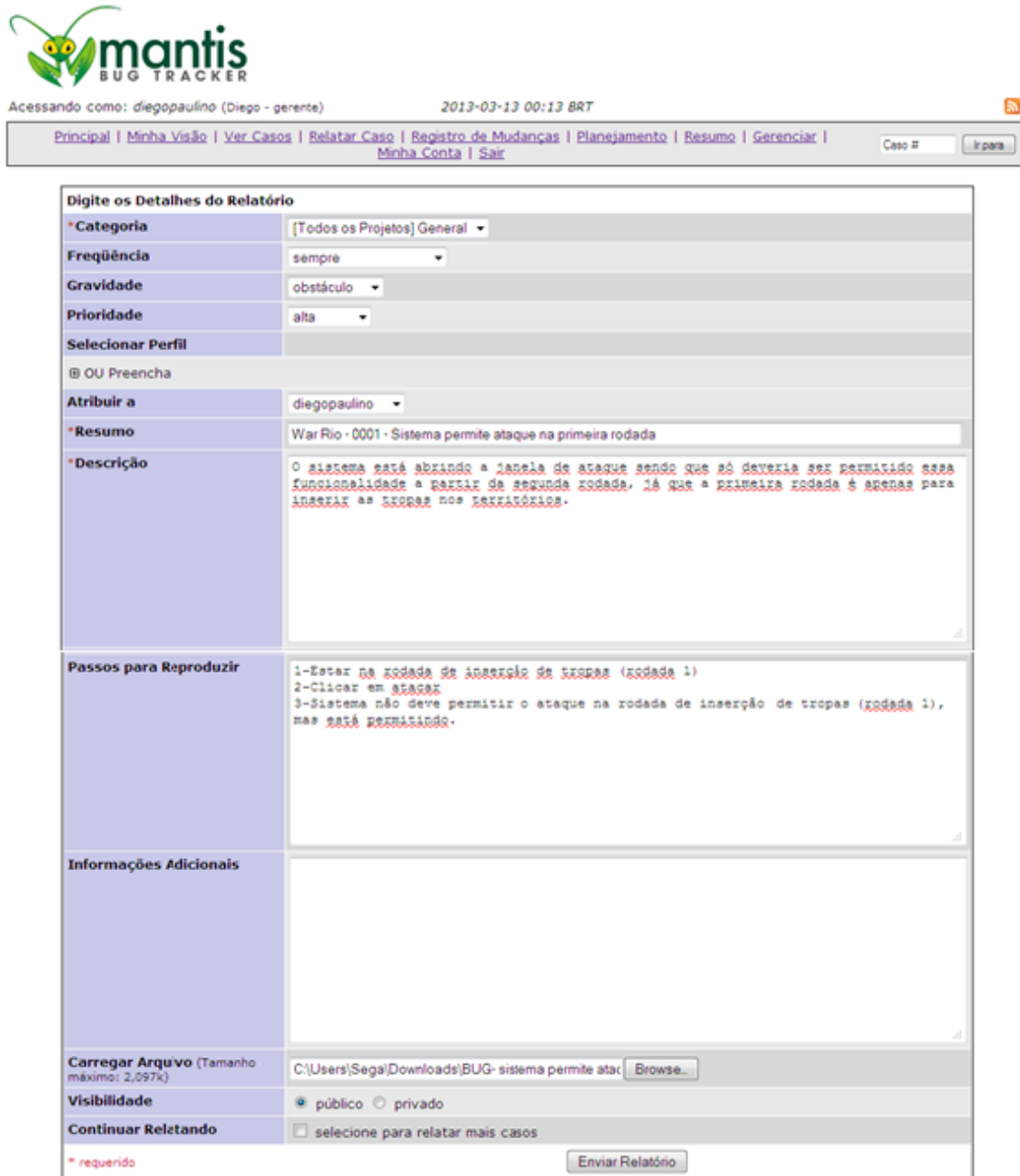
Change by Thiago Henriques (25/03/2013 18:58:38)

FIGURA 5.9 – HISTÓRICO DE UM DEFEITO

5.2. CADASTRO DO DEFEITO PELO MANTIS

A utilização do *Mantis* para gerenciar e controlar os erros encontrados em um sistema se dá da seguinte forma:

- a) Ao encontrar um fato que não esteja de acordo com o que foi determinado no caso de teste, o responsável pelo teste registrará o erro no *Mantis* indo na opção 'Relatar Caso', descrevendo os passos que realizou para chegar até o erro, bem como a frequência com que ele ocorre, a sua gravidade perante o sistema e qual seria a prioridade de correção dele (Figura 5.10).
Para facilitar o trabalho do desenvolvedor que irá trabalhar no erro, o responsável pelo teste poderá tirar cópias de cada passo e da tela do sistema no momento do erro. Essas cópias serão atribuídas ao erro no *Mantis* na forma de anexo.



Digite os Detalhes do Relatório

* **Categoria** [Todos os Projetos] General ▾

Freqüência sempre ▾

Gravidade obstáculo ▾

Prioridade alta ▾

Selecionar Perfil

OU Preencha

Atribuir a diegopaulino ▾

* **Resumo** War Rio - 0001 - Sistema permite ataque na primeira rodada

* **Descrição**
 O sistema está abrindo a janela de ataque sendo que só deveria ser permitido essa funcionalidade a partir da segunda rodada, lá mas a primeira rodada é apenas para inserir as tropas nos territórios.

Passos para Reproduzir
 1-Estar na rodada de inserção de tropas (rodada 1)
 2-Clicar em ATACAR
 3-Sistema não deve permitir o ataque na rodada de inserção de tropas (rodada 1), mas está permitindo.

Informações Adicionais

Carregar Arquivo (Tamanho máximo: 2,097K) C:\Users\Sega\Downloads\BUG- sistema permite atar

Visibilidade público privado

Continuar Relatando selecione para relatar mais casos

* requerido

FIGURA 5.10 – TELA COM DESCRIÇÃO DO ERRO GERADO PELO RESPONSÁVEL PELO TESTE

- b) Após terminar de descrever os dados do erro e enviá-lo para o gerente do projeto pela opção ‘Enviar Relatório’, o erro irá ser criado no *Mantis* e terá o status de Novo (Figura 5.11).



Acessando como: *diegopaulino* (Diego - gerente)

2013-03-13 00:25 BRT



[Principal](#) | [Minha Visão](#) | [Ver Casos](#) | [Relatar Caso](#) | [Registro de Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) | [Minha Conta](#) | [Sair](#)

Recentemente Visitado: 0000006, 0000005, 0000004, 0000003

Relator:	Monitorado Por:	Atribuído a:	Categoria:	Gravidade:	Resolução:	Perfil:
qualquer	qualquer	qualquer	qualquer	qualquer	qualquer	qualquer
Estado:	Ocultar Status:					Prioridade:
qualquer	fechado (e acima)					qualquer
Exibir:	Visibilidade:	Mostrar Casos "Pequenos":	Alterado(hrs):	Usar Filtros de Data:	Relações:	
50	qualquer	Sim	6	Não	qualquer	
Plataforma:	SO:	Versão SO:	Marcadores:			
qualquer	qualquer	qualquer				
Nota Por:	qualquer	Ordenar por:	Atualizado Descendente			
Match Type:	All Conditions					

[[Filtros Avançados](#)] [[Criar Link Permanente](#)]

Visualizando Casos (1 - 1 / 1) [[Imprimir Relatórios](#)] [[Exportar CSV](#)] [[Exportação para Excel](#)]

	P	Núm	#	@	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>		0000006			General	obstáculo	novo	2013-03-13	War Rio - 0001 - Sistema permite ataque na primeira rodada

Selecionar Tudo

novo retorno admitido confirmado atribuído resolvido fechado

FIGURA 5.11 – TELA DO MANTIS EXIBINDO O STATUS DO ERRO RECÉM-CRIADO

- c) Ao receber o relatório de informação sobre o novo erro, o gerente o irá analisar e o encaminhará para um dos desenvolvedores da sua equipe. Ao fazer isso o erro no *Mantis* irá trocar de status, que passará a ter o status de ‘Atribuído’ (Figura 5.12).



Acessando como: diegopaulino (Diego - gerente)

2013-03-13 00:26 BRT



Principal | Minha Visão | Ver Casos | Relatar Caso | Registro de Mudanças | Planejamento | Resumo | Gerenciar |

Recentemente Visitado: 000006, 000005, 000004, 000003

Ver Detalhes do Caso [[Ir para as Anotações](#)] [[Enviar um lembrete](#)] [[Histórico do Caso](#)] [[Imprimir](#)]

Núm	Projeto	Categoria	Visibilidade	Data de Envio	Última Atualização
0000006	Projeto Final	[Todos os Projetos] General	público	2013-03-13 00:24	2013-03-13 00:26

Relator: diegopaulino

Atribuído a: diegopaulino

Prioridade: alta Gravidade: obstáculo Frequência: sempre

Estado: atribuído Resolução: aberto

Plataforma: SO: Versão SO:

Resumo: 0000006: War Rio - 0001 - Sistema permite ataque na primeira rodada

Descrição: O sistema está abrindo a janela de ataque sendo que só deveria ser permitido essa funcionalidade a partir da segunda rodada, já que a primeira rodada é apenas para inserir as tropas nos territórios.

