



Maraká: Infra-estrutura Computacional para Apoiar o Planejamento e Controle dos Testes de Software

Arilo Claudio Dias Neto

Guilherme Horta Travassos

Universidade Federal do Rio de Janeiro – COPPE/Sistemas
Caixa Postal 68.511 – CEP 21.941-972 – Rio de Janeiro – RJ – Brasil
{acdn, ght}@cos.ufrj.br

***Abstract.** This paper describes Maraká, a computational infrastructure for supporting the software testing planning and control. Its requirements set has been derived from a survey applied in a real scenario of software development and knowledge acquired in the software testing technical literature. Maraká allows software engineers to follow the software testing process and the documentation of software testing activities using an international standard (IEEE Standard 829). The control of software testing activities is provided by artifacts showing charts, scheduling some information about the accomplished tests.*

***Resumo.** Este artigo descreve a infra-estrutura Maraká, uma infra-estrutura computacional que apóia o planejamento e controle de testes de software desenvolvida a partir de requisitos extraídos dos resultados obtidos em um survey, que avaliou o estado da prática das atividades de teste de software em um cenário real de desenvolvimento de software, e do conhecimento extraído na literatura técnica da área de testes de software. Maraká permite o acompanhamento do processo de testes de software e a documentação das atividades realizadas ao longo dos testes usando um padrão internacional (IEEE Standard 829). O controle das atividades de teste de software é suprido através de artefatos que descrevem gráficos de acompanhamento, cronogramas ou outras informações sobre os testes realizados.*

1. Introdução

Teste de Software é o processo de execução de um produto para determinar se ele atingiu suas especificações e funcionou corretamente no ambiente para o qual foi projetado [Whittaker 2000]. Os testes avaliam a existência de defeitos no produto através da revelação de falhas durante a sua execução.

No processo de desenvolvimento de software, todos os defeitos são humanos e, apesar do uso dos melhores métodos de desenvolvimento, ferramentas ou profissionais, permanecem presentes nos produtos [Howden 1987], o que torna a atividade de teste fundamental durante o desenvolvimento de um software, pois corresponde ao último recurso para avaliação do produto antes da sua entrega ao usuário final [Pressman 2005].

No entanto, apesar da sua eficiência, contribuindo para a qualidade do produto final, testes, assim como diversas outras práticas do processo de desenvolvimento de software, possuem problemas que precisam ser evitados durante a sua realização para garantir o seu sucesso. Segundo Mats (2001), os cinco principais problemas nas atividades de teste de software são:

- Atrasos no cronograma do projeto, deixando a equipe de teste impossibilitada de



completar os testes planejados devido à redução de recursos e tempo;

- Carência na rastreabilidade de casos de teste entre diferentes versões do sistema, dificultando o reuso e repetição dos testes após modificações nos requisitos;
- Teste manual ou não-padronizado, resultando em um grande esforço a cada início de uma nova atividade de teste;
- Incerteza sobre o que está sendo testado, devido à falta de definição dos objetivos e escopo para as atividades de teste;
- Ausência de critérios para seleção dos casos de teste, definição da sua completude e estabelecimento de um ponto de parada, dentre outros, dificultando a revelação de falhas no produto.

Como pode ser observado, esses problemas estão diretamente relacionados à ausência ou limitação das atividades de planejamento e controle dos testes de software.

Segundo Beizer (1990) e Juristo *et al.* (2004), Teste de Software é considerada a atividade mais custosa do processo de desenvolvimento, e desta forma, necessita de um bom planejamento e controle a fim de evitar perdas de recursos e atrasos no cronograma. O planejamento garante que os testes sejam preparados antes da conclusão do produto, e evita testes tendenciosos. A atividade de controle assegura que os testes planejados sejam monitorados constantemente e seus resultados sejam registrados. Com isso, a realização de testes de software sem um planejamento pode ser comparada ao desenvolvimento de um projeto sem a criação de um plano para ele, e isto ocorre, normalmente, pelos mesmos motivos: grande pressão para iniciar a fase de execução o quanto antes (no caso de testes de software a execução está relacionada à aplicação dos casos de teste, e no desenvolvimento de software está relacionada à codificação) [Mcgregor e Sykes 2001].

Apesar de notável importância das atividades de planejamento e controle de testes de software, observa-se na literatura técnica da área de testes pouca importância dada a elas. Poucas abordagens que apóiam a essas atividades são propostas, e normalmente para um contexto específico, e um número mais reduzido ainda é aplicado na indústria, caracterizando o estado da arte das atividades de planejamento e controle dos testes [Dias Neto 2004].

A partir da dificuldade de caracterização do estado da arte das atividades de planejamento e controle de teste de software, devido à carência de estudos científicos sobre este tema, torna-se necessário obter um entendimento de como estas atividades estão sendo aplicadas em ambientes reais de desenvolvimento de software para que a partir das reais necessidades das organizações, novas estratégias possam ser adotadas.

Com o intuito de caracterizar o estado da prática das atividades de teste em ambientes reais de desenvolvimento de software, foi realizado um *survey* com organizações de software localizadas em um pólo de desenvolvimento de software brasileiro cujo objetivo foi a avaliação de quais práticas de teste de software estão sendo utilizadas nesse cenário de desenvolvimento [Dias Neto *et al.* 2006]. Para isso, foi avaliado um conjunto de práticas de teste de software extraídas da literatura técnica da área quanto a sua aplicabilidade nas organizações de software e em relação à sua importância para os projetos da organização, no ponto de vista dos profissionais que atuavam nessas organizações. Os resultados indicaram uma carência e limitação na aplicação de práticas relacionadas ao planejamento e controle de teste de software



nessas organizações que participaram do *survey*, porém observou-se o interesse dessas organizações em relação a elas. Ao total, foram avaliadas 32 práticas de teste de software, mas na Tabela 1 só estão apresentadas, em ordem decrescente, as práticas consideradas importantes pelos profissionais das organizações que participaram do *survey*. Essas práticas dividem-se em categorias que expressam características específicas da atividade de teste. Podem ser práticas genéricas, que indicam um tipo de teste a ser realizado, ou práticas relacionadas à organização, planejamento, controle ou utilização de ferramentas para apoiar às atividades de teste de software.

Tabela 1. Lista de Práticas de Teste consideradas importantes no *survey* [Dias Neto *et al.* 2006].

Categoria	Práticas de Teste de Software consideradas importante por Organizações de Software que participaram do <i>Survey</i>
Organização	Existência de um responsável ou uma equipe alocada para as atividades de teste.
Genérica	Realização de Teste de Aceitação.
Planejamento	Documentação do Plano de Teste descrevendo objetivos/abordagens.
Controle	Registro de falhas e defeitos detectados durante os testes.
Organização	Re-execução dos testes quando o software é modificado.
Genérica	Realização de Teste de Unidade.
Genérica	Realização de Teste de Integração.
Genérica	Realização de Teste de Sistema.
Organização	Utilização de metodologia ou processo para sistematizar e organizar testes de software.
Planejamento	Documentação de Procedimentos e Casos de Teste e seus Resultados.
Organização	Separação das atividades de teste do desenvolvimento.
Organização	Realização de treinamentos sobre testes.
Organização	Existência de testadores em tempo integral para realização de testes.
Controle	Conservação dos dados dos testes para usos futuros.
Controle	Avaliação dos documentos de teste (testware) quanto à sua qualidade.
Ferramenta	Utilização de ferramentas de gerência de teste para rastrear e registrar os resultados da execução de teste.

Baseando-se no cenário acadêmico, que ressalta a importância das atividades de planejamento e controle para as atividades de teste de software, e nas necessidades reais de organizações de software, caracterizada a partir do *survey* realizado, este trabalho propõe a utilização de uma abordagem de apoio ao planejamento e controle de teste de software composta por dois elementos básicos: **documentação** e **sistematização** [Dias Neto e Travassos 2005]. Seguindo tal abordagem, foi desenvolvida a infra-estrutura computacional Maraká que apóia o planejamento e controle de testes de software, possibilitando o acompanhamento e monitoração de um processo de testes e a geração dos artefatos relevantes ao longo dos testes em uma organização.

Sendo assim, este artigo está organizado da seguinte forma: na seção 2 é descrita a abordagem de apoio ao planejamento e controle de testes de software e seus elementos básicos [Dias Neto e Travassos 2005]. Na seção 3 é descrita a infra-estrutura Maraká através de suas funcionalidades. Na seção 4 é apresentada uma avaliação da infra-estrutura Maraká em relação ao apoio fornecido às práticas de teste de software consideradas importantes para o desenvolvimento de software na opinião de profissionais de organizações de software que participaram do *survey* que caracterizou o estado da prática das atividades de teste em um cenário de desenvolvimento de software brasileiro [Dias Neto *et al.* 2006]. Por fim, na seção 5 são descritas as conclusões e trabalhos futuros que darão continuidade a este trabalho.



2. Abordagem de Apoio ao Planejamento e Controle de Testes de Software

A abordagem é composta por dois elementos básicos que visam apoiar as atividades de planejamento e controle dos testes de software: **documentação** e **sistematização** [Dias Neto *et al.* 2006] e foi baseada na metodologia descrita em [Crespo *et al.* 2004].

A seguir, será descrito cada elemento que compõe a abordagem definida e a estratégia adotada neste trabalho para sua aplicação em uma organização, de acordo com a estrutura definida na Figura 1.

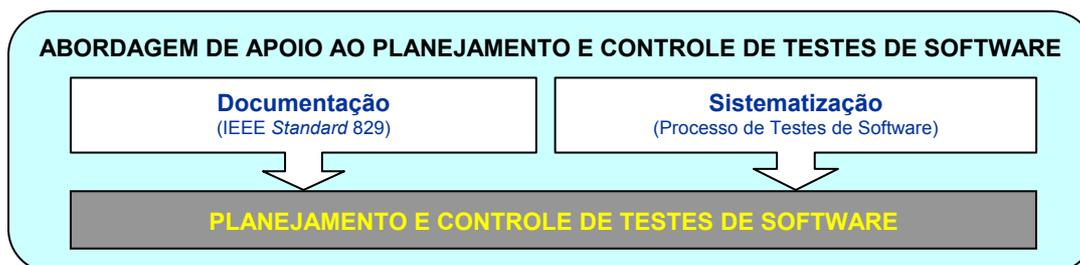


Figura 1. Composição da Abordagem de Apoio ao Planejamento e Controle de Testes de Software

2.1. Documentação dos Testes

A documentação tem como objetivo principal fornecer evidências (registros) sobre a realização de determinada atividade e facilita a comunicação entre as diferentes partes envolvidas. Este elemento da abordagem proposta contribui para o planejamento dos testes através da especificação antecipada do que deve ser feito, quando deve ser feito e quem deve fazer cada tarefa. Em relação ao controle dos testes, a documentação permite uma comparação posterior entre o que foi planejado previamente e o que ocorre efetivamente durante os testes através do registro das tarefas e dos resultados obtidos.

Como mecanismo adotado neste trabalho para documentação dos testes, será utilizado o IEEE *Standard* 829-1998 [IEEE-829 1998] – Padrão para Documentação dos Testes de Software, que foi escolhido por ser um padrão internacional, é gratuito e possui na sua estrutura um conjunto de documentos que possibilitam a documentação da preparação, execução e registro dos testes.

Os oito documentos definidos pelo padrão que cobrem as tarefas de teste são: *Plano de Teste*, *Especificação de Projeto de Teste*, *Especificação de Procedimento de Teste*, *Histórico dos Testes*, *Relatório de Incidente de Teste*, *Relatório de Resumo de Teste* e *Relatório de Encaminhamento de Item de Teste*. Uma descrição detalhada de cada documento que compõe o padrão pode ser obtida em [Crespo *et al.* 2004; IEEE-829 1998].

2.2. Sistematização dos Testes

A sistematização tem como objetivo principal organizar o conjunto de atividades ou tarefas a serem realizadas durante os testes. A partir disso, diversos benefícios podem ser obtidos, tais como:

- Definição do conjunto de atividades a serem realizadas durante os testes e a ordem em que devem ser executadas;
- Definição de papéis e responsabilidades para cada atividade a ser realizada;



- Definição de prazos e metas a serem atingidas a cada momento durante os testes;
- Definição dos artefatos a serem produzidos ao longo das atividades de teste, e o momento da sua produção;
- Melhor alocação dos recursos definidos para o projeto, e;
- Gerenciamento da equipe de teste para um determinado projeto.