Passos para Reproduzir: 1-Estar na rodada de inserção de tropas (rodada 1)
2-Clicar em atacar
3-Sistema não deve permitir o ataque na rodada de inserção de tropas (rodada 1), mas está permitindo.

Marcadores: Nenhum marcador aplicado.

Aplicar Marcadores (Separar por ','): Marcadores atuais:

Arquivos Anexados

Relações

Nova relação: Este caso está relacionado a

Carregar Arquivo

Selecionar Arquivo (Tamanho máximo: 2,097K)

Usuários monitorando este caso

Lista de Usuários: Não há usuários monitorando este caso.

Nome de usuário

Anotações

Não há anotações anexadas a este caso.

Adicionar Anotação

Anotação:

Visibilidade: privado

Histórico do Caso

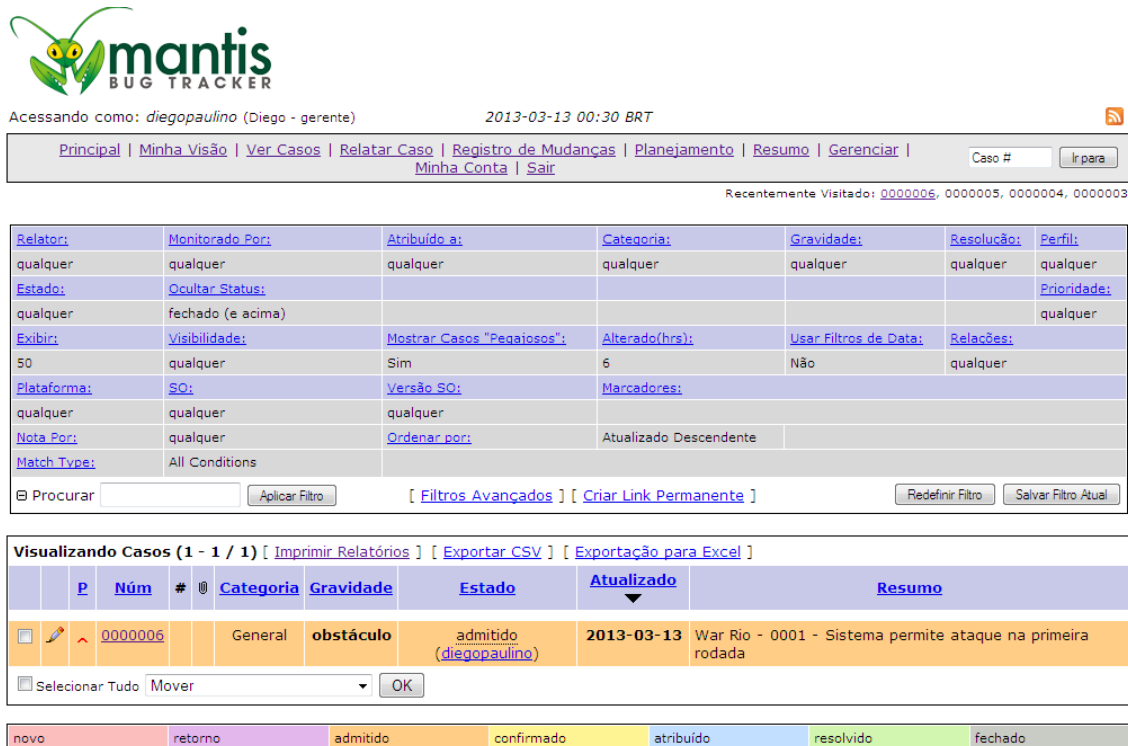
Data da Mudança	Nome de usuário	Campo	Alteração
2013-03-13 00:24	diegopaulino	Novo Caso	
2013-03-13 00:24	diegopaulino	Estado	novo => atribuído
2013-03-13 00:24	diegopaulino	Atribuído a	=> diegopaulino
2013-03-13 00:25	diegopaulino	Atribuído a	diegopaulino =>
2013-03-13 00:25	diegopaulino	Estado	atribuído => novo
2013-03-13 00:26	diegopaulino	Atribuído a	=> diegopaulino
2013-03-13 00:26	diegopaulino	Estado	novo => atribuído

Copyright © 2000 - 2012 MantisBT Group
webmaster@example.com



FIGURA 5.12 – TELA EXIBINDO AS NOVAS ATRIBUIÇÕES DO ERRO BEM COMO O SEU HISTÓRICO

- d) Ao começar a trabalhar no erro que lhe foi atribuído, o desenvolvedor deverá trocar o status do erro para Admitido, tornando o controle do gerente mais preciso sobre em que os desenvolvedores estão trabalhando no momento (Figura 5.13).



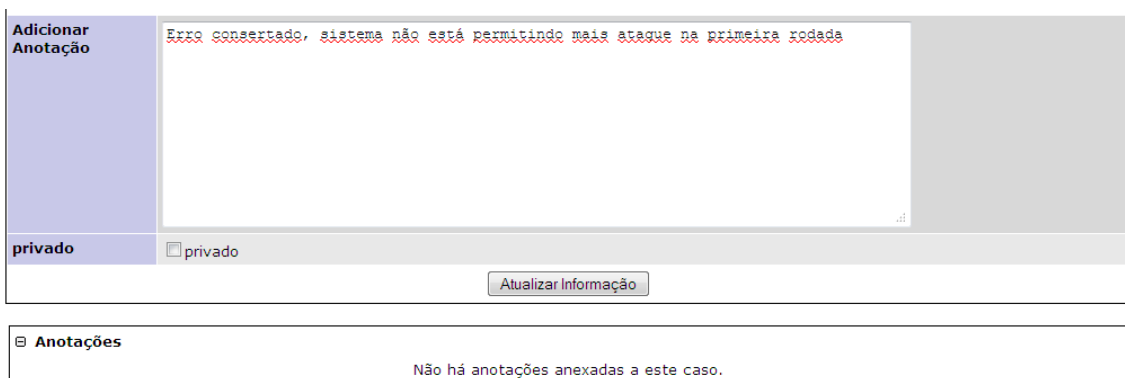
The screenshot shows the Mantis Bug Tracker interface. At the top, it says 'Acessando como: diegopaulino (Diego - gerente)' and '2013-03-13 00:30 BRT'. Below this is a navigation bar with links like 'Principal', 'Minha Visão', 'Ver Casos', etc. The main content area shows a filter table with columns for 'Relator:', 'Monitorado Por:', 'Atribuído a:', 'Categoria:', 'Gravidade:', 'Resolução:', and 'Perfil:'. Below the filter table is a table of cases. The first case is highlighted in orange and has the following details:

P	Núm	#	@	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>	0000006			General	obstáculo	admitido (diegopaulino)	2013-03-13	War Rio - 0001 - Sistema permite ataque na primeira rodada

Below the table, there are buttons for 'Selecionar Tudo', 'Mover', and 'OK'. At the bottom, there is a legend for bug statuses: novo, retorno, admitido, confirmado, atribuído, resolvido, and fechado.

FIGURA 5.13 – TELA EXIBINDO O ERRO EM QUE O DESENVOLVEDOR ESTÁ TRABALHANDO

- e) Ao terminar de realizar as mudanças necessárias para que o erro deixe de existir, o desenvolvedor irá descrever o que foi consertado bem como as evidências de como o sistema está em pleno funcionamento (Figura 5.14). Assim, ao termina-lo, o desenvolvedor deverá atualizar o status do erro para Confirmado (Figura 5.15).



The screenshot shows the 'Adicionar Anotação' (Add Note) form in the Mantis Bug Tracker. The form has a text area where a note has been entered: 'Erro consertado, sistema não está permitindo mais ataque na primeira rodada'. Below the text area, there is a checkbox for 'privado' (private) which is currently unchecked. At the bottom of the form, there is a button labeled 'Atualizar Informação' (Update Information). Below the form, there is a section for 'Anotações' (Notes) which currently shows 'Não há anotações anexadas a este caso.' (No notes attached to this case).

FIGURA 5.14 – TELA EXIBINDO AS INFORMAÇÕES SOBRE O ERRO RESOLVIDO PELO DESENVOLVEDOR

Acessando como: *diegopaulino* (Diego - gerente)

2013-03-13 00:34 BRT



[Principal](#) | [Minha Visão](#) | [Ver Casos](#) | [Relatar Caso](#) | [Registro de Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) |

Recentemente Visitado: [0000006](#), [0000005](#), [0000004](#), [0000003](#)

Relator:	Monitorado Por:	Atribuído a:	Categoria:	Gravidade:	Resolução:	Perfil:
qualquer	qualquer	qualquer	qualquer	qualquer	qualquer	qualquer
Estado:	Ocultar Status:					Prioridade:
qualquer	fechado (e acima)					qualquer
Exibir:	Visibilidade:	Mostrar Casos "Pegajosos":	Alterado(hrs):	Usar Filtros de Data:	Relações:	
50	qualquer	Sim	6	Não	qualquer	
Plataforma:	SO:	Versão SO:	Marcadores:			
qualquer	qualquer	qualquer				
Nota Por:	qualquer	Ordenar por:	Atualizado Descendente			
Match Type:	All Conditions					
<input type="checkbox"/> Procurar <input type="text"/> <input type="button" value="Aplicar Filtro"/> [Filtros Avançados] [Criar Link Permanente] <input type="button" value="Redefinir Filtro"/> <input type="button" value="Salvar Filtro Atual"/>						

Visualizando Casos (1 - 1 / 1) [\[Imprimir Relatórios \]](#) [\[Exportar CSV \]](#) [\[Exportação para Excel \]](#)

	P	Núm	#	@	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>		0000006	1		General	obstáculo	confirmado (diegopaulino)	2013-03-13	War Rio - 0001 - Sistema permite ataque na primeira rodada
<input type="checkbox"/> Selecionar Tudo <input type="text" value="Mover"/> <input type="button" value="OK"/>									

novo retorno admitido confirmado atribuído resolvido fechado

FIGURA 5.15 – TELA EXIBINDO OS ERROS QUE FORAM CORRIGIDOS PELOS DESENVOLVEDORES

- f) Ao receber o relato de que o erro no *Mantis* tem o seu status de acordo com o seu término, o gerente do projeto irá retornar o erro para o responsável do teste para que ele simule o erro, confirmando assim se o erro foi mesmo consertado na versão atual do sistema. Ao fazer isso o status do erro será atualizado para Resolvido (Figura 5.16).



Acessando como: *diegopaulino* (Diego - gerente) 2013-03-13 00:36 BRT

Principal | Minha Visão | Ver Casos | Relatar Caso | Registro de Mudanças | Planejamento | Resumo | Gerenciar | Minha Conta | Sair

Recentemente Visitado: 0000006, 0000005, 0000004, 0000003

Relatori:	Monitorado Por:	Atribuído a:	Categoria:	Gravidade:	Resolução:	Perfil:
qualquer	qualquer	qualquer	qualquer	qualquer	qualquer	qualquer
Estado:	Ocultar Status:					Prioridade:
qualquer	fechado (e acima)					qualquer
Exibir:	Visibilidade:	Mostrar Casos "Pequenos":	Alterado(hrs):	Usar Filtros de Data:	Relações:	
50	qualquer	Sim	6	Não	qualquer	
Plataforma:	SO:	Versão SO:	Marcadores:			
qualquer	qualquer	qualquer				
Nota Por:	qualquer	Ordenar por:	Atualizado Descendente			
Match Type:	All Conditions					

Procurar [Aplicar Filtro] [Filtros Avançados] [Criar Link Permanente] [Redefinir Filtro] [Salvar Filtro Atual]

Visualizando Casos (1 - 1 / 1) [Imprimir Relatórios] [Exportar CSV] [Exportação para Excel]

	P	Núm	#	U	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>		0000006	1		General	obstáculo	resolvido (thiagoholiveira)	2013-03-13	War Rio - 0001 - Sistema permite ataque na primeira rodada

Selecionar Tudo Mover OK

novos
retorno
admitido
confirmado
atribuído
resolvido
fechado

FIGURA 5.16 – TELA EXIBINDO OS ERROS QUE FORAM RESOLVIDOS E RETORNADOS PARA OS SEUS RESPONSÁVEIS PELO TESTE PARA TESTÁ-LOS DE NOVO

- g) Caso o responsável pelo teste ainda esteja com problemas com relação à realização do teste, o erro que ele informou ainda está ocorrendo, por exemplo, ou caso o desenvolvedor não tenha conseguido simular o erro da mesma forma que o responsável pelo teste, ele deverá atualizar a descrição do erro no *Mantis* informando que os passos realizados não simularam tal erro (Figura 5.17) e irá mudar o seu status para Retorno (Figura 5.18).

FIGURA 5.17 – POSSÍVEIS TIPOS DE MENSAGENS QUE PODERÃO SER ENCONTRADAS CASO O ERRO NÃO TENHA SIDO CONSERTADO OU NÃO TENHA SIDO SIMULADO



Acessando como: *diegopaulino* (Diego - gerente)

2013-03-13 00:41 BRT



[Principal](#) | [Minha Visão](#) | [Ver Casos](#) | [Relatar Caso](#) | [Registro de Mudanças](#) | [Planejamento](#) | [Resumo](#) | [Gerenciar](#) |

[Minha Conta](#) | [Sair](#)

Recentemente Visitado: [0000006](#), [0000005](#), [0000004](#), [0000003](#)

Relator:	Monitorado Por:	Atribuído a:	Categoria:	Gravidade:	Resolução:	Perfil:
qualquer	qualquer	qualquer	qualquer	qualquer	qualquer	qualquer
Estado:	Ocultar Status:					Prioridade:
qualquer	fechado (e acima)					qualquer
Exibir:	Visibilidade:	Mostrar Casos "Pequenos":	Alterado(hrs):	Usar Filtros de Data:	Relações:	
50	qualquer	Sim	6	Não	qualquer	
Plataforma:	SO:	Versão SO:	Marcadores:			
qualquer	qualquer	qualquer				
Nota Por:	qualquer	Ordenar por:	Atualizado Descendente			
Match Type:	All Conditions					

[\[Filtros Avançados \]](#)
[\[Criar Link Permanente \]](#)

Visualizando Casos (1 - 1 / 1) [[Imprimir Relatórios](#)] [[Exportar CSV](#)] [[Exportação para Excel](#)]

	P	Núm	#	👤	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>		0000006	3		General	obstáculo	retorno (thiagoholiveira)	2013-03-13	War Rio - 0001 - Sistema permite ataque na primeira rodada

Selecionar Tudo

novos	retorno	admitido	confirmado	atribuído	resolvido	fechado
-------	---------	----------	------------	-----------	-----------	---------

FIGURA 5.18 – TELA EXIBINDO OS ERROS QUE NÃO FORAM RESOLVIDOS OU NÃO FORAM SIMULADOS DA FORMA COM QUE FORAM TESTADOS

- h) Caso o responsável pelo teste não tenha encontrado resquícios do erro abordado e o sistema esteja funcionando de acordo com o estabelecido com o caso de teste, ele irá retornar para o gerente do projeto que o erro no *Mantis* foi resolvido. Assim o gerente do projeto poderá atualizar o status do erro para Fechado, fazendo com que o erro saia da lista de erros a serem corrigidos para que o sistema esteja em pleno funcionamento (Figura 5.19).



Acessando como: *diegopaulino* (Diego - gerente) 2013-03-13 00:43 BRT

Principal | Minha Visão | Ver Casos | Relatar Caso | Registro de Mudanças | Planejamento | Resumo | Gerenciar | Minha Conta | Sair

Recentemente Visitado: 0000006, 0000005, 0000004, 0000003

Relator:	Monitorado Por:	Atribuído a:	Categoria:	Gravidade:	Resolução:	Perfil:
qualquer	qualquer	qualquer	qualquer	qualquer	qualquer	qualquer
Estado:	Ocultar Status:					Prioridade:
qualquer	nenhum					qualquer
Exibir:	Visibilidade:	Mostrar Casos "Pegajosos":	Alterado(hrs):	Usar Filtros de Data:	Relações:	
50	qualquer	Sim	6	Não	qualquer	
Plataforma:	SO:	Versão SO:	Marcadores:			
qualquer	qualquer	qualquer				
Nota Por:		Ordenar por:		Atualizado Descendente		
Match Type:	All Conditions					

Procurar Aplicar Filtro [Filtros Avançados] [Criar Link Permanente] Redefinir Filtro Salvar Filtro Atual

Visualizando Casos (1 - 1 / 1) [Imprimir Relatórios] [Exportar CSV] [Exportação para Excel]

	P	Núm	#	U	Categoria	Gravidade	Estado	Atualizado	Resumo
<input type="checkbox"/>		0000006	3		General	obstáculo	fechado (thiagoholiveira)	2013-03-13	War Rio - 0001 - Sistema permite ataque na primeira rodada

Selecionar Tudo Mover OK

nov retorno admitido confirmado atribuido resolvido fechado

FIGURA 5.19 – TELA EXIBINDO OS ERROS QUE JÁ FORAM CONSERTADOS

6. VISÃO DE PROFISSIONAIS DA ÁREA SOBRE O RQM E O MANTIS

Este capítulo traz a visão de profissionais da área de ciência da computação sobre o *Mantis* e o *RQM*. Todos os profissionais ouvidos (desenvolvedores, analistas, testadores, estagiários e líderes de equipe) já haviam tido contato com uma ou as duas ferramentas.

O questionário utilizado se baseou no modelo de *SQFD* (*Software Quality Function Deployment*). O método de testes *SQFD* é uma variação voltada para produtos de *software* do método *QFD*. Esse método foi desenvolvido na década de 60 no Japão pelos professores Yoji Akao e Shigeru Mizuno com o intuito de capturar as maiores necessidades e desejos de clientes ou consumidores de um produto manufaturado, facilitando o trabalho de desenvolvedores em dar vida aos desejos dos clientes e consumidores (Akao, 1990).