Como mecanismo para sistematização dos testes adotado neste trabalho, foi especificado um processo de testes de software que será responsável por sistematizar as atividades de teste em uma organização e direcionará a construção dos documentos descritos no IEEE *Std* 829, seguindo a metodologia definida em [Crespo *et al.* 2004], apoiando, assim, o planejamento e controle dos testes.

Um processo de testes de software envolve todos os passos necessários ao planejamento, execução e controle das atividades de teste ao longo do processo de desenvolvimento de um software. O objetivo a ser alcançado com a definição de um processo de testes de software é a especificação das tarefas, artefatos a serem produzidos e consumidos, os critérios a serem adotados e os responsáveis por cada tarefa definida, permitindo a sistematização dos testes e possibilitando a sua implantação, através de uma adaptação inicial, em organizações de software [Crespo *et al.* 2004].

As atividades e sub-atividades do processo de testes foram definidas baseando-se nas recomendações do IEEE *Std* 829 [IEEE-829 1998], atividades descritas na literatura de teste software e de atividades de gerência de projetos de software, obtidas a partir de diversas fontes de conhecimento, como CMMI [CMMI 2000] e PMBOK [PMBOK 2000]. Já os artefatos produzidos ao longo do processo de testes são os documentos especificados pelo IEEE *Std* 829 [IEEE-829 1998].

Além disso, o processo de testes de software deve possuir papéis específicos em relação ao processo de desenvolvimento, de acordo com informações extraídas do SWEBOOK (*Software Engineering Body of Knowledge*) [Bourque e Dupuis 2001]. Os papéis associados ao processo de testes de software são:

- **Gerente de Teste.** Pessoa responsável pelo planejamento e controle dos testes.
- **Projetista de Teste.** Pessoa responsável pelo projeto dos testes, incluindo seleção de abordagens de teste e identificação e especificação dos casos e procedimentos de teste.
- **Testador.** Pessoa responsável pela execução dos procedimentos de teste especificados a fim de detectar falhas no software e registro dos incidentes ocorridos durante os testes.

Para especificação do processo de testes de software foi utilizada uma abordagem para modelagem de processos organizacionais apresentada em [Villela 2004] e descrita na Figura 2, que permite a representação gráfica dos itens que compõem um processo.

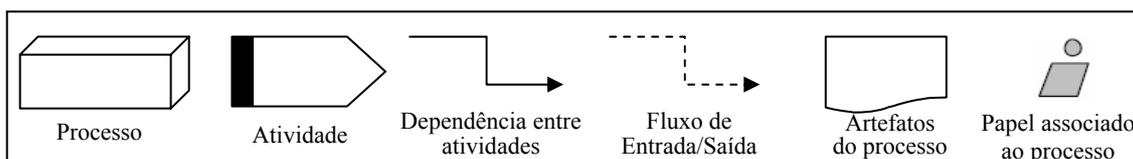


Figura 2. Notação usada para modelagem do Processo de Testes de Software



O processo de testes de software é composto por dois sub-processos: de planejamento e de execução dos testes, conforme visualizado na Figura 3.

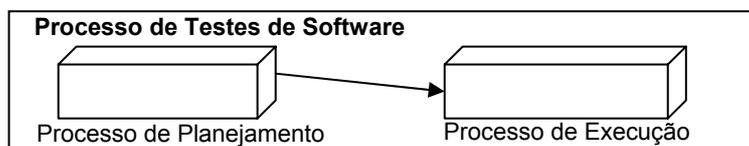


Figura 3. Sub-processos que compõem o Processo de Testes de Software

Cada sub-processo é composto por atividades, que por sua vez são responsáveis pela produção dos artefatos do processo de testes de software.

2.2.1. Processo de Planejamento dos Testes

O objetivo deste processo é planejar e definir todas as tarefas relacionadas às atividades de teste antes que comece a sua execução efetiva.

Este processo é composto por 4 macro-atividades, conforme visualizado na Figura 4: *Planejar Teste*, *Projetar Teste*, *Especificar Casos de Teste* e *Definir Procedimentos de Teste*. As atividades desse processo produzem, ao final, os seguintes documentos: *Plano de Teste*, *Especificação de Projeto de Teste*, *Especificação de Caso de Teste* e *Especificação de Procedimento de Teste*.

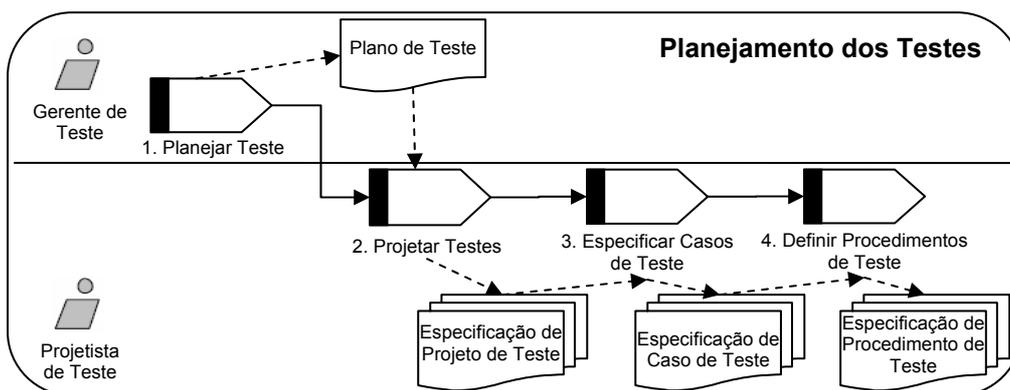


Figura 4. Processo de Planejamento dos Testes

2.2.2. Processo de Execução dos Testes

O objetivo deste processo é a execução dos testes a partir do que foi estabelecido durante o planejamento, monitorando as atividades realizadas, registrando os incidentes detectados e apresentando ao final os resultados dos testes.