O *SQFD* se baseia em adaptar os conceitos criados no *QFD* com o foco em produtos de *software*. Ele foi utilizado da seguinte forma (Velo, Neto et al, 2010):

- O questionário foi subdividido em três partes: Nível de avaliação, Função/Característica do produto e Perguntas; onde as perguntas são dependentes da função/característica do produto, que é dependente do nível de avaliação no momento.
- Ao estabelecer quais seriam os níveis de avaliação, foram debatidos quais seriam as funções/características mais pertinentes para o objetivo do questionário.
- Com as funções/características escolhidas deu-se início à escolha das perguntas que mais se encaixassem no que as funções/características pedem.

- Após se estabelecer o nível de avaliação, as funções/características a serem abordadas e as perguntas a serem feitas, foram definidos os que cada resposta dada teria. A todas as perguntas de uma função/característica foi atribuído um peso total de 100% e a todas as funções/características de um nível de avaliação também tiveram um peso total de 100%.
- Com as respostas em mãos, determinou-se a pontuação final do *RQM* ou do *Mantis* pela seguinte fórmula:

- I. -Pergunta = Soma (PesoPergunta * PesoResposta);
- II. -Função/Característica = Soma (PesoFunção/Característica * Pergunta);
- III. - Nível de avaliação = Soma (Função/Característica);
- IV. Total = Soma (Nível de avaliação).

Os questionários foram disponibilizados aos profissionais por cerca de 30 dias. Como esperado para esta abordagem, o número de respostas foi baixo (treze respostas foram recebidas), o que pode representar uma limitação deste trabalho. No entanto, foi possível gerar uma análise de cada uma das ferramentas e fazer uma comparação entre elas. De todas as respostas recebidas, 42% foram de usuários do *Mantis* e 58% delas do *RQM*. Com a pontuação obtida com as repostas relacionadas à Função/Características e ao Nível de avaliação de cada uma (Apêndice A), foi obtido o seguinte resultado final:

- I. Total (Mantis) = $(2.813 + 0.6 + 1.52 + 3.6 + 5.25 + 3.26 + 3.07 + 3.885) = 23.998$;
- II. Total (RQM) = $(5.783 + 5.95 + 6.14 + 5.7 + 5.25 + 5.13 + 5.13 + 5.555) = 44.638$.

Os resultados obtidos estão de acordo com as observações feitas durante o estudo das ferramentas. Por possuir uma pontuação maior em todos os Níveis de Avaliação abordado, o *RQM* acabou sendo a ferramenta melhor avaliada. E isso se deve ao fato de possuir uma grande integração com as outras ferramentas de teste da *IBM* e ter a possibilidade de criar um plano de teste completo, enquanto o *Mantis* executa apenas o gerenciamento de testes.

Isto reforça a observação de que o *RQM* é uma ferramenta mais completa, tendo inúmeros recursos para um gerenciamento de testes mais eficiente tendo, entretanto, o limitante de ser uma ferramenta paga. O *Mantis* tem a vantagem de ser gratuita, possui bem menos recursos, mas é apontada como uma ferramenta que executa seu objetivo de maneira eficiente. Assim, caberá ao líder de teste avaliar a possibilidade de utilizar uma ou outra

ferramenta em função da disponibilidade de recursos, da demanda em termos de gerenciamento de teste e qual das ferramentas seria essencial no momento atual do projeto.

As perguntas realizadas, seus pesos e os questionários respondidos pelos profissionais do ramo podem ser encontrados no Apêndice B, Apêndice C e Apêndice D.

7. CONSIDERAÇÕES FINAIS

Como foi apontado tanto por experiências pessoais quanto por profissionais com certo tempo de mercado, tanto o *Mantis* quanto o *RQM* são ferramentas que executam com competência aquilo que lhes é demandado. A maior gama de recursos do *RQM* permite além da análise completa do *bug* que foi encontrado e do desenvolvimento de relatórios de teste, criar um plano de teste com um alto nível de detalhamento. Assim, permite criar poderosos casos de teste e auxilia a execução de testes manuais. Outra característica relevante é a possibilidade de integração com outras ferramentas usadas para o mesmo projeto, coisa que o *Mantis* não realiza, focando somente no gerenciamento dos erros encontrados.

Já o *Mantis* se diferencia do *RQM* por ser uma ferramenta livre, fazendo com que os usuários possam customizá-la de acordo com suas necessidades. Mas seu uso acaba sendo recomendado para aqueles que já tiveram contato com ela antes, pois sua configuração desde o momento inicial até o seu funcionamento não é uma tarefa fácil e pode acabar afastando aqueles que só precisam utilizá-lo para controlar os erros em pequenos projetos.

Entretanto, apesar da análise feita comparando o registro de um defeito em cada ferramenta e do resultado da pesquisa com profissionais do mercado apontando o *RQM* a melhor comparada ao *Mantis*, fica a critério do líder de teste escolher a melhor ferramenta de acordo com as necessidades do seu projeto verificando critérios como custo, tamanho do projeto, facilidade de uso, experiência dos profissionais com a ferramenta e necessidade de integração com outras.

A metodologia de trabalho utilizado, apesar do número restrito de profissionais ouvido, permitiu evidenciar as principais características e vantagens de cada uma das ferramentas estudadas, podendo assim contribuir com o processo de escolha dos líderes de teste.

8. REFERÊNCIAS BIBLIOGRÁFICAS

AKAO, Y. Quality Function Deployment. Cambridge MA. Productivity Press. 1990.

BARTIE, Alexandre. Processo de Teste de Software – Parte 01. Disponível em: <<http://imasters.com.br/artigo/6102/software/processo-de-teste-de-software-parte-01/>>. Acesso em: 25 de fev. 2013.

CASO DE TESTE. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikipédia Foundation, 2013. Disponível em: <http://pt.wikipedia.org/w/index.php?title=Caso_de_teste>. Acesso em: 25 de fev. 2013.

ESPINHA, Rafael. Lab 01 – Rational Quality Manager. Prático em Teste de Software. Notas de Aula. CCE PUC-Rio.

FOUNDATION LEVEL SYLLABUS: BSTQB. Versão 2.1.1br. 2011. Disponível em: <<http://www.bstqb.org.br>>. Acesso em: 29 de nov. 2012.

GETEC SISTEMAS. Qualidade de software, uma necessidade Disponível em: <<http://www.devmedia.com.br/qualidade-de-software-uma-necessidade/9243>>. Acesso em: 25 de nov. 2012.

GLOSSÁRIO PADRÃO DE TERMOS UTILIZADOS EM TESTE DE SOFTWARE: BSTQB. Versão 2.2br. 2012. Disponível em: <<http://www.bstqb.org.br>>. Acesso em: 25 de fev. 2013.

MANTIS BUG TRACKER. 2013. Disponível em <<http://www.mantisbt.org>>. Acesso em: 15 de dez. 2012.

MARS CLIMATE ORBITER. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikipédia Foundation, 2013. Disponível em: <http://pt.wikipedia.org/w/index.php?title=Mars_Climate_Orbiter>. Acesso em: 25 de nov. 2012.

MURTA,L.G.P. Verificação, Validação e Testes. Engenharia de Software II. Notas de aula. Universidade Federal Fluminense. 2012. Disponível em: <<http://www.ic.uff.br/~leomurta/courses/2012.2/es2/aula6.pdf>> . Acesso em: 25 de fev. 2013.

RATIONAL QUALITY MANAGER. 2013. Disponível em < <https://jazz.net/products/rational-quality-manager/>>. Acesso em: 07 de jan. 2013

VELOSO, Janielton de Sousa; Neto, Pedro de Alcântara dos S. et al. Avaliação de Ferramentas de Apoio ao Teste de Sistemas de Informação. iSys, Revista Brasileira de Sistemas de Informação. PPGI / UNIRIO, v. 1.; p.1-17. 2010.

WEISZFLOG, W. Michaelis Moderno Dicionário da Língua Portuguesa. 2012. Editora Melhoramentos Ltda. – nova ortografia. Disponível em: <<http://michaelis.uol.com.br/moderno/portugues/index.php?lingua=portugues-portugues&palavra=testar>>. Acesso em: 25 de fev. 2013.

APÊNDICE A

Neste apêndice será exibida a pontuação do *Mantis* e do *RQM* relacionada às questões que cada uma obteve relacionada às suas Funções/Características e seus Níveis de avaliação:

- Pontuação obtida da Função/Característica do *Mantis*:

- I. Gestão dos dados do plano de teste (Mantis) = $\{0.15 * [1 * (4*1 + 2*0)]\} = 0.15 * 4 = 0.6$;
- II. Estimador de complexidade de caso de uso (Mantis) = $\{0.05 * [1 * (2*1 + 3*0 + 1*0.5)]\} = 0.05 * 2.5 = 0.125$;
- III. Estimar de prazos para execução de tarefas de teste (Mantis) = $\{0.05 * [1 * (2*1 + 3*0)]\} = 0.05 * 2 = 0.1$;
- IV. Base de dados histórica de projetos (Mantis) = $\{0.05 * \{[0.7 * (2*1 + 3*0 + 1*0.5)] + [0.3 * (3*1 + 2*0.5)]\}\} = \{0.05 * \{1.75 + 4\}\} = 0.05 * 4.75 = 0.238$;
- V. Calculador de produtividade (Mantis) = $\{0.2 * [1 * (1*1 + 3*0 + 1*0.5)]\} = 0.2 * 1.5 = 0.3$;
- VI. Registro de tarefas (Mantis) = $\{0.2 * [1 * (5*1 + 1*0)]\} = 0.2 * 5 = 1$;
- VII. Alocador inteligente de tarefa (Mantis) = $\{0.3 * [1 * (1*1 + 4*0 + 1*0.5)]\} = 0.3 * 1.5 = 0.45$;

- VIII. Integração com ferramentas de gestão de requisitos (Mantis) = $\{0.6 * \{[0.5 * (1*1 + 4*0)] + [0.5 * (1*1 + 3*0)]\}\} = \{0.6 * \{0.5 + 0.5\}\} = 0.6 * 1 = 0.6;$
- IX. Detector de alterações entre as visões, exibindo partes afetadas? (Mantis) = $\{0.4 * [1 * (4*0)]\} = 0.4 * 0 = 0;$
- X. Integração com ferramentas de acompanhamento de bugs (Mantis) = $\{0.4 * \{[0.7 * (4*0 + 1*0.5)] + [0.3 * (4*1 + 1*0)]\}\} = \{0.4 * \{0.35 + 1.2\}\} = 0.4 * 1.55 = 0.62;$
- XI. Integração com ferramentas de gestão de configuração (Mantis) = $\{0.6 * [1 * (1*1 + 3*0 + 1*0.5)]\} = 0.6 * 1.5 = 0.9;$
- XII. Gerador de relatório com formato definido pelo usuário (Mantis) = $\{0.4 * [1 * (2*1 + 4*0)]\} = 0.4 * 4 = 1.6;$
- XIII. Gerador de gráficos com fonte de dados e formato definido pelo usuário (Mantis) = $\{0.4 * [1 * (4*1 + 2*0)]\} = 0.4 * 4 = 1.6;$
- XIV. Uso de hiperlinks e agrupadores nos relatórios (Mantis) = $\{0.2 * [1 * (1*1 + 3*0 + 2*0.5)]\} = 0.2 * 2 = 0.4;$
- XV. Cadastro de usuários (Mantis) = $\{0.25 * [1 * (6*1)]\} = 0.25 * 6 = 1.5;$
- XVI. Cadastro de grupos (Mantis) = $\{0.25 * [1 * (5*1 + 1*0.5)]\} = 0.25 * 5.5 = 1.375;$
- XVII. Cadastro de equipe (Mantis) = $\{0.25 * [1 * (5*1 + 1*0.5)]\} = 0.25 * 5.5 = 1.375;$
- XVIII. Cadastro de perfis com possibilidade de definição de permissões por grupo, projeto, equipe (Mantis) = $\{0.25 * [1 * (4*1 + 1*0)]\} = 0.25 * 4 = 1;$
- XIX. Seguir um guia de estilo (Mantis) = $\{0.3 * [1 * (1*1 + 2*0 + 2*0.5)]\} = 0.3 * 2 = 0.6;$
- XX. Utilizar terminologia adequada ao contexto (Mantis) = $\{0.7 * \{[0.4 * (2*1 + 4*0)] + [0.6 * (5*1 + 1*0)]\}\} = \{0.7 * \{0.8 + 3\}\} = 0.7 * 3.8 = 2.66;$
- XXI. Help on-line (Mantis) = $\{0.3 * \{[0.6 * (3*1 + 2*0 + 1*0.5)] + [0.4 * (2*1 + 3*0 + 1*0.5)]\}\} = \{0.3 * \{2.1 + 1\}\} = 0.3 * 3.1 = 0.93;$
- XXII. Manual de usuário (Mantis) = $\{0.3 * [1 * (2*1 + 1*0 + 3*0.5)]\} = 0.3 * 3.5 = 1.05;$
- XXIII. Sítio de apoio com exemplos de uso (Mantis) = $\{0.2 * \{[0.3 * (3*1 + 1*0 + 2*0.5)] + [0.7 * (2*1 + 3*0 + 1*0.5)]\}\} = \{0.2 * \{1.2 + 1.75\}\} = 0.2 * 2.95 = 0.59;$

- XXIV. Curso de formação (Mantis) = $\{0.2 * [1 * (2*1 + 3*0 + 1*0.5)]\} = 0.2 * 2.5 = 0.5$;
- XXV. Executor de teste com possibilidade de pausa e retomada da execução (Mantis) = $\{0.3 * \{[0.4 * (2*1 + 2*0 + 1*0.5)] + [0.4 * (2*1 + 3*0 + 1*0.5)] + [0.2 * (5*1 + 1*0)]\}\} = \{0.3 * \{1 + 1 + 5\}\} = 0.3 * 7 = 2.1$;
- XXVI. Agrupador e escalonador de testes (Mantis) = $\{0.1 * \{[0.6 * (1*1 + 5*0)] + [0.4 * (3*1 + 2*0 + 1*0.5)]\}\} = \{0.1 * \{0.6 + 1.4\}\} = 0.1 * 2 = 0.2$;
- XXVII. Gerador de log de execução de testes (Mantis) = $\{0.2 * \{[0.2 * (3*1 + 2*0 + 1*0.5)] + [0.8 * (1*1 + 2*0 + 2*0.5)]\}\} = \{0.2 * \{0.7 + 1.6\}\} = 0.2 * 2.3 = 0.46$;
- XXVIII. Analisador de falhas (Mantis) = $\{0.3 * \{[0.5 * (2*1 + 1*0 + 2*0.5)] + [0.5 * (3*1 + 1*0 + 1*0.5)]\}\} = \{0.3 * \{1.5 + 1.75\}\} = 0.3 * 3.25 = 0.975$;
- XXIX. Cadastramento automático de falhas (Mantis) = $\{0.1 * [1 * (1*1 + 3*0 + 1*0.5)]\} = 0.1 * 1.5 = 0.15$.

- Pontuação obtida da Função/Característica do *RQM*:

- I. Gestão dos dados do plano de teste (RQM) = $\{0.15 * [1 * (6*1 + 1*0.5)]\} = 0.15 * 6.5 = 0.975$;
- II. Estimador de complexidade de caso de uso (RQM) = $\{0.05 * [1 * (3*1 + 1*0 + 2*0.5)]\} = 0.05 * 4 = 0.2$;
- III. Estimar de prazos para execução de tarefas de teste (RQM) = $\{0.05 * [1 * (5*1 + 1*0.5)]\} = 0.05 * 5.5 = 0.275$;
- IV. Base de dados histórica de projetos (RQM) = $\{0.05 * \{[0.7 * (6*1 + 1*0.5)] + [0.3 * (7*1)]\}\} = \{0.05 * \{4.55 + 2.1\}\} = 0.05 * 6.65 = 0.333$;
- V. Calculador de produtividade (RQM) = $\{0.2 * [1 * (5*1 + 2*0.5)]\} = 0.2 * 6 = 1.2$;
- VI. Registro de tarefas (RQM) = $\{0.2 * [1 * (6*1 + 1*0.5)]\} = 0.2 * 6.5 = 1.3$;
- VII. Alocaador inteligente de tarefa (RQM) = $\{0.3 * [1 * (4*1 + 2*0.5)]\} = 0.3 * 5 = 1.5$;