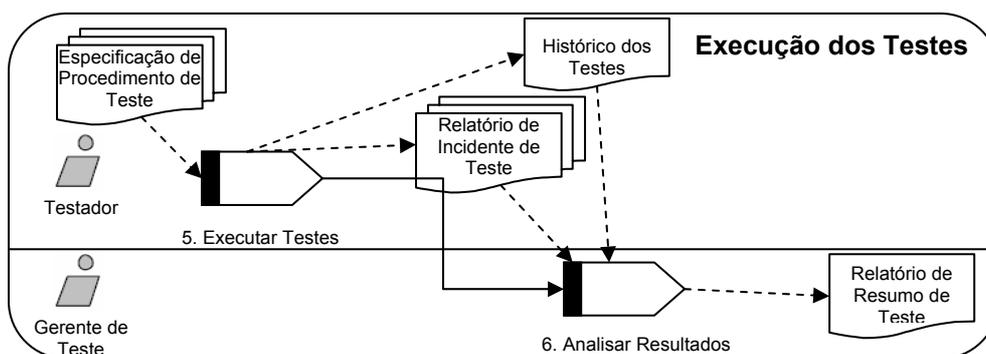


Figura 5. Processo de Execução dos Testes



O processo de Execução dos Testes é composto por 2 macro-atividades, conforme visualizado na Figura 5: *Executar Testes* e *Analisar Resultados*. Ao final deste processo serão estabelecidos os seguintes documentos: *Histórico dos Testes*, *Relatório de Incidente de Teste* e *Relatório de Resumo de Teste*.

Apesar da definição de uma abordagem com os elementos que apóiam o planejamento e controle dos testes (documentação e sistematização), a implantação e gerenciamento de um processo complexo, que envolve várias pessoas, atividades e tarefas repetitivas de forma manual, ou seja, sem um apoio automatizado ou semi-automatizado, torna-se uma tarefa muito custosa em um projeto de software.

Com o objetivo de minimizar o esforço no planejamento e controle dos testes de software em um projeto, foi construída uma infra-estrutura computacional, denominada Maraká¹, que visa à aplicação da abordagem proposta para apoiar o planejamento e controle dos testes de software. Esta infra-estrutura fornece um apoio automatizado para o acompanhamento das atividades do processo de testes de software e a construção dos diversos artefatos a serem produzidos ao longo deste processo.

3. A Infra-estrutura Maraká

Esta seção descreve os atributos da infra-estrutura e sua arquitetura, provendo apoio ao planejamento e controle dos testes de software. Por fim, será apresentado o seu funcionamento.

3.1 Atributos de Maraká

Maraká possui os seguintes atributos:

- Permite a criação de novas atividades de teste a serem gerenciadas;
- Permite o acesso aos dados das atividades de teste de acordo com os diferentes papéis associados ao processo de testes de software;
- Permite a construção, visualização e impressão de todos os artefatos do processo de testes de software citados na seção 2.2.;
- Possibilita o gerenciamento da equipe de teste de uma organização e o papel de cada membro da equipe no processo de testes, e a alocação dos membros da equipe em diferentes atividades de teste;
- Possibilita o controle dos testes através da visualização de gráficos que indicam o andamento e os resultados dos testes. Os gráficos são de:
 - **Análise de Procedimentos de teste:** visualiza o estado da execução dos procedimentos de teste em um gráfico de pizza agrupando de acordo com o seu resultado (aprovado, rejeitado ou não-executado);
 - **Incidentes Detectados:** visualiza o número de defeitos detectados durante os testes a cada dia, a fim de indicar um aumento ou diminuição do número de defeitos encontrados;
 - **Gantt:** visualiza graficamente o cronograma dos testes (planejado e real) através das atividades e sub-atividades do processo de testes, suas dependência e datas de início e término;

¹ Maraká é um instrumento usado pelo Pajé em rituais indígenas para expulsar os males que assombram a tribo. No nosso contexto, os males seriam as falhas reveladas em um produto e que tentamos eliminá-las.



- **Incidentes por Item de Teste:** visualiza número de incidentes em cada item de teste avaliado, a fim de indicar itens com maior probabilidade de falhas;
- Funciona através da Internet, possibilitando inclusive o acesso simultâneo de múltiplos usuários, e;
- Está disponível nos idiomas Português e Inglês, mas possibilita a adição de novos idiomas através de um mecanismo de localização.

Para utilização da infra-estrutura e suas funcionalidades, é necessária a criação de uma conta de acesso. Isso pode ser feito através de contato com os membros do grupo que a desenvolveu. Maraká está disponível no endereço <http://lens.cos.ufjf.br/maraka>.

3.2 Arquitetura de Maraká

A arquitetura de Maraká é representada por quatro componentes que controlam o acesso à infra-estrutura, a equipe de teste, as atividades de teste e o acompanhamento do processo de testes, conforme visualizado na Figura 6. Os componentes são:

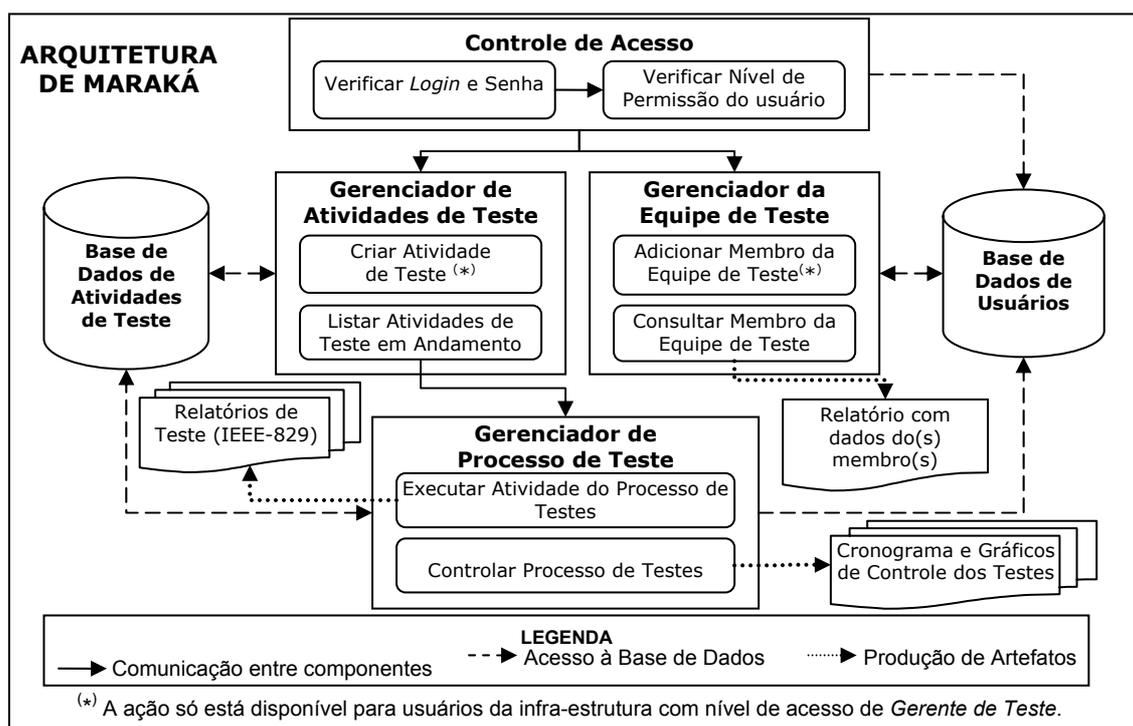


Figura 6. Arquitetura de Maraká

- **Controle de Acesso:** responsável por viabilizar o acesso à infra-estrutura e verifica o nível de permissão dos usuários, permitindo o acesso às suas funcionalidades somente por usuários previamente cadastrados na *Base de Dados de Usuários*. Após a validação do *login*, a infra-estrutura permitirá o acesso às funcionalidades dos componentes *Gerenciador da Equipe de Teste* e *Gerenciador de Atividades de Teste*.
- **Gerenciador da Equipe de Teste:** responsável pelo gerenciamento (adição, alteração e consulta) da equipe de teste (pessoas que terão acesso à infra-



estrutura) na *Base de Dados de Usuários*.

- **Gerenciador de Atividades de Teste:** responsável pela criação e consulta de atividades de teste na *Base de Dados de Atividades de Teste*. Este componente também lista todas as atividades de teste que ainda não foram concluídas nas quais o usuário que efetuou *login* está alocado, possibilitando o gerenciamento individualizado de cada atividade de teste que esteja em andamento através do componente *Gerenciador de Processo de Teste*.
- **Gerenciador de Processo de Teste:** responsável pelo gerenciamento do processo de testes de software a ser seguido para uma atividade de teste que esteja em andamento, respeitando suas dependências e os papéis associados ao processo. Permite, ainda, a consulta aos artefatos do processo, ao cronograma da atividade de teste, a informações sobre os membros da equipe de teste alocados para a atividade de teste e aos gráficos de acompanhamento dos testes. Este componente é o responsável pela documentação e sistematização dos testes através da infra-estrutura.

A partir da necessidade de utilização da infra-estrutura através da Internet e do controle do acesso dos usuários, foi cogitada a utilização de um *framework* para apoiar o atendimento a esses requisitos. Com isso, o *framework* Mambo² foi escolhido como base para o desenvolvimento da infra-estrutura por possuir diversas características:

- é um gerente de conteúdo dinâmico em *sites* ou aplicações que utilizam a Internet como plataforma, gerenciando páginas, menus, *links*, usuários, nível de acesso, *templates* (interface gráfica) da aplicação e idiomas, dentre outros;
- disponibiliza um mecanismo para controle de acesso aos usuários e ao conteúdo de uma aplicação;
- permite a criação e instalação de novos componentes na aplicação para a realização de novas funcionalidades que não são atendidas originalmente no *framework*;
- possui uma camada de interface com a base de dados, de forma que os dados armazenados na base possam ser acessados através desta interface, reduzindo o esforço de inserção, alteração, remoção e consulta aos dados na base, e;
- possui código aberto e possui diversos componentes disponíveis na sua comunidade (www.mamboforge.net) que podem ser instalados para acrescentar funcionalidades à aplicação.

Sendo assim, na arquitetura definida para a infra-estrutura (Figura 6), o *framework* Mambo instancia o componente *Controle de Acesso* e a *Base de dados de Usuários*. Dessa forma, foi necessário o desenvolvimento dos demais componentes e da *Base de Dados de Atividades de Teste*. Porém, devido à facilidade fornecida pelo *framework* Mambo, novos componentes podem ser facilmente acoplados à infra-estrutura através da interface provida por este *framework*, possibilitando a adição de novas funcionalidades e, conseqüentemente, a evolução da infra-estrutura Maraká.

O *framework* Mambo é desenvolvido na plataforma de desenvolvimento PHP e utiliza o banco dados MySQL como repositório dos seus dados. Dessa forma, todos os componentes na infra-estrutura foram desenvolvidos na plataforma PHP + MySQL.

² Mais informações sobre o *framework* Mambo podem ser obtidas em www.mamboforge.net.



3.3 Funcionamento de Maraká

A seguir, estão listadas as funcionalidades fornecidas por Maraká e serão apresentadas telas capturadas, descrevendo o apoio de Maraká a estas funcionalidades.

- **Gerenciando a Equipe de Teste**

Maraká permite a visualização/impressão de relatórios com os dados de um ou todos os membros da equipe de teste (ver Figura 7A), a adição de novos membros e a consulta aos dados dos membros cadastrados (Figura 7B).



Relatório de Membro da Equipe de Teste

(A) Gerado por Maraká - 11/02/2006

Consultar Membros da Equipe de Teste

Nome: Arilo Claudio Dias Neto

Função no Processo de Testes: Gerente de Teste

Posição: Gerente de Projeto

Telefone: 2542-6935

E-mail: ariloclaudio@hotmail.com

Website: www.cos.ufrj.br/~acd

Idade: 22

Número de Testes que Participou: 3

Habilidade: Liderança

Conhecimento: C++

Gerente de Teste							
Nome	Posição	Telefone	E-Mail	Website	Testes que participou	Idade	
Arilo Claudio Dias Neto	Gerente de Projeto	2542-6935			3	22	
Guilherme Travassos	Gerente da Equipe	2345678			20	50	

Projeta de Teste

Nome	Posição	Telefone	E-Mail	Website	Testes que participou	Idade	
Dessana	Analista	6543210			3	21	(B)

Testador

Nome	Posição	Telefone	E-Mail	Website	Testes que participou	Idade	
Rafael Barcelos	Programador	2345678			1	23	

Figura 7. Telas de Gerenciamento da Equipe de Teste

- **Gerenciando Atividades de Teste**

Maraká permite a criação de novas atividades de teste a serem acompanhadas através da infra-estrutura e consulta aos seus artefatos produzidos (Figura 8).

Informações sobre a Atividade de Teste

>> Identificador	<input type="text" value="SCE"/>
>> Título	<input type="text" value="Sistema de Controle de Estoque da Empresa X"/>
>> Descrição	Este projeto é responsável pelos testes funcionais do SCE 2.1, visando definidos pelos usu...
>> Versão	<input type="text" value="2.3"/>
>> Responsável	Guilherme Travassos
>> Data de Criação	<input type="text" value="10/03/2006"/>
>> Data de Previsão	<input type="text" value="13/09/2006"/>

Atividades de Teste de Software

Escolha uma das atividades de teste listadas e clique em seguida no botão CONSULTAR para visualizar os artefatos produzidos ao longo dessa Atividade de Teste.

Parâmetros: Identificador: Título:

Identificador	Título	Data de Criação	Data de Previsão	Versão
PT-11	SIGDEM 2.0	02/09/2005	13/10/2005	
TESTE01	Sistema DIAS LTDA	26/09/2005	31/12/1969	

Figura 8. Telas de Gerenciamento da Atividade de Teste

- **Gerenciando o Processo de Testes em uma Atividade de Teste**

No gerenciamento do processo para uma atividade de teste de software, Maraká



possibilita o acompanhamento do processo de testes de software para cada atividade de teste criada através da infra-estrutura (Figura 9A) e o controle dos testes através de relatórios (Figura 9B), cronograma (Figura 10A) ou gráficos (Figura 10B).



Figura 9. (A) Acompanhamento do Processo de Testes e (B) Plano de Testes produzido

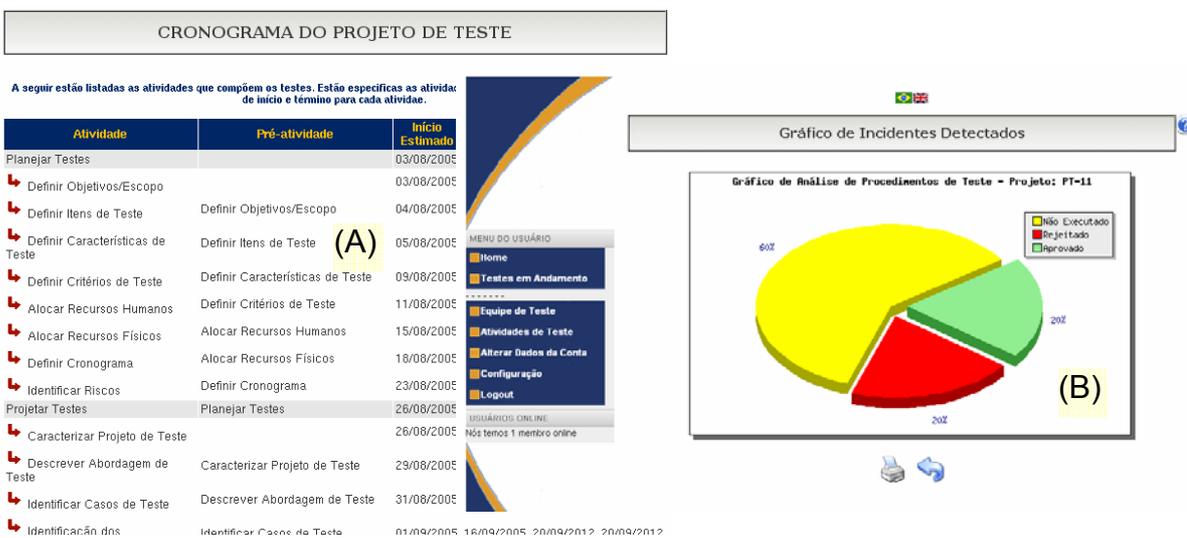


Figura 10. (A) Cronograma dos Testes, (B) Gráfico de Controle dos Testes

4. Avaliação e Limitações de Maraká

4.1 Análise das Práticas de Teste de Software Atendidas por Maraká

Maraká foi desenvolvida visando atender a necessidades reais de organizações de software, em relação às atividades de planejamento e controle de testes de software. Para analisar o apoio provido pela infra-estrutura, as sua funcionalidades foram avaliadas em relação ao apoio fornecido ao conjunto de práticas de teste de software consideradas importantes por profissionais de organizações de software, caracterizadas a partir da realização de um *survey* que avaliou o estado da prática de teste de software em um cenário de desenvolvimento de software específico [Dias Neto *et al.* 2006].



Apesar dos resultados obtidos neste *survey* não poderem ser generalizados para um contexto mais amplo, ele apresenta indícios do que está sendo aplicado e das reais necessidades em relação às atividades de teste de software em organizações de software brasileiras localizadas em um cenário específico.

Com isso, a Tabela 2 apresenta a avaliação de Maraká em relação ao apoio fornecido a essas práticas de teste. Para cada prática analisada, é apresentado um indicador do nível de atendimento e uma descrição do apoio fornecido pela infraestrutura à prática de teste de software. Nesta tabela, o ícone indica que a prática de teste foi atendida totalmente por Maraká, o ícone indica que a prática foi atendida parcialmente pela infra-estrutura e o ícone indica que a prática não foi atendida.

Pode ser observado que as duas práticas de teste de software atendidas parcialmente por Maraká estão relacionadas à organização dos testes dentro da empresa, e correspondem a decisões da gerência. Sendo assim, a sua realização não pode ser garantida pela infra-estrutura, porém elas podem ser aplicadas caso a gerência adote esta estratégia em seus projetos de software.

As duas práticas de teste de software não são atendidas pela infra-estrutura, *Re-execução dos testes quando o software é modificado* e *Avaliação dos documentos de teste (testware) quanto à sua qualidade*, representam limitações reais da infra-estrutura. As demais práticas consideradas importantes no *survey* estão atendidas por Maraká.