- VIII. Integração com ferramentas de gestão de requisitos (RQM) = $\{0.6 * \{[0.5 * (5*1 + 2*0.5)] + [0.5 * (4*1 + 2*0 + 1*0.5)]\}\} = \{0.6 * \{3 + 2.25\}\} = 0.6 * 5.25 = 3.15;$
- IX. Detector de alterações entre as visões, exibindo partes afetadas? (RQM) = $\{0.4 * [1 * (7*1)]\} = 0.4 * 7 = 2.8;$
- X. Integração com ferramentas de acompanhamento de bugs (RQM) = $\{0.4 * \{[0.7 * (6*1 + 1*0.5)] + [0.3 * (5*1 + 2*0.5)]\}\} = \{0.4 * \{4.55 + 1.8\}\} = 0.4 * 6.35 = 2.54;$
- XI. Integração com ferramentas de gestão de configuração (RQM) = $\{0.6 * [1 * (5*1 + 2*0.5)]\} = 0.6 * 6 = 3.6;$
- XII. Gerador de relatório com formato definido pelo usuário (RQM) = $\{0.4 * [1 * (5*1 + 2*0.5)]\} = 0.4 * 6 = 2.4;$
- XIII. Gerador de gráficos com fonte de dados e formato definido pelo usuário (RQM) = $\{0.4 * [1 * (5*1 + 1*0 + 1*0.5)]\} = 0.4 * 5.5 = 2.2;$
- XIV. Uso de hiperlinks e agrupadores nos relatórios (RQM) = $\{0.2 * [1 * (5*1 + 1*0 + 1*0.5)]\} = 0.2 * 5.5 = 1.1;$
- XV. Cadastro de usuários (RQM) = $\{0.25 * [1 * (4*1 + 2*0.5)]\} = 0.25 * 5 = 1.25;$
- XVI. Cadastro de grupos (RQM) = $\{0.25 * [1 * (5*1 + 2*0.5)]\} = 0.25 * 6 = 1.5;$
- XVII. Cadastro de equipe (RQM) = $\{0.25 * [1 * (4*1 + 2*0.5)]\} = 0.25 * 5 = 1.25;$
- XVIII. Cadastro de perfis com possibilidade de definição de permissões por grupo, projeto, equipe (RQM) = $\{0.25 * [1 * (4*1 + 2*0.5)]\} = 0.25 * 5 = 1.25;$
- XIX. Seguir um guia de estilo (RQM) = $\{0.3 * [1 * (4*1 + 2*0 + 1*0.5)]\} = 0.3 * 4.5 = 1.35;$
- XX. Utilizar terminologia adequada ao contexto (RQM) = $\{0.7 * \{[0.4 * (5*1 + 2*0.5)] + [0.6 * (3*1 + 4*0.5)]\}\} = \{0.7 * \{2.4 + 3\}\} = 0.7 * 5.4 = 3.78;$
- XXI. Help on-line (RQM) = $\{0.3 * \{[0.6 * (3*1 + 1*0 + 2*0.5)] + [0.4 * (3*1 + 3*0.5)]\}\} = \{0.6 * \{2.4 + 1.8\}\} = 0.6 * 4.2 = 2.52;$
- XXII. Manual de usuário (RQM) = $\{0.3 * [1 * (2*1 + 1*0 + 3*0.5)]\} = 0.3 * 3.5 = 1.05;$
- XXIII. Sítio de apoio com exemplos de uso (RQM) = $\{0.2 * \{[0.3 * (4*1 + 1*0 + 1*0.5)] + [0.7 * (3*1 + 2*0 + 1*0.5)]\}\} = \{0.2 * \{1.35 + 2.45\}\} = 0.2 * 3.8 = 0.76;$
- XXIV. Curso de formação (RQM) = $\{0.2 * [1 * (3*1 + 1*0 + 2*0.5)]\} = 0.2 * 4 = 0.8;$

- XXV. Executor de teste com possibilidade de pausa e retomada da execução (RQM)
 $= \{0.3 * \{[0.4 * (6*1 + 1*0.5)] + [0.4 * (5*1 + 2*0.5)] + [0.2 * (6*1 + 1*0.5)]\}\} = \{0.3 * \{2.6 + 2.4 + 1.3\}\} = 0.3 * 6.3 = 1.89;$
- XXVI. Agrupador e escalonador de testes (RQM) = $\{0.1 * \{[0.6 * (5*1 + 2*0.5)] + [0.4 * (4*1 + 3*0.5)]\}\} = \{0.1 * \{3.6 + 2.2\}\} = 0.1 * 5.8 = 0.58;$
- XXVII. Gerador de log de execução de testes (RQM) = $\{0.2 * \{[0.2 * (4*1 + 2*0 + 1*0.5)] + [0.8 * (5*1 + 1*0 + 1*0.5)]\}\} = \{0.2 * \{0.9 + 4.4\}\} = 0.2 * 5.3 = 1.06;$
- XXVIII. Analisador de falhas (RQM) = $\{0.3 * \{[0.5 * (3*1 + 1*0 + 3*0.5)] + [0.5 * (4*1 + 2*0.5)]\}\} = \{0.3 * \{2.25 + 2.5\}\} = 0.3 * 4.75 = 1.425;$
- XXIX. Cadastramento automático de falhas (RQM) = $\{0.1 * [1 * (5*1 + 2*0.5)]\} = 0.1 * 6 = 0.6.$

- Pontuação obtida do Nível de avaliação do *Mantis*:

- I. Gerador de plano de teste (Mantis) = $(0.6 + 0.125 + 0.1 + 0.238 + 0.3 + 1 + 0.45) = 2.813;$
- II. Rastreador (Mantis) = $(0.6 + 0) = 0.6;$
- III. Integrador (Mantis) = $(0.62 + 0.9) = 1.52;$
- IV. Gerador de relatórios (Mantis) = $(1.6 + 1.6 + 0.4) = 3.6;$
- V. Suporte da ferramenta (Mantis) = $(1.5 + 1.375 + 1.375 + 1) = 5.25;$
- VI. Arquitetura da ferramenta (Mantis) = $(0.6 + 2.66) = 3.26;$
- VII. Auxílio da ferramenta (Mantis) = $(0.93 + 1.05 + 0.59 + 0.5) = 3.07;$
- VIII. Executor de teste (Mantis) = $(2.1 + 0.2 + 0.46 + 0.975 + 0.15) = 3.885.$

- Pontuação obtida do Nível de avaliação do *RQM*:

- I. Gerador de plano de teste (RQM) = $(0.975 + 0.2 + 0.275 + 0.333 + 1.2 + 1.3 + 1.5) = 5.783;$
- II. Rastreador (RQM) = $(3.15 + 2.8) = 5.95;$
- III. Integrador (RQM) = $(2.54 + 3.6) = 6.14;$

- IV. Gerador de relatórios (RQM) = $(2.4 + 2.2 + 1.1) = 5.7$;
- V. Suporte da ferramenta (RQM) = $(1.25 + 1.5 + 1.25 + 1.25) = 5.25$;
- VI. Arquitetura da ferramenta (RQM) = $(1.35 + 3.78) = 5.13$;
- VII. Auxílio da ferramenta (RQM) = $(2.52 + 1.05 + 0.76 + 0.8) = 5.13$;
- VIII. Executor de teste (RQM) = $(1.89 + 0.58 + 1.06 + 1.425 + 0.6) = 5.555$.

APÊNDICE B

Neste apêndice será exibido o questionário utilizado na pesquisa de opinião com os profissionais da área de Tecnologia da Informação que já tiveram experiências no uso de testes em seus projetos.

Questionário Mantis e RQM

Questionário para auxiliar a confecção do projeto final "A visão sobre gerenciamento de testes através das ferramentas *Mantis* e *RQM*" para Universidade Federal Fluminense por Diego Paulino e Thiago Henriques´

Sobre qual ferramenta você responderá o questionário?

Caso deseje responder sobre as duas ferramentas, escolha uma delas agora e ao fim, escolha a outra.

- Mantis
- RQM (Rational Quality Manager)

A ferramenta auxilia a geração de um plano de teste?

- Sim
- Não
- Satisfatório

A ferramenta consegue estimar a complexidade de um caso de uso?

- Sim
- Não
- Satisfatório

A ferramenta auxilia a estimativa de tempo para execução de uma tarefa de desenvolvimento de teste?

- Sim
- Não
- Satisfatório

A ferramenta registra dados sobre esforço envolvido na execução das atividades de teste?

- Sim
- Não
- Satisfatório

A ferramenta permite a busca de itens no histórico através de parâmetros como data, dia, funcionalidade testada e outras?

- Sim
- Não
- Satisfatório

A ferramenta é capaz de calcular a produtividade de um testador?

- Sim
- Não
- Satisfatório

A ferramenta possui um registro das tarefas a serem executadas?

- Sim
- Não
- Satisfatório

A ferramenta auxilia na alocação de tarefas, sugerindo uma alocação ótima, levando em consideração produtividade dos testadores, complexidade, esforço e tempo disponível?

- Sim

- Não
- Satisfatório

A ferramenta possui facilidade de integração com ferramentas de cadastro de requisitos?

- Sim
- Não
- Satisfatório

A ferramenta verifica mudanças no cadastro de requisitos?

- Sim
- Não
- Satisfatório

A ferramenta possui acesso aos artefatos do sistema e ao cadastro de requisitos verificando alterações entre itens relacionados?

- Sim
- Não
- Satisfatório

A ferramenta acessa uma ferramenta de acompanhamento de falhas, possibilitando o cadastro automático das mesmas?

- Sim
- Não
- Satisfatório

A ferramenta utiliza dados de cadastro de falhas para estatísticas de teste?

- Sim
- Não
- Satisfatório

A ferramenta consegue acessar um repositório central para consultar e alterar os artefatos de testes existentes?

- Sim
- Não
- Satisfatório

A ferramenta gera relatórios sobre os testes, com especificações definidas pelo usuário, incluindo as fontes de dados a serem utilizadas?

- Sim
- Não
- Satisfatório

A ferramenta permite gerar gráficos com o resultado dos testes e especificação de formato definido pelo usuário e seleção de fontes de dados?

- Sim
- Não
- Satisfatório

A ferramenta gera relatórios organizando os testes agrupados?

- Sim
- Não
- Satisfatório

A ferramenta possibilita o cadastramento de usuários?

- Sim
- Não
- Satisfatório

A ferramenta possibilita o cadastramento de grupos de usuários?

- Sim
- Não
- Satisfatório

A ferramenta possibilita o cadastro de equipes de teste?

- Sim
- Não
- Satisfatório

A ferramenta permite cadastro de perfis configurando restrições de acesso por usuários, grupos ou projetos?

- Sim

- Não
- Satisfatório

A ferramenta utiliza um padrão para o desenho da interface com o usuário?

- Sim
- Não
- Satisfatório

A ferramenta usa termos conhecidos, apoiados por padrões da indústria como IEEE, ISO e ABNT?

- Sim
- Não
- Satisfatório

A ferramenta possui uma linguagem de fácil entendimento?

- Sim
- Não
- Satisfatório

A ferramenta possui ajuda on-line através de documentação?

- Sim
- Não
- Satisfatório

A ferramenta mantém um help-desk?

- Sim
- Não
- Satisfatório

A ferramenta apresenta documentação ou manual de utilização inteligível?

- Sim
- Não
- Satisfatório

A ferramenta possui um site, blog, etc., de apoio com manuais e exemplos de uso?

- Sim
- Não
- Satisfatório

A ferramenta explica funcionalidade utilizando exemplos passo a passo?

- Sim
- Não
- Satisfatório

A ferramenta possui treinamento formal explicando como utilizar o software?

- Sim
- Não
- Satisfatório

A ferramenta permite parar a execução do teste a qualquer instante?

- Sim
- Não
- Satisfatório

A ferramenta permite reiniciar a execução dos testes após a execução ter sido suspensa?

- Sim
- Não
- Satisfatório

A ferramenta apresenta a possibilidade de adicionar comentários aos casos de testes criados?

- Sim
- Não
- Satisfatório

A ferramenta permite escalonar quais casos de testes serão executados dentre uma bateria de testes?

- Sim
- Não
- Satisfatório

A ferramenta permite agrupar os casos de testes?

- Sim
- Não
- Satisfatório

A ferramenta gera log de execução permitindo visualização gráfica?

- Sim
- Não
- Satisfatório

A ferramenta gera log de execução permitindo visualização gráfica?

- Sim
- Não
- Satisfatório

O log de execução da ferramenta permite a identificação do momento e causas possíveis da falha?

- Sim
- Não
- Satisfatório

A ferramenta faz avaliação e classificação das falhas descobertas?

- Sim
- Não
- Satisfatório

A ferramenta faz comparativos de falhas descobertas com as mantidas no histórico?

- Sim
- Não
- Satisfatório

A ferramenta consegue cadastrar falhas através da integração com ferramentas de acompanhamentos de falhas?

- Sim
- Não

Satisfatório

Trabalhou com qual das duas ferramentas? Como foi sua experiência com ela(s)? Você a(s) recomendaria para ser (em) utilizada em futuros projetos?

Quais foram os maiores desafios na utilização da ferramenta? Ela conseguiu ser útil no projeto em que você está ou estava alocado?

Quais pontos positivos e negativos você poderia apontar nela comparada com outras ferramentas com que você já trabalhou ou trabalha?

Qual sua profissão?

Por quanto tempo você utiliza/utilizou a ferramenta?

APÊNDICE C

Neste apêndice será exibido o modelo SQFD utilizado, com os pesos escolhidos para o Nível de Avaliação, a Função/Característica do produto, as Perguntas e as Respostas.

Nível de avaliação	Função/Característica do produto	Perguntas
Gerador de plano de teste	Gestão dos dados do plano de teste (0,15)	A ferramenta auxilia a geração de um plano de teste? (1,0)
	Estimador de complexidade de caso de uso (0,05)	A ferramenta consegue estimar a complexidade de um caso de uso? (1,0)
	Estimar de prazos para execução de tarefas de teste (0,05)	A ferramenta auxilia a estimativa de tempo para execução de uma tarefa de desenvolvimento de teste? (1,0)
	Base de dados histórica de projetos (0,05)	A ferramenta registra dados sobre esforço envolvido na execução das atividades de teste? (0,7)
		A ferramenta permite a busca de itens no histórico através de parâmetros como data, dia, funcionalidade testada e outras? (0,3)
	Calculador de produtividade (0,2)	A ferramenta é capaz de calcular a produtividade de um testador? (1,0)
	Registro de tarefas (0,2)	A ferramenta possui um registro das tarefas a serem executadas? (1,0)
Alocador inteligente de tarefa (0,3)	A ferramenta auxilia na alocação de tarefas, sugerindo uma alocação ótima, levando em consideração produtividade dos testadores, complexidade, esforço e tempo disponível? (1,0)	
Rastreador	Integração com ferramentas de gestão de requisitos (0,6)	A ferramenta possui facilidade de integração com ferramentas de cadastro de requisitos? (0,5)
		A ferramenta verifica mudanças no cadastro de requisitos? (0,5)

	Detector de alterações entre as visões, exibindo partes afetadas? (0,4)	A ferramenta possui acesso aos artefatos do sistema e ao cadastro de requisitos verificando alterações entre itens relacionados? (1,0)
Integrador	Integração com ferramentas de acompanhamento de bugs (0,4)	A ferramenta acessa uma ferramenta de acompanhamento de falhas, possibilitando o cadastro automático das mesmas? (0,7) A ferramenta utiliza dados de cadastro de falhas para estatísticas de teste? (0,3)
	Integração com ferramentas de gestão de configuração (0,6)	A ferramenta consegue acessar um repositório central para consultar e alterar os artefatos de testes existentes? (1,0)
Gerador de relatórios	Gerador de relatório com formato definido pelo usuário (0,4)	A ferramenta gera relatórios sobre os testes, com especificações definidas pelo usuário, incluindo as fontes de dados a serem utilizadas? (1,0)
	Gerador de gráficos com fonte de dados e formato definido pelo usuário (0,4)	A ferramenta permite gerar gráficos com o resultado dos testes e especificação de formato definido pelo usuário e seleção de fontes de dados? (1,0)
	Uso de hiperlinks e agrupadores nos relatórios (0,2)	A ferramenta gera relatórios organizando os testes agrupados? (1,0)
Suporte da ferramenta	Cadastro de usuários (0,25)	A ferramenta possibilita o cadastramento de usuários? (1,0)
	Cadastro de grupos (0,25)	A ferramenta possibilita o cadastramento de grupos de usuários? (1,0)
	Cadastro de equipe (0,25)	A ferramenta possibilita o cadastro de equipes de teste? (1,0)
	Cadastro de perfis com possibilidade de definição de permissões por grupo, projeto, equipe (0,25).	A ferramenta permite cadastro de perfis configurando restrições de acesso por usuários, grupos ou projetos? (1,0)
Arquitetura de ferramenta	Seguir um guia de estilo (0,3)	A ferramenta utiliza um padrão para o desenho da interface com o usuário? (1,0)
	Utilizar terminologia adequada ao contexto (0,7)	A ferramenta usa termos conhecidos, apoiados por padrões da indústria como IEEE, ISO e ABNT? (0,4) A ferramenta possui uma linguagem de fácil entendimento? (0,6)
Auxílio da ferramenta	Help on-line (0,3)	A ferramenta possui ajuda on-line através de documentação? (0,6) A ferramenta mantém um help-desk? (0,4)

	Manual de usuário (0,3)	A ferramenta apresenta documentação ou manual de utilização inteligível? (1,0)
	Sítio de apoio com exemplos de uso (0,2)	A ferramenta possui um site, blog, etc., de apoio com manuais e exemplos de uso? (0,3)
		A ferramenta explica funcionalidade utilizando exemplos passo a passo? (0,7)
	Curso de formação (0,2)	A ferramenta possui treinamento formal explicando como utilizar o software? (1,0)
Executor de teste	Executor de teste com possibilidade de pausa e retomada da execução (0,3)	A ferramenta permite parar a execução do teste a qualquer instante? (0,4)
		A ferramenta permite reiniciar a execução dos testes após a execução ter sido suspensa? (0,4)
		A ferramenta apresenta a possibilidade de adicionar comentários aos casos de testes criados? (0,2)
	Agrupador e escalonador de testes (0,1)	A ferramenta permite escalonar quais casos de testes serão executados dentre uma bateria de testes? (0,6)
		A ferramenta permite agrupar os casos de testes? (0,4)
	Gerador de log de execução de testes (0,2)	A ferramenta gera log de execução permitindo visualização gráfica? (0,2)
		O log de execução da ferramenta permite a identificação do momento e causas possíveis da falha? (0,8)
	Analisador de falhas (0,3)	A ferramenta faz avaliação e classificação das falhas descobertas? (0,5)
		A ferramenta faz comparativos de falhas descobertas com as mantidas no histórico? (0,5)
	Cadastramento automático de falhas (0,1)	A ferramenta consegue cadastrar falhas através da integração com ferramentas de acompanhamentos de falhas? (1,0)

Onde Sim vale 1,0; Não vale 0,0; Satisfatório vale 0,5.

APÊNDICE D

Neste apêndice será exibido o questionário respondido pelos profissionais da área de Tecnologia da Informação que já tiveram experiências no uso de testes em seus projetos.