Tabela 2. Apoio de Maraká às práticas de teste importantes, segundo o *survey* [Dias Neto et al. 2006].

Práticas de Teste de Software consideradas Importante por Organizações de Software que participaram do <i>Survey</i>	Descrição do apoio fornecido	Atendimento
Existência de um responsável ou uma equipe alocada para as atividades de teste.	Apoio à especificação da equipe de teste de software para uma organização, de forma que somente essas pessoas possam ser alocadas nas atividades de teste.	
Realização de Teste de Aceitação.	Possibilidade de uso da infra-estrutura para apoiar o planejamento e controle das atividades de teste independente do tipo de teste a ser realizado.	
Realização de Teste de Unidade.		
Realização de Teste de Integração.		
Realização de Teste de Sistema.		
Documentação do Plano de Teste descrevendo objetivos/abordagens.	Apoio à criação do documento <i>Plano de Teste</i> de acordo com o IEEE Std 829, onde está inclusa uma seção com a definição dos objetivos e escopo dos testes a serem realizados.	
Registro de falhas e defeitos detectados durante os testes.	Apoio ao registro dos incidentes detectados nos testes, gerando um <i>Relatório de Incidente de Teste</i> para cada incidente de acordo com o IEEE Std 829.	
Re-execução dos testes quando o software é modificado.	Não possui um mecanismo para re-execução dos testes planejados. Para re-executar os testes e registrar os novos resultados obtidos, é necessário criar uma nova atividade de teste e seguir o processo de testes novamente.	
Utilização de metodologia ou processo para sistematizar e organizar testes de software.	Existe um processo de testes de software inserido na infra-estrutura, de forma que todas as atividades de teste a serem realizadas devem seguir este processo (seus papéis, dependências e critérios).	
Documentação de Procedimentos e Casos de Teste e seus Resultados.	Apoio à criação dos documentos <i>Especificação de Caso de Teste</i> e <i>Especificação de Procedimento de Teste</i> para cada caso ou procedimento de teste, e registro dos resultados dos procedimentos de teste executados no documento <i>Histórico dos Testes</i> , todos de acordo com o IEEE Std 829.	
Separação das atividades de teste do desenvolvimento.	Segue um processo de testes totalmente independente do processo de desenvolvimento de software.	
Realização de treinamentos sobre testes.	Apoio, durante a atividade de planejamento dos testes, à especificação dos treinamentos necessários para a realização dos testes para um projeto, indicando carga horária e a previsão de dias a serem gastos com o treinamento. No entanto, não pode ser garantida a realização do treinamento ou a sua qualidade.	



Existência de testadores em tempo integral para realização de testes.	Apoio à especificação da equipe de teste para uma organização, de forma que somente essas pessoas definidas possam ser alocadas nas atividades de teste. No entanto, não pode ser garantida a utilização dessas pessoas nas atividades de teste em tempo integral, pois isso depende da estrutura da organização.	
Conservação dos dados dos testes para usos futuros.	Armazenamento de todos os dados gerados ao longo dos testes e permite a qualquer momento a consulta a esses dados, como uma forma de apoio durante a realização de novas atividades de teste.	
Avaliação dos documentos de teste (testware) quanto à sua qualidade.	Apoio à criação dos documentos de teste, mas não possui mecanismo que apóie a avaliação da qualidade desses documentos.	
Utilização de ferramentas de gerência de teste para rastrear e registrar os resultados da execução de teste.	Maraká é uma ferramenta de apoio à gerência dos testes, e permite, entre outras tarefas, o registro dos resultados obtidos na execução dos testes.	

4.2 Trabalhos Relacionados

Algumas ferramentas que apóiam certas funcionalidades providas por Maraká em relação ao planejamento/controle de testes puderam ser identificadas e podem ser aplicadas ao longo de um projeto, como Rational Test Manager (www.ibm.com), ApTest Manager (www.aptest.com/atm2), TestLog (www.testlog.com) ou QATrac³ (<http://sourceforge.net/projects/qatraq/>) ou. No entanto, essas ferramentas possuem propósitos diferentes de Maraká, pois a maioria foi desenvolvida a fim de comercialização, ou apóiam somente um ponto específico do processo de testes ou requer uma forte adequação da organização (e suas atividades de teste) à metodologia inserida na ferramenta para a realização dos testes, ou seja, é difícil adaptar a ferramenta à organização. Isso torna uma eventual comparação entre essas ferramentas e Maraká dependente de critérios adequados e justos, pois deve mesclar aspectos relacionados a uma ferramenta de comercialização e uma ferramenta acadêmica.

É preciso ressaltar que uma das propostas e funcionalidades previstas em Maraká é a possibilidade de ajustar os dois elementos que compõem a abordagem descrita na Seção 2 (documentação e sistematização). O processo de testes seguido por Maraká pode ser modificado pela organização e escolhido para um projeto específico de acordo com suas necessidades (ver Figura 11, apesar de nessa tela só existir um processo para ser escolhido). O mesmo ocorre para os roteiros de documentos de teste utilizado por Maraká, que pode ser substituído ou modificado a qualquer momento a partir da necessidade da organização. Ambas as alternativas descritas na Seção 2 são pontos iniciais a serem utilizados por Maraká, mas podem ser adaptados.



Figura 11. Selecionando o Processo de Testes de Software em Maraká

5. Conclusões e Trabalhos Futuros

Neste artigo foi apresentada a infra-estrutura computacional Maraká, como mecanismo para simplificar através do apoio automatizado, à aplicação de uma abordagem de apoio

³ Esta ferramenta é gratuita.



ao planejamento e controle de testes de software composta por dois elementos básicos: **documentação e sistematização**.