Sobre qual ferramenta você responderá o questionário?	Por quanto tempo você utiliza/utilizou a ferramenta?	Qual sua profissão?	A ferramenta auxilia a geração de um plano de teste?
Mantis	2 anos	Tester	Sim
Mantis			Não
RQM	1 ano e 2 meses	Analista de Teste	Sim
RQM	4 anos	Engenheiro	Sim
RQM	30 dias	Analista de Teste	Sim
RQM	10 meses	Analista de testes	Sim
RQM	1 mês	Analista de Testes	Satisfatório
RQM	2 anos	Líder de Teste	Sim
Mantis	Dois anos.	Analista de requisitos	Não
RQM	6 meses	Gerente de Projetos	Sim
Mantis	Um ano e meio	Analista de Sistemas Java	Sim
Mantis	um ano	Programador Junior	Sim
Mantis			Sim

A ferramenta consegue estimar a complexidade de um caso de uso?	A ferramenta auxilia a estimativa de tempo para execução de uma tarefa de desenvolvimento de teste?	A ferramenta registra dados sobre esforço envolvido na execução das atividades de teste?	A ferramenta permite a busca de itens no histórico através de parâmetros como data, dia, funcionalidade testada e outras?	A ferramenta é capaz de calcular a produtividade de um testador?
Não	Não	Não		Não
Sim	Não	Satisfatório	Satisfatório	
Sim	Sim	Sim	Sim	Sim
Não	Sim	Sim	Sim	Satisfatório
Satisfatório	Satisfatório	Sim	Sim	Sim
Sim	Sim	Sim	Sim	Satisfatório
Satisfatório	Sim	Satisfatório	Sim	Sim
Sim	Sim	Sim	Sim	Sim
Não	Não	Não	Sim	Satisfatório
		Sim	Sim	Sim
Não	Sim	Não	Sim	Não
Satisfatório	Sim	Sim	Sim	Sim
Sim	Não	Sim	Satisfatório	Não

A ferramenta possui um registro das tarefas a serem executadas?	A ferramenta auxilia na alocação de tarefas, sugerindo uma alocação ótima, levando em consideração produtividade dos testadores, complexidade, esforço e tempo disponível?	A ferramenta possui facilidade de integração com ferramentas de cadastro de requisitos?	A ferramenta verifica mudanças no cadastro de requisitos?	A ferramenta possui acesso aos artefatos do sistema e ao cadastro de requisitos verificando alterações entre itens relacionados?
Não	Não			
Sim	Satisfatório	Não		Não
Sim	Sim	Sim	Não	Sim
Sim	Satisfatório	Sim	Sim	Sim
Sim	Sim	Satisfatório	Satisfatório	Sim
Sim	Satisfatório	Sim	Sim	Sim
Satisfatório	Sim	Satisfatório	Não	Sim
Sim	Sim	Sim	Sim	Sim
Sim	Não	Não	Não	Não
Sim		Sim	Sim	Sim
Sim	Não	Sim	Sim	Não
Sim	Sim	Não	Não	Não
Sim	Não	Não	Não	Não

				Sim
	Satisfatório	Sim	Satisfatório	Não
	Sim	Sim	Sim	Sim
	Sim	Sim	Sim	Sim
	Sim	Sim	Sim	Satisfatório
	Sim	Sim	Sim	Sim
	Satisfatório	Satisfatório	Satisfatório	Sim
	Sim	Sim	Sim	Sim
	Sim	Sim	Sim	Sim
	Satisfatório	Satisfatório	Satisfatório	Satisfatório
	Sim	Sim	Sim	Sim
	Não	Não	Não	Não
	Não	Sim	Não	Não
	Não	Sim	Não	Não

A ferramenta permite gerar gráficos com o resultado dos testes e especificação de formato definido pelo usuário e seleção de fontes de dados?	Sim	A ferramenta gera relatórios organizando os testes agrupados?	Satisfatório	A ferramenta possibilita o cadastramento de usuários?	Sim	A ferramenta possibilita o cadastramento de grupos de usuários?	Sim	A ferramenta possibilita o cadastro de equipes de teste?	Sim	A ferramenta permite cadastro de perfis configurando restrições de acesso por usuários, grupos ou projetos?	Não
	Sim		Satisfatório		Sim		Sim		Sim		
	Sim		Não		Sim		Satisfatório		Satisfatório		
	Sim		Sim		Sim		Sim		Sim		Sim
	Sim		Sim		Sim		Sim		Sim		Sim
	Sim		Satisfatório		Sim		Sim		Sim		Sim
	Sim		Sim		Sim		Satisfatório		Satisfatório		Satisfatório
	Não		Sim		Sim		Sim		Sim		
	Sim		Não		Sim		Sim				Sim
	Sim		Sim		Sim		Sim		Sim		Sim
	Não		Sim		Sim		Sim		Sim		Sim
	Sim		Não		Sim		Sim		Sim		Sim
	Não		Não		Sim		Sim		Sim		Sim

A ferramenta utiliza um padrão para o desenho da interface com o usuário?	A ferramenta usa termos conhecidos, apoiados por padrões da indústria como IEEE, ISO e ABNT?	A ferramenta possui uma linguagem de fácil entendimento?	A ferramenta possui ajuda on-line através de documentação?	A ferramenta mantém um help-desk?	A ferramenta apresenta documentação ou manual de utilização inteligível?
	Não	Não	Sim	Não	Sim
Satisfatório	Sim	Sim	Não	Sim	Satisfatório
Sim	Sim	Sim	Sim	Sim	Sim
Não	Satisfatório	Satisfatório	Sim	Sim	Satisfatório
Satisfatório	Sim	Sim	Satisfatório	Satisfatório	Satisfatório
Não	Satisfatório	Satisfatório	Satisfatório	Satisfatório	Satisfatório
Sim	Sim	Satisfatório	Não	Satisfatório	Não
Sim	Sim	Sim	Sim	Sim	Sim
Satisfatório	Sim	Sim	Satisfatório	Satisfatório	Satisfatório
Sim	Sim	Satisfatório			
Não	Não	Sim	Sim	Não	Satisfatório
Sim	Não	Sim	Sim	Não	Sim
Não	Não	Sim	Não	Sim	Não

A ferramenta possui um site, blog, etc., de apoio com manuais e exemplos de uso?	A ferramenta explica funcionalidade utilizando exemplos passo a passo?	A ferramenta possui treinamento formal explicando como utilizar o software?	A ferramenta permite parar a execução do teste a qualquer instante?	A ferramenta permite reiniciar a execução dos testes após a execução ter sido suspensa?	A ferramenta apresenta a possibilidade de adicionar comentários aos casos de testes criados?
Satisfatório	Não	Não	Não	Não	Sim
Satisfatório	Sim	Sim	Sim	Sim	Não
Sim	Sim	Sim	Sim	Sim	Sim
Sim	Sim	Sim	Sim	Sim	Satisfatório
Sim	Satisfatório	Satisfatório	Sim	Sim	Sim
Não	Não	Satisfatório	Satisfatório	Satisfatório	Sim
Satisfatório	Não	Não	Sim	Satisfatório	Sim
Sim	Sim	Sim	Sim	Sim	Sim
Sim	Satisfatório	Satisfatório	Satisfatório	Satisfatório	Sim
			Sim	Sim	Sim
Não	Não	Não	Sim	Não	Sim
Sim	Sim	Sim	Não	Sim	Sim
Sim	Não	Não		Não	Sim

A ferramenta permite escalonar quais casos de testes serão executados dentre uma bateria de testes?	A ferramenta permite agrupar os casos de testes?	A ferramenta gera log de execução permitindo visualização gráfica?	A ferramenta gera log de execução permitindo visualização gráfica?	O log de execução da ferramenta permite a identificação do momento e causas possíveis da falha?	A ferramenta faz avaliação e classificação das falhas descobertas?
Não	Não	Não		Sim	Satisfatório
Não	Satisfatório	Satisfatório		Satisfatório	Satisfatório
Sim	Sim	Sim	Sim	Sim	Sim
Satisfatório	Satisfatório	Sim	Sim	Sim	Satisfatório
Sim	Satisfatório	Satisfatório	Satisfatório	Satisfatório	Satisfatório
Sim	Sim	Não	Não	Sim	Sim
Satisfatório	Satisfatório	Não	Não	Não	Não
Sim	Sim	Sim	Sim	Sim	Sim
Sim	Sim	Sim	Sim	Satisfatório	Sim
Sim	Sim	Sim	Sim	Sim	Satisfatório
Não	Sim	Sim	Satisfatório	Não	Não
Não	Sim	Sim	Não	Não	Sim
Não	Não	Não	Sim		

A ferramenta faz comparativos de falhas descobertas com as mantidas no histórico?		A ferramenta consegue cadastrar falhas através da integração com ferramentas de acompanhamentos de falhas?
Sim		Sim
Sim		Não
Sim		Sim
Sim		Sim
Satisfatório		Sim
Satisfatório		Satisfatório
Sim		Sim
Satisfatório		Satisfatório
		Sim
Não		Não
Sim		Não