Sendo assim, a infra-estrutura foi projetada e desenvolvida baseando-se em dois cenários: acadêmico, caracterizado através de revisões na literatura da área de teste de software, e prático, através dos resultados obtidos em um *survey* que avaliou o estado atual de práticas de teste de software em um cenário real de desenvolvimento de software brasileiro, e ao final identificou carências e limitações das organizações de software que compõem este cenário em relação à aplicação de práticas exclusivas de apoio ao planejamento e controle dos testes [Dias Neto *et al.* 2006]. Com isso, duas expectativas principais foram criadas para que a infra-estrutura proposta possa efetivamente trazer benefícios para o planejamento e controle de teste de software:

- Possibilitar a realização de atividades de teste de software de forma mais sistematizada a partir de um processo de testes de software que servirá como guia para a realização das tarefas necessárias em uma atividade de teste.
- Fornecer um apoio adequado para o controle e a documentação dos testes realizados, utilizando um padrão internacional para documentação dos testes, como forma de evidenciar as atividades realizadas ao longo do processo de testes, permitindo aos envolvidos nessas atividades acompanhar seus resultados ao longo do processo, e podendo, ainda, reaproveitar essas informações em novos projetos de software.

Como próximo passo deste trabalho está prevista a construção de novas funcionalidades que apoiarão outras práticas relacionadas ao planejamento e controle dos testes de software, como:

- Criação de um mecanismo de integração de ferramentas externas, a fim de aproveitar dados gerados por outras ferramentas;
- Criação de um mecanismo para reutilização de dados gerados em atividades de teste anteriores através de Maraká em novas atividades de teste, possibilitando, inclusive, a realização de testes de regressão, e;
- Implantação de um mecanismo para gerência de configuração dos artefatos produzidos por Maraká;
- Implantação de um mecanismo para gerência de conhecimento em Maraká, possibilitando o registro de lições aprendidas a partir de sua utilização.

Além disso, está prevista a realização de estudos experimentais com o objetivo de avaliar a viabilidade de utilização da infra-estrutura em ambientes reais de desenvolvimento de software. No entanto, o cenário atual indica que as organizações de software possuem dificuldades em gerenciar suas atividades de desenvolvimento e de testes. A aplicação de Maraká em ambientes reais de desenvolvimento requer uma organização inicial das atividades de desenvolvimento de software, pois os artefatos produzidos ao longo do desenvolvimento são insumos necessários para o planejamento e execução dos testes. Atualmente, a infra-estrutura está sendo aplicada em um projeto real de desenvolvimento de software baseado na web no contexto do grupo de Engenharia de Software Experimental da COPPE. Os dados obtidos por meio deste uso serão utilizados para a construção do plano do estudo de viabilidade de Maraká.

A partir dos resultados deste estudo de viabilidade, poderá ser observada a hipótese inicial que indica se a implantação de Maraká causa um impacto menor em organizações que não sejam completamente imaturas em relação a práticas de Engenharia de Software e que estão familiarizadas com a utilização de processos, pois o passo fundamental para a sua implantação é a utilização do processo de testes inserido na infra-estrutura. Sendo assim, as demais organizações, não tão familiarizadas com esses procedimentos, terão inicialmente dificuldades em utilizá-la, mas podem aplicá-la



como uma estratégia para sistematização das suas atividades de teste.

Agradecimentos

Agradecemos à FAPEAM e CNPq pelo apoio fornecido ao longo deste trabalho. Os resultados deste trabalho estão inseridos no contexto do Projeto “6.34: Infra-estrutura Computacional para apoiar o Planejamento e Controle de Testes de Software”, submetido ao Programa Brasileiro de Qualidade e Produtividade em Software – Ciclo 2005.

Referências Bibliográficas

- Beizer, B., “Software testing techniques”, 2nd ed., Van Nostrand Reinhold Co., New York, NY, 1990.
- Bourque, P., Dupuis, R., (2001) “Guide to the Software Engineering Body of Knowledge - SWEBOK”, Stoneman Trial Version 1.0, <http://www.swebok.org>, Maio.
- CMMI Product Development Team (2000): “CMMI-SE/SW: Capability Maturity Model – Integrated for Systems Engineering/Software Engineering”, version 1.0 continuous representation. Technical Report 2000-TR-019, Software Engineering Institute, Carnegie Mellon University, USA.
- Crespo, A. N., Silva, O. J., Borges, C. A., Salviano, C. F., Argollo, M., Jino, M. (2004) “Uma metodologia para teste de Software no Contexto da Melhoria de Processo”, In: III Simpósio Brasileiro de Qualidade de Software (SBQS 2004), Brasília.
- Dias Neto, A. C. (2004) “Caracterização de Abordagens de Planejamento e Controle de Testes de Software”, Monografia do Curso de Engenharia de Software Experimental (CPS820), COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.
- Dias Neto, A.C., Travassos, G.H. (2005) “Towards a Computerized Infrastructure for Software Testing Planning and Control”. In: 6º IEEE Latin-American Test Workshop (LATW), Salvador.
- Dias Neto, A. C., Natali, A.C., Rocha, A.R., Travassos, G.H. (2006) “Caracterização do Estado da Prática das Atividades de Teste de em um Cenário de Desenvolvimento de Software Brasileiro”, In: 5º Simpósio Brasileiro de Qualidade de Software (SBQS), Vila Velha, ES.
- Howden, W.E., “Functional program testing and analysis”. Nova York, NY, McGrall-Hill, 1987.
- IEEE *Standard 829-1998: Standard for Software Test Documentation*, IEEE Press, 1998.
- Juristo, N., Moreno, A. M., Vegas, S. (2004) “Reviewing 25 years of testing technique experiments”. *Empirical Software Engineering: An International Journal*, 9(1), Março.
- Mats, L. (2001) “The top five software-testing problems and how to avoid them”, *EDN Europe*, Fevereiro, Vol. 46 Issue 2, p37, 3p; (AN 4121152).
- Mcgregor, J. D., Sykes, D. A., “A practical guide to testing object-oriented software”, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 2001.
- PMBOK – “Project Management Body of Knowledge” (2000) PMI – Project Management Institute.
- Pressman, R. S., “Software Engineering: A Practitioner’s Approach”, McGraw-Hill, 6th ed, Nova York, NY, 2005.
- Rocha, A. R. C., Maldonado, J. C., Weber, K. C., et al., “Qualidade de software – Teoria e prática”, Prentice Hall, São Paulo, 2001.
- Villela, K. (2004) “Definição e Construção de Ambientes de Desenvolvimento de Software Orientados à Organização”, Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, maio. Disponível em <http://www.cos.ufrj.br/ta>.
- Whittaker, J.A. (2000) “What Is Software Testing? And Why Is It So Hard?”, *IEEE Software*, January/February.